Comments on "Privacy-Preserving Public Auditing Protocol for Regenerating-Code-Based Cloud Storage"

Jindan Zhang, Rongxing Lu, *Senior Member, IEEE*, Baocang Wang, and Xu An Wang

***Abstract*— Public auditing protocol is crucial for the success of cloud computing, as it can ensure the outsourced data in cloud server are not tampered by attackers. Due to its importance, public auditing protocol has received considerable attention in the past years. In 2015, Liu *et al.* proposed a privacy-preserving public auditing protocol for regenerating-code-based cloud storage (IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, 10(7):1513–1528, 2015) and claimed it is secure under the considered security model. However, in this article, we will show that their protocol is not as secure as they claimed, i.e., the proxy delegated by the data owner can forge an authenticator for any data block, which obviously invalidates their protocol's security. We hope that by identifying the design flaw, similar weaknesses can be avoided in future protocol design.**

***Index Terms*— Cloud storage, public auditing, authenticator, forgery attack.**

I. INTRODUCTION

THE potential cost savings, flexibility, and scalability of cloud computing continue to attract more and more enterprises and individual users to outsource their data to cloud servers. However, as the cloud servers are not fully trustable, how to ensure those outsourced data intact becomes a challenging issue. Aiming at addressing the challenge, Atenesis *et al.* [1] proposed the concept of provable data possession. Specifically, in the paradigm of provable data possession, a data owner first computes tags or authenticators for data blocks, and then outsources both data blocks and the corresponding tags to the cloud server. Later, when an auditing party challenges the cloud server on random positions of the outsourced data blocks, the cloud server will respond a proof, which is formed by the aggregated data blocks and aggregated tags. If the proof can pass the verification, the auditing party can ensure the data being kept integrity. Due to its importance to cloud computing security, many public auditing protocols have been proposed in the past years.

In 2015, Liu *et al.* [2] proposed a privacy-preserving public auditing protocol for regenerating-code-based cloud storage, and claimed it is secure under the considered security model. However, in this article, we will show that their protocol is not as secure as they claimed, i.e., the proxy delegated by the data owner can forge an authenticator for any data block, which is obviously beyond the proxy's permissible ability.

Manuscript received February 21, 2020; accepted October 11, 2020. Date of publication October 19, 2020; date of current version November 9, 2020. This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB0802000; in part by the National Natural Science Foundation of China under Grant U19B2021, Grant U1736111, and Grant 61972457; and in part by the Key Research and Development Program of Shaanxi under Grant 2020ZDLGY08-04. *(Corresponding author: Rongxing Lu.)*

Jindan Zhang is with the Xianyang Vocational Technical College, Xianyang 712000, China, and also with the State Key Laboratory of Integrated Service Networks, School of Telecommunications Engineering, Xidian University, Xi'an 710071, China (e-mail: zhangjindan83@163.com).

Rongxing Lu is with the Faculty of Computer Science, University of New Brunswick, Fredericton, NB E3B 5A3, Canada (e-mail: rlu1@unb.ca).

Baocang Wang is with the State Key Laboratory of Integrated Service Networks, School of Telecommunications Engineering, Xidian University, Xi'an 710071, China (e-mail: bcwang79@aliyun.com).

Xu An Wang is with the Key Laboratory of Network and Information Security, Engineering University of People's Armed Police, Xi'an 710086, China, and also with the Guizhou Provincial Key Laboratory of Public Big Data, Guizhou University, Guiyang 550000, China (e-mail: wangxazjd@163.com).

Digital Object Identifier 10.1109/TIFS.2020.3032283

II. REVIEW OF LIU *et al.*'S PUBLIC AUDITING PROTOCOL

Liu *et al.*'s public auditing protocol consists of three procedures, namely, Setup, Audit, and Repair [2]. Since the Audit and Repair are not directly related to our attack, here we just briefly review the Setup procedure.

Setup. Considering the regenerating-code-based cloud storage with parameters (n, k, l, α, β), where the parameter β is assumed as 1 for simplicity. Let G and G_T be multiplicative cyclic groups of the same large prime order p, and $e : G \times G \to G_T$ be an efficiently computable bilinear pairing map. Let g be a generator of G and $H(\cdot) : \{0,1\}^* \to G$ be a secure hash function that maps strings uniformly into group G. The related parameters are initialized as follows.

