**Available Online at www.ijcsmc.com**

**International Journal of Computer Science and Mobile Computing**

**A Monthly Journal of Computer Science and Information Technology**

**ISSN 2320–088X**
**IMPACT FACTOR: 6.017**

***IJCSMC, Vol. 7, Issue. 4, April 2018, pg.82 – 86***

# SECURITY OF CLOUD DATA UNDER EXPOSURE OF SECRET KEY

**Shanthi. K. N, Radhakrishna Dodmane**
[1]Computer Science and Engineering, NMAMIT Nitte, VTU, India
[2]Associate professor, Computer Science and Engineering, NMAMIT Nitte, VTU, India
[1] shanthikn1990@gmail.com; [2] radhakrishna@nitte.edu.in

---

***Abstract— A powerful attacker can break data confidentiality by acquiring cryptographic keys, by means of coercion or backdoors in cryptographic software. Once the encryption key is exposed, the only viable measure to preserve data confidentiality is to limit the attacker's access to the ciphertext. This may be achieved, for example, by spreading ciphertext blocks across servers in multiple administrative domains—thus assuming that the adversary cannot compromise all of them. Nevertheless, if data is encrypted with existing schemes, an adversary equipped with the encryption key, can still compromise a single server and decrypt the ciphertext blocks stored therein. In this paper, we study data confidentiality against an adversary which knows the encryption key and has access to a large fraction of the ciphertext blocks. To this end, we propose Bastion, a novel and efficient scheme that guarantees data confidentiality even if the encryption key is leaked and the adversary has access to almost all ciphertext blocks. We analyze the security of Bastion, and we evaluate its performance by means of a prototype implementation. We also discuss practical insights with respect to the integration of Bastion in commercial dispersed storage systems. Our evaluation results suggest that Bastion is well-suited for integration in existing systems since it incurs less than 5% overhead compared to existing semantically secure encryption modes.***

***Keywords— Key exposure, cryptography, Security, Data confidentiality, Dispersed storage.***

---

# I. INTRODUCTION

The world recently witnessed a massive surveillance program aimed at breaking users' privacy. Perpetrators were not hindered by the various security measures deployed within the targeted services. For instance, although these services relied on encryption mechanisms to guarantee data confidentiality, the necessary keying material was acquired by means of backdoors, bribe, or coercion. If the encryption key is exposed, the only viable means to guarantee confidentiality is to limit the adversary's access to the ciphertext, e.g., by spreading it across multiple administrative domains, in the hope that the adversary cannot compromise all of them. However, even if the data is encrypted and dispersed across different administrative domains, an adversary equipped with the appropriate keying material can compromise a server in one domain and decrypt ciphertext blocks stored therein.

In this paper, we study data confidentiality against an adversary which knows the encryption key and has access to a large fraction of the ciphertext blocks. The adversary can acquire the key either by exploiting flaws or backdoors in the key-generation software, or by compromising the devices that store the keys (e.g., at the user-side or in the cloud). As far as we are aware, this adversary invalidates the security of most cryptographic solutions, including those that protect encryption keys by means of secret-sharing (since these keys can be leaked as soon as they are generated). To counter such an adversary, we propose Bastion, a novel and efficient scheme which ensures that plaintext data cannot be recovered as long as the adversary has access to at most all but *two* ciphertext blocks, even when the encryption key is exposed. Bastion achieves this by combining the use of standard encryption functions with an efficient linear transform. In this sense, Bastion shares similarities with the notion of all-or-nothing transform. An AONT is not an encryption by itself, but can be used as a pre-processing step before encrypting the data with a block cipher. This encryption paradigm—called AON encryption— was mainly intended to slow down brute-force attacks on the encryption key. However, AON encryption can also preserve data confidentiality in case the encryption key is exposed, as long as the adversary has access to at most all but one ciphertext blocks. Existing AON encryption schemes, however, require *at least* two rounds of block cipher encryptions on the data: one preprocessing round to create the AONT, followed by another round for the actual encryption. Notice that these rounds are sequential, and cannot be parallelized. This results in considerable—often unacceptable—overhead to encrypt and decrypt large files. On the other hand, Bastion requires only one round of encryption—which makes it well-suited to be integrated in existing dispersed storage systems. We evaluate the performance of Bastion in comparison with a number of existing encryption schemes. Our results show that Bastion only incurs a negligible performance deterioration (less than 5%) when compared to symmetric encryption schemes, and considerably improves the performance of existing AON encryption schemes . We also discuss practical insights with respect to the possible integration of Bastion in commercial dispersed storage systems. Our contributions in this paper can be summarized as follows:

- A new scheme is proposed called Bastion, an efficient scheme which ensures data confidentiality against an adversary that knows the encryption key and has access to a large fraction of the ciphertext blocks.
- The security of Bastion is analyzed, it prevents leakage of any plaintext block as long as the adversary has access to the encryption key and to all but two ciphertext blocks.

