National College of Ireland

Dual Image Encryption Technique Using Pixel Shuffling Algorithm and Bitwise operations for Colour images

MSc Research Project
Cloud Computing

Mukul Gopinath
Student ID: x17123739

School of Computing
National College of Ireland

Supervisor: Dr. Sachin Sharma

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Mukul Gopinath
Student ID:	x17123739
Programme:	Cloud Computing
Year:	2018
Module:	MSc Research Project
Supervisor:	Dr. Sachin Sharma
Submission Due Date:	20/12/2018
Project Title:	Dual Image Encryption Technique Using Pixel Shuffling Algorithm and Bitwise operations for Colour images
Word Count:	6076
Page Count:	17

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	18th December 2018

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Dual Image Encryption Technique Using Pixel Shuffling Algorithm and Bitwise operations for Colour images

Mukul Gopinath
x17123739

Abstract

The rapid adoption of Cloud in various domains like Education, Science, Engineering, Marketing and Business, face issues when it comes to Medical and Healthcare industry adopting the Cloud. The Compliance and Security of Medical images and Confidential images needs to be secured using suitable approaches so that the image confidentiality is not sacrificed at any instance. The Image encryption aims at achieving ideal NPCR and Entropy values when comparing with the original image, the encryption should be loss-less and identical to original image. The approach towards securing the image can be preserved through modified Fisher Yates Pixel-shuffling algorithm which distorts the structure of the image and increasing the randomness using Binary Transformation further. The prototype was evaluated and the average Entropy was observed to be 7.76 and NPCR was 100% which produces better results in terms of NPCR and compartitively similar for Entropy to that of the other methods.
Keywords: NPCR (Number of Pixel Change Rate), RGB (Red-Blue-Green).

1 Introduction

Data and Technology have evolved rapidly in the recent years, which has led to the multifold data explosion. The challenges apart from performance and latency include Security and Privacy. An approach towards ensuring Security and Privacy, started with the implementation of Principle of Least Privilege, Firewalls and stronger passwords. In recent years, there are many consumers who depend on cloud platform by utilizing the SaaS cloud offerings of the providers. CSPM (Cloud Security and Privacy Model) emphasizes on the five-layer security - Access Control and Privacy Management, Data Privacy and loss, Network Security, Cloud Infrastructure Security, and Physical and Environment Security. CSPM helps the users to adopt suitable countermeasures to Intrusion Detection System and Intrusion Prevention System. Various attacks that might occur are Insider attack, Session Hijacking and Cryptographic attacks (El Makkaoui et al.; 2016).

It is also noted that not all consumers demand same level of security and privacy, as the level of confidentiality differs between the consumers. Multi-tenant architecture implementation of Cloud offerings has many advantages like replication of client settings and setting up environment for multiple consumers, from the provider's point of view. The challenges that arise due to the lack of due diligence may result in the Data breach or data loss of other clients. To counter this, the need for Data Encryption was introduced so that,

the data would be meaningless even if the access is compromised, or at least to internally secure data. The evolution of Encryption algorithms has been around in the past decade and they have been continuously improving in multi-facets. In the recent years, the need to secure Data other than textual format is emphasized, Image stenography is an approach towards distortion of images which secures the images by data hiding concept (Kaur and Kaur; 2015)

The layers at the Data Level may be classified into Storage Layer, Management Layer, Application Interface Layer, Access Layer. Risk of Transmission is due to the network related attacks that may result in interception of network communication. Data storage risk is determined by the provider's SLA and privacy policies which determine the confidentiality of the consumer's data. Cloud Terminal risk might occur due to the loopholes in technological advancements which result in terminal infections. In order to counter the issues, there is a need for standardization, laws and regulations (Zhe et al.; 2017).

Can a pixel-shuffling encryption algorithm increase the security of confidential and medical images?

