Mix&Slice: Efficient Access Revocation in the Cloud

Enrico Bacis [1], Sabrina De Capitani Di Vimercati [2], Sara Foresti [2], Stefano Paraboschi [1], Marco Rosa [1], Pierangela Samarati [2]
[1] Università degli Studi di Bergamo (Italy), [2] Università degli Studi di Milano (Italy)

CSAW Europe 2017 (originally presented at ACM CCS'16)

Scenario

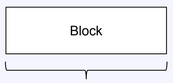
- **Encryption** is increasingly used to protect stored data from both other users and cloud providers
- In presence of multiple users, the **access policy** maps to the ability of each user to **get/derive the encryption key** for the resources she should be allowed to access
- **Problem**: enforcement of authorization revocation

Alternatives to Manage User Revocation

- The key becomes unavailable to the revoked user
  - efficient, but the user may have kept a copy of the key
- The resource is re-encrypted with a fresh key
  - inefficient, the entire resource has to be re-encrypted
  - especially inefficient in a cloud environment (download → re-encrypt → re-upload)
- The resource is protected with an additional encryption layer (i.e., over-encryption)
  - it requires support by the server
- Our solution: Mix&Slice

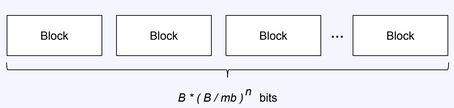
Basic Concepts

- **Block**: A sequence of bits input to a block cipher (it corresponds to the classical block concept). AES is today the most common option ($B = 128$ bits).



- **Mini-block**: A portion of a block that defines the atomic unit of information (either available in its entirety or missing). Its size must be a divisor of the size of the block.

- **Macro-block**: A sequence of blocks. It allows extending the application of the block cipher on sequences of bits larger than individual blocks.



Our approach partitions the resource in distinct, equally sized, macro-blocks. Then it performs two processes: mixing and slicing.

Mixing

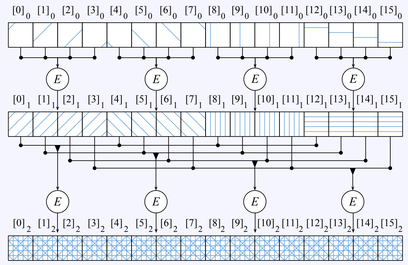


Figure: The mixing process creates the inter-dependency

A symmetric block cipher guarantees **complete dependency of the encrypted result**. This guarantee is not provided when the data to encrypt is larger than a block.

In our mixing process the content of each macro-block is processed by an **iterative application of multiple encryption rounds** together with a carefully designed **mini-block mixing**, that ensures that every mini-block of the input has had impact on each of the mini-blocks in the encrypted output.

Unmixing

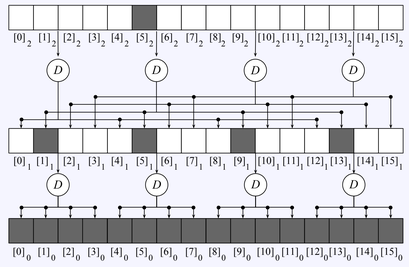
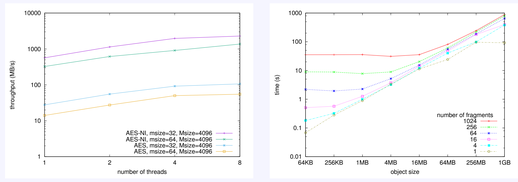


Figure: The lack of a micro-block protects the whole macro-block

If a mini-block in the encrypted representation is missing, the encryption cannot be inverted, even with knowledge of the key. This is an All-or-Nothing Transform (*AONT*).

Other AONT techniques derive a key from the encrypted representation. Storing this key permits to bypass the protection. In our scenario a revoked user must not be able to bypass the protection with a computational and local storage cost significantly lower than the cost of storing the resource itself.

