|  |
| --- |
|  |
| AES, Twofish - Which Symmetric-key encryption algorithm is preferable in terms of security and which in performance? |
| Computer Science |
|  |
|  |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |
| **Word count: 3924** |

**Abstract:**

This investigation explores “AES, Twofish - Which Symmetric key encryption algorithm is preferable in terms of security and which in performance?”

The importance of encryption in networking and computer systems is increasing substantially due to which many encryption algorithms have been made. The essay explores the types of encryption, their security issues as well as the factors that need to be considered for an algorithm to be secure and efficient. Although different encryption algorithms are used in different businesses, companies, and networks, each one has its own advantages and disadvantages. The strength of the algorithm’s key schedule determines the security of the cipher. Therefore, to compare their security, the key schedule of both the algorithms was analyzed in depth. Following this, the efficiency of AES and Twofish was compared by an experiment where both the algorithms were run on a computer with different sizes of text file data. Furthermore, the data was graphed to measure the relationship between the size of the text file and the time taken for each algorithm in the decryption and encryption of the data. The equations were formulated for the comparison and the limitations of this experiment were discussed.

From the analysis and findings in this essay, it has been concluded that Twofish is a better encryption algorithm with respect to security as it has a stronger key schedule but AES is a preferable algorithm with respect to performance as it is much faster. As encryption is a growing research field, there are many opportunities for further investigation regarding encryption algorithms. By comparison of both aspects, there seems to be a trade-off between security and performance efficiency. However, in the future, it will be possible to optimize the algorithm and get the required security as well as efficiency.

(Word count: 293)

Contents

[Introduction: 1](#_Toc476131004)

[Important Terminology related to encryption: 2](#_Toc476131005)

[Types of encryption: 2](#_Toc476131006)

[Symmetric encryption 2](#_Toc476131007)

[Security issues regarding Symmetric Key Encryption: 3](#_Toc476131008)

[Asymmetric encryption: 3](#_Toc476131009)

[Security Issues regarding Symmetric Key Encryption: 4](#_Toc476131010)

[Other security issues in both types of Encryption 5](#_Toc476131011)

[Features that make an encryption algorithm secure: 5](#_Toc476131012)

[Features that make an encryption algorithm more efficient in terms of performance: 6](#_Toc476131013)

[Key Generation: 6](#_Toc476131014)

[Key scheduling 7](#_Toc476131015)

[The AES algorithm: 7](#_Toc476131016)

[Key expansion: The AES algorithm 8](#_Toc476131017)

[The Twofish Algorithm: 14](#_Toc476131018)

[Key Scheduling of the Twofish Algorithm: 14](#_Toc476131019)

[AES vs Twofish in terms of security: 19](#_Toc476131020)

[AES vs Twofish in terms of performance efficiency 20](#_Toc476131021)

[Experiment: 21](#_Toc476131022)

[Encryption time processed data: 22](#_Toc476131023)

[Decryption processed data: 23](#_Toc476131024)

[Graphical representation of encryption time: 24](#_Toc476131025)

[Analysis of Data: 25](#_Toc476131026)

[Limitations of this experiment: 26](#_Toc476131027)

[Conclusion: 27](#_Toc476131028)

[Appendix: 28](#_Toc476131029)

[Bibliography: 33](#_Toc476131030)

# Introduction:

The growing use of networking has changed the way we live and work due to its active use in our daily lives. The usage of any network is done so that data can be transmitted or retrieved easily and efficiently. For instance, indulging in online business transactions, social networking, surfing through the internet to read blogs etc. When we are sending private information from one terminal to another, security of data is a major concern. One of the ways in which we can ensure data security is through the process of *encryption.*

Ancient cryptography was first found in the early 1900BC in Egypt. It showed no advancement till the middle ages. During World War 2, the Germans used encryption in the enigma machine. After that was deciphered, modern day cryptography was developed with many algorithms. After reading the book “Alan Turing: The enigma.”, I was struck by how important cryptography is to computer science which led me to this investigation.

Encryption is the technique of converting electronic data (plain text) into a form that cannot be easily understood (cipher text) using mathematical algorithms.[[1]](#footnote-1) Even though there are strong encryption algorithms in our systems and across a network, any encrypted data transmitted through the internet can still be attacked. With the latest advances in technology, ciphers using keys to encrypt and decrypt are becoming easier to decipher. Furthermore, when transmitting large sizes of data across a network, we should take the efficiency into consideration as well. This brought me to my research question – ***“AES, Twofish - Which Symmetric key encryption algorithm is preferable in terms of security and which in performance?”***

# Important Terminology related to encryption:

**Key:** A key is a string of data that is fed into the algorithm for encryption. It can be numeric or alpha-numeric.

**Key size:** The measure of the key in bits is called the key size.

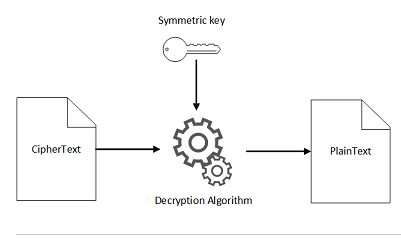
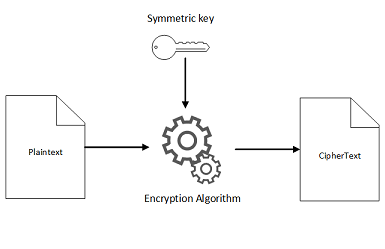
**Block size:** fixed sizes of strings encrypt Data. This fixed length varies from one algorithm to another. This fixed length is called block size.

**Round:** This is the process and amount of time taken for the encryption function to give cipher-text as its output.

# Types of encryption:

## Symmetric encryption

Symmetric key encryption is the technique of the encryption and decryption of data with the same key. The sender of the encrypted data must send the private key to the receiving terminal so that the data can be accessed. The key must remain confidential to prevent attacks. Symmetric key encryption is used to encrypt bulk data in organizations and governments. AES is an example of a form of symmetric encryption that is used officially by the American government. This is depicted in Figure 1.1

[[2]](#footnote-2)

**Figure 1.1**

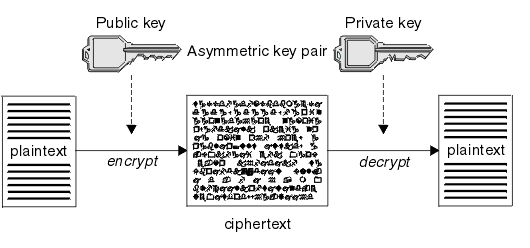
### Security issues regarding Symmetric Key Encryption:

**Key search or brute force attacks:** In this type of attack, every possible key is used to attempt to decrypt the message. Depending on the key size (n), there are 2n possible keys. Brute force attacks are not efficient and in some cases, not feasible. If this type of attack is done, the attacker cannot be stopped from decrypting the message.