- KeyGen$(1^k) \to (pk, sk)$: The data owner generates a random signing key pair (spk, ssk) and two random elements $x, y \leftarrow_R Z_p$ and computes $pk_x = g^x$, $pk_y = g^y$. Then, the secret parameter is $sk = (x, y, ssk)$ and the public parameter is set as $pk = (pk_x, pk_y, spk)$.
- Delegation$(sk) \to (x)$: The data owner sends the encrypted x to the proxy by using the proxy's public key. Upon receiving it, the proxy decrypts and stores x locally.
- SigAndBlockGen$(sk, F) \to (\Phi, \Psi, t)$: The data owner uniformly chooses a random identifier $ID \leftarrow_R \{0,1\}^*$, a random symbol $u \leftarrow_R G$, one set $\Gamma = \{w_1, w_2, \cdots, w_m\}$ with elements $w_i \leftarrow G$ and a file tag $t = (ID||u||w_1||\cdots||w_m)Sig_{ssk}(ID||u||w_1||\cdots||w_m)$ for F, where $Sig()$ is one standard signature scheme. Recall that the original file F is split into m blocks, $\{\bar{w_i}\}_{i=1}^m$; the data

owner will compute and store $n\alpha$ coded blocks among n cloud servers. Viewing each segment of the blocks as a single symbol for simplicity, our signature is generated simultaneously with the encoding process as follows

1) Augmentation: The data owner first augments the native m blocks in a proper way.
2) Signing for Native Blocks: The data owner views the data parts of the augmented blocks as a set of segments and computes an authenticator for each segment as
$$\sigma^{**}_{j_0k_0} = (u^{\bar{w}_{j_0k_0}} \prod_{\lambda=1}^{m} w_\lambda^{\bar{w}_{j_0(s+\lambda)}})^y = (u^{\bar{w}_{j_0k_0}} w_{j_0})^y$$
where $1 \le j_0 \le m, 1 \le k_0 \le s$.
3) Combination and Aggregation: The data owner randomly chooses m elements $\{\epsilon_{ij\lambda}\}_{\lambda=1}^m$ from $GF(p)$ to be coefficients and linearly combines the augmented native blocks to generate coded blocks $v_{ij} (1 \le i \le n, 1 \le j \le \alpha)$, as follows:
$$v_{ij} = \sum_{\lambda=1}^{m} \epsilon_{ij\lambda} w_\lambda \in GF(p)^{s+m}$$
and apparently each symbol can be obtained by
$$v_{ijk} = \sum_{\lambda=1}^{m} \epsilon_{ij\lambda} \bar{w}_{\lambda k} \in GF(p)$$
with $1 \le k \le s+m$. After that, we can get the aggregated tags as
$$\sigma^*_{ijk} = \prod_{\lambda=1}^{m} (\sigma^{**}_{\lambda k})^{\epsilon_{ij\lambda}} = (u^{v_{ijk}} \cdot \prod_{\lambda=1}^{m} w_\lambda^{\epsilon_{ij\lambda}})^y$$
with $1 \le k \le s$.
4) Authenticator Generation: The data owner generates an authenticator for v_{ijk} as:
$$\begin{aligned}\sigma_{ijk} &= H(ID||i||j||k)^x \sigma^*_{ijk} \\ &= H(ID||i||j||k)^x (u^{v_{ijk}} \cdot \prod_{\lambda=1}^{m} w_\lambda^{\epsilon_{ij\lambda}})^y\end{aligned}$$
where the symbols i, j, k respectively denote the index of the server, the index of the block at a server, and the index of a segment in a certain coded block. Then, authenticator set $\Phi = \{\Phi_i = \{\sigma_{ijk}\}_{1\le j\le\alpha, 1\le k\le s}\}_{1\le i\le n}$ and coded block set $\Psi = \{\Psi_i = \{v_{ij}\}_{1\le j\le\alpha}\}_{1\le i\le n}$ are obtained. Finally, the data owner outsources these two sets across n cloud servers, specifically sending $\{\Phi_i, \Psi_i, t\}$ to server i and deleting them from the local storage.

