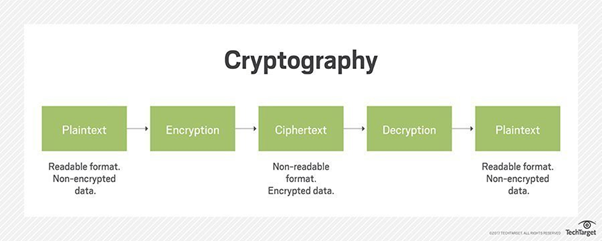
**Introduction**

**What is Cryptography and Why do we need it?**

Cryptography is a method of protecting information and communications through the use of codes, so that only those for whom the information is intended can read and process it. The prefix “crypt-” means “hidden” or “vault” — and the suffix “-graphy” stands for “writing”. Cryptography provides a way of secure communication in the presence of malicious third-parties—known as adversaries.

It uses techniques derived from mathematical concepts and a set of rule-based calculations called algorithms, to transform messages in ways that are hard to decipher. Encryption uses an algorithm and a key to transform an input (plaintext) into an encrypted output (ciphertext).

Algorithms are considered secure if an attacker cannot determine any properties of the plaintext or key, given the ciphertext. An attacker should not be able to determine anything about a key given a large number of plaintext/ciphertext combinations which used the key.



**Cryptography in Everyday life:**

### Authentication/Digital Signatures

### Time Stamping

### Electronic Money

### Secure Network Communications

### Anonymous Remailers

### Disk Encryption

**Encryption:**

Encryption is the process of converting human-readable plaintext to incomprehensible text, also known as ciphertext. In simpler terms, encryption takes readable data and alters it so that it appears random. Encryption requires the use of a cryptographic key: a set of mathematical values.

**Decryption:**

The conversion of encrypted data into its original form is called decryption. It is generally a reverse process of encryption. It decodes the encrypted information so that an authorized user can only decrypt the data because decryption requires a secret key or password.

**Types of encryption:**

The two main kinds of encryption are **symmetric encryption** and **asymmetric encryption**. Asymmetric encryption is also known as public key encryption.

* In symmetric encryption, there is only one key, and all communicating parties use the same key for both encryption and decryption.
* In asymmetric or public key encryption, there are two keys: one key is used for encryption, and a different key is used for decryption. The decryption key is kept private, while the encryption key is shared publicly for anyone to use. Asymmetric encryption is a foundational technology for Transport Layer Security (TLS).

**Why data encryption?**

1. **Privacy:** Encryption ensures that no one can read communications or data at rest except the intended recipient or the rightful data owner. This prevents attackers, ad networks, Internet service providers, and in some cases governments from intercepting and reading sensitive data.
2. **Security:** Encryption helps prevent data breaches, whether the data is in transit or at rest. If a corporate device is lost or stolen and its hard drive is properly encrypted, the data on that device will still be secure. Similarly, encrypted communications enable the communicating parties to exchange sensitive data without leaking the data.
3. **Data integrity:** Encryption also helps prevent malicious behaviour, such as on-path attacks. When data is transmitted across networks, encryption ensures that what the recipient receives has not been tampered with on the way.
4. **Authentication:** Public key encryption, among other things, can be used to establish that a website's owner owns the private key listed in the website's TLS certificate. This allows users of the website to be sure that they are connected to the real website.
5. **Regulations:** For all the reasons stated above, many industry and government regulations require companies that handle user data to keep that data encrypted. In order to comply with these regulations, it is mandatory to implement encryption protocols even if it is not desired. Examples of regulatory and compliance standards that require encryption include HIPAA, PCI-DSS, and the GDPR.

**What is an encryption algorithm?**

An encryption algorithm is the method used to transform data into ciphertext. An algorithm will use the encryption key in order to alter the data in a predictable way, so that even though the encrypted data will appear random, it can be turned back into plaintext by using the decryption key.

Some common symmetric encryption algorithms are:

* AES
* DES
* 3DES
* Blowfish
* Twofish

Some common asymmetric encryption algorithms are:

* RSA
* Elliptic curve cryptography

**Encryption Algorithms:**

1. AES

The **AES algorithm** (also known as the **Rijndael algorithm**) is a symmetrical block cipher algorithm that takes plain text in blocks of 128 bits and converts them to ciphertext using keys of 128, 192, and 256 bits. Since the AES algorithm is considered secure, it is in the worldwide standard.

