RFC: HDF5 Encryption VFD Overview

Kyle Lofthus kyle.lofthus@lifeboat.llc
John Mainzer john.mainzer@lifeboat.llc

This document described Encryption VFD implemented for enabling data encryption in HDF5.

1   Introduction

The purpose of the Encryption VFD is to keep HDF5 files encrypted at all times, ensuring sensitive data remains secure. The VFD decrypts data from the HDF5 file only as required to satisfy read requests, and data in write requests is encrypted before it is written to file.

2   Encryption

Initially, the Encryption VFD supports two encryption ciphers: AES256[1] and Twofish. Both ciphers have block sizes of 128 bits and key sizes of 256 bits, providing high security without compromising performance. Future updates may add more ciphers and key size options.

This implementation uses Cipher Block Chaining (CBC) mode. Both AES256 and Twofish encrypt 128 bit blocks of data. To prevent repetition patterns in the encrypted data, CBC incorporates randomization using an Initialization Vector (IV), the same size as the cipher blocks, which the VFD generates when encrypting a page.

Since the Encryption VFD works with paged data, encrypting a single page requires multiple cipher blocks. CBC XORs[2] the encryption key with the IV to encrypt the first block of the page, and each subsequent plaintext[3] block is XOR'd with the newly encrypted block. To further avoid repetition a new IV is generated for each page also ensuring every page is independent from other pages.

Two different page types are used, plaintext pages and ciphertext pages. The ciphertext page size is the plaintext page size plus any additional size needed for the encryption[4]. Ciphertext pages are used for the file itself to store the IV for each page in the first block of that ciphertext page unencrypted. IVs

---

[1] AES keys can also be of size 128 and 196, but currently just support 256 due to being more secure.

[2] XOR (exclusive or) is a boolean operator that compares two input bits and outputs one bit.

[3] Plaintext specifically refers to non-encrypted data that that will be encrypted or has been decrypted.

[4] If the encryption cipher increases the size of data or requires additional information, such as an IV.

Lifeboat

are not encrypted and are cleartext[5], because they are random values and cannot be used alone to decrypt a page.

3   Encrypted Files and Encryption VFD Operations

To prevent incorrect cipher configurations, when a file is created the first page stores the file's cipher configuration[6], and the second page contains an encrypted test phrase to verify the configuration with the provided key. These pages are protected from accidental tampering by incorporating an encryption offset for paged I/O requests. Thus, any I/O operations to the first or second pages are redirected by treating the third page in the file as the first page, ensuring the integrity of the cipher configuration and test phrase.

3.1   Design

The first page is cleartext, though future iterations will explore making it plaintext and encrypting it. For strong ciphers like AES and Twofish, leaving the cipher configuration as cleartext does not weaken the security of the file, especially when the mode of operation is cipher block chaining (CBC). The first page is not encrypted because the VFD needs to know the cipher configuration used to encrypt the file to be able to decrypt the file. Encrypting the first page with a user-selected cipher poses challenges, as there is no reliable method to determine the cipher used.

However, if a user-selected cipher with known vulnerabilities is used, knowing the cipher greatly decreases the file's security, necessitating encryption of the first page. One solution we plan to explore is to use a standard cipher to encrypt the first page of all files, allowing users to select a different cipher for the rest of the file. This ensures an encrypted first page while giving users flexibility in choosing their cipher. However, this solution may require two keys if the ciphers have different key sizes.

