Cryptoanalysis of an Identity-Based Provable Data Possession Protocol with Compressed Cloud Storage*

Lidong Han[1], Guangwu Xu[2], Qi Xie[1], Xiao Tan[1], and Chengliang Tian[3]

[1] Key Laboratory of Cryptography Technology of Zhejiang Province, Hangzhou Normal University,Hangzhou, China ldhan@hznu.edu.cn, qixie68@126.com, xiaotan.cs@gmail.com ,
[2] School of Cyber Science and Technology, Shandong University, Qingdao, China, gxu4sdq@sdu.edu.cn
[3] College of Computer Science and Technology, Qingdao University, Qingdao, China tianchengliang@qdu.edu.cn

Abstract. In this letter, we analyze the security of identity-based provable data possession protocol with compressed cloud storage (published in IEEE TIFS, doi:10.1109/TIFS.2022. 3159152). An adversary can recover the ephemeral secret keys from only two encrypted blocks and obtain the original data with a high probability under Dirichlet's basic result. Moreover, he can impersonates a data owner to outsource any file to the cloud in an unwanted way.

Keywords: Cryptoanalysis · provable data possession · ephemeral secret key · data auditing

1 Introduction

growing demand for computing resources, more and more users prefer to store their data into cloud. Since the data owner loses control of their data and the cloud is not completely trusted, it is important for users to audit the integrity of their data outsourced on cloud. There are two techniques to enable the cloud to produce proof of outsourced data: proof of retrievability (POR) proposed by Juels and Kaliski[1] and provable data possession (PDP) by Ateniese et al.[2]. In Ateniese et al.'s scheme, they utilized RSA-based homomorphic tags to verify the data integrity with a probabilistic algorithm. After that, many research work[3–5] were proposed to discuss lower computational complexity, improved security and dynamic operations for public auditing of outsourced data. On the other hand, several previous schemes in [6–8] focused on users' public key generation without the help of public key infrastructure. They constructed identity-based PDP scheme to facilitate certificate management.

* This work was supported by the National Natural Science Foundation of China (Grant No.U21A20466, No.61972124) (*Corresponding author: Lidong Han.)*

Recently, Yang et al. proposed an identity-based PDP scheme, called IBPDP-CCS, to support compressed cloud storage[9]. They utilized the basic algebraic operations to design a concrete protocol which can provide lower storage, communication,and computation costs. Specially, the data owner only needs to upload the encrypted blocks and a tag to the cloud without his original file. In [9], it was claimed that IBPDP-CCS can provide data privacy, unforgeability.

In this letter, we analyze IBPDP-CCS and point out that the identity-based provable data possession protocol is insecure to an adversary. Specifically, we demonstrate that in IBPDP-CCS, an attacker can first ecxtract $\hat{a}$ from the file tag τ. Next, he computes the ephemeral secret value c from two encrypted blocks y_i, y_j and τ by utilizing the Euclidean algorithm. The probability of success in our attack is one that two random integers are coprime. From c, the adversary is able to get other private key a, b and therefore he can execute an impersonation attack.

2 Review IBPDP-CCS

In this section, we briefly review the underlying identity based PDP protocol which achieves compressed cloud storage [9] proposed by Yang et al.. Their scheme contains four entities: data owner, cloud, and third-party auditor (TPA), and key generation center (KGC). KGC generates the system parameters and the secret key for a user. Data owners store the encrypted blocks of file and the file tag into the cloud. In data auditing, TPA transfers a challenge message to cloud for the audit on behalf of the users. Upon receiving the challenge, the cloud produces the proof as responce to TPA. TPA velidates the responce and returns the result to data owner. During data recovery, the owner can decrypt the given encrypted file.

The IBPDP-CCS scheme consists of seven algorithms: **Setup**, **Extract**, **Outsource**, **Challenge**, **ProofGen**, **Verify**, **Recover**. We ignore other algorithms and the readers are referred to [9] for more details.

