ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Audit-Free Cloud Storage via Deniable Attribute based Encryption

G.Prashanth
B. Tech Scholar
Dept. of Information Technology,
Vignana Bharathi Institute of Technology,
Aushapur, Hyderabad-501301.

L.Nithish Reddy
B. Tech Scholar
Dept. of Information Technology,
Vignana Bharathi Institute of Technology,
Aushapur, Hyderabad-501301.

Mrs V.Ambica
Assistant Professor,
Dept. of Information Technology,
Vignana Bharathi Institute of Technology,
Aushapur, Hyderabad-501301.

Abstract
Cloud storage services have become increasingly popular. Because of the importance of privacy, many cloud storage encryption schemes have been proposed to protect data from those who do not have access. All such schemes assumed that cloud storage providers are safe and cannot be hacked; however, in practice, some authorities (i.e., coercers) may force cloud storage providers to reveal user secrets or confidential data on the cloud, thus altogether circumventing storage encryption schemes. In this paper, we present our design for a new cloud storage encryption scheme that enables cloud storage providers to create convincing fake user secrets to protect user privacy. Since coercers cannot tell if obtained secrets are true or not, the cloud storage providers ensure that user privacy is still securely protected.

Keywords—***Deniable Encryption, Composite Order Bilinear Group, Attribute-Based Encryption, Cloud Storage.***

INTRODUCTION
Cloud storage services have rapidly become increasingly popular. Users can store their data on the cloud and access their data anywhere at any time. Because of user privacy, the data stored on the cloud is typically encrypted and protected from access by other users. Considering the collaborative property of the cloud data, attribute-based encryption (ABE) is regarded as one of the most suitable encryption schemes for cloud storage. There are numerous ABE schemes that have been pro-posed.

Most of the proposed schemes assume cloud storage service providers or trusted third parties handling key management are trusted and cannot be hacked; however, in practice, some entities may intercept communications between users and cloud storage providers and then compel storage providers to release user secrets by us-ing government power or other means. In this case, encrypted data are assumed to be known and storage providers are requested to release user secrets. As an example, in 2010, without notifying its users, Google re-leased user documents to the FBI after receiving a search warrant . In 2013, Edward Snowden disclosed the ex-istence of global surveillance programs that collect such cloud data as emails, texts, and voice messages from some technology companies. Once cloud stor-age providers are compromised, all encryption schemes lose their effectiveness. Though we hope cloud storage providers can fight against such entities to maintain user privacy through legal avenues, it is seemingly more and more difficult. As one example, Lavabit was an email service company that protected all user emails from outside coercion; unfortunately, it failed and decided to shut down its email service.

Since it is difficult to fight against outside coercion, we aimed to build an encryption scheme that could help cloud storage providers avoid this predicament. In our approach, we offer cloud storage providers means to create fake user secrets. Given such fake user secrets, outside coercers can only obtained forged data from a user’s stored ciphertext. Once coercers think the received secrets are real, they will be satisfied and more impor-tantly cloud storage providers will not have revealed any real secrets. Therefore, user privacy is still protected.

This concept comes from a special kind of encryption scheme called **deniable encryption**, first proposed in Deniable encryption involves senders and receivers creating convincing fake evidence of forged data in ciphertexts such that outside coercers are satisfied. Note that deniability comes from the fact that coercers cannot prove the proposed evidence is wrong and therefore have no reason to reject the given evidence. This ap-proach tries to altogether block coercion efforts since coercers know that their efforts will be useless. We make use of this idea such that cloud storage providers can provide audit-free storage services. In the cloud storage scenario, data owners who store their data on the cloud are just like senders in the deniable encryption scheme. Those who can access the encrypted data play the role of receiver in the deniable encryption scheme, including the cloud storage providers themselves, who have system-wide secrets and must be able to decrypt all encrypted data[1].

In this work, we describe a deniable ABE scheme for cloud storage services. We make use of ABE character-istics for securing stored data with a fine-grained access control mechanism and deniable encryption to prevent outside auditing. Our scheme is based on Waters cipher-text policy-attribute based encryption (CP-ABE) scheme. We enhance the Waters scheme from prime order bilinear groups to composite order bilinear groups. By the subgroup decision problem assumption, our scheme enables users to be able to provide fake secrets that seem legitimate to outside coercers.

