Audit-Free Cloud Storage via Deniable Attribute-Based Encryption

DOMMETI SATYA VENKATA KARTHIK
PG Scholar, Department of M.C.A,
S.K.B.R P.G College,
Amalapuram, E.G.Dt., A.P, India.
dsvkarthik1@gmail.com

Mr. NAGA. SRINIVASA RAO*
Asst. Professor, Dept of M.C.A,
S.K.B.R P.G College,
Amalapuram, E.G.Dt., A.P, India.
naagaasrinu@gmail.com

Abstract-Cloud storage services are becoming increasingly popular. Due to the importance of privacy, many cloud storage encryption schemes have been proposed to protect data from those who do not have access. All of these schemes assume that cloud storage providers are secure and cannot be hacked; In practice, however, some authorities (i.e. kennels) can force cloud storage providers to disclose user secrets or confidential data in the cloud, thereby completely bypassing storage encryption systems. In this white paper, we present our design for a new cloud storage encryption scheme that enables cloud storage providers to create compelling fake user secrets to protect user privacy. Since blackmailers cannot detect whether received secrets are true or not, cloud storage providers ensure that users' privacy is still securely protected.

Index Terms—Deniable Encryption, Composite order Bilinear Group, Attribute-Based Encryption, Cloud Storage.

I. Introduction

Cloud storage services are rapidly growing in popularity. Users can store their data in the cloud and access their data anytime, anywhere. For data protection reasons, the data stored in the cloud is typically encrypted and protected from access by other users. Considering the collaborative nature of cloud data, attribute-based encryption (ABE) is considered one of the most suitable encryption methods for cloud storage. Most of the proposed schemes assume that cloud storage service providers or trusted third parties that handle key management are trustworthy and cannot be hacked; In practice, however, some entities can intercept communications between users and cloud storage providers and then force storage providers to disclose user secrets using government powers or other means.

In this case, encrypted data is assumed to be known and storage providers are asked to disclose user secrets. For example, in 2010, Google released user documents to the FBI after receiving a search warrant without notifying its users. In 2013, Edward Snowden revealed the existence of global surveillance programs that collect such cloud data as emails, texts and voice messages from some tech companies. Once cloud storage providers are compromised, all encryption schemes lose their effectiveness. While we hope cloud storage providers can fight such companies to legally protect user privacy, it seems it's getting harder and harder. An example: Lavabit was an email services company that protected all user emails from outside duress; unfortunately it failed and decided to discontinue its email service.

Since it is difficult to fight against external coercion, we wanted to develop 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. Faced with such fake user secrets, external extortionists can only obtain fake data from a user's stored ciphertext. Once the extortionists believe that the secrets received are real, they will be satisfied and more importantly, cloud storage providers will not have revealed any real secrets. Therefore, user privacy is still protected. This concept comes from a special type of encryption scheme called deniable encryption. With deniable encryption, the sender and receiver create compelling fake evidence of fake data in ciphertexts, leaving outside extortionists satisfied. Note that the denial stems from the fact that the blackmailers cannot prove the proposed evidence is false and therefore have no reason to reject the evidence given. This approach seeks to block coercive action entirely, as the coercive forces know their efforts will be useless.

This work leverages this idea to enable cloud storage providers to provide audit-free storage services. In the cloud storage scenario, data owners who store their data in 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 hold system-wide secrets and need to be able to decrypt all encrypted data. This work describes a deniable ABE scheme for cloud storage services. This work uses ABE features to secure stored data with a fine-grained access control mechanism and deniable encryption to prevent external auditing. Our scheme is based on Waters' CP-ABE (Ciphertext Policy-Attribute Based Encryption) scheme. This work extends the Waters scheme from first-order bilinear groups to composite-order bilinear groups. By adopting the subgroup decision problem, our scheme allows users to provide fake secrets that appear legitimate to outside extortionists.

II. RELATED WORK

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 [1]. That is, only those who meet the owner conditions can successfully decrypt stored data. We note here that ABE is encryption for privileges, not users. This makes ABE a very useful tool for cloud storage services as 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 with pairwise keys. In addition, it is also impractical for many people to encrypt data many times. With ABE, data owners only decide what kind of users can access their encrypted data.

