

**COMP9337 - Securing Fixed and Wireless Networks**

**T1 2022**

**LAB 1**

**Group: T18B 7**

**Name:** Yuhua Zhao – **ZID:** Z5404443

**Name:** Yasin Khan – **ZID:** z5265047

**Environment:**

* **Programming Language:** Python
* **Version:** 3.9.2

## **Part A: DES Encryption and Decryption:**

* **Please run the following argument to test the Outcome**

python tempdes.py fecdba9876543210 0123456789abcdef test.txt mytest.des

**Argument Explanation:**

* + Argument 1: Initialization Vector (IV)
  + Argument 2: Key to be use for encryption
  + Argument 3: Input file path of the file that will be read
  + Argument 4: Encrypted file will be generated based on the argument 4 name.

### **Part A: Code Explanation:**

Text

Description automatically generated

* + DES encryption’s input shall be multiple of 8 octets. If the Length of the input is less than that, padding is required.
  + Firstly, line 14 will read the file according to Argument 3.
  + Secondly, the program will check whether the Length of the input is multiple of 8 bytes. If yes, will return the input itself. Else it will detect how many missing bytes are and add the Padding accordingly.

Text

Description automatically generated

* + “Encrypt\_DES” function is used to encrypt the plain text
  + “Decrypt\_DES” Function is used to decrypt the Cipher test that encrypt by the Encrypt\_DES function.

Text

Description automatically generated

* + Line 47: Detect arguments when this python script is being called and assigned to the Variable respectively.
  + Line 50: Call the Read File to get the text that needs to be encrypted and encode it with the selected Format.
  + Line 53 – 55: Record the time used for DES CBC encryption.
  + Line 57 – 59: Generate a file named by Argument 4 and write Ciphertext in.
  + Line 62 – 64: Record the time used for DES CBC decryption.
  + Line 67 – 73: Print the necessary value for Lab Part B.

## **Part B: Performance Measures for Various Algorithm:**

### **File Explanation:**

|  |  |
| --- | --- |
| **File Name:** | **Explanation:** |
| TestFile | Include all the Plaintext files with different Lengths that are used to test the Algorithm Performance. |
| tempdes.py | Include DES CBC Algorithm |
| tempaes.py | Include AES CBC Algorithm |
| temprsa.py | Include RSA Algorithm |
| tempsha1.py | Include SHA-1 Algorithm |
| tempHMAC.py | Include HMAC Signature Algorithm |
| algPerformance.py | Run all the Algorithms by entering the Plaintext with different Sizes and printing out the performance the results. |