**System based attacks:** This occurs when the attack is done on the cryptographic system and not the algorithm itself. The attacker aims at the system in which the data must be received or is sent from. For example, in early satellite TV systems, system based attacks on VC-I video encryption were done.

## **Asymmetric encryption:**

In asymmetric or public key encryption, data is encrypted and decrypted using different keys and not the same key. When the data is encrypted, a public key is used. A public key is a key that is published and available to everyone. For decryption, a private key is used which should be kept secure. Many online transactions use this type of encryption for security. This can be seen in figure 1.2

[[3]](#footnote-3)

**Figure 1.2**

To make sure that the public key is authentic, digital signatures can be used. A digital signature can be created through a software which creates a hash of the electronic data that must be signed. The hash is encrypted with the use of the user’s private key. Even if a single bit of data is changed, a different hash value will be returned. Thus, even a small change will invalidate the digital signature. RSA is the most widely used asymmetric encryption algorithm.

### Security Issues regarding Symmetric Key Encryption:

***Man in the middle attack (MIM):*** In this method of attack, the attacker intercepts the request of the host and sends his public key instead of the required recipients. This way, whatever the host sends, the attacker can read it and decrypt it.

***Birthday Attack –*** This technique is similar to brute force, but it is used to counter the hash function in asymmetric encryption. For a hash function that gives 64-bit hash values, the number of possible values will be 264. The hash value can be called broken if the attacker can find two different inputs that give the one hash value.

***Brute force, cryptanalysis and system based attacks.***

### Other security issues in both types of Encryption

***Fault analysis Attacks* –** The attacker studies the output of the useful information of the cryptosystem (A cryptosystem consists of key generation, encryption, and decryption algorithms) after errors are deliberately induced so that their internal states can be revealed.

***Side Channel Attack (SCA)*** – The hardware and physical implementation of the system is exploited in this attack. The exploitation of the weakness of the hardware/physical implementation of the system is done for this attack.

***Power Analysis Attacks –*** Like the name suggests, the attacker studies the nature of the computations of the processor from the power consumed. For example, if more power is consumed, the key may be longer.

***Timing Attacks –*** This method of attack exploits the time taken by the algorithm to encrypt the data. The attacker can analyze the details of processor by measuring the timing. [[4]](#footnote-4)

### Features that make an encryption algorithm secure:

***Larger key length*:** The encryption’s strength is directly related to the probability of attaining the key to decrypt it. If an algorithm uses larger key sizes (measured in bits), there are more possible keys to be used to decrypt it. Therefore, encryption strength is directly related to the size of the keys. Longer keys provide stronger encryption.

**Key management:** This refers to the methods and tasks used in protecting, backing up, storing, and organizing the encryption keys. If this is not done, the key may end up in the wrong hands.

**Number of keys:** The more the number of keys, depending on the type of encryption, the more secure the algorithm is. For symmetric key encryption, communication is done between n number of people. The number of keys needed for secure transmission of information is n(n-1)/2 keys. For asymmetric encryption, it is 2n keys needed for secure connection.

**Type of encryption:** The security of the algorithm depends on what type of encryption is used as well. There are some cases where asymmetric encryption is more secure and vice versa.

### Features that make an encryption algorithm more efficient in terms of performance:

**Time efficiency:** This refers to the amount of time taken for the algorithm to execute. An algorithm is said to be more efficient if the amount of time taken is less as processor time is saved.

**Space efficiency:** This refers to the amount of memory needed for the algorithm to execute. An algorithm is said to be more efficient if the space required for execution is less.

**The structure of the algorithm:** The Big O notation is often used to determine the complexity of an algorithm’s structure. Big O notation is used to determine the performance of an algorithm and describes the worst-case scenario of the algorithm.

### Key Generation:

For keys, Integers are used in modern cryptographic systems. In some cases, the Random or pseudorandom number generator (RNG or PRNGs) can generate the keys. PRNGs are used preferably as the data is seeded generally by system entropy (The randomness which is done by the operating system). This makes the key more difficult to deduce. For public key encryption, the private key is generally computed before or from the public key. Generally, the public key is created from the private key as they must be strongly related. But it is never done vice versa. If the public key was computed before, the attacker could also do the same, and it may not be secure. Separate key generation algorithms are made with the use of random generators.

### Key scheduling

The key scheduling algorithm is one of the most important parts of a block cipher algorithm. The strength of this algorithm determines the security of the cipher. Therefore, I have explained the key schedule for each of the algorithms. I will be comparing the security based on which key schedule is better. A key schedule is an algorithm which generates round keys from the cipher key. It helps in resisting attacks in which a part of the cipher key is known or can be interpreted by a cryptanalyst. The safest key schedules should create round keys in such a way that they seem to be random or mutually independent of each other. For a secure algorithm, the key schedule is so strong that knowing one round key will not help in finding the other round keys or even the cipher key. In the following, I will attempt to compare the AES and Twofish algorithms.

### The AES algorithm:

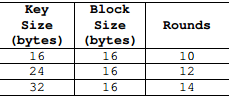
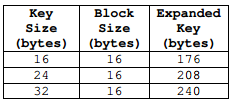
AES is the most widely used symmetric encryption algorithm and it was chosen to replace DES which became outdated and insecure.

In 2001, AES was approved as the Federal Information Processing Standards Publication by the NIST. Since then it has become the US government standard for encrypting and decrypting data. It was first called Rijndael. The AES uses a structure of substitution-permutation not a Feistel cipher Key expansion (This is a symmetric structure in which block ciphers are constructed) is the core of the algorithm security.

### Key expansion: The AES algorithm

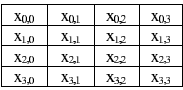
The AES algorithm is symmetric which means that all the bytes are treated similarly so that they undergo the same operations. It is parallel which means that the operations on each byte can be executed parallelly. It has modularity which accounts for creating the rounds that consists of the iterations

In AES, the key schedule comprises two parts: *key expansion* and *key selection.* In key expansion, an input key is taken by the algorithm of 4 \* Nk bytes where Nk is at either 4, 6 or 8. In each round, the actual key is expanded. The final output of this is an expanded key which is 4 \* Nb \* (Nr+1) bytes. Here, Nb has a value of 4 and Nr refers to the number of rounds available in the algorithm. The key expansion is independent of the value of the plaintext or the cipher-text. AES is an iterative block cipher meaning that the same operations are performed on a fixed number of bytes. The number of rounds of the algorithm depends on the key size. A round refers to one iteration of the operations. For each round, the AES generates an expanded key from the cipher key. Figure 1.3 shows the number of rounds based on the block size and key size in bytes along with the size of the expanded key. The expanded key is based on the block and key size.

