Verifiable Message-Locked Encryption

Sébastien Canard, Fabien Laguillaumie and Marie Paindavoine

ABSTRACT

One of today's main challenge related to cloud storage is to maintain the functionalities and the efficiency of customers' and service providers' usual environments while protecting the confidentiality of sensitive data. Deduplication is one of those functionalities: it enables cloud storage providers to save a lot of memory by storing only once a file uploaded several times. However, classical encryption schemes block deduplication. One needs to use a "message-locked encryption" scheme (MLE), which allows the detection of duplicates and the storage of only one encrypted file on the server, which can be decrypted by any owner of the file. However, in most existing scheme, a user can bypass this deduplication protocol. In this article, we provide servers verifiability for MLE schemes: the servers can verify that the ciphertexts are well-formed. This property forces a customer to prove that she complied to the deduplication protocol, thus preventing her to deviate from *the prescribed functionality* of MLE. Then, we provide an MLE scheme satisfying this new security property. To achieve the deduplication consistency, our construction primarily relies on zero-knowledge proofs. Unlike Abadi et al.'s MLE, we instantiate those proofs, so that we obtain a more efficient scheme, secure in the random oracle model.

1. INTRODUCTION

Cloud computing is often promoted towards companies as a way to reduce their costs while increasing accessibility and flexibility. It is common sense to have one large computing infrastructure that companies would share instead of replicating smaller ones. This saves money and is an eco-friendlier way to distribute resources. But cloud platforms are neither cheap nor eco-friendly. The larger amount of data these platforms host, the more expensive they become. Impact on the environment grows as well. One way to address this issue is to delete identical files stored on the server by different users. This method, called *deduplication*, is widely used by cloud providers.

However, some of the cloud storage users may want to encrypt their data, distrusting an honest-but-curious cloud provider. If they use a classical secure encryption scheme, deduplication is not possible anymore: two encryptions of the same plaintext under different keys naturally yield indistinguishable ciphertexts. New kind of encryption is needed, under which it is possible to determine whether two different ciphertexts are *locked* to the same message or not.

Previous Work.

The work on the *message-locked encryption* model has been initiated by Douceur et al. [5] with their *convergent encryption* (CE) scheme. The main idea is very simple: everyone that encrypts the same message m will obtain the *same* ciphertext c. It is worth describing it, since it is simple and illustrates pretty well the different security issues. The convergent encryption protocol CE given in [5] uses a hash function H (which is modelled as a random oracle for the security proof) and a deterministic symmetric encryption scheme SE: it sets the encryption and decryption key as $K = H(M)$, where M is the message to be encrypted, and the ciphertext C is computed as $\mathsf{SE.Encrypt}(M, K)$. The ciphertext is concatenated to a tag $\tau = H(C)$ which allows the server to easily detect duplicates. When the server receives a new ciphertext, it discards the file if the tag equals one already in its database.

In [3], authors point out the lack of a formal security investigation of this emerging model. They formally introduce the concept of *message-locked encryption* (MLE) and provide a complete security analysis. In particular, they show that a secure MLE does not need to be deterministic to achieve its goal. It is sufficient (and more general) to provide an equality testing procedure that publicly checks if two ciphertexts encrypt the same plaintext, as shown in [1]. The interactive case has recently been studied in [2].

Security.

As other kinds of "searchable encryption", MLE stands at the boundary of deterministic and probabilistic encryption worlds. As such, it cannot provide the standard notions of semantic security. Likewise, security can only be achieved for unpredictable data (having

high min-entropy). If one can guess a possible message, one can encrypt it and then easily test ciphertexts for equality. But, as pointed out previously, deterministic encryption is too restrictive. In previous works two privacy properties (PRV for short) were defined, both stating that privacy should hold for an adversary being able to choose the distribution where the messages are drawn (hence the notion of CDA, for Chosen Distribution Attack). In the PRV$-CDA [3] experiment, the adversary has to distinguish a ciphertext according to a distribution of its choice from a random bit sequence. The PRV-CDA2 [1] adds the parameter dependence setting, for which the security should hold even for messages that depend on the public parameters. They are then given to the adversary who chooses a distribution. Abadi et al. [1] have also slightly modified the security experiment, compared to [3], introducing a real-or-random oracle that gives to the encryption algorithm either a set of (unpredictable) messages drawn from the adversary's chosen distribution, or a true randomly chosen set of (unpredictable) messages. The adversary has to distinguish between both cases.

In addition, authors in [3], introduce the natural requirement of *tag consistency*, whose goal is to make it impossible to undetectably replace a message by a fake one. It states that if two tags are equal, then underlying messages should be equal.