Users who meet the conditions can 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 the policy validation. KP-ABE is an ABE where the policy is embedded in the user secret key and the attribute set is embedded in the ciphertext. Conversely, CP-ABE embeds the policy in the ciphertext and the user secret has the attribute set.

Goya et al. proposed the first KPABE in [2]. They constructed an expressive way to relate any monotonic formula as a guideline for user secrets. Bethencourt et al. proposed the first CP-ABE in [3]. This scheme used a tree access structure to express any monotonic formula over attributes as a guide in the ciphertext. The first fully expressive CP-ABE was proposed by Waters in [4], which used Linear Secret Sharing Schemes (LSSS) to

build a ciphertext policy. Lewko et al. extended the Waters scheme in [5] to a fully safe CP-ABE, albeit with some loss of efficiency.

Recently, Attrapadung et al. constructed in [6] and Tysowski et al. designed their CP-ABE scheme for resource-constrained users in [7]. The concept of deniable encryption was first proposed in [8]. Like normal encryption schemes, deniable encryption 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 a few important deniable public key encryption schemes.

Canetti et al. used translucent sets in [8] to construct deniable encryption schemes. A translucent set is a set that contains a trapdoor subset. It is easy to randomly choose an element from the universal set or from the subset; However, without the trapdoor, it is difficult to determine whether a given element belongs to the subset. Canetti et al. showed that any trapdoor permutation can be used to construct the translucent sentence. To create a deniable public key encryption scheme from a transparent set, the transparent set is the public key and the trapdoor is the private key. The translucent sentence is used to represent an encrypted bit. Elements in the subset are represented by 1, while other elements that are not a subset are represented by 0. The sender can encode 1 by sending an element in the subset, but can assert that the element was chosen from the universal set (i.e., 0). The above is a basic scheme that denies the sender.

Canetti et al. also proved that a sender-deniable scheme can be transformed into 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 scannable encryption in [9]. ONeill et al. designed the bi-translucent set from a lattice in [10] that can build a native bi-deniable scheme. In addition to the bi-translucent set, there are other proposed approaches for creating deniable encryption schemes.

ONeill et al. proposed a new deniable method by a simulatable public key system [10]. The simulatable public key system provides a forgotten key generation function and a forgotten ciphertext function. When sending an encrypted bit, the sender sends a set of encrypted data, which can normally be encrypted or unnoticed. Therefore, the sender can claim that some sent messages go unnoticed, when in reality this is not the case. The idea can be applied on the receiver side such that the schema is an ambiguous schema. In [11] Gasti et al. proposed another deniable scheme in which a public-private key pair is established for each user, when in fact there are two pairs. The sender can send a real message encrypted with one key with a fake message encrypted with the other key. The sender decides which key to release according to the enforcer's identity.

Gastie et al. also applied this idea to cloud storage services. There are other deniable encryption schemes, including [12] and [13]. Aside from the above deniable schemas, there are studies that examine the limitations of the deniable schemas. . In [14], Nielsen states that it is impossible to encrypt unlimited messages with a short key in non-committing schemes, including deniable schemes. In [15] Bendlin et al. shows that non-interactive and fully receiver-deniable schemes cannot be achieved simultaneously.

III. PROCESS MODEL

We construct a deniable CP-ABE scheme that can make cloud storage services secure