The next iteration of the Encryption VFD will store the key(s) in a secure memory pool to protect them from being exposed to potential vulnerabilities like swapping[7] or unauthorized access. For this iteration the keys are stored in regular memory for testing purposes. Key distribution is something we have explored a few options for, as an example, allowing users to create a password for the encrypted file and using a unique salt to randomize the password, and then a Password-Based Key Derivation Function (PBKDF) to convert the password into a key of appropriate size for the cipher used. However, we will review with cybersecurity experts and modify any options and their priorities as we continue to investigate the issue and gather requirements. Additionally, the current test phrase is hard coded into the Encryption VFD. Using CBC mode the plaintext is randomized before encryption and generating a new Initialization Vector for each page to further prevent repetition, makes it so strong ciphers are

---

[5] Cleartext is different than plaintext. Cleartext is data stored or sent in an unencrypted form and is expected to have never been encrypted and to not be encrypted in the future.
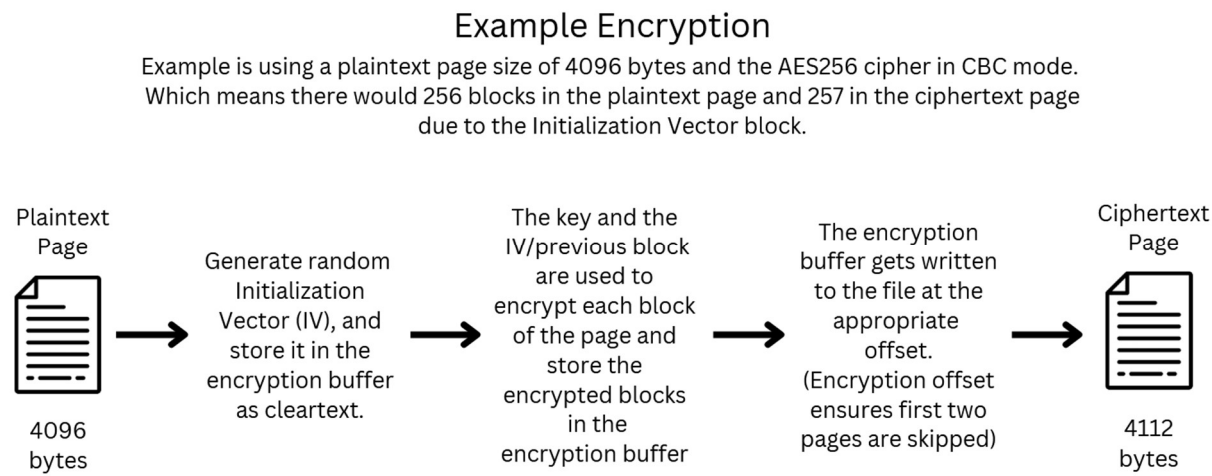
[6] Example cipher configuration shown on next page.

[7] Swapping: is a technique used by operating systems to manage physical memory, and leads to data stored in memory to be moved to a special area of disk space.

resistant to a known-plaintext attack (KPA). However, for further security future versions may incorporate a method for generating a test phrase so there will be no known plaintext.

3.2 Example: Cipher Configuration

Example Cipher Configuration
(Size of data is in bytes)