### **Working:**

The AES algorithm uses a substitution-permutation, or SP network, with multiple rounds to produce ciphertext. The number of rounds depends on the key size being used. A 128-bit key size dictates ten rounds, a 192-bit key size dictates 12 rounds, and a 258-bit key size has 14 rounds. Each of these rounds requires a round key, but since only one key is inputted into the algorithm, this key needs to be expanded to get keys for each round, including round 0.

1. DES

**Data Encryption Standard (DES)** is a block cipher algorithm that takes plain text in blocks of 64 bits and converts them to ciphertext using keys of 48 bits. It is a symmetric key algorithm, which means that the same key is used for encrypting and decrypting ​data.

**Working:**

* The process begins with the 64-bit plain text block getting handed over to an initial permutation (IP) function.
* The initial permutation (IP) is then performed on the plain text.
* Next, the initial permutation (IP) creates two halves of the permuted block, referred to as Left Plain Text (LPT) and Right Plain Text (RPT).
* Each LPT and RPT goes through 16 rounds of the encryption process.
* Finally, the LPT and RPT are rejoined, and a Final Permutation (FP) is performed on the newly combined block.
* The result of this process produces the desired 64-bit ciphertext.

1. Triple DES

**3DES** is an encryption cipher that was derived from the original Data Encryption Standard (DES).Although it’s officially known as the Triple Data Encryption Algorithm (3DEA), it is most commonly referred to as 3DES. This is because the 3DES algorithm uses the Data Encryption Standard (DES) cipher three times to encrypt its data.

1. Blowfish

**Blowfish** is a symmetric encryption algorithm, meaning that it uses the same secret key to both encrypt and decrypt messages. Blowfish is also a block cipher, meaning that it divides a message up into fixed length blocks during encryption and decryption. The block length for Blowfish is 64 bits; messages that aren't a multiple of eight bytes in size must be padded.

Blowfish is public domain, and was designed for use in performance-constrained environments such as embedded systems.

**Working:**

* Subkey Generation: This process converts the key up to 448 bit long to subkeys totaling 7168 bits.
* Data Encryption : This process involves the iteration of a simple function 16 times. Each round contains a key-dependent permutation and key and data substitution.

1. Twofish

**Twofish** is a symmetric block cipher, i.e, a single key is used for encryption and decryption. Twofish has a block size of 128 bits and accepts a key of any length up to 256 bits. Twofish is fast on both 32-bit and 8-bit CPUs (smart cards, embedded chips, etc), and in hardware. It is flexible; it can be used in network applications where keys are changed frequently and in applications where there is little or no RAM and ROM available.

**Working:**

* In each round of the Twofish algorithm, two 32-bit words serve as input into the F function.
* Each word is broken up into four bytes. Those four bytes are sent through four different key-dependent S-boxes.
* The four output bytes (the S-boxes have 8-bit input and output) are combined using a Maximum Distance Separable (MDS) matrix and combined into a 32-bit word.
* Then, the two 32-bit words are combined using a Pseudo-Hadamard Transform (PHT).
* The two 32-bit words are then added to two round subkeys.
* Finally, the two 32-bit words are [XORed](https://www.educative.io/edpresso/how-to-implement-xor-gate-in-python) with the right half of the text.

**Important Terms**

**Key**

A cryptographic key (key) is a parameter that determines the operation of a cryptographic function such as: (a) the transformation from plain text to cipher text and vice versa, (b) synchronized generation of keying material, (c) digital signature computation or validation.

**Key length**

Key length is equal to the number of bits in an encryption algorithm’s key. A short key length means poor security. However, a long key length does not necessarily mean good security. The key length determines the maximum number of combinations required to break an encryption algorithm. If a key is n bits long, then there are two to the nth power (2n) possible keys. For example, if the key is one bit long, and that one bit can either be a zero or a one, there are only two possible keys, 0 or 1. However, if the key length is 40 bits long, then there are 240 possible keys

**Block size**

[Block ciphers](https://en.wikipedia.org/wiki/Block_cipher) operate on a fixed length string of [bits](https://en.wikipedia.org/wiki/Bit). The length of this bit string is the block size. Both the input ([plaintext](https://en.wikipedia.org/wiki/Plaintext)) and output ([ciphertext](https://en.wikipedia.org/wiki/Ciphertext)) are the same length; the output cannot be shorter than the input

**Rounds**

A round is defined by each cipher and typically consists of a number of building blocks that are composed together to create a function that is run multiple times.