III. Our Attack

In Liu *et al.*’s public auditing protocol [2], their security model indicates that *“The proxy is semitrusted, it will not collude with the servers but might attempt to forge authenticators for some specified invalid blocks to pass the following verification.”* However, in this section, we will show that the proxy in their protocol can forge an authenticator for any data block. The detailed steps of our attack are as follows:

1) Besides the delegation key x, the proxy also has the authenticator set $\Phi = \{\Phi_i = \{\sigma_{ijk}\}_{1\le j\le\alpha, 1\le k\le s}\}_{1\le i\le n}$ and the coded block set $\Psi = \{\Psi_i = \{v_{ij}\}_{1\le j\le\alpha}\}_{1\le i\le n}$.
2) Therefore, the proxy can pick up the authenticators σ_{ijk} and $\sigma_{ij(k+1)}$, where
$$\sigma_{ijk} = H(ID||i||j||k)^x (u^{v_{ijk}} \cdot \prod_{\lambda=1}^{m} w_\lambda^{\epsilon_{ij\lambda}})^y$$
$$\sigma_{ij(k+1)} = H(ID||i||j||k+1)^x (u^{v_{ij(k+1)}} \cdot \prod_{\lambda=1}^{m} w_\lambda^{\epsilon_{ij\lambda}})^y$$
Note that, for σ_{ijk} and $\sigma_{ij(k+1)}$, $\prod_{\lambda=1}^{m} w_\lambda^{\epsilon_{ij\lambda}}$ is identical.
3) With delegation key x, the proxy can compute
$$A = \frac{\sigma_{ijk}}{H(ID||i||j||k)^x} = (u^y)^{v_{ijk}} \cdot (\prod_{\lambda=1}^{m} w_\lambda^{\epsilon_{ij\lambda}})^y$$
$$B = \frac{\sigma_{ij(k+1)}}{H(ID||i||j||(k+1))^x} = (u^y)^{v_{ij(k+1)}} \cdot (\prod_{\lambda=1}^{m} w_\lambda^{\epsilon_{ij\lambda}})^y$$
4) With A and B, the proxy can easily compute
$$\begin{aligned}\frac{A}{B} &= \frac{(u^y)^{v_{ijk}} \cdot (\prod_{\lambda=1}^{m} w_\lambda^{\epsilon_{ij\lambda}})^y}{(u^y)^{v_{ij(k+1)}} \cdot (\prod_{\lambda=1}^{m} w_\lambda^{\epsilon_{ij\lambda}})^y} \\ &= (u^y)^{(v_{ijk} - v_{ij(k+1)})}\end{aligned}$$
and thus obtain
$$C = (u^y) = (\frac{A}{B})^{(v_{ijk} - v_{ij(k+1)})^{-1}}$$
Note that $(v_{ijk} - v_{ij(k+1)})^{-1} \bmod p$ can be easily computed and publicly known to all.
5) With A and C, the proxy can easily compute
$$D = \prod_{\lambda=1}^{m} (w_\lambda^{\epsilon_{ij\lambda}})^y = \frac{A}{C^{v_{ijk}}}$$
6) With C, D, and x, the proxy can forge an authenticator for any data block. Assume the encoded data block is v_{ijk}', and the forged authenticator will be
$$\begin{aligned}\sigma_{ijk} &= H(ID||i||j||k)^x (C^{v_{ijk}'}) \cdot D \\ &= H(ID||i||j||k)^x (u^{v_{ijk}'} \cdot \prod_{\lambda=1}^{m} w_\lambda^{\epsilon_{ij\lambda}})^y\end{aligned}$$
Obviously, σ_{ijk} is a valid authenticator for the encoded data block v_{ijk}'.

IV. Conclusion

In this article, we have showed that Liu *et al.*’s protocol is not as secure as they claimed, i.e., the proxy delegated by the data owner can forge an authenticator for any data block [2]. We hope that by identifying the design flaw, similar weaknesses can be avoided in future public auditing protocol design.

References

[1] G. Ateniese *et al.*, “Provable data possession at untrusted stores,” in *Proc. 14th ACM Conf. Comput. Commun. Secur. (CCS)*, 2007, pp. 598–609.

[2] J. Liu, K. Huang, H. Rong, H. Wang, and M. Xian, “Privacy-preserving public auditing for Regenerating-Code-Based cloud storage,” *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 7, pp. 1513–1528, Jul. 2015.