Cryptanalysis 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. This letter addresses some security issues of an identity-based provable data possession protocol with compressed cloud storage (published in IEEE TIFS, doi:10.1109/TIFS.2022. 3159152). Some serious flaws are identified and an attack to the protocol is designed. This attack is able to recover the ephemeral secret keys from two encrypted blocks with high probability to reveal the original plaintext file completely. Moreover, an adversary can impersonate a data owner to outsource any file to the cloud in a malicious way. The main ingredients of the attack is some classical number theoretic results.

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

1 Introduction

With growing demand for computing resources, storing a user's data in the cloud has become a popular choice. Since in this case data owners lose 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 [2], the authors utilized RSA-based homomorphic tags to verify the data integrity with a probabilistic algorithm. After that, many research work[3–5] discussed how to lower computational complexity to improve security

* This work was supported by the National Natural Science Foundation of China (Grant No.U21A20466, No.61972124) and the National Key Research and Development Program of China (Grant No. 2018YFA0704702) (*Corresponding author: Lidong Han and Guangwu Xu.*)

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 by constructing identity-based PDP schemes to facilitate certificate management.

Recently, Yang et al. proposed an identity-based PDP scheme, called IBPDP-CCS, to support compressed cloud storage[9]. The design of their protocol only relies on the basic algebraic operations and the costs of storage, communication, and computation are lowered. Specially, a data owner only needs to upload the encrypted blocks and a tag to the cloud without including his original file. In [9], it was claimed that IBPDP-CCS provides data privacy and unforgeability.

In this letter, some serious security flaws of the identity-based provable data possession protocol IBPDP-CCS are identified. More specifically, we demonstrate that in IBPDP-CCS, an attacker is able to use a piece of public information in the file tag to derive an ephemeral secret value by using two encrypted blocks with high probability. With this, other ephemeral private key values can be obtained as well. In particular, the original plaintext file is revealed completely and impersonating a data owner is possible. The attack is based on some classical result in number theory.

2 A Review of IBPDP-CCS

In this section, we briefly review the underlying identity-based PDP protocol proposed by Yang et al. [9]. This schemes achieves compressed cloud storage and contains four entities: data owner, cloud, 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 a 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 response to TPA. TPA validates the response 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 will explain these seven procedures here. For other related information, the readers are referred to [9] for more details.

1. $Setup(\lambda) \rightarrow (MSK, PK)$. With the security parameter λ, KGC determines a large prime p, generates a random number q with q being much smaller than p, and two random elements $g, \sigma \in Z_p$. A hash function $H : \{0,1\}^* \rightarrow Z_p$ is fixed. The master secret key MSK of the KGC is σ, and the public key is $PK = \{p, q, g, g^\sigma, H\}$.
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 random number $\zeta \in Z_p$ and compute $a' = \zeta + \sigma H(ID) \mod (p-1)$. KGC transmits $SK_{ID} = a'$ to the user, together 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 $\hat{a} = a/r$. Note $ql \ll \hat{a}$.
   - The user divides 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 the 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 key are spk and ssk respectively.
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 the 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.
6. *Verify*$(chal, \Gamma, \tau) \rightarrow v$. On receiving Γ, TPA verifies 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 $x_i = (y_i - floor(y_i/\hat{a}) \cdot \hat{a})/c$.

Remark: We would like to make some remarks on the definition and process of IBPDP-CCS. First we note that in [9], the authors' usage of $\hat{a} = a/r$ seems to be that of $floor$ as they also used the division / to result the *mod* operation [4]. However, the operator $floor$ was defined and used in many places. The second remark is about the decryption formulas $x_i = (y_i - floor(y_i/\hat{a}) \cdot \hat{a})/c$, $i = 1, 2, \cdots, m$ in [9]. We believe that some conditions need to be specified in order to make these equalities hold. In our rest discussion, we shall not need to involve those conditions. The decryption formulas are sufficient for us to perform analysis on IBPDP-CCS.

[4] If '/' were for rational division, the secret a would be easily derived from the public piece $\hat{a}$.

3 Security Analysis of IBPDP-CCS

In this section, we present an analysis of the IBPDP-CCS scheme. We have identify some flaws in the design of this scheme, including some serious ones that compromise the security. With achieving compressed cloud storage in mind, a data owner only transmits encrypted values to cloud without the original file to support integrity auditing by TPA and decryption by the data owner. However, we are able to describe an attack to IBPDP-CCS in which an adversary can decrypt all encrypted blocks of files. To be more precise, an attacker is able to recover the ephemeral private key the data owner used to perform decryption. The main ingredients of the attack are some basic number theoretic primitives such as computing the greatest common divisor (GCD) and a classical result of Dirichlet [10] which states

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

As usual, an adversary is assumed to have the ability of eavesdropping the information from a communication channel. Therefore, the encrypted blocks and the file tag are available to the adversary in our attack.

Now let us describe the attack in details. With the set of encrypted blocks $T = \{y_1, y_2, \cdots, y_m\}$ and a file tag τ, the adversary performs the following steps.

- *Step* 1. The adversary selects 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 $\hat{a}$, the adversary 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 adversary calculates $c' = \gcd(\alpha_i, \alpha_j)$ using the Euclidean algorithm and then calculates $x'_k = \big(y_k - floor(y_k/\hat{a}) \cdot \hat{a}\big)/c'$ for $k = 1, 2, \cdots, m$.
- *Step* 4. The adversary checks whether the file $\{x'_1, x'_2, \cdots, x'_m\}$ is meaningful. If yes, then the private key c' is valid. Otherwise, go to Step 1.

A proof of the correctness of the above procedure goes as follows. According to the design and requirements of the IBPDP-CCS protocol, the equality $y_i = floor(y_i/\hat{a}) \cdot \hat{a} + cx_i$ holds for each $1 \leq i \leq m$. So

$$\gcd(\alpha_i, \alpha_j) = c \gcd(x_i, x_j).$$

It can be assumed that the integers x_i (converted from the actual file blocks) exhibit some randomness. Thus with bigger probability $\gcd(x_i, x_j) = 1$ holds true by Theorem 1.

Remark: In fact, one can do better than the described protocol. Computing $c'_{i,j} = \gcd(\alpha_i, \alpha_j)$ for all $1 \leq i, j \leq m, i \neq j$, then with overwhelming probably $c = \min_{1 \leq i,j \leq m, i \neq j} c'_{i,j}$. The worst situation is that all $\gcd(x_i, x_j) > 1$, then $\min_{1 \leq i,j \leq m, i \neq j} c'_{i,j}$ is still likely to be a small multiple of c.

We would like to further remark that, once an adversary obtains the valid value c, he can recover the other private key a, b using the similar technique as

above. More specifically, 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 a malicious manner.

4 Conclusion

This letter presents an analysis of the PDP protocol IBPDP-CCS [9]. Some serious security flaws are identified and an attack to IBPDP-CCS is described. An attacker is able to recover all encrypted blocks with high probability without knowing the secret key of the owner. 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, *Über* die Bestimmung der mittleren Werthe der Zahlentheorie, Abhandlungen der K*öi*glichen Akademie der Wissenchaften zu Berlin, 1849.