For example, in AES, a round consists of the operations *SubBytes, ShiftRows, MixColumns, AddRoundKey.* All these Operations happen in a single round and all these operations put together is a function and this function is executed multiple time. In AES,the number of rounds varies depending on the key length

**Plain Text**

*Plaintext* is a term used in cryptography that refers to a [message](http://www.linfo.org/message.html) before *encryption* or after *decryption*. That is, it is a message in a form that is easily readable by humans.

**Cipher Text**

Ciphertext is encrypted text transformed from [plaintext](https://searchsecurity.techtarget.com/definition/plaintext) using an [encryption](https://searchsecurity.techtarget.com/definition/encryption) algorithm. Ciphertext can't be read until it has been converted into plaintext (decrypted) with a [key](https://searchsecurity.techtarget.com/definition/key). The decryption cipher is an algorithm that transforms the ciphertext back into plaintext.

**Project Scope**

Implementation, Analysis and Comparison of Cryptographic algorithms - Encryption and Decryption of Text using AES, DES, TripleDES, Blowfish.and Twofish.

**Objectives**

* Defining the Algorithms to be Implemented
* Implementation and Execution of Algorithms
* Graphical representation of results
* Analysis and Comparison of results

**Theoretical Comparison**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Factors** | **AES** | **DES** | **3DES** | **Blowfish** | **Twofish** |
| **Created by** | Vincent Rijmen , Joan Daemen in 2001 | IBM in 1975 | IBM in 1978 | Bruce Schneier in 1993 | Counterpane Labs in 1998 |
| **Key Length** | 128,192 or 256 bits | 56 bits | 56 bits | 32 – 448 bits | 128, 192 or 256 |
| **Block Size** | 128 bits | 64 bits | 64 bits | 64 bits | 128 bits |
| **Rounds** | 10 - 128bit key  12 - 192bit key  14 - 256bit key | 16 | 48 | 16 | 16 |
| **Speed** | Fast | Slow | Very Slow | Very Fast | Fast |
| **Security** | Excellent Security | Not Secure Enough | Adequate Security | Less Secure than DES | Excellent Security |

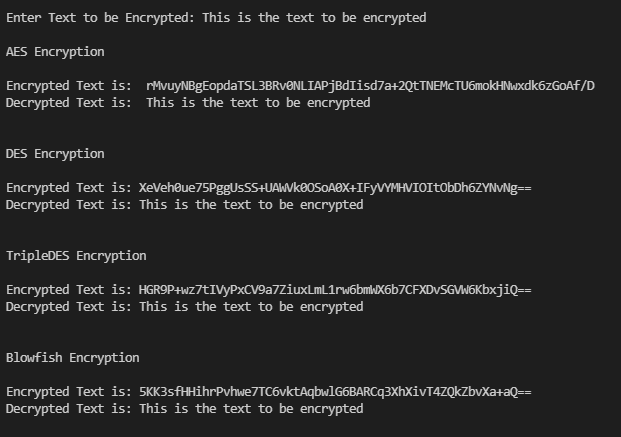
**Implementation**

**Input text:** *“This is the text to be encrypted”*

|  |  |
| --- | --- |
| **Algorithm** | **Encrypted Text** |
| AES | rMvuyNBgEopdaTSL3BRv0NLIAPjBdIisd7a+2QtTNEMcTU6mokHNwxdk6zGoAf/D |
| DES | XeVeh0ue75PggUsSS+UAWVk0OSoA0X+IFyVYMHVIOItObDh6ZYNvNg== |
| 3DES | HGR9P+wz7tIVyPxCV9a7ZiuxLmL1rw6bmWX6b7CFXDvSGVW6KbxjiQ== |
| Blowfish | 5KK3sfHHihrPvhwe7TC6vktAqbwlG6BARCq3XhXivT4ZQkZbvXa+aQ== |

**Decrypted Text:** *“This is the text to be encrypted”*

**Program Output**

******

**Twofish:**

Note: For Twofish, the length of the text should be less than 16 bytes. Hence, it will be implemented separately.