Implementation



(Mixing throughput) (Fragment size impact)

Figure: Experimental results of Mix&Slice based on AES / AES-NI

- For the implementation and experimental evaluation, the components that have to be considered are the **client**, which decrypts resources to access them, and the **algorithm parameters** such as *macro-block size* and *number of fragments*, which affect the interaction between client and server.
- The client implementation has been written in C and is able to leverage hardware implementation of AES (AES-NI), with **excellent performance** (up to 2.5GB/s). Experimenting with the **Swift server**, we observed that the number of fragments has to be selected based on the resource size, as the fragmentation overhead is significant for small resources.
- `https://github.com/unibg-seclab/aesmix`

All-or-Nothing Transform for User Revocation

Use an encryption structure that slices the resource and then mixes all pieces within each slice, making **the decryption dependent on the availability of all the pieces** (i.e., All-or-Nothing Transform). When the policy changes, update the encrypted representation of a randomly chosen piece. If there are n pieces, **the resource can become unavailable with the update of $1/n$-th of the resource**.

Slicing

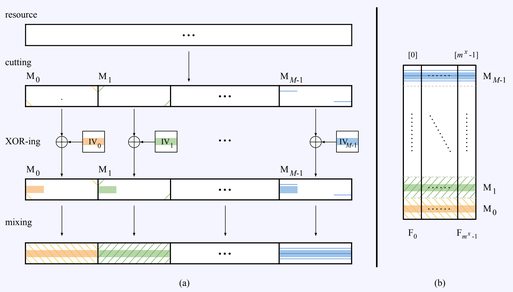


Figure: Macro-blocks can be sliced into fragments

Large macro-blocks introduce shortcomings such as: **reduction in decryption efficiency**, and **data transfer overhead** when only a small portion of the macro-block is needed.

These problems can be mitigated by limiting the macro-block size: (*a*) a resource can be split into macro-blocks, (*b*) all the mini-blocks in the same position are part of the same **fragment**. Each fragment is necessary to reconstruct the resource (*or even portions of it*).

Policy Update

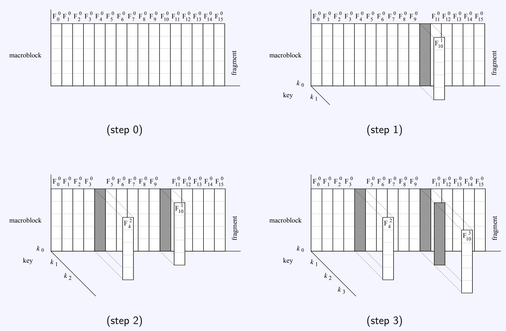


Figure: An example of fragments evolution

The base fragments F_0^i are produced by the application of Mix&Slice (step 0). When a user is revoked access to the resource, only a randomly chosen fragment needs to be re-encrypted (steps 1-3). At step j, the fragment F_0^i is replaced by fragment $F_j^i = E_{k_j}(F_0^i)$. **Only Authorized users will be able to access k_j.**

Using a *key regression* technique based on RSA, all keys k_i can be derived from $k_j, \forall i < j$, with no need to store additional information.

Conclusions

- Mix&Slice efficiently enforces access revocation on encrypted resources stored at external providers even for users locally maintaining copies of previously-used keys.
- The implementation and experimental evaluation confirm the efficiency and effectiveness of the proposal, and confirms its compatibility with current cloud storage solutions.

References

Enrico Bacis, Sabrina De Capitani di Vimercati, Sara Foresti, Stefano Paraboschi, Marco Rosa, and Pierangela Samarati.
Mix&slice: Efficient access revocation in the cloud.
In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016.

Sarah M Diesburg and An-I Andy Wang.
A survey of confidential data storage and deletion methods.
ACM Computing Surveys (CSUR), 43(1):2, 2010.

Ronald L Rivest.
All-or-nothing encryption and the package transform.
In *International Workshop on Fast Software Encryption*. Springer, 1997.

Acknowledgments

This work was supported by the EC within the H2020 under grant agreement 644579 (ESCUDO-CLOUD).