**Securing Cloud Data under Key Exposure**

**ABSTRACT:**

Recent news reveal a powerful attacker which breaks 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.

**EXISTING SYSTEM:**

* 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.
* Ramp schemes constitute a trade-off between the security guarantees of secret sharing and the efficiency of information dispersal algorithms. A ramp scheme achieves higher “code rates” than secret sharing and features two thresholds t1, t2. At least t2 shares are required to reconstruct the secret and less than t1 shares provide no information about the secret; a number of shares between t1 and t2 leak “some” information.
* Resch et al. combine AONT and information dispersal to provide both fault-tolerance and data secrecy, in the context of distributed storage systems. In existing system, however, an adversary which knows the encryption key can decrypt data stored on single servers.

**DISADVANTAGES OF EXISTING SYSTEM:**

* 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.

**PROPOSED SYSTEM:**

* 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). 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.

**ADVANTAGES OF PROPOSED SYSTEM:**

* 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 propose 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.
* We analyze the security of Bastion, and we show that 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.
* We evaluate the performance of Bastion analytically and empirically in comparison to a number of existing encryption techniques. Our results show that Bastion considerably improves (by more than 50%) the performance of existing AON encryption schemes, and only incurs a negligible overhead when compared to existing semantically secure encryption modes (e.g., the CTR encryption mode).
* We discuss practical insights with respect to the deployment of Bastion within existing storage systems, such as the HYDRAstor grid storage system.

**SYSTEM ARCHITECTURE:**



**MODULES:**

* Data Owner
* Data User
* Admin

**MODULES DESCRIPTION:**

**Data Owner:**

In Data Owner module, Initially Data Owner must have to register their detail and admin will approve the registration by sending signature key and private key through email. After successful login he/she have to verify their login by entering signature and private key. Then data Owner can upload files into cloud server with Polynomial key generation. He/she can view the files that are uploaded in cloud by entering the secret file key.

**Data User:**

In Data User module, Initially Data Users must have to register their detail and admin will approve the registration by sending signature key and private key through email. After successful login he/she have to verify their login by entering signature and private key. Data Users can search all the files upload by data owners. He/she can send search request to admin then admin will send the search key. After entering the search key he/she can view the file

**Admin:**

In Admin module, Admin can view all the Data owners and data user’s details. Admin will approve the users and send the signature key and private key to the data owners and data users. Also admin will send the search request key to the users. Admin can able see the files in cloud uploaded by the data owners.

**SYSTEM REQUIREMENTS:**

**HARDWARE REQUIREMENTS:**

* System : Core i3.
* Hard Disk : 120 GB.
* Monitor : 15’’ LED
* Input Devices : Keyboard, Mouse
* Ram : 4 GB

**SOFTWARE REQUIREMENTS:**

* Operating system : Windows 10.
* Coding Language : JAVA/J2EE
* Tool : Eclipse Photon
* Database : MYSQL 5.5/5.6

**REFERENCE:**

Ghassan O. Karame, *Member, IEEE*, Claudio Soriente, *Member, IEEE*, Krzysztof Lichota, Srdjan Capkun, *Senior Member, IEEE*, “Securing Cloud Data under Key Exposure”, **IEEE Transactions on Cloud Computing, 2017.**