 [[5]](#footnote-5)

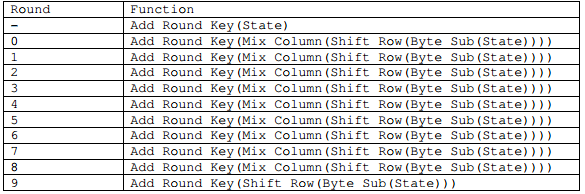
**Figure 1.3**

For example, if a 128-bit (or 16 byte) key is input into the algorithm, It will generate 10 keys of 128 bits. The key is divided into 4 x 4 arrays which are 128 bits each as shown below:

[[6]](#footnote-6)

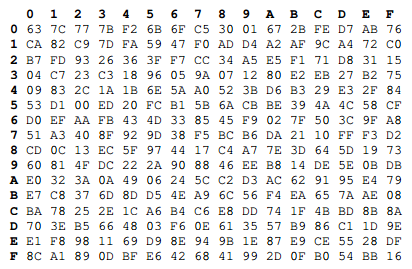
**Figure 1.4**

Each of these is processed in 10 rounds in a recursive pattern. In the diagram below, the Add Round Key function is the round function which expands the Key that is input.

[[7]](#footnote-7)

**Figure 1.4**

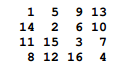
In each of the “Add Round” functions, each of the 16 bytes pass through the XOR gate with the expanded key. This is done so that the same bytes for the expanded key are not used and a pattern would not be visible for any cryptanalyst to use the expanded key to find the input key. After all the rounds are completed, the code is then generated by replacing the values of the state with the following values.

[[8]](#footnote-8)

**Figure 1.5**

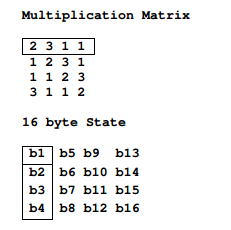
After this, each byte is substituted in an S (Substitution)-box. Substitution refers to the disorientation of bits which are performed by nonlinear transformations. This is crucial in modern encryption algorithms and helps in preventing differential cryptanalysis. The state is then arranged in a matrix and then there is a function that shifts each row circularly. The function moves each byte by one place. So, if a byte was in the 2nd position, it would go to the 3rd position after operation. The matrix is formed vertically but the shift of the row is performed horizontally. For the bytes 1 to 16, the shift row function is done as shown below.

Before: After:

  [[9]](#footnote-9)

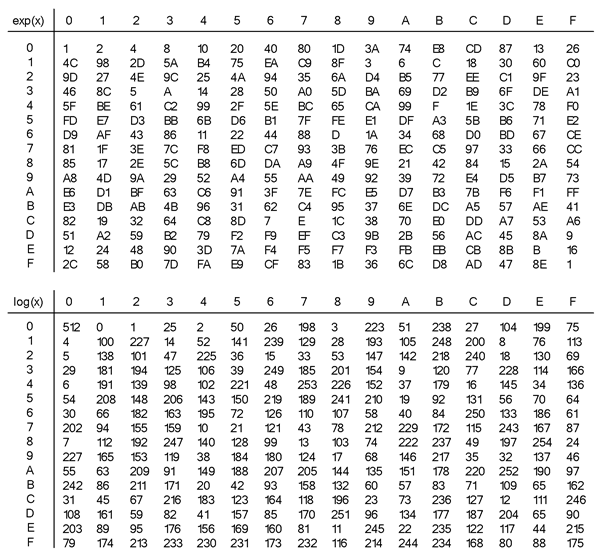
After this step is done, the function to mix the columns of the matrix is executed. The two steps to this are: Matrix multiplication and the Galois field multiplication. Matrix multiplication is performed 4 bytes at a time (a column).

A total of 16 multiplications is done as each value of the column is multiplied with all the values in the matrix. The final result will then again be passed through the XOR (exclusive or) operation to provide 4 bytes for the next state.

[[10]](#footnote-10)

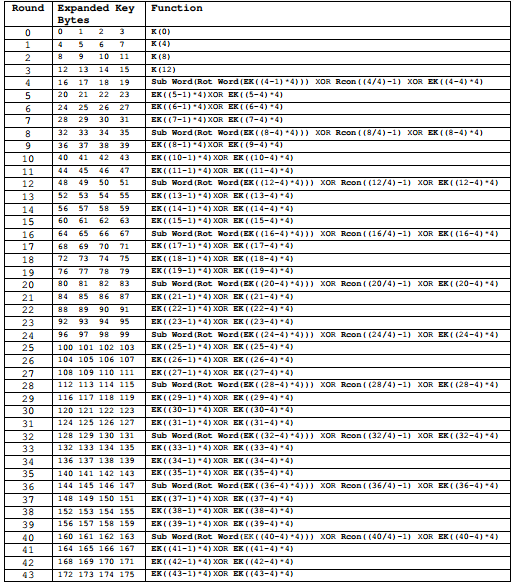
It is then multiplied with the elements of the Galois Field function GF (28). The Galois field is a finite field, in which there exists finitely many elements.

The tables below illustrate the Galois Field:

[[11]](#footnote-11)

***Figure 1.6***

The summary of the 128 but key expansion is shown below:

[[12]](#footnote-12)

**Figure 1.7**

The functions (besides the first 3 rounds) take the output of the previous functions as input. Therefore, it is recursive.

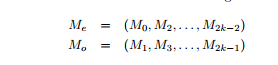
The security level of this algorithm can be increased if it is executed many times with random and different sub-keys. For the decryption of the plain text, all these steps are done in reverse order.

### The Twofish Algorithm:

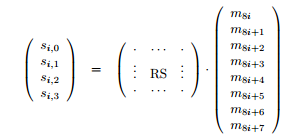
Twofish is a symmetric encryption algorithm which accepts key lengths up to 256 bits. The network structure is that of a Feistel network unlike AES. It comprises 16 rounds, a function which is consists of four key dependent substitution S-boxes called bijective F function, a pseudo-Hadamard transform, a 4 x 4 separable matrix over the GF (2^8), bitwise rotations, and a good key schedule. Twofish was designed by the computer scientists: Bruce Schneier, David Wagner, Niels Ferguson and many more.