Previous Work on ABE

Sahai and Waters first introduced the concept of ABE in which data owners can embed how they want to share data in terms of encryption . That is, only those who match the owner's conditions can successfully decrypt stored data. We note here that ABE is encryption for privileges, not for users. This makes ABE a very useful tool for cloud storage services since data sharing is an important feature for such services. There are so many cloud storage users that it is impractical for data owners to encrypt their data by pairwise keys. Moreover, it is also impractical to encrypt data many times for many people. With ABE, data owners decide only which kind of users can access their encrypted data. Users who satisfy the conditions are able to decrypt the encrypted data.

There are two types of ABE, CP-ABE and Key-Policy ABE (KP-ABE). The difference between these two lies in policy checking. KP-ABE is an ABE in which the policy is embedded in the user secret key and the attribute set is embedded in the ciphertext. Conversely, CP-ABE embeds the policy into the ciphertext and the user secret has the attribute set. Goyal et al. proposed the first KP-ABE .They constructed an expressive way to relate any monotonic formula as the policy for user secret keys. Bethencourt et al. proposed the first CP-ABE in . This scheme used a tree access structure to express any monotonic formula over attributes as the policy in the ciphertext. The first fully expressive CP-ABE was proposed by Waters which used Linear Secret Sharing Schemes (LSSS) to build a ciphertext policy. Lewko et al. enhanced the Waters scheme to a fully secure CP-ABE, though with some efficiency loss, in Recently, Attrapadung et al. constructed a CP-ABE with a constant-size ciphertext and Tysowski et al. designed their CP-ABE scheme for resource-constrained .

Previous Work on Deniable Encryption

The concept of deniable encryption was first proposed in . Like normal encryption schemes, deniable encryp-tion can be divided into a deniable shared key scheme and a public key scheme. Considering the cloud storage scenario, we focus our efforts on the deniable public key encryption scheme.

There are some important deniable public key en-cryption schemes[2]. Canetti et al. used translucent sets to construct deniable encryption schemes .A translucent set is a set containing a trapdoor subset. It is easy to randomly pick an element from the universal set or from the subset; however, without the trapdoor, it is difficult to determine if a given element belongs to the subset. Canetti et al. showed that any trapdoor permutation can be used to construct the translucent set. To build a deniable public key encryption scheme from a translucent set, the translucent set is the public key and the trapdoor is the private key. The translucent set is used to represent one encrypted bit. Elements in the subset are represented by 1 whereas other non-subset elements are represented by 0. The sender can encrypt 1 by sending an element in the subset, but can claim the element is chosen from the universal set . The above is a basic sender-deniable scheme. Canetti et al. also proved that a sender-deniable scheme can be transformed to a receiver-deniable scheme or a bi-deniable scheme with the help of intermediaries. There is research on how best to design a translucent set. Durmuth et al. designed the translucent set from the samplable encryption in . ONeill et al. designed the bi-translucent set from a lattice , which can build a native bi-deniable scheme.

In addition to the bitranslucent set, there are other proposed approaches to building deniable encryption schemes. ONeill et al. proposed a new deniable method through a simulatable public key system . The sim-ulatable public key system provides an oblivious key generation function and an oblivious ciphertext function. When sending an encrypted bit, the sender will send a set of encrypted data which may be normally encrypted or oblivious. Therefore, the sender can claim some sent messages are oblivious while actually they are not. The idea can be applied to the receiver side such that the scheme is a bi-deniable scheme. In Gasti et al. proposed another deniable scheme in which one public-private key pair is set up for each user while there are actually two pairs. The sender can send a true message encrypted by one key with a fake message encrypted by the other key. The sender decides which key is released according to the coercer’s identity. Gasti et al. also applied this idea to cloud storage services. There are still other deniable encryption schemes.

Aside from the above deniable schemes, there is research investigating the limitations of the deniable schemes. In Nielsen states that it is impossible to encrypt unbounded messages by one short key in non-committing schemes, including deniable schemes. In Bendlin et al. shows that noninteractive and fully receiver-deniable schemes cannot be achieved simultane-ously. We construct our scheme under these limitations.

Our contribution

In this work, we construct a deniable CP-ABE scheme that can make cloud storage services secure and audit-free. In this scenario, cloud storage service providers are just regarded as receivers in other deniable schemes.

Unlike most previous deniable encryption schemes, we do not use translucent sets or simulatable public key systems to implement deniability. Instead, we adopt the idea proposed in with some improvements. We construct our deniable encryption scheme through a multidimensional space. All data are encrypted into the multidimensional space. Only with the correct com-position of dimensions is the original data obtainable. With false composition, ciphertexts will be decrypted to predetermined fake data. The information defining the dimensions is kept secret. We make use of composite order bilinear groups to construct the multidimensional space. We also use chameleon hash functions to make both true and fake messages convincing.