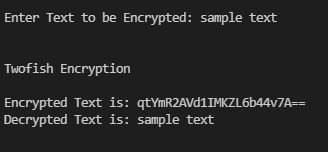
**Input text:** *“Sample text”*

|  |  |
| --- | --- |
| **Algorithm** | **Encrypted Text** |
| Twofish | qtYmR2AVd1IMKZL6b44v7A== |

**Decrypted Text:** *“Sample text”*

*This is the text to be encrypted*

**Program Output**

****

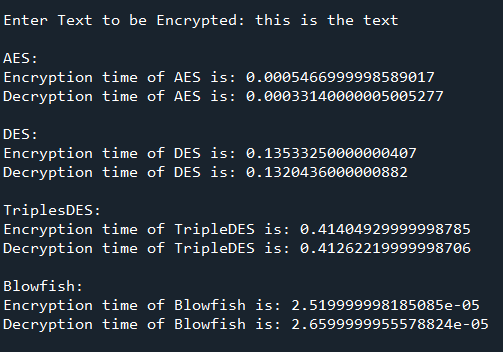
**Result Analysis**

**Input text:** *“This is the text”*

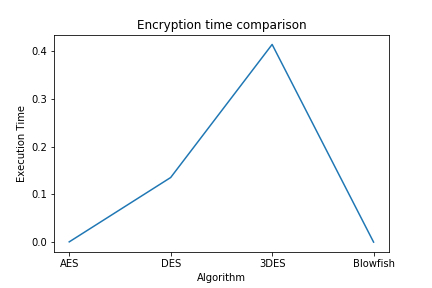
**Time Analysis:**

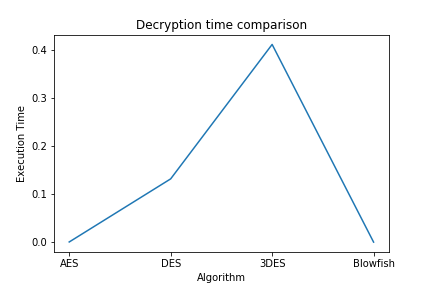
|  |  |  |
| --- | --- | --- |
| **Algorithm** | **Encryption Time (s)** | **Decryption Time (s)** |
| AES | 0.0005466999998589017 | 0.00033140000005005277 |
| DES | 0.13533250000000407 | 0.1320436000000882 |
| 3DES | 0.41404929999998785 | 0.41262219999998706 |
| Blowfish | 2.519999998185085e-05 | 2.6599999955578824e-05 |

**Program Output**

****

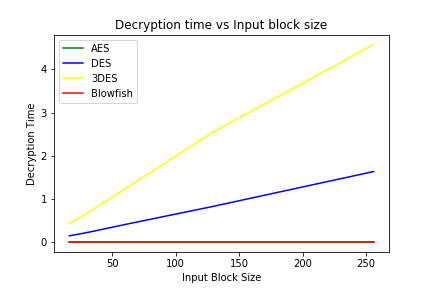
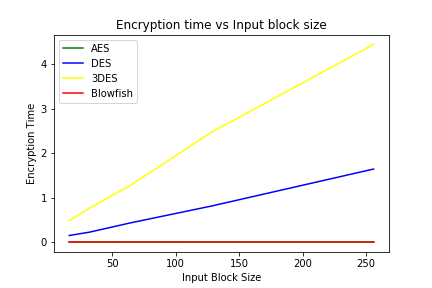
**Encryption and Decryption time comparisons:**





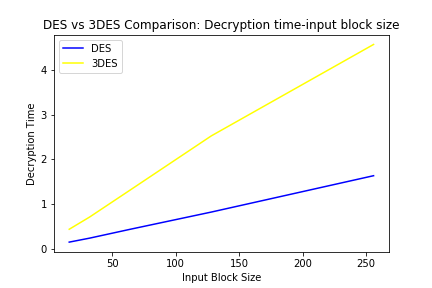
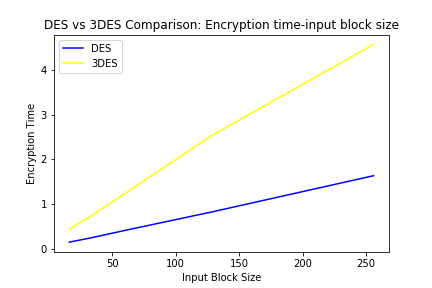
**Inference:** The execution time for encryption and decryption for AES and Blowfish is similar, while the time for 3DES is around three times that of DES. Blowfish is slightly faster than AES.

**Encryption time and Decryption time vs Input size:**



**Inference:** The encryption time and decryption time is proportional to the input block size. As the input size increases, the encryption time and decryption time also increases. The execution time in 3DES is around three times that of DES, similar to the previous graphs. As AES and Blowfish have similar execution times for encryption and decryption, their line graph is overlapping. For AES and Blowfish, the increase in execution time with increase in input block size is very minimal compared to DES and 3DES, hence it is not visible in the graph.