## Key Scheduling of the Twofish Algorithm:

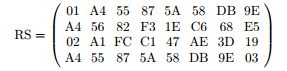
Twofish key scheduling consists of 16 rounds. In each round, Input of two 32 bit words are taken in function F, wherein F= GF (28). Let M represent the key which consists of 8k bytes, where k=N/64 and N represents the bits of the key. It is then input in two separate vectors as illustrated below:

[[13]](#footnote-13)

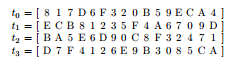
A third vector is formulated by taking the key in groups of 8 bytes then it is interpreted over the Galois Field. It is then multiplied in a 4 x 8 matrix derived by an RS code (Reed-Solomon code. Generally used for correcting errors or erasure codes in computer science). This can be shown below:

[[14]](#footnote-14)

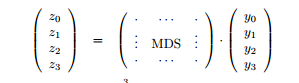
It is then interpreted as a 32-bit word. Now, for the RS matrix multiplication, a polynomial is taken over the GF field given: x8+x 6+x 3+x 2+1. The RS matrix is given by:

[[15]](#footnote-15)

Each of the 32-bit words is then split up into 4 bytes. They are sent to different key dependent substitution S-boxes used in the G function. The 4 bit S box is given by

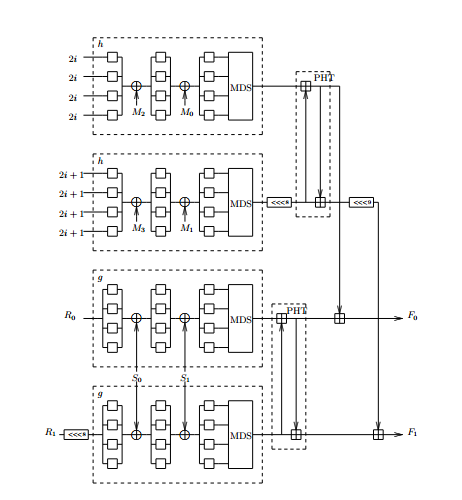
[[16]](#footnote-16)

The S-boxes have a four-bit input and output, the result is then combined using an MDS matrix. (MDS stands for Maximum Distance Separable). Then, it is assembled back into 32 bit words.

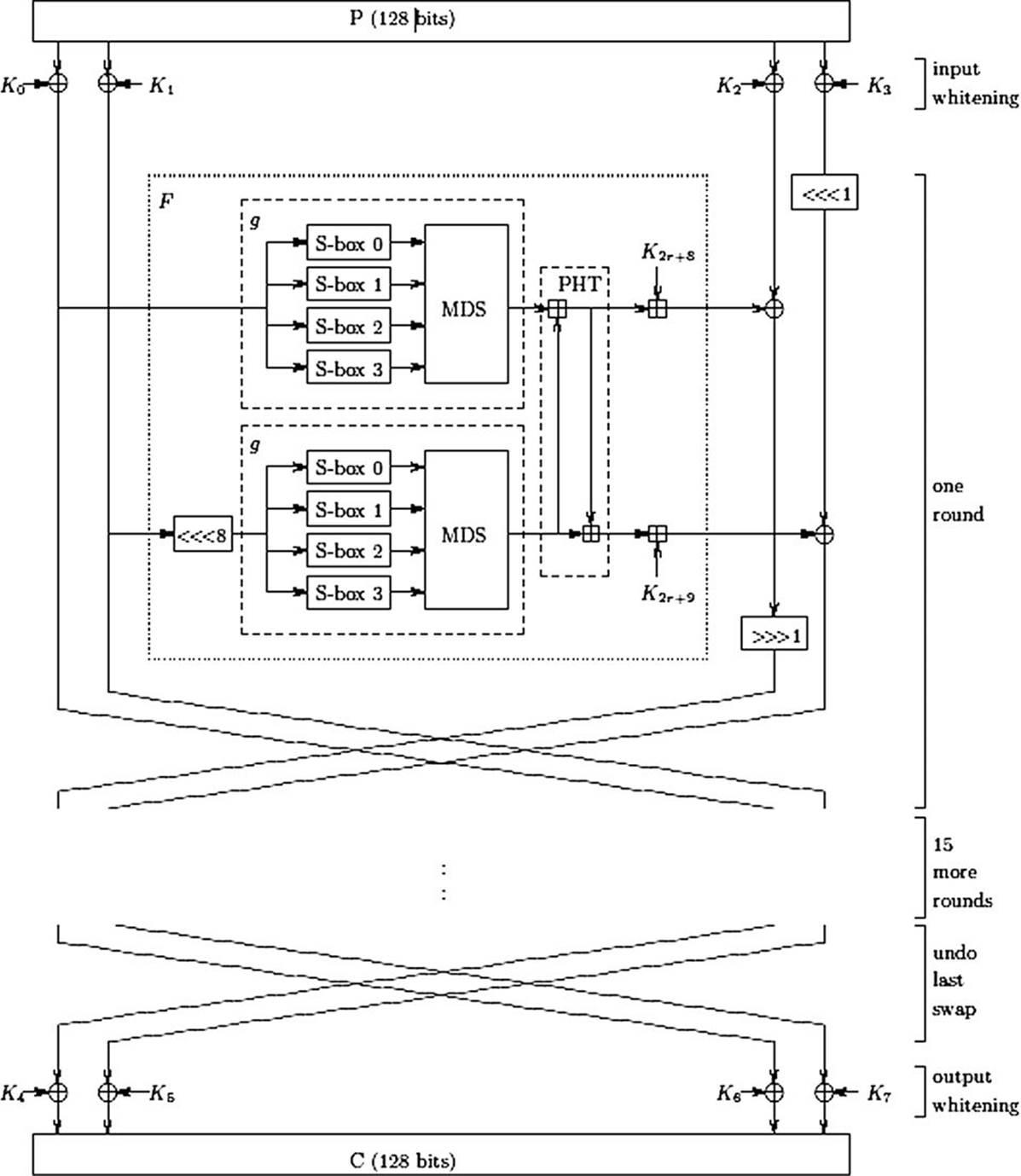
[[17]](#footnote-17)

The Pseudo-Hadamard Transform (PHT) is used to combine the 32-bit words and two keywords are added. Simultaneously, two 1-bit rotations are going on. They are then passed through the XOR operation with the right half of the plain text.