Our deniable ABE has the advantages described below over previous deniable encryption schemes.

Blockwise Deniable ABE

Most deniable public key schemes are bitwise, which means these schemes can only process one bit a time; therefore, bitwise deniable encryption schemes are inefficient for real use, especially in the cloud storage service case. To solve this problem, O'Neil et al. designed a hybrid encryption scheme that simultaneously uses symmetric and asymmetric en-cryption. They use a deniably encrypted plan-ahead symmetric data encryption key, while real data are encrypted by a symmetric key encryption mecha-nism. This reduces the repeating number from the block size to the key size. Though bitwise deniable encryption is more flexible than blockwise deniable encryption in ”cooking” fake data, when consider-ing cloud storage services, blockwise encryption is much more efficient in use.

Unlike those techniques used in previous deniable encryption schemes, we build two encryption en-vironments at the same time, much like the idea proposed in . We build our scheme with multiple dimensions while claiming there is only one dimension. This approach removes obvious redundant parts in . We apply this idea to an existing ABE scheme by replacing prime order groups with composite order groups. Since the base ABE scheme can encrypt one block each time, our deniable CP-ABE is certainly a blockwise deniable encryption scheme. Though the bilinear operation for the com-posite order group is slower than the prime order group, there are some techniques that can convert an encryption scheme from composite order groups to prime order groups for better computational per-formance,. We use composite order groups to describe our idea in Section 4 and transform it to prime order groups

Consistent Environment

Most of the previous deniable encryption schemes are inter-encryption-independent. That is, the encryption parameters should be totally different for each encryption op-eration. If two deniable encryptions are performed in the same environment, the latter encryption will lose deniability after the first encryption is coerced, because each coercion will reduce flexibility. For example, once coercers get private keys, which are the most common receiver proofs, these keys should be convincing not only under some particular files, but also under all related stored data. Otherwise, the coercers will know that these keys are fake; however, all proposed schemes only provide con-vincing proofs for particular transmissions. In the secure cloud storage service, this is not practical. It is impossible for a cloud storage service provider to prepare a unique encryption environment for each file, much less to maintain the access control mechanism at the same time.

In this work, we build a consistent environment for our deniable encryption scheme. By consistent environment, we means that one encryption envi-ronment can be used for multiple encryption times without system updates. The opened receiver proof should look convincing for all ciphertexts under this environment[3], regardless of whether a ciphertext is normally encrypted or deniably encrypted. The deniability of our scheme comes from the secret of the subgroup assignment, which is determined only once in the system setup phase. By the canceling property and the proper subgroup assignment, we can construct the released fake key to decrypt nor-mal ciphertexts correctly.

Deterministic Decryption

Most deniable encryp-tion schemes have decryption error problems. These errors come from the designed decryption mecha-nisms. Canetti et al. uses the subset decision mechanism for decryption. The re-ceiver determines the decrypted message according to the subset decision result. If the sender chooses an element from the universal set but unfortunately the element is located in the specific subset, then an er-ror occurs. The same error occurs in all translucent-set-based deniable encryption schemes. Another ex-ample is in , which uses a voting mechanism for decryption. Decryption is correct if and only if the correct part overwhelms the false part. Otherwise, the receiver will get the error result.

The concept of our deniable scheme is different than these schemes described above. Our scheme extends a pairing ABE, which has a deterministic decryption algorithm, from the prime order group to the composite order group. The decryption algorithm in our scheme is still deterministic; therefore, there is no decryption errors using our scheme.

Organization

In additional to this introductory section, we introduce preliminaries used in this paper in Section 2. In Section 3, we formally define deniable CP-ABE and its properties. In Section 4, we show how to set up a basic deniable CP-ABE scheme and prove security, deniability and other features of our scheme. In Section 5, we transform our basic scheme from composite order groups to prime order groups. We then enhance our scheme to be chosen-ciphertext attack (CCA) secure in Section 6. In section 7, we implement our deniable schemes and evaluate their performance. Finally, we present our conclusions in Section 8.

DEFINITION

Deniable CP-ABE Scheme

Deniable encryption schemes may have different prop-erties and we provide an introduction to many of these properties below.