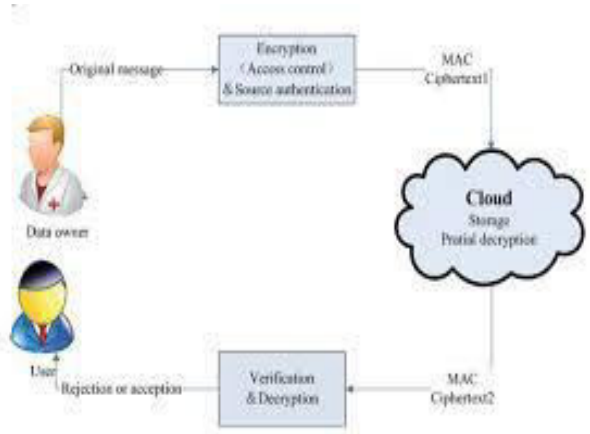
and audit-free. In this scenario, cloud storage service providers are only considered as receivers in other deniable schemes. Unlike most previous deniable encryption schemes, we do not use translucent sentences or simulatable public key systems to implement the denial. We construct our deniable encryption scheme through a multidimensional space. All data is encrypted in the multidimensional space. The original data can only be obtained if the dimensions are correctly compiled. If assembled incorrectly, ciphertexts are decrypted into predetermined forged data. The dimensions are kept secret. We use composite order bilinear groups to construct the multidimensional space. We also use chameleon hash functions to make both true and fake news persuasive. 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 handle one bit at a time; Therefore, bitwise deniable encryption schemes are inefficient for real-world use, especially in the case of cloud storage services. To solve this problem, ONeil et al. developed a hybrid encryption scheme that uses symmetric and asymmetric encryption at the same time. They use a dubious encrypted plan-ahead symmetric data encryption key, while real data is encrypted by a symmetric key encryption mechanism. This reduces the repeating number from the block size to the key size. Although bitwise deniable encryption is more flexible than blockwise deniable encryption when cooking fake data, when looking at cloud storage services, blockwise deniable encryption is much more efficient to use. Contrary to the techniques used in previous deniable encryption schemes, we build two encryption environments at the same time. We build our schema with multiple dimensions while claiming that there is only one dimension. We apply this idea to an existing ABE scheme by replacing primary ordering groups with composite ordering groups. Since the base ABE scheme can encrypt one block at a time, our deniable ABE is certainly a block-by-block deniable encryption scheme. Although the bilinear operation for the composite-order group is slower than the first-order group, there are some techniques that can convert an encryption scheme from composite-order groups to first-order groups for better computational performance.

• Uniform environment. Most of the previous deniable encryption schemes are encryption independent. That is, the encryption parameters should be completely different for each encryption operation. . If two deniable ciphers are performed in the same environment, the latter cipher will lose its deniability after the first cipher is enforced, since each cipher enforced reduces flexibility. For example, when extortionists obtain private keys, which are the most common recipient evidence, these keys should be convincing not only among specific files, but also among all related stored data. Otherwise, the blackmailers will know that these keys are fake; however, all proposed schemes only provide convincing evidence for specific transmissions. In 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, let alone 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 mean that one encryption environment can be used for multiple encryption times without system updates. The opened recipient proof should look convincing for all cipher texts under this environment3, regardless of whether a cipher text is normally encrypted or deniably encrypted. The deniability of our scheme stems from the secret of the subgroup

assignment, which is set only once in the system to decrypt normal cipher texts correctly.

• **Deterministic Decryption**. Most deniable encryption schemes have problems with decryption errors. These errors come from the designed decryption mechanisms. For example in [12], Canetti et al. uses the subset decision mechanism for decryption. The recipient determines the decrypted message according to the subset decision result. If the sender selects an item from the universal set, but unfortunately the item is in the specific subset, an error occurs. The same error occurs in all translucentset-based deniable encryption schemes. Which uses a voting mechanism for decryption? The decryption is correct if and only if the right part overwhelms the wrong part. Otherwise, the receiver gets the error result. Rated SMOTE to balance the data set.



IV. MODULES

1. Key Generation
2. Encryption
3. Decryption
4. Verification

Key Generation:

- Setup (1 power lamda) → (PP, MSK): This algorithm takes security parameter lamda as input and returns public parameter PP and system master key MSK.
- KeyGen(MSK, S) → SK: Given set of attributes S and MSK, this algorithm outputs private key SK.
- DenSetup(1 power lamda) → (PP,MSK, PK): This algorithm takes security parameter lamda as input and returns public parameters PP, system master key MSK, and system public key PK. PK is known by all system users and is kept secret to outsiders.
- DenKeyGen(MSK, S) → (SK, FK): Given set of attributes S and MSK, this algorithm outputs private key SK as well as FK for the user, where FK will be used for generating fake proof later.