The following figure shows the view of one single round function.

[[18]](#footnote-18)

The left half and the right half are then the "prewhitening" and "postwhitening" subkeys are XORed to the text block before the first round as well as after the last round. Each step of the round function is bijective. A bijective function is a one to one corresponding function where there are two sets and each element of one set is paired exactly with only one element from the other pair. This whole process can be illustrated by the diagram shown in figure 2:

[[19]](#footnote-19)

***Figure 2***

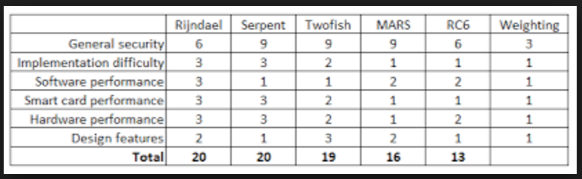
The round function in this algorithm is responsible for mixing up operations from the S-box substitution, the MDS matrix in GF (28), the 1 bit rotations, and the XORed result. This is the reason why the algorithm is difficult to theoretically attack.

The key dependent S-boxes help in making the algorithm resistant to differential cryptanalysis, linear cryptanalysis, as well as any other future attacks. To ensure the strength of the key schedule, the s-box construction rules were tested with the possible keys (128 bit). Therefore, Twofish does not have any weak keys.

The MDS matrix gives good diffusion and preserves its MDS property subsequently after the one bit rotations. The PHT provides diffusion (The property the statistical redundancy of the plain text is different from the cipher-text statistics). between the sub blocks and the key. The subkeys of each round are calculated very carefully using a similar method involving the s-box construction rules. This provides highly-efficient key mixing and can also prevent attacks based on related keys. The purpose of the one bit rotations is to break up the byte structure. If this is not done, then everything will be operated on in bytes. It becomes harder to cryptanalyze Twofish.

### AES vs Twofish in terms of security:

The table below shows the comparison of various encryption algorithms. Looking at the comparison of Rijndael (AES) and Twofish, in general security, Twofish is given a score of 9 while AES is given only 6. However, due to the other benefits of AES, it received a higher total score.

**[[20]](#footnote-20)**

The security factor is calculated by considering the minimal secure rounds in the algorithms key schedule. i.e., how many rounds are secure before we can break into the algorithm. A safety factor of 1 means that the cipher can be broken. Twofish consists of a 6-round attack with a safety factor of 2.67. AES has a safety factor which varies from 1.11/1.33/1.56. Until now, the best-known attacks can break only 6 rounds of Twofish and 9 rounds of AES.

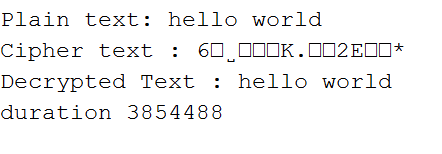
### AES vs Twofish in terms of performance efficiency

I ran the java implementation of the two algorithms, which I got from an internet source, on my computer system which has the processor specifications: Intel core i5; 4GB RAM. Operating system used: Windows 10. The output of the time taken for the different text sizes to encrypt and decrypt was recorded in Nano seconds. The graphs plotted after the experiment illustrates the efficiency difference.

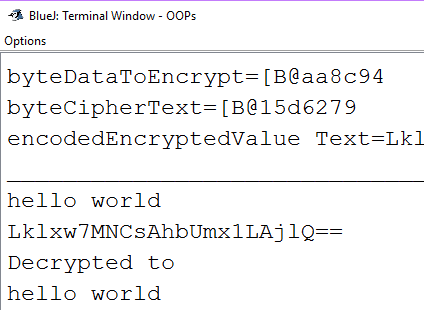
## Experiment:

The following pictures is the input and output I ran for the program (the program code is available in the appendix followed by its source).

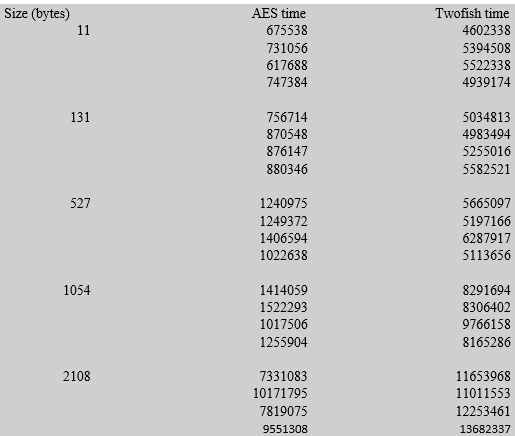
Twofish running program sample: Input and output.



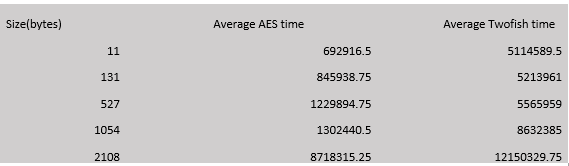
AES Running program sample: Input and output.



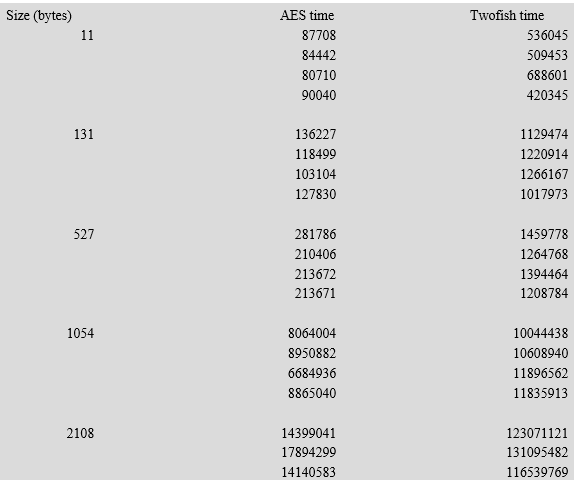
**Encryption time raw data:**



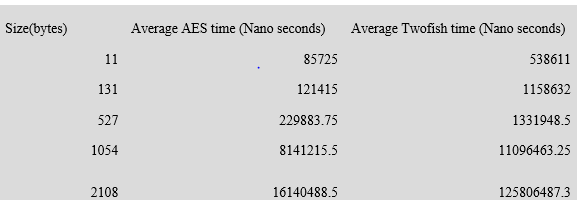
### Encryption time processed data:



**Decryption time raw data:**

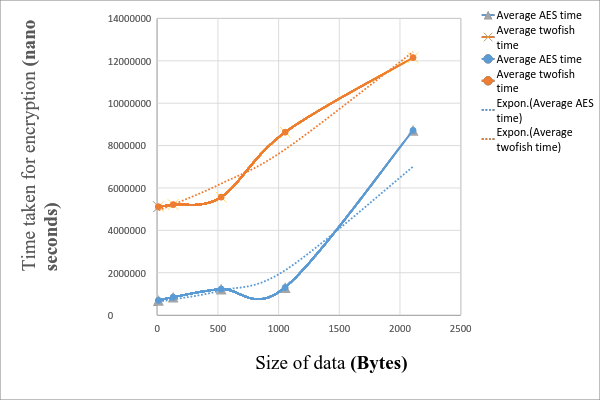


### Decryption processed data:

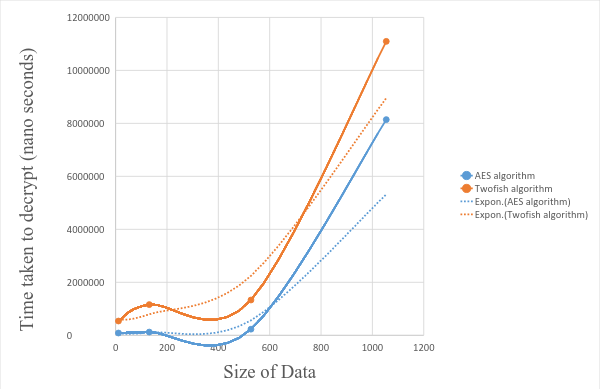


### Graphical representation of encryption time:

**Encryption time comparison**



#### Graphical representation of decryption time:



**Decryption time comparison**

# **Analysis of Data:**

Due to the trend, we can see that the relation between size and data was exponential. The equation AES encryption was: y = 636876e0.0011x and for Twofish encryption it was: y = 5E+06e0.0004x. Moreover, the equation that I got for AES decryption was: y = 59589e0.0043x and for Twofish decryption was: y = 562262e0.0026x

When considering the time taken for encryption and decryption, Twofish takes longer than AES for all data sizes.

In the graph depicting the encryption and decryption time in relation to the size of data, the curve that represents the time taken for the Twofish algorithm to execute (in orange) is above the curve which represents the time taken for the AES algorithm to execute (in blue). The execution time is essential when encrypting bulk data. For encrypting files, the AES algorithm would take less time, therefore, it would be preferable to the Twofish algorithm.

## **Limitations of this experiment:**

**Different processors** have different levels of performance for different algorithms when the algorithms are run on them. Thus, this data is not applicable to all processors.

The time taken for an algorithm to execute also depends on **the system state and RAM usage**.

**Different operating systems** manage memory and allocate processor time differently. Therefore, the operating system used can cause variances in performance. It can be different for Mac OS and Windows and Unix-based operating systems.

Different computer systems have **different register size** which in turn affects processing speed.

Twofish has a fixed number of rounds in its key schedule whereas AES does not have a fixed number of rounds as it depends on the size of the key input by the user. This leads to Twofish performing slower than AES. Due to this, AES was selected as the Advanced Encryption Standard of the US government to Twofish, which came in second place.

# **Conclusion:**

*I conclude from my findings that Twofish is better than AES with respect to security, but AES is better than Twofish with respect to efficiency*. Twofish has a more complex key schedule which makes it more secure than AES. The Twofish algorithm has one of the fewest published attacks. This factor indeed speaks of the strength of Twofish. The Twofish algorithm is unbreakable from a theoretical perspective as well. However, performance efficiency is important if large sizes of data must be encrypted quickly. It was found that the AES was much faster than Twofish. This is the primary reason AES was chosen as an Encryption standard by the US government over Twofish.

Throughout this investigation, it can be seen that there is a trade-off between security and performance. As Twofish has more rounds, it is more secure and performs less efficiently. Therefore, the following question arises: Is it possible to attain both security and performance in an encryption algorithm? With further investigation into encryption algorithms, it would be possible to minimize this trade off and make networks more secure.

# Appendix:

**Source code Twofish available at:** <https://codeload.github.com/shivani1091/twofish/zip/master>

**Source code AES available at:** <https://gist.github.com/bricef/2436364>

**AES:**

import java.security.MessageDigest;

import java.util.Arrays;

import javax.crypto.KeyGenerator;

import javax.crypto.SecretKey;

import javax.crypto.spec.SecretKeySpec;

import javax.crypto.spec.IvParameterSpec;

import javax.crypto.Cipher;

import javax.crypto.spec.IvParameterSpec;

import javax.crypto.spec.SecretKeySpec;

public class AES {

static String IV = "AAAAAAAAAAAAAAAA";

static String plaintext = "test text 123\0\0\0"; /\*Note null padding\*/

static String encryptionKey = "0123456789abcdef";

public static void main(String [] args) {

try {

System.out.println("==Java==");

System.out.println("plain: " + plaintext);

double starttime= System.nanoTime();

byte[] cipher = encrypt(plaintext, encryptionKey);

double endtime=System.nanoTime();

System.out.print("cipher: ");

System.out.println("time taken");

System.out.println(endtime-starttime);

for (int i=0; i<cipher.length; i++)

System.out.print(new Integer(cipher[i])+" ");

System.out.println("");

String decrypted = decrypt(cipher, encryptionKey);

System.out.println("decrypt: " + decrypted);

} catch (Exception e) {

e.printStackTrace(); } }

public static byte[] encrypt(String plainText, String encryptionKey) throws Exception {

Cipher cipher = Cipher.getInstance("AES/CBC/NoPadding", "SunJCE");

SecretKeySpec key = new SecretKeySpec(encryptionKey.getBytes("UTF-8"), "AES");

cipher.init(Cipher.ENCRYPT\_MODE, key,new IvParameterSpec(IV.getBytes("UTF-8")));

return cipher.doFinal(plainText.getBytes("UTF-8")); }

public static String decrypt(byte[] cipherText, String encryptionKey) throws Exception{

Cipher cipher = Cipher.getInstance("AES/CBC/NoPadding", "SunJCE");

SecretKeySpec key = new SecretKeySpec(encryptionKey.getBytes("UTF-8"), "AES");

cipher.init(Cipher.DECRYPT\_MODE, key,new IvParameterSpec(IV.getBytes("UTF-8")));

return new String(cipher.doFinal(cipherText),"UTF-8"); } }

**Twofish:**

import java.util.Scanner;

public class MyTwofish {

public static void main(String[] args) {

try (Scanner in = new Scanner(System.in)){

System.out.print("Enter the key : ");