Ad hoc deniability vs. plan-ahead deniability: The for-mer can generate a fake message (from the entire message space) when coerced, whereas the latter re-quires a predetermined fake message for encryption. Undoubtedly, all bitwise encryption schemes are ad hoc.

Sender-, receiver-, and bi-deniability: The prefix here in each case implies the role that can fool the coercer with convincing fake evidence. In sender-deniable encryption schemes and receiver-deniable schemes, it is assumed that the other entity cannot be coerced. Bi-deniability means both sender and receiver can generate fake evidence to pass third-party coercion.

Full deniability vs. multi-distributional deniability: A fully deniable encryption scheme is one in which there is only one set of algorithms, i.e., a key-generation algorithm, an encryption algorithm and so on. Senders, receivers and coercers know this set of algorithms and a sender and a receiver can fool a coercer under this condition. As for multi-distributional deniable encryption schemes, there are two sets of algorithms, one being a normal while the other is a deniable set. The outputs of algorithms in these two sets are computationally in-distinguishable. The normal set of algorithms cannot be used to fool coercers, whereas the deniable set can be used. A sender and a receiver can use the deniable algorithm set, but claim that they use the normal algorithm set to fool coercers..

Interactive encryption vs. non-interactive encryption: The difference between these two types of encryp-tion is that the latter scheme does not need interac-tion between sender and receiver.

According to the above definitions, the ideal deniable encryption scheme is ad hoc, full, bi-deniability and non-interactive deniability; however, there is research focused on determining the limitations of the deniable schemes. IN Nielsen stated that it is impossible to encrypt un-bounded messages by one short key in non-committing schemes, including deniable schemes.

Since we want our scheme to be blockwise deniable with a consistent encryption environment, we design our scheme to be a plan-ahead deniable encryption scheme. In [21], Bendlin et al. showed that non-interactive and fully receiver-deniable properties cannot be achieved simultaneously. We prefer our scheme to have the non-interactive prop-erty for ease of use. Therefore, our scheme is multi-distributional. In summary, our deniable scheme is plan-ahead, bi-deniable, and multi-distributional. Below, we provide the definition of this kind of deniable CP-ABE scheme.

Definition 7 (Deniable CP-ABE): Our plan-ahead, bi-deniable, and multi-distributional CP-ABE scheme is composed of the following algorithms:
Setup(1^{λ}) → (P P, MSK): This algorithm takes security parameter λ as input and returns public parameter P P and system master key MSK.

KeyGen(MSK, S) → SK: Given set of attributes S and MSK, this algorithm outputs private key SK.

Enc(P P, **M**, A) → C: This encryption algorithm takes as input public parameter P P , message **M**, and LSSS access structure A = (M, ρ) over the universe of attributes. This algorithm encrypts **M** and outputs a ciphertext C, which can be decrypted by those who possess an attribute set that satisfies access structure A. Note that A is contained in C.

Dec(P P, SK, C) → {**M**, ⊥}: This decryption algorithm takes as input public parameter P P , private key SK with its attribute set S, and ciphertext C with its access structure A. If S satisfies A, then this algorithm returns M ; otherwise, this algorithm returns ⊥.

OpenEnc(P P, C, **M**) → P_E : This algorithm is for the sender to release encryption proof P_E for (M, C).

OpenDec(P P, SK, C, **M**) → P_D : This algorithm is for the receiver to release decryption proof P_D for (M, C).

Verify(P P, C, M, P_E , P_D) → {T, F }: This algorithm is used to verify the correctness of P_E and P_D.

DenSetup(1^{λ}) → (P P, MSK, P K): This algorithm takes security parameter λ as input and returns public parameters P P , system master key MSK, and system public key P K. P K is known by all system users and is kept secret to outsiders.

DenKeyGen(MSK, S) → (SK, F K): Given set of attributes S and MSK, this algorithm outputs private key SK as well as F K for the user, where F K will be used for generating fake proof later.

DenEnc(P P, P K, **M**, **M**$'$, A) → C$'$: Aside from the inputs of the normal encryption algorithm, this deniable encryption algorithm needs public key P K and fake message **M**$'$. The output ciphertext must be indistinguishable from the output of **Enc**.

DenOpenEnc(P P, C$'$, **M**$'$) → P_E' : This algorithm is for the sender to release encryption proof P_E' for fake message M. The output must be indistinguish-able from the result of **OpenEnc** and must pass the **Verify** algorithm.

DenOpenDec(P P, SK, F K, C$'$, **M**$'$) → P_D': This al-gorithm is for the receiver to release decryption proof P_D' for fake message **M**$'$. The output must be indistinguishable from the result of **OpenDec** and must pass the **Verify** algorithm.