## II. RELATED WORK

Sharedkey deniable encryption [1], [2], [4]. An encryption scheme is "deniable" if—when coerced to reveal the encryption key—the legitimate owner reveals "fake keys" thus forcing the ciphertext to "look like" the encryption of a plaintext different from the original one—hence keeping the original plaintext private. Deniable encryption therefore aims to deceive an adversary which does not know the "original" encryption key but, e.g., can only acquire "fake" keys.

Information dispersal based on erasure codes [5], [6] has been proven as an effective tool to provide reliability in a number of cloud-based storage systems [3].

Erasure codes enable users to distribute their data on a number of servers and recover it despite some servers failures.

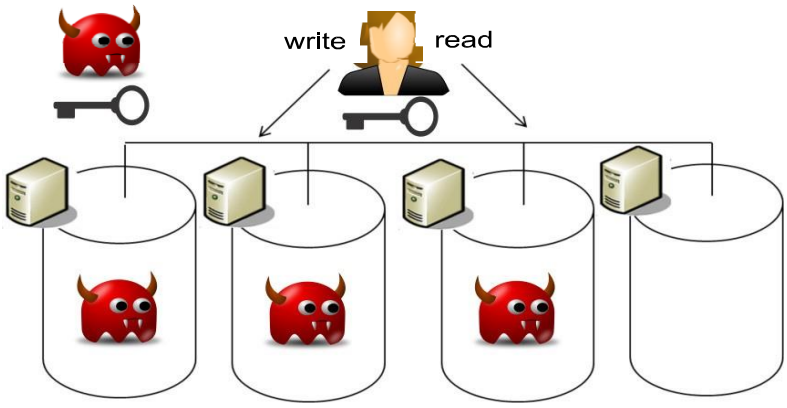
All-or-nothing transformations (AONTs) were first introduced in [11] and later studied in [8], [9]. The majority of AONTs leverage a secret key that is embedded in the output blocks. Once all output blocks are available, the key can be recovered and single blocks can be inverted. AONT, therefore, is not an encryption scheme and does not require the decryptor to have any key material.

Leakage-resilient cryptography aims at designing cryptographic primitives that can resist an adversary which learns partial information about the secret state of a system, e.g., through side-channels [10]. Different models allow to reason about the "leaks" of real implementations of cryptographic primitives [7].

## III. SYSTEM DESIGN

In this section, we start by detailing the system and security models that we consider in the paper. We then argue that existing security definitions do not capture well the assumption of key exposure, and propose new security definition that captures this notion.

### *A. System Model*



It is assumed that an adversary which can acquire all the cryptographic secret material, and can compromise a large fraction (up to all but one) of the storage servers.

Consider a multi-cloud storage system which can leverage a number of commodity cloud providers (e.g., Amazon, Google) with the goal of distributing trust across different administrative domains. This "cloud of clouds" model is receiving increasing attention nowadays, with cloud storage providers such as EMC, IBM, and Microsoft, offering products for multicloud systems . In particular, we consider a system of s storage servers S1, . . . , Ss, and a collection of users. We assume that each server appropriately authenticates users. For simplicity and without loss of generality, we focus on the read/write storage abstraction of [21] which exports two operations: write This routine splits v into s pieces {v1, . . . , vs} and sends hvji to server Sj , for j ∈ [1 . . . s]. read( · ) The read routine fetches the stored value v from the servers. For each j ∈ [1 . . . s], piece vj is downloaded from server Sj and all pieces are combined into v. We assume that the initial value of the storage is a special value ⊥, which is not a valid input value for a write operation.