String k = "ssssssssssssssss";

System.out.print("Enter the plain text : ");

String pt = br.readline();

byte tempKey[] = k.getBytes();

byte tempPlainText[] = pt.getBytes();

byte key[] = new byte[32];

byte plainText[] = new byte[128];

int i;

for(i=0; i<32;i++){

if(i<tempKey.length) key[i] = tempKey[i];

else key[i] = (byte)0;}

for(i=0; i<128;i++){

if(i<tempPlainText.length) plainText[i] = tempPlainText[i];

else plainText[i] = (byte)0; }

Object K = Twofish\_Algorithm.makeKey(key);

byte[] ct = Twofish\_Algorithm.blockEncrypt(plainText, 0, K);

String cipherText = new String(ct);

System.out.println("Cipher text : "+cipherText);

long StartTime=System.nanoTime();

System.out.print("Enter the key : ");

String k2 = "ssssssssssssssss";

byte tempKey2[] = k2.getBytes();

byte key2[] = new byte[32];

for(i=0; i<32;i++){

if(i<tempKey2.length) key2[i] = tempKey2[i];

else key2[i] = (byte)0; }

Object K2 = Twofish\_Algorithm.makeKey(key2);

byte[] cpt = Twofish\_Algorithm.blockDecrypt(ct, 0, K2);

String ot = new String(cpt);

System.out.println("Decrypted Text : "+ot);

long EndTime=System.nanoTime();

long duration=EndTime-StartTime;

System.out.println("duration "+duration); } } }

**Appendix word count: 171**

**Word count: 3924**

# Bibliography:

1. What is encryption, Tech target.

Available at: <http://searchsecurity.techtarget.com/definition/encryption>

Accessed on: 10/06/2016

1. How symmetric key cryptography works, amazon web services.

<http://docs.aws.amazon.com/kms/latest/developerguide/crypto_overview.html>

Accessed on: 10/06/2016

1. Cryptography, IBM knowledge center.

<http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.sec.doc/q009800_.htm>

Accessed on: 10/06/2016

1. Tutorials point

<http://www.tutorialspoint.com/>

Accessed on: 10/06/2016

1. Advanced Encryption Standard by example, ABI software development, Written By: Adam Berent

<http://www.adamberent.com/documents/AESbyExample.pdf>

Accessed on: 10/06/2016

1. Modern Cryptography: Applied Mathematics for Encryption and Information Security (2016), Kids and teens programming classes.

<http://apprize.info/security/cryptography/7.html>

Accessed on: 15/07/2016

1. How encryption works, how stuff works, Jeff Tyson.

Available at: <http://computer.howstuffworks.com/encryption.htm>

Accessed on: 15/07/2016

1. Understanding the 3 main types of encryption, Atomic object, Matt Behrens

Available at: <https://spin.atomicobject.com/2014/11/20/encryption-symmetric-asymmetric-hashing/>

Accessed on: 15/07/2016

1. Twofish, A 128 Block Cipher, Bruce Schneier

https://www.schneier.com/academic/paperfiles/paper-twofish-paper.pdf

Accessed on: 15/07/2016

1. Brief history of encryption, Technewsworld, Barry K. Shelton and Chris R. Johnson.

Available at: <http://www.technewsworld.com/story/70437.html>

Accessed on: 16/07/2016

1. Advantages and disadvantages of Asymmetric key encryption, *Steve Lander*

Available at*:* <http://techin.oureverydaylife.com/advantages-disadvantages-symmetric-key-encryption-2609.html>

Accessed on: 16/07/2016

1. What is encryption and why is it important, Scrypt blog.

<https://www.scrypt.com/blog/encryption-important/>

Accessed on: 16/07/2016

1. Why encryption is crucial to your organization, healthcareITnews, Tim Maliyi

<http://www.healthcareitnews.com/blog/why-encryption-crucial-your-organization>

Accessed on: 25/07/2016

1. The importance of strong encryption to security, Schneier on security.

<https://www.schneier.com/blog/archives/2016/02/the_importance_.html>

Accessed on: 25/07/2016

1. Performance analysis on Data encryption algorithms

Available at: <http://www.cse.wustl.edu/~jain/cse567-06/ftp/encryption_perf/>

Accessed on: 25/07/2016

1. The key to strong encryption, William Jackson.

Available at: <https://gcn.com/articles/2009/04/20/cryptography-best-practices.aspx>

Accessed on: 25/07/2016

1. Asymmetric cryptography, Teach target.

Available at: <http://searchsecurity.techtarget.com/definition/asymmetric-cryptography>

Accessed on: 12/08/2016

1. Symmetric vs Asymmetric information, ciphercloud, [Michael Higashi](https://www.ciphercloud.com/blog/author/michaelhiga/)

Available at: <https://www.ciphercloud.com/blog/cloud-information-protection-symmetric-vs-asymmetric-encryption/>

Accessed on: 12/08/2016

1. Attacks on symmetric key, Clemson.edu

<https://www.cs.clemson.edu/course/cpsc424/material/Cryptography/Attacks%20on%20Symmetric%20Key.pdf>

Accessed on: 12/08/2016

1. A review and comparative analysis of various encryption algorithms, sersc.

Available at: <http://www.sersc.org/journals/IJSIA/vol9_no4_2015/27.pdf>

Accessed on: 25/08/2016

1. Encryption key management, Techtarget.

Available at: <http://searchdatabackup.techtarget.com/definition/encryption-key-management>

Accessed on: 25/08/2016

1. Algorithm efficiency, jcsites.juniata.edu

Available at: <http://jcsites.juniata.edu/faculty/rhodes/cs2/ch12a.htm>

Accessed on: 25/08/2016

1. The design of Rijndael – The advanced encryption standard, 14 February 2002, Joan Daemen and Vincent Rijmen.

Available at: <https://books.google.com/books?id=tfjd6icCUoYC&pg=PA77&lpg=PA77&dq=key+expansion+definition&source=bl&ots=zpBeo-xts9&sig=fPbC68XEeDPuUVwf9-P5fNUgDUE&hl=en&sa=X&ved=0ahUKEwjk9bnWx4zQAhVpilQKHSv9DPAQ6AEIQDAG#v=onepage&q=key%20expansion%20definition&f=false>

Accessed on: 13/09/2016

1. Algorithm efficiency, jcsites.juniata.edu

Available at: <http://jcsites.juniata.edu/faculty/rhodes/cs2/ch12a.htm>

Accessed on: 13/09/2016

1. Algorithm efficiency, jcsites.juniata.edu

Available at: <https://pdfs.semanticscholar.org/dbb0/48954cea8033c4f51e7af535be17d9dacdf3.pdf>