We require the following properties:
Security: The tuple **{Setup**,**KeyGen**,**Enc**,**Dec}** must form a secure CP-ABE scheme in a security model. In this work, we propose a CPA secure scheme and a CCA secure scheme. These two security models are defined in Section 3.2.

Bi-deniability: The CP-ABE is bi-deniable if, given public parameter P P, the two distribution tuples (M, C, P_E, P_D) and (M', C', P_E', P_D') are computational indistinguishable, where M, M ′ are claimed messages, C, C′ are normally and deniably encrypted ciphertexts, respectively, and PE , PD , PE′ , PD′ are proofs generated from the nor-mal and deniable open algorithms, respectively. That is, there is no PPT algorithm A for which is non-negligible.

$$Adv_A := \left| P[A(P\ P, (M, C, P_E, P_D)) = 1] - P[A(P\ P, (M', C', P_E', P_D')) = 1] \right|$$

Deniable Receiver Proof Consistency: The deni-able CP-ABE is deniable receiver proof consistent if a deniable receiver proof is convincing even when considering all ciphertexts in the system. That is, given set of ciphertexts C, including normally encrypted ciphertexts and deniably encrypted cipher-texts, normal proof P_D and deniable proof $P_D{}'$, there is no PPT algorithm A for which
$Adv_A := |P [A(C, P_D) = 1] − P [A(C, P_D{}') = 1]|$ is non-negligible.

We note that the last requirement is unusual for deni-able encryption schemes. We build our scheme with this requirement for practicality. In a cloud it is impractical to frequently update security parame-ters. Therefore, coercers are able to check proofs with all stored encrypted files. For normal provided proofs, there will be no problems. So, our scheme must ensure deniable proofs to pass coercer checks, or coercers will know cheating has occurred. We also note that not all stored files are deniably encrypted. Some files are nor-mally encrypted. A proposed receiver proof, regardless of whether it is normal or deniable, should be convincing for both normally and deniably encrypted files. We focus on receiver proofs instead of sender proofs because in most cases, senders add randomness during encryption. Therefore, any two sender proofs are usually indepen-dent, and sender proof consistency is unnecessary. For the above reasons, we build our scheme such that it adheres to the **Deniable Receiver Proof Consistency** requirement.

IS A CONFIDENTIAL PK PRACTICAL?

In the above definition, our scheme assumes that P K will be kept secret from the coercer. Some may argue that it is impractical, stating that coercers can pretend to be users in cloud storage services and obtain the P K. Once the P K is released to coercers, they can easily generate deniably encrypted ciphertexts and use these ciphertexts to determine the types of receiver proofs. To address this question, we must return to the basic assumption of deniable encryption schemes, i.e., **senders and receivers want to hide their communication messages from outside coercers**. Like all other cryptographic schemes, secrets must be assumed to be unknown to adversaries and our scheme is no exception. Therefore assuming that the P K is kept secret to coercers is acceptable and unavoidable.

To keep P K secret, cloud service providers can in-tegrate deniable CP-ABE schemes with their own user authentication mechanisms. Note that in our definition, a deniable CP-ABE scheme can enable cloud storage service providers to offer two kinds of storage services, one being normal storage service, the other being audit-free storage service. So a user can choose to enjoy normal cloud storage services through a basic authenti-cation process or enjoy audit-free cloud storage services through a much more sincere authentication process. Therefore, we believe our idea can be used to build practical cloud storage services, especially for those communities who currently have serious authentication processes.

Chosen-Plaintext-Attack (CPA) Security Model and Chosen-Ciphertext-Attack (CCA) Security Model

Here we describe the secure model for a CP-ABE scheme. An adversary is given a challenge question and is al-lowed to query an oracle for some information. The adversary wins the game if it can correctly answer the question. The formal security game is described as follows:
Setup: The challenger first runs **Setup** and outputs P P to the adversary.