### *B. Adversarial Model*

We assume a computationally-bounded adversary A which can acquire the long-term cryptographic keys used to encrypt the data. The adversary may do so either *(i)* by leveraging flaws or backdoors in the key-generation software, or *(ii)* by compromising the device that stores the keys (in the cloud or at the user). Since ciphertext blocks are distributed across servers hosted within different domains, we assume that the adversary cannot compromise all storage servers. In particular, we assume that the adversary can compromise all but one of the servers and we model this adversary by giving it access to all but λ ciphertext blocks. Note that if the adversary also learns the user's credentials to log into the storage servers and download all the ciphertext blocks, then no cryptographic mechanism can preserve data confidentiality. We stress that compromising the encryption key does not necessarily imply the compromise of the user's credentials. For example, encryption can occur on a specific-purpose device , and the key can be leaked, e.g., by the manufacturer; in this scenario, the user's credentials to access the cloud servers are clearly not compromised.

### *C. Proposed System*

In this section, we present our scheme, dubbed Bastion, which ensures that plaintext data cannot be recovered as long as the adversary has access to all but two ciphertext blocks—even when the encryption key is exposed.

Bastion departs from existing AON encryption schemes. Current schemes require a pre-processing round of block cipher encryption for the AONT, followed by another round of block cipher encryption. Differently, Bastion first encrypts the data with one round of block cipher encryption, and then applies an efficient linear post-processing to the ciphertext. By doing so, Bastion relaxes the notion of all-or-nothing encryption at the benefit of increased performance.

## IV. IMPLEMENTATION

This concept is implemented in java with the following hardware and software requirements

*A. Hardware Requirements*

- Any Processor above 500 MHz.
- 128Mb RAM
- 15 Gb Hard disk
- Standard Mouse and keyboard
- VGA and HR Monitor

*B. Software Requirements*

- Back End: Java Netbeans
- Front End: JSP
- Database: MySQL

## V. CONCLUSION

In this paper, the problem of securing data outsourced to the cloud against an adversary which has access to the encryption key is addressed. For that purpose, a novel security definition is introduced that captures data confidentiality against the new adversary. New approach is proposed, it is a scheme which ensures the confidentiality of encrypted data even when the adversary has the encryption key, and all but two ciphertext blocks.

This paper is most suitable for settings where the ciphertext blocks are stored in multi-cloud storage systems. In these settings, the adversary would need to acquire the encryption key, and to compromise all servers, in order to recover any single block of plaintext.

# REFERENCES

[1] Beimel, *"Secret-sharing schemes: A survey,"* in Interna- tional Workshop on Coding and Cryptology (IWCC), 2011, pp. 11–46.
[2] H. Krawczyk, *"Secret Sharing Made Short,"* in Advances in Cryptology (CRYPTO), 1993, pp. 136–146.
[3] C. Charnes, J. Pieprzyk, and R. Safavi-Naini, *"Conditionally secure secret sharing schemes with disenrollment capability,"* in ACM Conference on Computer and Communications Security (CCS), 1994, pp. 89–95
[4] A. Shamir, *"How to Share a Secret?"* in Communications of the ACM, 1979, pp. 612–613.
[5] R. Canetti, C. Dwork, M. Naor, and R. Ostrovsky, *"Deniable Encryption,"* in Proceedings of CRYPTO, 1997.
[6] J. H. van Lint, *Introduction to Coding Theory*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1982.
[7] M. Abd-El-Malek, G. R. Ganger, G. R. Goodson, M. K. Reiter, and J. J. Wylie, *"Fault-Scalable Byzantine Fault-Tolerant Services,"* in ACM Symposium on Operating Systems Principles (SOSP), 2005, pp. 59–74.
[8] M. K. Aguilera, R. Janakiraman, and L. Xu, *"Using Erasure Codes Efficiently for Storage in a Distributed System,"* in International Conference on Dependable Systems and Networks (DSN), 2005, pp. 336–345.
[9] G. R. Blakley and C. Meadows, *"Security of ramp schemes,"* in Advances in Cryptology (CRYPTO), 1984, pp. 242–268.
[10] S. Micali and L. Reyzin, *"Physically observable cryptography (extended abstract),"* in Theory of Cryptography Conference (TCC), 2004, pp. 278–296.
[11] R. L. Rivest, *"All-or-Nothing Encryption and the Package Transform,"* in International Workshop on Fast Software Encryption (FSE), 1997, pp. 210–218.