1. $Setup(\lambda) \rightarrow (MSK, PK)$. With the security paramete λ, KGC generates two random primes p, q where q is much smaller than p, two elements $g, \sigma \in Z_p$, and hash function $H : \{0,1\}^* \rightarrow Z_p$. The master secret key MSK of KGC is σ, and the public key PK is $\{p, q, g, g^\sigma, H\}$. And $floor(\cdot)$ is a function of rounding down to the nearest integer.
2. $Extract(ID) \rightarrow SK_ID$. In this algorithm, KGC outputs the secret key SK_ID for a user whose identity is ID and the user validate it.
   - From a user identity ID, KGC selects a rondom number $\zeta \in Z_p$ and compute $a' = \zeta + \sigma H(ID) \mod (p-1)$. KGC transmits $SK_ID = a'$ to the user with g^ζ.
   - After receiving a' and g^ζ, the user determines $g^{a'} = g^\zeta \cdot g^{\sigma H(ID)} \mod p$ to judge the correctness of his secret key.
3. $Outsource(F, SK_{ID}, PK) \rightarrow (T, \tau)$. The user encrypts all blocks of the file F and generates the file tag.

- The data owner randomly chooses $a'' \in Z_p$, $b, c, r, l \in Z_q$ and computes $a = a' + a''$, and $\hat{a} = a/r$.
- The user devides the file into $\{x_1, x_2, \cdots, x_m\}$ and generates the encrypted block y_i by computing $y_i = a(x_i + bH(name||i)) + cx_i$, where $x_i \in Z_l$, $name$ is the identifier of file F.
- Define $T = \{y_1, y_2, \cdots, y_m\}$ as a set of all encrypted blocks. The owner generates the file tag $\tau = name||l||m||\hat{a}||g^a||g^c||g^{abc}||spk||SSig(name||l||m||\hat{a}||g^a||g^c||g^{abc}, ssk)$, where $SSig$ is an identity-based secure digital signature whose public key and secret keyare spk and ssk.

4. *Challenge*$(\cdot) \rightarrow chal$. TPA produces a challenge $chal$ when he wants to perform data audit.
   - TPA first checks the validity of the file tag τ with public key of ID-based signature. If invalid, TPA terminates the audit; otherwise, TPA extracts the values $m, l, \hat{a}, g^a, g^c, g^{abc}$ from the tag τ.
   - TPA chooses random indices of the challenged block $\{i_1, i_2, \cdots, i_n\}$ from $[1, \cdots, m]$ and random numbers $\{e_1, e_2, \cdots, e_n\}$ such that $\sum_{j=1}^{n} e_j ql < \hat{a}$.
   - TPA sends to cloud the challenge sequence as $chal = \{i_1, i_2, \cdots, i_n; e_1, e_2, \cdots, e_n\}$.
5. *ProofGen*$(T) \rightarrow \Gamma$. The cloud generates a proof by computing $\Gamma = \sum_{j=1}^{n} e_{i_j} y_{i_j}$ as response to TPA's challenge.
6. *Verify*$(chal, \Gamma, \tau) \rightarrow v$. On receiving Γ, TPA virifies the equation

$$g^{cfloor(\Gamma/\hat{a})\cdot\hat{a}} \stackrel{?}{=} g^{a(\Gamma - floor(\Gamma/\hat{a})\cdot\hat{a})} \cdot g^{abc\sum_{j=1}^{n} e_j H(i_j)} \mod p$$

   If yes, TPA returns $v = 1$; otherwise, $v = 0$.
7. *Recover*$(y_i) \rightarrow x_i$. From y_i, the user can recover the original data block x_i by calculating $x_i = (y_i - floor(y_i/\hat{a}) \cdot \hat{a})/c$.

3 Security Analysis of IBPDP-CCS

As shown in Section 2, IBPDP-CCS explores the basic algebraic operation to achieve compressed cloud storage. In other words, the owner only transmits encrypted values to cloud without the original file to support integrity auditing by TPA and decryption by a data owner. However, we demonstrate that, in IBPDP-CCS, an adversary can decrypt all encrypted blocks of files even if he does not know the master private key. Specifically, an attacker is able to recover the ephemeral private key which the data owner utilizes to encrypt. The main idea of our attack is to compute the greatest common divisor (GCD) of two values from two encrypted blocks. Before describing our attack, we first introduce a random number theory, which is a seminal result of Dirichlet [10].

Theorem 1 *If α and β are two random integers, the probability that $gcd(\alpha, \beta) = 1$ is $\frac{6}{\pi^2} \approx 0.608$*

In our attack, an adversary has the ability of eavesdropping the information from a communication channel, such as encrypted blocks and a file tag. We assume that the encrypted blocks are independent random variables.

