Eliatra Suite Encrypted Indices

Native Encryption at Rest for OpenSearch

# Abstract

What’s currently missing in the OpenSearch ecosystem is a first-class citizen functionality for Encryption at Rest. That is encrypting the data stored on the filesystem which can only be decrypted with the right cryptographic key. Beside data privacy there is also data integrity we want to assure for the stored data. Protecting the integrity of data means that malicious modification of data will not remain unnoticed.

# Why?

The typical solution to achieve Encryption at Rest is to use a technique called “whole disk encryption” (WDE) or use storage level encryption provided by a cloud provider like encrypted AWS EBS volumes. Whole Disk Encryption guards against hard disk stealing or, when used in cloud environments, against unauthorized access to an EBS volume. It’s probably a very good idea to use WDE generally if you deal with sensitive data but it guards against a specific attack vector which might not be enough when using OpenSearch.

This is because you want to control which data will be encrypted and how. Furthermore, you want to have a key management functionality to grant and revoke access to encrypted data. Finally, you likely want to also encrypt your backups or data exports.

The solution we offer does all this by encrypt data before it gets stored and indexed into OpenSearch. Encryption is configured per index and every field value in such an index which contains text or binary data will be encrypted. Your applications remain unchanged. All you need to do is to send an encryption key, as Http header, along with the indexing requests.

The initial creator of an encrypted index can add other encryptions keys to allow encryption and therefore share the data with other users or client applications.

Searching in encrypted indices works similar. Just add an Http header containing the encryption key to your search request and you are ready. For OpenSearch Dashboards this will be done automatically after an initial setup.

Eliatra Suite Encrypted Indices ensures that data values for encrypted indices are never stored in plaintext on the hard disk or storage system.

# Protection levels

There are four main threats for which Eliatra Suite Encrypted Indices can provide a mitigation:

1. Attacker gains read/write access to RBAC protected OpenSearch Indices via HTTPS because of a weak or stolen password or misconfigured permissions

Because an attacker lacks the knowledge of the encryption key an index request will fail and a search request will yield only results in their encrypted form.

This attack vector can be mitigated well and truly if the encryption keys remain private.

1. Attacker gains access to OpenSearch backups (i.e., snapshots)

The data of encrypted indices which are contained in a snapshot is encrypted.

Therefore, it is not possible for an attacker to decrypt the data without knowledge of the encryption key.

This attack vector can also be mitigated well and truly if the encryption keys remain private.

1. Attacker gains read/write access to the storage system (i.e., hard disk)

All data of encrypted indices are never stored unencrypted on the storage system.

So, it is not possible for an attacker to decrypt the data without knowledge of the encryption key. It is also not possible to modify the encrypted data because this would be noticed during decryption as tempered data.

This attack vector can be mitigated full-on if the encryption keys remain private.

1. Attacker gains root access to the running operating system

This is the worst case that could happen. Even then our solution can dramatically improve data privacy and integrity because it makes it very hard for an attacker to locate and retrieve the necessary key material from memory (RAM) to decrypt the data stored on the hard disk.

Sensitive key material is stored in off-heap memory and only if a request, which targets encrypted data, is executing. So, an attacker needs to wait for such a request and then locate the data in native memory before the request completes.

Eliatra Suite Encrypted Indices is not an End-to-End encryption solution but as close as possible without having the implications to encrypt, analyze and decrypt data on the client side. Therefore, this attack vector cannot be mitigated full-on but in practice it’s almost impossible for an attacker to get access to the encrypted data.

# How it works?

The solution will contain a command line tool which helps with creating and managing encrypted indices. There will also be a configuration user interface in OpenSearch dashboards for this.

To get started one need to create an encrypted index. This works like creating a normal index but a few additional index settings and creating a set of keys. This can be done by hand using the normal OpenSearch API’s but also by a simplified dedicated “create encrypted index” API or via the graphical user interface in OpenSearch Dashboards. The person or process which initially creates the encrypted index will be the data owner of data stored in this index. Only the data owner can grant and revoke access for other keys (i.e., other users and client applications).

The data owner (as well as others when granted access) can ingest and search data by providing parts of the previously generated key material as an HTTP header in every request.

Without a proper key it is impossible to ingest and search for data. Any data returned by search request (for a match all query for example) will be encrypted and therefore useless for the requestor.

The encryption functionalities can be used together with the general Eliatra Suite security features (like RBAC) but also without them. In the latter case security would only be based solely on cryptography.

# How to get it

There is already a technical preview available. If you are interested, please contact us.

The solution will be put under a commercial license, but the code will be made public nevertheless for review purposes. We offer also support and custom feature development.

Some parts of the solution will be released under the Apache 2 license to contribute back to the Open-Source ecosystem.

# Register as beta tester

If you like to help us testing this functionality register as beta tester and receive a free license.

**INTERNAL NOTES**

See diagrams in attached power point presentation

I suggest redrawing the diagrams to match eliatra CI.