Phase 1: The adversary generates queries $q_1, . . . , q_m$ to the challenger. Query q_i can be one of the follow-ing two types of queries:

DENIABLE CP-ABE CONSTRUCTION

To build an audit-free secure cloud storage service, we use a deniable CP-ABE scheme as our core technology. We construct our basic deniable CP-ABE scheme, which is based on , as follows:

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Setup(1^{λ}) → (P P, MSK): This algorithm generates bilinear group G of order N = $p_1p_2p_3$, where p_1, p_2, p_3 are distinct primes with bilinear map func-tion e : G × G → G_T . G_T is also order N . We let G_{p1} , G_{p2} , G_{p3} denote three orthogonal subgroups in G of order p_1, p_2, p_3, respectively. This algorithm then picks generators $g_1 \in G_{p1}$, $g_3 \in G_{p3}$, and randomly picks α, a ∈ Z_N . This algorithm also chooses hash function H_1 : **{**0, 1**}*** → G_{p3} . Public pa-rameter P P is **{**G, e, H_1, g_1g_3, $(g_1g_3)^{\mathbf{a}}$, $e(g_1g_3, g_1g_3)^{\alpha}$**}** and system secret key MSK is $(g_1g_3$

KeyGen(MSK, S) → SK: Given set S of attributes, this algorithm chooses t ∈ Z_N randomly and out-puts private key SK as:
SK = **{**$(g_1g_3)^{\alpha+at}$, $(g_1g_3)^{t}$, **{**$H_1(x)^{t}$**}**$_{\forall x \in S}$ **}**
= **{**K, L, **{**K_x**}**$_{\forall x \in S}$ **}**.

Enc(P P, **M**, A = (M, ρ)) → C: Given message **M** and LSSS access structure (M, ρ). Let M be a

l × n matrix and M_i denote the ith row of M . This algorithm first chooses two random vectors v = (s, y_2, . . . , y_n) ∈ Z_N^n and r = (r_1, . . . , r_l) ∈ Z_N^l . This algorithm then calculates $\lambda_i = v M_i$, ∀ i ∈ {1, . . . , l} .

In addition, this algorithm sets up one-way hash function $H(\cdot, \cdot)^4$ with two inputs. Note that hash function H can be any kind of one-way function and is determined during encryption. Each transaction may have different H. This algorithm flips two coins b_0, b_1 and picks two random string t_0, t_1. The output ciphertext C will be:
C = **{**A_0, A_1, B, (C_1, D_1), . . . , (C_l, D_l), H, t_0, t_1, V **}**, where,
αs $^{\mathbf{R}}$ A_{b0} = **M** · $e(g_1g_3, g_1g_3)$, A_{1-b0} ←− G_T , B = $(g_1g_3)^s$,
$C_i = (g_1g_3)^{a\lambda i} H_1(\rho(i))^{-ri}$, $D_i = (g_1g_3)^{ri}$, i = 1 . . . l, V = H(**M**, t_{b1}) **6**= H(A_{1-b0} · $e(g_1g_3, g_1g_3)^{-\alpha s}$, t_{1-b1}).
Access structure A is also attached to C.

Dec(P P, SK, C) → **{M**, ⊥**}**: To decrypt ciphertext C for access structure A = (M, ρ), this algorithm first checks if attribute set S of SK satisfies A. Suppose S satisfies A and let I ⊂ **{**1, 2, . . . , l**}** be defined as I = **{**i : ρ(i) ∈ S**}**. Then this algorithm finds a set **P** of constants **{**w ∈ $Z_{\mathbf{p}}$**}** such that $_{\mathbf{i \in I}}$ $w_{\mathbf{i}}\lambda_{\mathbf{i}}$ = s. This algorithm computes $\mathbf{M}_0$, $\mathbf{M}_1$ as follows:

$$M[0,1] = A(0,1) \cdot \frac{\prod_{i \in I} (e(C_i, L)e(D_i, K_{\rho(i)}))^{w_i}}{e(B, K)}$$

This algorithm then calculates

$$v_{i,j} = H(\mathbf{M}_i, t_j), \forall i, j \in \{0, 1\}.$$

If $v_{\mathbf{i,j}}$ is equal to V , then $\mathbf{M}_{\mathbf{i}}$ is the true message and is returned. Otherwise, this algorithm returns ⊥.

OpenEnc(P P, C, **M**) → $P_{\mathbf{E}}$: This algorithm returns two coins b_0, b_1 as proof $P_{\mathbf{E}}$.

OpenDec(P P, SK, C, **M**) → $P_{\mathbf{D}}$: This algorithm di-rectly returns SK as proof $P_{\mathbf{D}}$ since this is the most persuasive proof.