We describe our attack in details as follows. An adversary has obtained the set of encrypted blocks $T = \{y_1, y_2, \cdots, y_m\}$ and a file tag τ.

- *Step* 1. The adversary first select two random blocks y_i, y_j, and extracts the value $\hat{a}$ from the file tag τ.
- *Step* 2. From y_i, y_j and $\hat{a}$, he computes
  $\alpha_i = y_i - floor(y_i/\hat{a}) \cdot \hat{a}$
  $\alpha_j = y_j - floor(y_j/\hat{a}) \cdot \hat{a}$
- *Step* 3. The attacker generates $c' = \gcd(\alpha_i, \alpha_j)$ under Eulidean algorithm and then calculates $x' = \alpha_i/c'$.
- *Step* 4. Finally, he tests if the block x' is decoded to be meaningful or not. If yes, then the private key c' is valid to decrypt all encrypted block. Otherwise, go to Step 1.

From Theorem 1, it is clear that our attack is successful with a high probility. Moreover, we can enhance the successful probability of our attack using three or more encrypted blocks. Note that the probabilty that three random integers are coprime is 0.832.

On the other hand, once an adversary can obtain the valid value c, he can recover the other private key a, b using the similar technique as above. Specially, from the equation $y_i = a(x_i + bH(name||i)) + cx_i$, an adversary has known the values of y_i and $cx_i = y_i - floor(y_i/\hat{a})$. Then, for a pair (y_i, y_j), he generates $\gcd(y_i - cx_i, y_j - cx_j)$ which is probably the values of a from the aforementioned analysis. Then b is recovered using $a, c, x_i, y_i, H(name||i)$. Therefore, an adversary can impersonate the owner to outsource any file which has same file tag τ in an unwanted way.

4 Conclusion

In this letter, we give an analysis and present an attack on IBPDP-CCS [9] proposed by Yang et al. An adversary in our attack can decrypt all encrypted blocks although it does not know the master key of a data owner. In fact, the proposed attack can recover the secret encrypting key from encrypted blocks with a high probability. Furthermore, an adversary can impersonate the data owner to outsource files to the cloud.

References

1. A. Juels and B.S. Kaliski Jr., "PORs: Proofs of Retrievability for Large Files," in *Proc. 14th ACM Conf. Comput. Commun. Secur.(CCS)*, 2007, pp. 584–597.
2. G. Ateniese et al., "Provable data possession at untrusted stores", in *Proc. 14th ACM Conf. Comput. Commun. Secur.(CCS)*, 2007, pp. 598–609.

3. C. Erway, A. Kupcu, C. Papamanthou, R. Tamassia,"Dynamic provable data possession", in *Prod. of CCS'09*, 2009,pp.213–222.
4. Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in *Proc. of ESORICS'09*, vol.5789, 2009, pp. 355–370.
5. Du et al., "Enabling Secure and Efficient Decentralized Storage Auditing with Blockchain," *IEEE Trans. Inf. Forensics Security*, vol.1,2021, PP.1-1.
6. H. Wang, Q. Wu, B. Qin, and J. Domingo-Ferrer, "Identity-based remote data possession checking in public clouds",*IET Inf. Secur.*, 2014, vol. 8, no. 2, pp.114–121.
7. Y. Zhang, J. Yu, R. Hao, C. Wang, and K. Ren, "Enabling efficient user revocation in identity-based cloud storage auditing for shared big data," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 3, 2020,pp. 608–619.
8. J. Li, H. Yan, and Y. Zhang, "Certificateless public integrity checking of group shared data on cloud storage," *IEEE Trans. Services Comput.,* vol.14, no.1, 2021, pp. 71–81.
9. Y.Yang, Y. Chen, F. Chen, J. Chen, "An efficient identity-based provable data possession protocol with compressed cloud storage", *IEEE Trans. Inf. Forensics Security*, vol.17, 2022, pp. 1359–1371.
10. G. Lejeune Dirichlet, *Ü*ber die Bestimmung der mittleren Werthe der Zahlentheorie, Abhandlungen der K*ö*iglichen Akademie der Wissenchaften zu Berlin, 1849.