Our contributions.

In this article, we investigate the converse: if two messages are equal, does the server always perform deduplication? Strangely enough, in almost all previous CE and MLE schemes [5, 3, 2], it is straightforward for a user to avoid the deduplication process altogether. The main feature for which those schemes were introduced, namely deduplication, is not achieved. In those schemes, the server does not actually verify that the key has actually been computed as required. In order to achieve verifiability in MLE, we introduce a new notion of *deduplication consistency*. It states that an equality test run on two valid ciphertexts with the same underlying plaintext will output 1 with overwhelming probability. Verifiability is a classical notion to prevent denial-of-service attacks, but this can be also useful in some scenarios. A court could oblige a cloud service provider to delete *all* copies of a given file, for example a newspaper article (right-to-be-forgotten) or a media file (for copyright infringement). If users are able to escape the deduplication process, the cloud service provider would not be able to prove that he complied to the court decision.

Another issue in cloud storage is efficiency, as people usually expect instant uploading and responses from the cloud storage provider. The equality testing should be fast enough to fulfil this requirement. Moreover, the ciphertexts' expansion should be carefully controlled, as the deduplication main goal is to save space storage. As such, neither the generic non-interactive zero-knowledge proof (NIZK) used in [1] nor fully homomorphic encryption used in [2] could be considered as acceptable solutions.

In this paper, we propose a new, deduplication-consistent construction for message-locked encryption in the random-oracle model. As we use NIZK to ensure verifiability, the PRV$-CDA property is not adapted to our scheme. However, as we are not parameter-independent, we modify the PRV-CDA2 property to suit our scheme. Thus, we define the parameter independant PRV-piCDA security notion, that requires as PRV-CDA2 that encryptions of random messages should be undistinguishable from encryptions of messages drawn from a chosen distribution, but, the public parameters of the scheme are hidden from the adversary (as in the PRV$-CDA game).

As explained in [1], a natural way to provide the verifiability of a ciphertext in a scheme of e.g. [5], is to provide a NIZK proof that the key $K = H(M)$ is correctly computed from the message M, and the ciphertext $C = \mathsf{SE.Encrypt}(M, K)$ is also consistent w.r.t. the same message M and key K. But a NIZK on a non-NP language is not possible, and the hash function H here needs to be a random oracle, which is not an NP language. The solution proposed in [1] consists in using a costly cut-and-choose technique to overcome the random oracle and a generic NIZK over circuits [6], applied to a hash function which does not need to be a random oracle. To achieve our goal, we propose a different strategy. Firstly, we do not use a standard hash function, but one having algebraic properties, for which we prove that the resulting output (the key in our case) is indistinguishable from a random one, using the leftover-hash lemma. We then use a variant of the ElGamal encryption scheme for small messages and also apply a tag construction similar to the one given in [1]. All this eventually makes it possible to use efficient NIZK over discrete logarithm relation sets [9, 7] to prove that these computations are all consistent one with each other.

Organization of the paper.

The paper is now organized as follows. In the next section, we introduce our new notion of deduplication consistency and Section 3 describes our new construction.

2. DEDUPLICATION CONSISTENCY

In precedent works, the main security requirement, besides privacy, was tag consistency, meaning that if the equality test $\mathsf{EQ}(c_1, c_2)$ outputs 1 on two ciphertexts, then the underlying plaintexts are the same. As sketched in the introduction, we tackle here the converse case: if two ciphertexts c_1 and c_2 are meant to encrypt

the same message, we require that $\mathsf{EQ}(c_1, c_2)$ outputs 1 with overwhelming probability. To capture such security issue, we introduce in the following a new security notion for message-locked encryption, called *deduplication consistency*. This notion ensures that a MLE schemes *provably* provides deduplication.

2.1 Overview

The main point of deduplication consistency is to make a MLE scheme "verifiable". In fact, if a server makes use of an MLE scheme for which it cannot be convinced that deduplication is actually enforced, he will loose the benefit he has expected from deduplication. In most existing schemes indeed (see below), only users are responsible for a smooth deduplication process. Then these schemes can easily be "deviate[d] from [their] prescribed functionality"[1].

In addition to save space storage, verifiable deduplication is a functionality that can have an interest of its own. Today, a really hot topic is the right-to-be-forgotten. An important question related to this topic is how a server can prove that he really deleted some given files. The problem is even more difficult if the files are encrypted on the server: the right to privacy of a user cannot prevail over the right to privacy of other users. It can happen however that a court asks a cloud service provider to remove a defamatory newspaper article or video from its storage space. Then the server's manager could encrypt this specific file with a verifiable MLE scheme and match it against the other files in the server. If the equality test returns one, deleting the corresponding file will be be sufficient to prove that no user can now access to this file, as no user can bypass the deduplication procedure.