Encryption: Enc (PP, M, A) → C: This encryption algorithm takes as input public parameter PP, message M, and LSSS access structure A = (M, p) over the universe of attributes. This algorithm encrypts M and outputs a cipher text C, which can be decrypted by those who possess an attribute, set that satisfies access structure A.

- OpenEnc(PP,C,M) → PE: This algorithm is for the sender to release encryption proof PE for (M,C).
- DenEnc(PP, PK,M,M′,A) → C′: Aside from the inputs of the normal encryption algorithm, this deniable encryption algorithm needs public key PK and fake message M′. The output ciphertext must be indistinguishable from the output of Enc.
- DenOpenEnc(PP,C′,M′) → P′ E : This algorithm is for the sender to release encryption proof P′ E for fake message M′. The output must be indistinguishable from the result of

OpenEnc and must pass the Verify algorithm.

Decryption:

- Dec(PP, SK,C) → {M,⊥}: This decryption algorithm takes as input public parameter PP, 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 ⊥.
- OpenDec(PP, SK,C,M) → PD: This algorithm is for the receiver to release decryption proof PD for (M,C).
- DenOpenDec(PP, SK, FK,C′,M′) → P′ D: This algorithm 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.

Verification: Verify (PP, C, M, PE, and PD) → {T, F}: This algorithm is used to verify the correctness of PE and PD.

V. CONCLUSION

This work proposed a deniable CP-ABE scheme to build an audit-free cloud storage service. The deniability feature invalidates coercion, and the ABE property ensures secure cloud data sharing with a fine-grained access control mechanism. This proposed system offers a possible way of tackling immoral interference with the right to privacy. This work hopes that more schemes can be created to protect the privacy of cloud users.

REFERENCES

[1] A. Sahai and B. Waters, “Fuzzy identity-based encryption,” in Eurocrypt, 2005, pp. 457–473.

[2] V. Goyal, O. Pandey, A. Sahai, and B. Waters, “Attribute-based encryption for fine-grained access control of encrypted data,” in ACM Conference on Computer and Communications Security, 2006, pp. 89–98.

[3] J. Bethencourt, A. Sahai, and B. Waters, “Ciphertext-policy attribute-based encryption,” in IEEE Symposium on Security and Privacy, 2007, pp. 321–334.

[4] B. Waters, “Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization,” in Public Key Cryptography, 2011, pp. 53–70.

[5] A. B. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, “Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption,” in Eurocrypt, 2010, pp. 62–91.

[6] N. Attrapadung, J. Herranz, F. Laguillaumie, B. Libert, E. de Panafieu, and C. R`afols, “Attribute-based encryption schemes with constant-size ciphertexts,” Theor. Comput. Sci., vol. 422, pp. 15–38, 2012.

[7] P. K. Tysowski and M. A. Hasan, “Hybrid attribute- and reencryption- based key management for secure and scalable mobile applications in clouds.” IEEE T. Cloud Computing, pp. 172–186, 2013.

[8] R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky, “Deniable encryption,” in Crypto, 1997, pp. 90–104.

[9] M. D¨urmuth and D. M. Freeman, “Deniable encryption with negligible detection probability: An interactive construction,” in Eurocrypt, 2011, pp. 610–626.

[10] A. O’Neill, C. Peikert, and B. Waters, “Bi-deniable public-key encryption,” in Crypto, 2011, pp. 525–542.

[11] P. Gasti, G. Ateniese, and M. Blanton, “Deniable cloud storage: sharing files via public-key deniability,” in WPES, 2010, pp. 31– 42.

[12] M. Klonowski, P. Kubiak, and M. Kutylowski, “Practical deniable encryption,” in SOFSEM, 2008, pp. 599–609.

[13] M. H. Ibrahim, “A method for obtaining deniable public-key encryption,” I. J. Network Security, vol. 8, no. 1, pp. 1–9, 2009.

[14] J. B. Nielsen, “Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case,” in Crypto, 2002, pp. 111–126.

[15] R. Bendlin, J. B. Nielsen, P. S. Nordholt, and C. Orlandi, “Lower and upper bounds for deniable public-key encryption,” Cryptology ePrint Archive, Report 2011/046, 2011, http://eprint.iacr. org/.