Verify(P P, C, **M**, $P_{\mathbf{E}}$, $P_{\mathbf{D}}$) → **{**T, F **}**: To verify $P_{\mathbf{E}}$ and $P_{\mathbf{D}}$, this algorithm first runs **Dec**(P P, $P_{\mathbf{D}}$, C) and checks if the output is equal to declared input **M**. Then, this algorithm checks $P_{\mathbf{E}}$ with correct

We use H to represent a hash function's public information and H(·, ·) to represent the hash operation. coins b_0, b_1 derived in the decryption process. If both requirements are satisfied, this algorithm returns T ; otherwise, it returns F .

- **DenSetup**(1^{λ}) → (P P, MSK, P K): This algo-rithm runs **Setup**(1^{λ}) and obtains P P. Sys-tem public key P K is {g_2g_3, $(g_2g_3)^{a}$, $e(g_3, g_3)^{\alpha}$, $e(g_2g_3, g_2g_3)^{\alpha}$} and system secret key MSK is {$(g_1g_3)^{\alpha}$, $g_1g_2g_3$, $(g_1g_2g_3)^{\alpha}$}.

Correctness

In this subsection, we show the correctness of this deni-able CP-ABE scheme. There are four cases here:

1) When using ormal key SK to decrypt normally encrypted ciphertext C, the decryption process will be:

$$\frac{Q_{i\in I}(e(C_i, L)e(D_i, K_{\rho(i)}))^{w_i}}{e(B, K)}$$

$$= \frac{Q_{i\in I}\ (e((g_1g_3)^{a\lambda_i}\ ,\ (g_1g_3)^{t}))^{w_i}}{e((g_1g_3)^{s}, (g_1g_3)^{\alpha+at})}$$

$$= \frac{e(g_1g_3, g_1g_3)^{at\ \sum_{i\in I}\lambda_i w_i}}{e(g_1g_3, g_1g_3)^{s(\alpha+at)}}$$

$$= e(g_1g_3, g_1g_3)^{-\alpha s}.$$

With the hash function in C and V , the receiver can derive message **M**.

2) When using normal key SK to decrypt deniable ciphertext C', the decryption process will be:

$$\frac{Q_{i\in I}(e(C_i', L)e(D_i', K_{\rho(i)}))^{w_i}}{e(B', K)}$$

$$= \frac{Q_{i\in I}\ (e((g_2g_3)^{a\lambda_i}\ ,\ (g_1g_3)^{t}))^{w_i}}{e((g_2g_3)^{s}, (g_1g_3)^{\alpha+at})}$$

$$= \frac{e(g_3, g_3)^{at\ \ i\in I\ \lambda_i w_i}}{e(g_3, g_3)^{s(\alpha+at)}}$$

$$= e(g_3, g_3)^{-\alpha s}.$$

With chameleon hash CH and V in C′, the receiver can derive true message M. Therefore, via the normal key, the receiver can obtain the correct mes-sage regardless of whether the message is normally encrypted.

3) When using deniable key F K to decrypt deniable ciphertext C', which is the case for fooling the coercer, the decryption process will be:

$$\frac{Q_{i\in I}(e(C_i', L')e(D_i', K'_{\rho(i)}))^{w_i}}{e(B', K)}$$

$$= \frac{Q_{i\in I}\ (e((g_2g_3)^{a\lambda_i}\ ,\ (g_1g_2g_3)^{t}))^{w_i}}{e((g_2g_3)^{s}, (g_1g_2g_3)^{\alpha+at})}$$

$$= \frac{e(g_2g_3, g_2g_3)^{at\ \ i\in I\ \lambda_i w_i}}{e(g_2g_3, g_2g_3)^{s(\alpha+at)}}$$

$$= e(g_2g_3, g_2g_3)^{-\alpha s}.$$

With chameleon hash CH and V in C', the receiver can derive fake message M'. Therefore, the coercer will be convinced by M and F K.

4) When using deniable key F K to decrypt normal ciphertext C, which is the key compatible property, the decryption process will be:

$$\frac{Q_{i\in I}(e(C_i, L')e(D_i, K'_{\rho(i)}))^{w_i}}{e(B, K')}$$

$$= \frac{Q_{i\in I}\ (e((g_1g_3)^{a\lambda_i}\ ,\ (g_1g_2g_3)^{t}))^{w_i}}{e((g_1g_3)^{s}, (g_1g_2g_3)^{\alpha+at})}$$

$$= \frac{e(g_1g_3, g_1g_3)^{at\ \ i\in I\ \lambda_i w_i}}{e(g_1g_3, g_1g_3)^{s(\alpha+at)}}$$

$$= e(g_1g_3, g_1g_3)^{-\alpha s}.$$

Therefore, correct message **M** will be derived from normal ciphertext C, even though the key is deni-able.