2.2 Deduplication Consistency in Related Work

Considering the different solutions given by Bellare et al. [3], it is clear that all the CE, HCE1, HCE2, RCE, XHC and SXH schemes do not achieve such a property. In CE, the tag is computed as $\tau = H(C)$, where $K = H(M)$ and $C = \mathcal{SE}(K, M)$. In all the other schemes, the tag τ is equivalent to some $\mathcal{F}(\mathcal{F}(M))$, where $\mathcal{F}$ can here be either a hash function (see HCE1, HCE2, RCE), an extraction (see XHC) or a projection (see SXH) function. Then, an attacker can instead just randomly choose the key K so that she will be able to decrypt, although the tag comparison will always output false. In all these schemes, the server never verifies that the key used to encrypt the message is the same as the one used to compute the tag: a user can always cheat with the deduplication protocol.

Abadi et al. [1] have proposed two schemes. Their fully randomized scheme bonds together the key, the message, the tag and the ciphertexts, making each one of them consistent with all the others, with a zero-knowledge proof. We have thus the intuition that this scheme is deduplication consistent. There is no such relation between the message, the key, the ciphertext, and the tag in the proposed deterministic scheme.

[1]Oded Goldreich, The Foundations of Cryptography, Preface.

Experiment $\mathbf{Exp}^{DC}_{\Pi,\mathcal{A}}(\lambda)$

$\mathsf{pp} \leftarrow \mathsf{PPGen}(1^\lambda)$;
$(M, c_0, c_1) \leftarrow \mathcal{A}(1^\lambda, \mathsf{pp})$;
If $(\mathsf{Valid}(\mathsf{pp}, c_0) = 0) \vee (\mathsf{Valid}(\mathsf{pp}, c_1) = 0)$ then return 0;
If $\mathsf{EQ}(\mathsf{pp}, c_0, c_1) = 1$ then return 0;
$k_M \leftarrow \mathsf{KD}(\mathsf{pp}, M)$;
$M_0 \leftarrow \mathsf{Dec}(\mathsf{pp}, k_M, c_0)$; $M_1 \leftarrow \mathsf{Dec}(\mathsf{pp}, k_M, c_1)$;
If $M \neq M_0 \vee M \neq M_1$ then return 0;
Return 1;

Figure 1: Deduplication Security Game : $\mathbf{Exp}^{DC}_{\Pi,\mathcal{A}}(\lambda)$

2.3 Formal Definition

We define the deduplication experiment $\mathbf{Exp}^{DC}_{\Pi,\mathcal{A}}(1^\lambda)$ described on Figure 1.

Definition 1 (Deduplication consistency). *An MLE scheme* Π *is* deduplication consistent *if for any probabilistic polynomial-time* $\mathcal{A}$*, there exists a negligible function* $\nu(\lambda)$ *such that:*

$$\mathsf{Adv}^{DC}_{\Pi,\mathcal{A}}(\lambda) = Pr\left[\mathbf{Exp}^{DC}_{\Pi,\mathcal{A}} = 1\right] \leq \nu(\lambda),$$

where the random experiment $\mathbf{Exp}^{DC}_{\Pi,\mathcal{A}}(1^\lambda)$ *is described in Figure 1.*

3. A MESSAGE LOCKED ENCRYPTION WITH DEDUPLICATION CONSISTENCY

In this section, we describe our construction of a deduplication consistent MLE. Compared to the fully randomized message-locked encryption from [1], the main difference is that the secret key is derived from the message using a hash function which has algebraic properties. Thus, we avoid generic NIZK [6], gaining efficiency. More precisely, the message M is cut into small blocks $(m_1 \| \ldots \| m_\ell)$ of ρ bits, and the key is computed as $k_M = \prod_{i=1}^{\ell} a_i^{m_i} \mod p$ for publicly known a_i's. By using a variant of the leftover hash lemma [4], we prove that if the messages come from a source with high enough min-entropy, the key k_M is indistinguishable from a uniform key.

Each block m_i of the message is then encrypted using the ElGamal encryption with messages in the exponent, and the key k_M:

$$T_{1,i} = \ g_i^{r_i} \text{ and } T_{2,i} = \ h^{m_i} \cdot g_i^{r_i \cdot k_M}. \qquad (1)$$

These blocks m_i are chosen small enough to be efficiently decrypted.