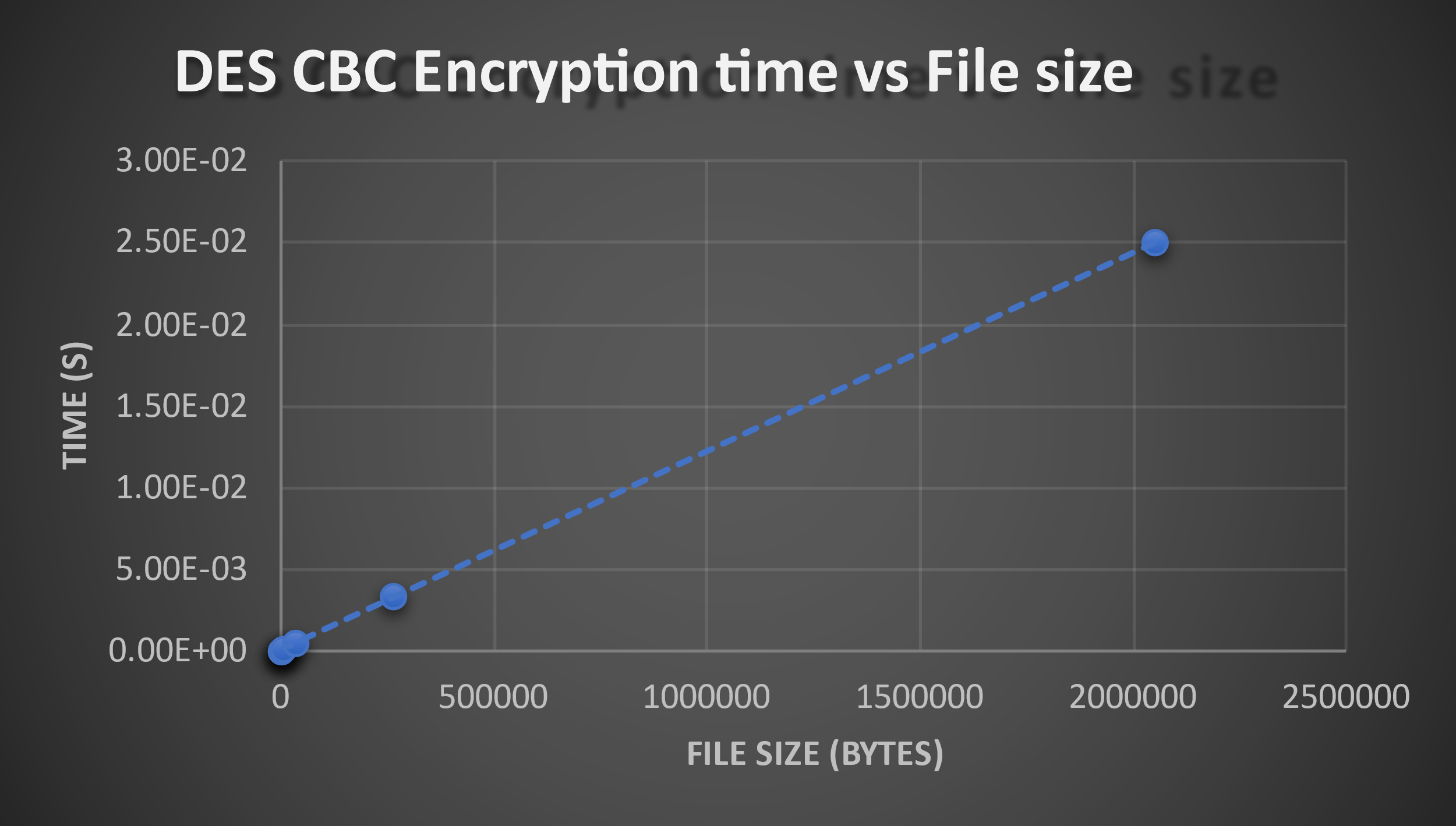
### **Code Testing:**

**Please run the following argument to test the Outcome**

**python algPerformance.py**

The command above will run all the encryption algorithms that have specified on the requirement and print out the Encryption time and Decryption according to the Algorithm.