Therefore, correct message **M** will be derived from normal ciphertext C, even though the key is deni-able.

From the above, our scheme has two important prop-erties. First, a user can obtain the true message with a valid secret key, regardless of whether the ciphertext is normally encrypted or deniably encrypted. Second, the fake key can be used to decrypt the normally encrypted ciphertext.

Theorem 2: Our CP-ABE system is receiver proof con-sistent.

Proof: In our scheme, we use keys as receiver proofs since keys are the most immediate proofs available. As shown above, both P_D and P_D' can be used to ”correctly” decrypt these ciphertexts. By ”correctly” here, we mean that a ciphertext can be decrypted to a meaningful message, which may be true or a pre-determined fake message. With P_D , regardless of whether a message is normally encrypted or deniably encrypted, the true message can be derived. As for P_D', the decryption outputs are true messages when they are normally en-crypted and are fake messages when true messages are deniably encrypted. Therefore, anyone who can differentiate between $(C_1, \ldots, C_n, P_D)$ and $(C_1, \ldots, C_n, P_D')$ can also differentiate between true and pre-determined fake messages. In other words, these two tuples are indistinguishable. □

Security Proof

To prove that our deniable encryption scheme is secure requires this scheme to be a valid encryption scheme. For a multi-distributional deniable encryption scheme, it is only necessary to prove the security from the normal algorithm set. That is, we only need to prove the security of a scheme composed of the following four algorithms **Setup**, **KeyGen**, **Enc**, and **Dec**. As for the deniable algorithms, since deniable keys and ciphertexts are indistinguishable from normal keys and ciphertexts, which will be proved in the next subsection, deniable algorithms will be treated as normal algorithms which are proved to be secure. In other words, if the normal algorithm set can form a secure scheme, but the deniable set cannot, the security test will be a tool to distinguish these two sets of algorithms and there will be no denia-bility in our scheme. For proving security, we will reduce Waters CP-ABE to our deniable ABE scheme.

Theorem 3: Our proposed CP-ABE scheme is CPA se-cure if Waters CP-ABE is CPA secure.

Proof: Let **A** be an adversary that breaks the above ..deniable CP-ABE scheme. We can construct algorithm **B** that can break Waters CP-ABE as follows. **B** is given public parameters through the Waters CP-ABE scheme's **Setup** algorithm from challenger **X**

P P_{p3} := {g_3, g_3^{a3}, $e(g_3, g_3)^{\alpha 3}$ },

with prime number p_3, G_{p3} , $e(\cdot, \cdot)$ and $H_1(\cdot)$. For convenience, we use the suffix to represent different sub-groups in our proof. Algorithm **B** proceeds as follows.

Setup: **B** first picks two different prime numbers p_1 and p_2. Next, **B** generates group G with order $N = p_1p_2p_3$. Note that the subgroup with p_3 order in G should be the same as G_{p3} . **B** sets up P P_{p1} with the Waters CP-ABE **Setup** algorithm from G_{p1} and outputs {g_1, g_1^{a1} , $e(g_1, g_1)^{\alpha 1}$ }, where a_1, α_1 are in Z_{p1} . Next, **B** shows P P := {g_1g_3, g_1^{a1} g_3^{a3} , $e(g_1, g_1)^{\alpha 1}$ $e(g_3, g_3)^{\alpha 3}$ } which comes from Z_{p3} and Z_{p1} can be treated as $(g_1g_3)^{a}$, where a ∈ Z_N from the Chinese remainder theorem. For the same reason, $e(g_1, g_1)^{\alpha 1}$ $e(g_3, g_3)^{\alpha 3}$ can be treated as $e(g_1g_3, g_1g_3)^{\alpha}$, where α ∈ Z_N

Phase 1: When **B** receives a key generation query for attribute set S from **A**, **B** simply relays the query to **X** and obtains SK_{p3} = {K_{p3} , L_{p3} , $\{K_x\}_{\forall x \in S}$ }. **B** generates K_{p1} , L_{p1} with the same algorithm. Next, **B** replies **A** the secret key SK as follows:

SK = {Kp_1 Kp_3 , Lp_1 Lp_3 , {Kx}∀ x∈ S }.

Challenge: A outputs two messages M0, M1 with M1 and (M, ρ) to X as the challenge and obtains

P P_{p3} := {g_3, g_3^{a3} , $e(g_3, g_3)^{\alpha 3}$ }.