**3DES vs DES:**

****

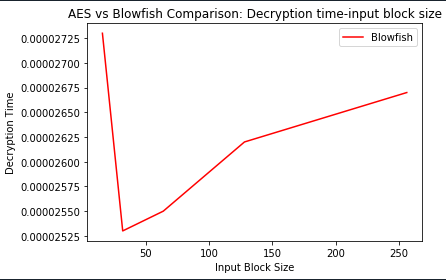
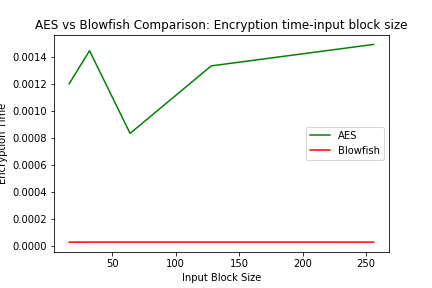
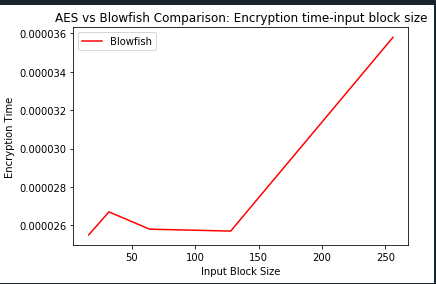
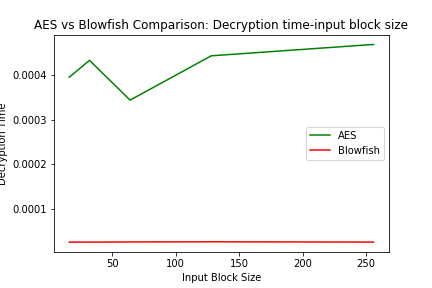
**Inference:** 3DES takes around 3 times that of DES.

For example: for input size 256, DES encryption time - 1.6445564000000559

3DES encryption time - 4.448912500000006

So, we can see that the encryption time for 3DES is approximately three times that of DES.

**AES vs Blowfish:**

****

**Inference:** The graphs clearly show that the encryption and decryption times for AES are greater than that of Blowfish. AES becomes slightly faster towards a block size of 64, while Blowfish becomes slightly faster towards a block size of 32. For both, the execution time increases with input size.

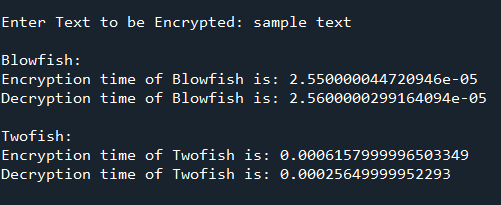
**Twofish:**

**Input text:** *“sample text”*

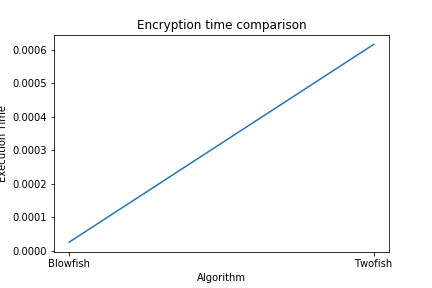
**Time Analysis:**

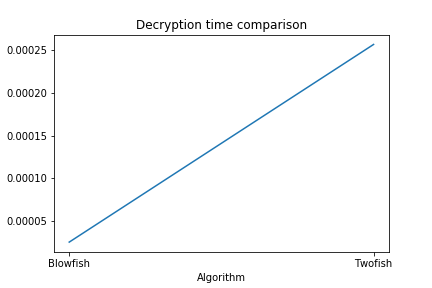
|  |  |  |
| --- | --- | --- |
| **Algorithm** | **Encryption Time (s)** | **Decryption Time (s)** |
| Twofish | 0.0006157999996503349 | 0.00025649999952293 |

**Program Output**

****

**Twofish vs Blowfish:**

****

****

**Inference:** As seen in the graph above, Blowfish is faster than Twofish in both encryption and decryption. However, Twofish is preferred because it has better security, with only minimal extra time taken for encryption and decryption.

**Conclusion**

The algorithms used were defined and implemented.

Five different cryptography algorithms for encryption and decryption of an input text were - AES, DES, 3DES, Blowfish and Twofish.

Graphical representation and result analysis was performed for each algorithm. The results of the analysis are:

* Order of execution time was - Blowfish, Twofish, AES, DES and 3DES.
* 3DES was three times slower than DES.
* Blowfish was faster than AES.
* Twofish was slightly slower than Blowfish, but faster than AES.
* Twofish only accepts input strings less than length of 16 bytes.