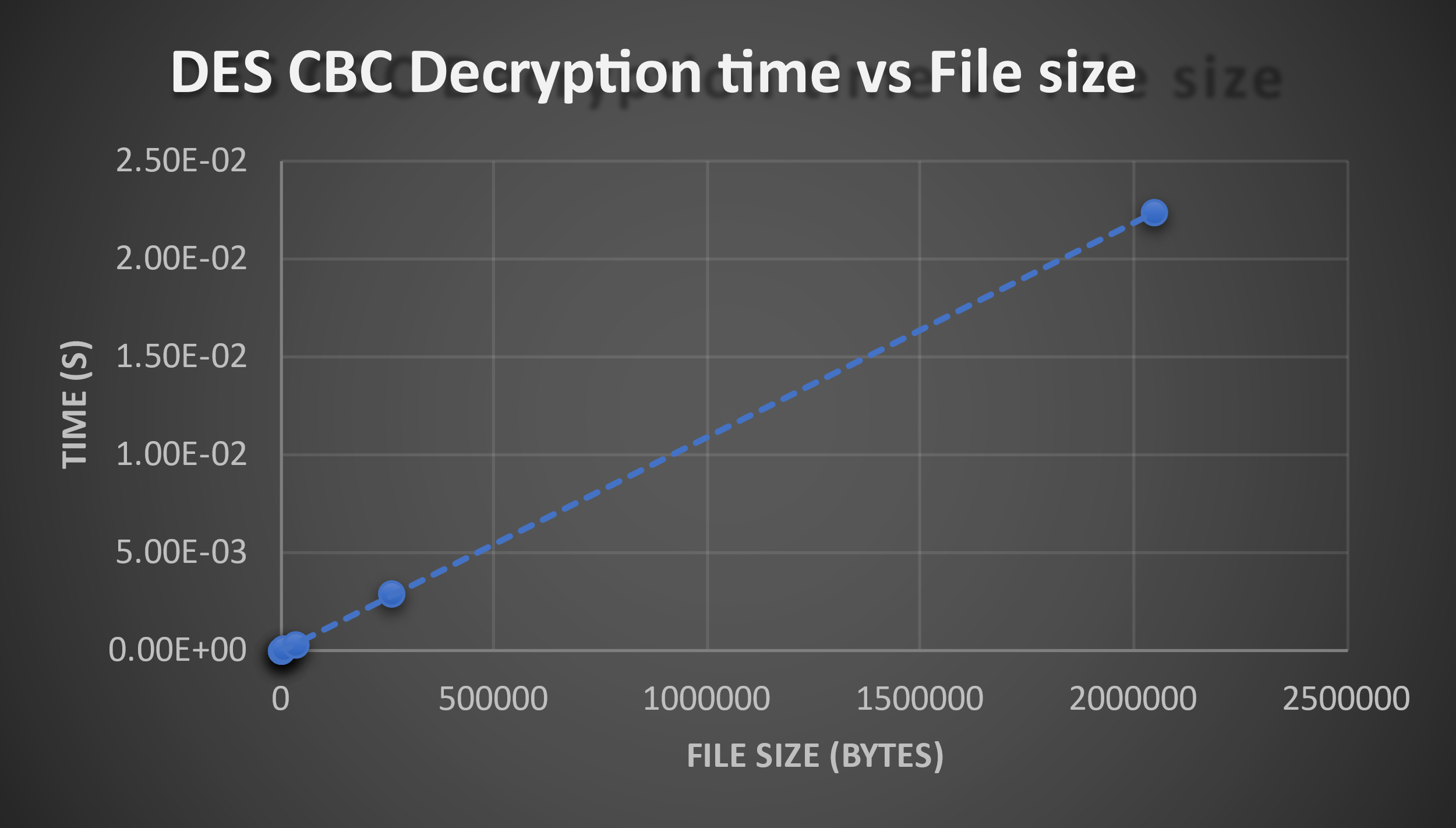
### **Algorithm Performance Co**mparison:



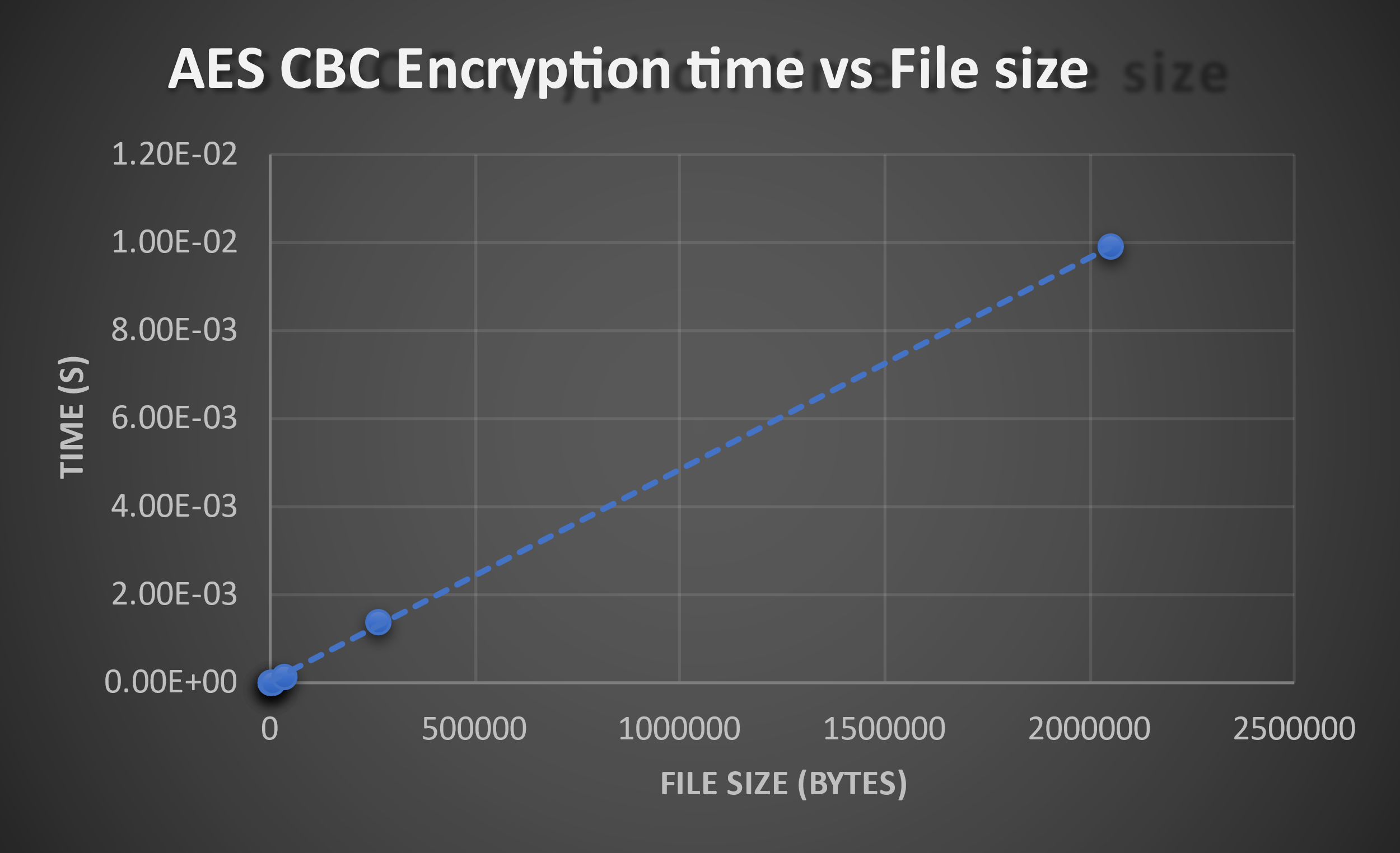
DES CBC:

Both Encryption and Decryption times seem to have a linear trend.

Decryption seems to be slightly faster at times, but times are generally comparable