Accessed on: 13/09/2016

1. Key dependent S-BOX generation in AES block cipher system, Kazys KAZLAUSKAS, Jaunius KAZLAUSKAS.

Available at: <https://pdfs.semanticscholar.org/dbb0/48954cea8033c4f51e7af535be17d9dacdf3.pdf>

Accessed on: 13/09/2016

1. How does the encryption algorithm (Rijndael) work, Acebit.

Available at: <https://www.password-depot.com/know-how/blowfish_and_rijndael.htm>

Accessed on: 20/09/2016

1. How does the encryption algorithm (Rijndael) work, Acebit.

Available at: <https://www.password-depot.com/know-how/blowfish_and_rijndael.htm>

Accessed on: 20/09/2016

1. The selection of the Advanced Encryption Standard.

Available at: <http://web.mit.edu/6.933/www/Fall2000/aes/AES_final_6933.pdf>

Accessed on: 20/09/2016

1. On the Twofish key schedule, Bruce Schneier, John Kelsey, Doug Whiting, David Wagnery, Chris Hallz, and Niels Ferguson (writers of the program itself).

Available at: <https://www.schneier.com/academic/paperfiles/paper-twofish-keysched.pdf>

Accessed on: 20/09/2016

1. On the Twofish key schedule, Bruce Schneier, John Kelsey, Doug Whiting, David Wagnery, Chris Hallz, and Niels Ferguson (writers of the program itself).

Available at: <https://www.schneier.com/academic/paperfiles/paper-twofish-keysched.pdf>

Accessed on: 20/09/2016

1. Galois field in Cryptography, Christophorus Juan Benvenuto, May 31 2012.

Available at: <https://www.math.washington.edu/~morrow/336_12/papers/juan.pdf>

Accessed on: 20/09/2016

1. Advanced Encryption standard, techopedia.

Available at: <https://www.techopedia.com/definition/1763/advanced-encryption-standard-aes>

Accessed on: 27/09/2016

1. Twofish Encryption algorithm, Horatiu Paul Stancu.

Available at: <https://users.cs.jmu.edu/abzugcx/Public/Student-Produced-Term-Projects/Cryptology-2002-SPRING/Twofish-Encryption-Algorithm-by-Horatiu-Paul-Stancu.doc>

Accessed on: 27/09/2016

1. The Twofish Encryption algorithm, Bruce Schneider, December 01 1998

Available at: <http://www.drdobbs.com/security/the-twofish-encryption-algorithm/184410744>

Accessed on: 27/09/2016

1. Implementation of the Twofish algorithm, International Journal of Computer Applications (0975 – 8887) Volume 70– No.13, May 2013

Available at: <http://research.ijcaonline.org/volume70/number13/pxc3888087.pdf>

Accessed on: 27/09/2016

1. The Twoﬁsh Team’s Final Comments on AES Selection, Bruce Schneier, John Kelsey Doug Whiting, David Wagner, Chris Hall, Niels Fergusonk Tadayoshi Kohno, Mike Stay, May 15 2000.

Available at: <http://csrc.nist.gov/archive/aes/round2/comments/20000515-bschneier.pdf>

Accessed on: 27/09/2016

1. Security performance of Rijndael, 100tb blog.

Available at: <https://www.100tb.com/blog/security-performance-serpent-cipher-rijndael/>

Accessed on: 27/09/2016

1. AES encryption in java, Devesh Sharma.

Available at: <https://deveshsharma.info/2012/12/13/aes-encryption-in-java/>

Accessed on: 27/09/2016

1. Java File Encryption Decryption using AES Password Based Encryption (PBE)

Available at: <http://javapapers.com/java/java-file-encryption-decryption-using-aes-password-based-encryption-pbe/>

Accessed on: 14/10/2016

1. Source code Twofish in Java, GitHub, Shivani.

Available at: <https://codeload.github.com/shivani1091/twofish/zip/master>

Accessed on: 21/10/2016

1. Source code AES in Java, Github

Available at: <https://gist.github.com/bricef/2436364>

Accessed on: 21/10/2016

1. http://searchsecurity.techtarget.com/definition/encryption [↑](#footnote-ref-1)
2. <http://docs.aws.amazon.com/kms/latest/developerguide/crypto_overview.html> [↑](#footnote-ref-2)
3. <http://www.ibm.com/support/knowledgecenter/SSFKSJ_8.0.0/com.ibm.mq.sec.doc/q009800_.htm> [↑](#footnote-ref-3)
4. <http://www.tutorialspoint.com/> [↑](#footnote-ref-4)
5. http://www.adamberent.com/documents/AESbyExample.pdf [↑](#footnote-ref-5)
6. http://www.adamberent.com/documents/AESbyExample.pdf [↑](#footnote-ref-6)
7. http://www.adamberent.com/documents/AESbyExample.pdf [↑](#footnote-ref-7)
8. http://www.adamberent.com/documents/AESbyExample.pdf [↑](#footnote-ref-8)
9. http://www.adamberent.com/documents/AESbyExample.pdf [↑](#footnote-ref-9)
10. http://www.adamberent.com/documents/AESbyExample.pdf [↑](#footnote-ref-10)
11. https://msdn.microsoft.com/en-us/library/cc245449.aspx [↑](#footnote-ref-11)
12. http://www.adamberent.com/documents/AESbyExample.pdf [↑](#footnote-ref-12)
13. https://www.schneier.com/academic/paperfiles/paper-twofish-paper.pdf [↑](#footnote-ref-13)
14. https://www.schneier.com/academic/paperfiles/paper-twofish-paper.pdf [↑](#footnote-ref-14)
15. https://www.schneier.com/academic/paperfiles/paper-twofish-paper.pdf [↑](#footnote-ref-15)
16. https://www.schneier.com/academic/paperfiles/paper-twofish-paper.pdf [↑](#footnote-ref-16)
17. https://www.schneier.com/academic/paperfiles/paper-twofish-paper.pdf [↑](#footnote-ref-17)
18. https://www.schneier.com/academic/paperfiles/paper-twofish-paper.pdf [↑](#footnote-ref-18)
19. http://apprize.info/security/cryptography/7.html [↑](#footnote-ref-19)
20. http://crypto.stackexchange.com/questions/11104/how-exactly-was-the-finalist-chosen-in-the-nist-aes-competition [↑](#footnote-ref-20)