It remains to create a suitable tag, which is done by using the same technique as in [1]. More precisely, we provide a pair $(\tau_1 = t_1^u, \tau_2 = t_2^{u \cdot k_M})$, which will make it possible to detect a duplication using a pairing computation. The consistency of the tag computation is done by verifying the following equations:

$$\tau_1 = t_1^u \text{ and} \quad (2)$$

$$e(\tau_1, t_2)^{k_M} = e(t_1, \tau_2). \quad (3)$$

To achieve deduplication consistency, we finally provide a NIZK proof that everything is well-formed: the key (according to the message), the tag (according to the key) and the ElGamal ciphertexts (according to both the message and the key). For this purpose, we provide an additional commitment C of the m_i's:

$$C = \prod_{i=1}^{\ell} a_i^{m_i} \cdot x^s = k_M \cdot x^s \mod p. \quad (4)$$

The main point regarding our NIZK is that we need to prove that the secret k_M (as an exponent for the groups $\mathbb{G}_1$ and $\mathbb{G}_T$) involved in the tag, the commitment and the ciphertexts is the same secret k_M as the one (as an element of the group $\mathbb{Z}_p^*$ of order p) computed from the message. Regarding the tag and the ElGamal ciphertext (equations (1) to (3)), the key k_M is seen as an exponent, and we can thus use standard and efficient ZK proofs *à la* Schnorr, making them non-interactive using the Fiat-Shamir heuristic.

As $C = k_M \cdot x^s = \prod_{i=1}^{\ell} a_i^{m_i} \cdot x^s \mod p$, we can efficiently prove the correctness of the commitment. It remains to make the link between the message and the key. For this purpose, we make use of both equations (3) and (4). More precisely, equation (3) can be rewritten as:

$$e(\tau_1, t_2)^C = e(t_1, \tau_2)^{x^s}. \quad (5)$$

Proving equation (5) is true involves the use of a double discrete logarithm. We use the techniques from [8] which slightly alter the efficiency of our construction.

4. REFERENCES

[1] Martín Abadi, Dan Boneh, Ilya Mironov, Ananth Raghunathan, and Gil Segev. Message-locked encryption for lock-dependent messages. In *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, pages 374–391, 2013.

[2] Mihir Bellare and Sriram Keelveedhi. Interactive message-locked encryption and secure deduplication. In *Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings*, volume 9020, pages 516–538. Springer, 2015.

[3] Mihir Bellare, Sriram Keelveedhi, and Thomas Ristenpart. Message-locked encryption and secure deduplication. In *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, pages 296–312, 2013.

[4] Kai-Min Chung and Salil P. Vadhan. Tight bounds for hashing block sources. In Ashish Goel, Klaus Jansen, José D. P. Rolim, and Ronitt Rubinfeld, editors, *Approximation, Randomization and Combinatorial Optimization. Algorithms and Techniques, 11th International Workshop, APPROX 2008, and 12th International Workshop, RANDOM 2008, Boston, MA, USA, August 25-27, 2008. Proceedings*, volume 5171 of *Lecture Notes in Computer Science*, pages 357–370. Springer, 2008.

[5] John R. Douceur, Atul Adya, William J. Bolosky, Dan Simon, and Marvin Theimer. Reclaiming space from duplicate files in a serverless distributed file system. In *ICDCS*, pages 617–624, 2002.

[6] Jens Groth. Short non-interactive zero-knowledge proofs. In Masayuki Abe, editor, *Advances in Cryptology – ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 341–358, Singapore, December 5–9, 2010. Springer, Berlin, Germany.

[7] Aggelos Kiayias, Yiannis Tsiounis, and Moti Yung. Traceable signatures. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 571–589, Interlaken, Switzerland, May 2–6, 2004. Springer, Berlin, Germany.

[8] Toru Nakanishi and Yuji Sugiyama. Unlinkable divisible electronic cash. In Josef Pieprzyk, Eiji Okamoto, and Jennifer Seberry, editors, *ISW 2000: 3rd International Workshop on Information Security*, volume 1975 of *Lecture Notes in Computer Science*, pages 121–134, Wollongong, NSW, Australia, December 20–21, 2000. Springer, Berlin, Germany.

[9] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In Gilles Brassard, editor, *Advances in Cryptology – CRYPTO'89*, volume 435 of *Lecture Notes in Computer Science*, pages 239–252, Santa Barbara, CA, USA, August 20–24, 1989. Springer, Berlin, Germany.