HDFS Encryption and Hadoop Key Management Server (KMS) – A Detailed Guide

Introduction to HDFS Encryption

HDFS Transparent Data Encryption (TDE) ensures that data at rest is encrypted while maintaining seamless access for authorized users. It allows data to be automatically encrypted before storage on HDFS and decrypted on-the-fly when accessed by authorized users.

HDFS encryption relies on two key components:

HDFS Encryption Zones – Special directories where all files are encrypted.
Hadoop Key Management Server (KMS) – Manages encryption keys securely.

How HDFS Client Writes Data to an HDFS Secured (Encryption) Zone

When an HDFS client writes data to an HDFS Encryption Zone, the data is automatically encrypted before it is stored. The encryption and decryption processes are transparent to users and applications, ensuring security without requiring modifications to their code.

Step-by-Step Sequence of Writing Data to an HDFS Encryption Zone

1️⃣ HDFS Client Requests to Write Data

The HDFS client initiates a write operation inside an HDFS Encryption Zone.

(The client does not directly handle encryption keys; all encryption is handled transparently)

2 NameNode Retrieves Encryption Zone Key (EZ Key)

The NameNode checks if the target directory is part of an Encryption Zone (EZ).
If it is, the NameNode retrieves the Encryption Zone Key (EZ Key) reference (but never the actual key).
The actual EZ Key is securely stored in the Hadoop Key Management Server (KMS). The NameNode records the Encrypted Data Encryption Key (EDEK) in the file metadata.

3 Client Requests a Data Encryption Key (DEK)

The HDFS client initiates a request for a Data Encryption Key (DEK) from the NameNode. In response, the NameNode contacts the Key Management Server (KMS) to generate a unique DEK for the specified file. The KMS then encrypts the DEK using the EZ Key, resulting in an Encrypted Data Encryption Key (EDEK). Finally, the NameNode sends the EDEK back to the client.

4 Client Encrypts Data Using the DEK
The HDFS client sends the Encrypted Data Encryption Key (EDEK) to the Key Management Server (KMS) to obtain the actual Data Encryption Key (DEK). The client then uses this DEK to encrypt the data before transmitting it to HDFS. This client-side encryption ensures that data is never stored in an unencrypted form.

5 Encrypted Data Blocks are Sent to DataNodes

The HDFS client divides the encrypted data into blocks and writes them to HDFS DataNodes.

DataNodes store only the encrypted data – they never see decrypted data.

Step-by-Step Sequence of Reading Data from an HDFS Encryption Zone

1️⃣ HDFS Client Requests to Read a File

The HDFS client initiates a read operation for a file inside an HDFS Encryption Zone.

(The client is unaware of encryption, meaning no application code changes are required.)

2️⃣ Client Retrieves Encrypted Data Encryption Key (EDEK)

The NameNode checks if the file is in an Encryption Zone. If yes, it retrieves the Encrypted Data Encryption Key (EDEK) from file metadata.

3️⃣ Client Sends the EDEK to KMS for Decryption

The HDFS client sends the EDEK to the Key Management Server (KMS).

4️⃣ KMS Decrypts the EDEK and Returns the DEK

If the client has the correct permissions, KMS decrypts the EDEK using the Encryption Zone Key (EZ Key).

KMS sends the decrypted DEK back to the HDFS client.

5️⃣ HDFS Client Uses the DEK to Decrypt Data

The HDFS client retrieves the encrypted data blocks from HDFS DataNodes.