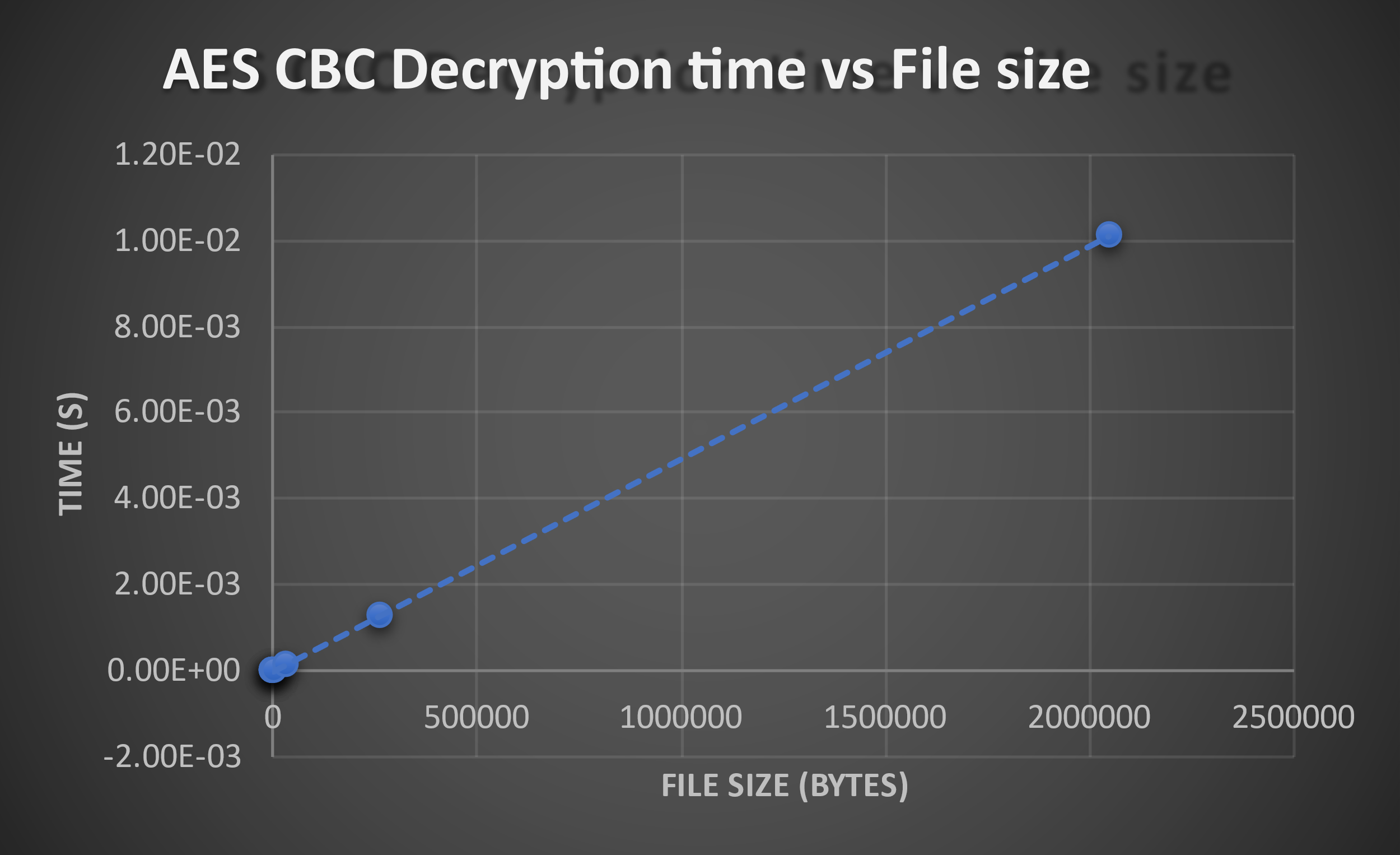
Encryption and decryption seem to be faster than RSA but slower than AES as well as the HASH algorithms



AES CBC:

Both Encryption and Decryption times seem to have a linear trend

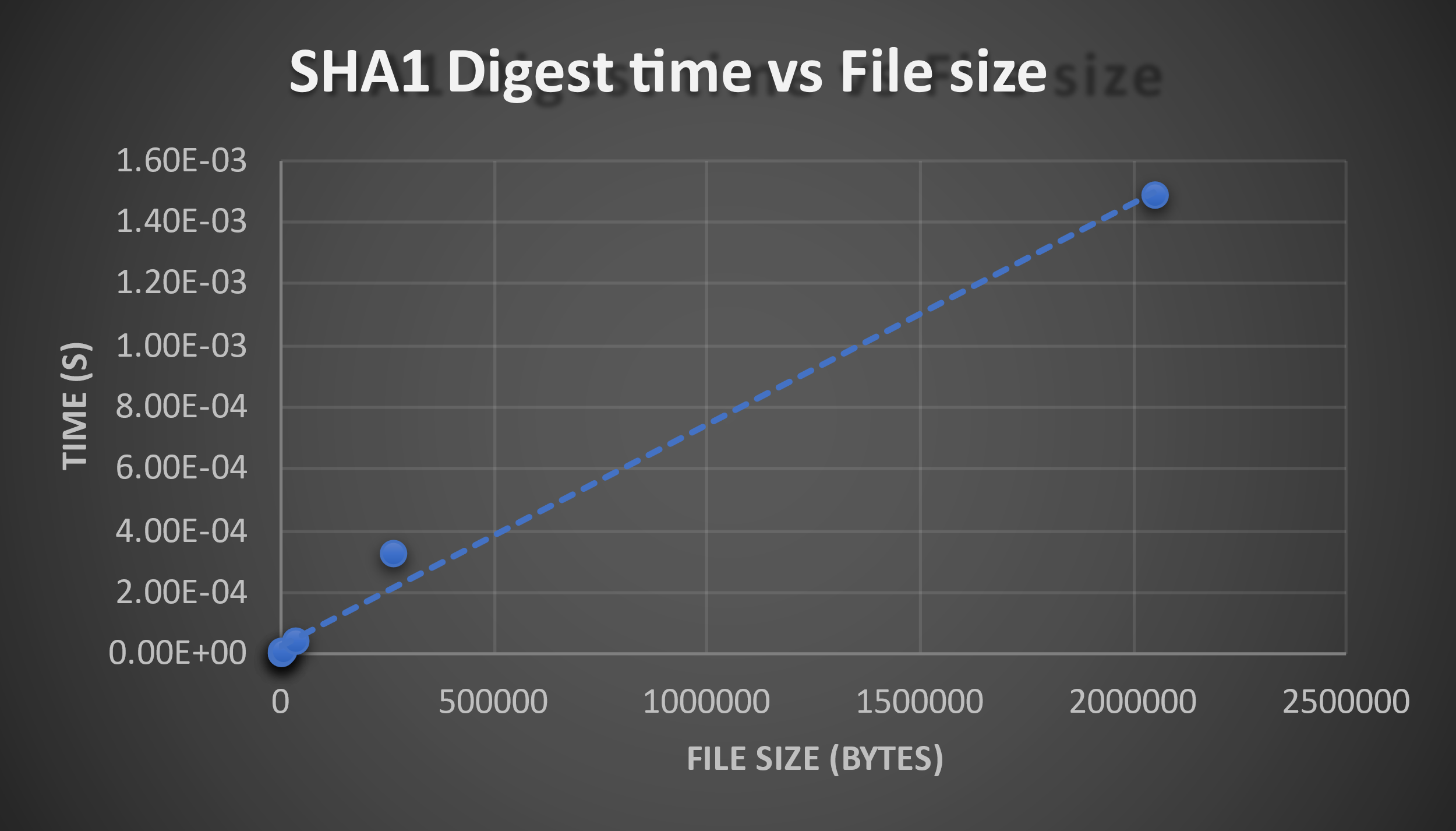
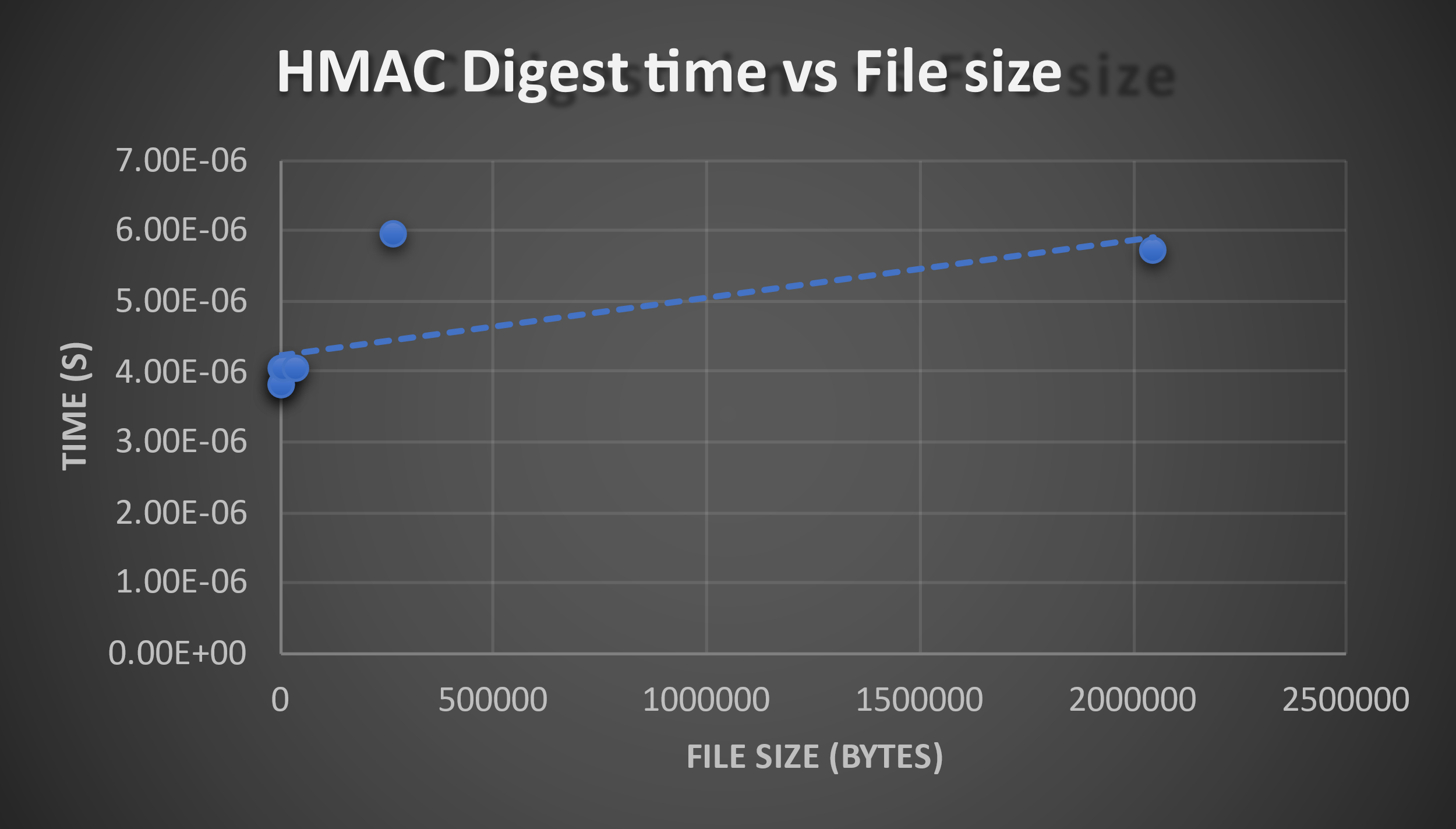
Decryption time is slightly faster than encryption for smaller file sizes and longer for larger



Encryption and Decryption times faster than RSA, DES but slower than HASH algorithms

SHA1 and HMAC both have a generally positive trend with time vs file size

It's worth noting that SHA1 shapes better to a linear trend however HMAC is seemingly more random.

**Hash Algorithm Comparison (SHA1 & HMAC):**

SHA1 and HMAC both have a generally positive trend with time vs file size.

It's worth noting that SHA1 shapes better to a linear trend however HMAC is seemingly more random.

HMAC appears to be the fastest algorithm of them all followed by SHA-1. Overall HASH algorithms outperformed counterparts.

**RSA Algorithm:**

There doesn’t seem to be a trend for RSA encryption.

However, one thing that is interesting is that the times are all in a similar range for all the file sizes despite some being much larger than others.

For RSA decryption there doesn’t seem to be any pattern.

Times look random for File sizes.

RSA looks like the slowest, having a comparable time to others despite having only being tested with small files.