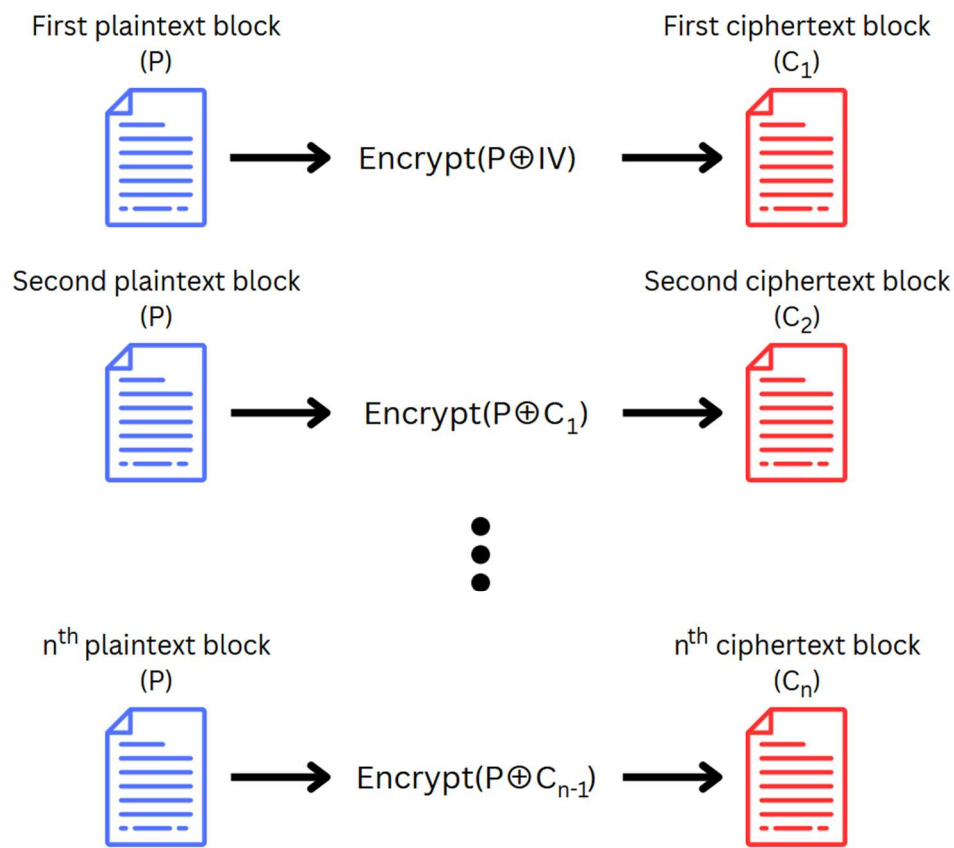| Cipher Configuration		Description
cipher:	0	Integer value which represents the cipher that the VFD needs to use to encrypt/decrypt the file (0 = AES cipher).
key size:	32	Size of the key used to encrypt/decrypt the file.
cipher block size:	16	The block size the cipher can encrypt/decrypt at a time.
mode:	0	Integer value which represents the mode of operation for handling successive cipher blocks in page (0 = CBC).
iv size:	16	Size of the initialization vector (IV). If the mode of operation uses an IV, generally it is the same size as the cipher block size.
plaintext page size:	4096	Size of the plaintext pages.
ciphertext page size:	4112	Size of the ciphertext pages (larger to store the IV)
encryption buffer size:	65792	Size of the encryption buffer, which is limited to ensure there is enough space set in memory, and must be a multiple of the ciphertext page size.

3.3 Example: Encryption

Example Encryption
Example is using a plaintext page size of 4096 bytes and the AES256 cipher in CBC mode. Which means there would 256 blocks in the plaintext page and 257 in the ciphertext page due to the Initialization Vector block.

Plaintext Page — 4096 bytes → Generate random Initialization Vector (IV), and store it in the encryption buffer as cleartext. → The key and the IV/previous block are used to encrypt each block of the page and store the encrypted blocks in the encryption buffer → The encryption buffer gets written to the file at the appropriate offset. (Encryption offset ensures first two pages are skipped) → Ciphertext Page — 4112 bytes

Lifeboat

3.4   Cipher Block Chaining

Cipher Block Chaining (CBC)

Lets say all plaintext blocks are the same (P). The first block is XORed with the IV to create (C_1). Subsequent blocks are XORed with the previous block (which is now encrypted) to create the subsequent unique ciphertext blocks.

First plaintext block
(P)

First ciphertext block
(C_1)

$\text{Encrypt}(P \oplus IV)$

Second plaintext block
(P)

Second ciphertext block
(C_2)

$\text{Encrypt}(P \oplus C_1)$

$\vdots$

n^{th} plaintext block
(P)

n^{th} ciphertext block
(C_n)

$\text{Encrypt}(P \oplus C_{n-1})$

Lifeboat

Acknowledgement

Revision History

August 2, 2024:	Version 1 circulated for comment within Lifeboat, LLC.
August 27, 2024:	Version 2 incorporates feedback and checked into GitHub repo.
September 20, 2024:	Version 3 adds more description to address questions.

Lifeboat