The Cloud has attracted the Health industry towards the adoption of the suitable Cloud offerings in order to benefit one another. As the health industry, unlike most consumers of cloud, demand a higher level of confidentiality and privacy as a part of the Compliance. This limits the Cloud domain to be unadaptable for the Health industry due to the lack of standardization. The health-care environment aims at exploiting the potential of the Cloud to easily store, process and retrieve the data from the Cloud seamlessly without any security hazard or data loss (Daman et al.; 2016).

Figure 1 below presents the Strength, Weakness, Opportunity and Threat analysis of Health-care in Cloud Computing, which represents the gaps and match for Cloud computing to be adaptable for Health-care.

Strength	Weakness
Cost Effectiveness, Innovate, Flexible, Cost Reduction, Energy Saving, Better Control, Ability to expand	High Speed bandwidth, Integration with local software, Lack of control and QoS, Legal, Ethical and Privacy Laws
Opportunity	**Threat**
Latest technology, Minimum Investment, Adaptive to future requirement	Data Security, Loss of Connectivity, Integration with another platform is hard, Lack of Standards

Figure 1: SWOT analysis of Cloud Computing

The latter part of the paper is orchestrated as: Section 2 which emphasizes the past solutions by analyzing the algorithms and methodologies adopted. Section 3 aims to provide a detailed overview of methodology. Section 4 describes the implementation of the algorithm. Section 5 focuses on the evaluation of the research and to contrast the results quantitatively with reference to the past work. Section 6 is for the conclusion and future work for the research work implemented.

2 Related Work

2.1 Introduction

This section is a brief summarization of the past work that has been carried out with relation to encryption and security in the domain of Computing. The entire section is divided into three other subsections. Subsection 2 elaborates the need for security and confidentiality in Cloud Computing, followed by Subsection 3 which focuses on the various encryption approaches and critically analyses each approach. Subsection 4 helps to understand on how to quantitatively analyse the encryption methodology against the efforts of implementation. Subsection 5 concludes the section.

2.2 Importance of security, confidentiality in Cloud Computing

Cloud Computing has gained importance in the decade and thus evolving the Cloud domain to suit all needs of the users have become the main concern. As per the IDC Enterprise Panel, Security stands as the most significant issue in Cloud Computing. The various threats that may potentially harm the data in Cloud, which results in DoS, data breach and loss. The two domains in Security includes Cloud Security Control and Security Compliance Model. This paper proposes to provide an application which help users to be aware of the providers who support Compliance models (Nayak et al.; 2017).

Traditional cryptographic algorithms like RSA (Rivest, Shamir and Adleman), DES (Data Encryption Standard), AES (Advanced Encryption Scheme) based on Number Theory and demand large computational power. The importance of encryption of data to ensure data confidentiality. The proposed work aims to evaluate various security breach and to address them with suitable techniques (Hendre and Joshi; 2015).

2.3 Encryption Algorithms

2.3.1 Chaotic Logistic Map

Double Chaotic Logistic Map is better when compared to a One Step Logistic Map which shows a detailed approach to encrypting and decrypting the images, where the correlation between the original image and proposed approach is positive which means that the retrieval of the image during decryption results in an image that is same as the original one whereas the One step Logistic map results in a negative correlation which results in an image which decrypts into a closer to original image with data loss or loss of clarity compared to original image. This approach suits different sized images also which is better than having encryption dependent on a size image which suffers data loss due to compression (Safi and Maghari; 2017).

Multiple stage logistic map encryption uses three functions to encrypt the sample image which are Read, Pixelate, Change, the same series of process is done in reverse order to decrypt the image. The process is series and requires one function to be completed before the next one can be initiated. There are no insights about the decryption security. Although the proposed approach can withstand differential and statistical attacks (Brindha; 2017).

Lorenz equation and Henon map is said to generate high entropy and low correlation between neighboring pixels which are supportive to the process of Encryption. The NPCR and UACI values suggest that the encrypted image is close to a completely random

image. The histogram at various levels of encryption suggest that the chosen approach is far better. The process of Confusion and Diffusion is adopted in this scenario and the resulting Entropy value is higher on the benchmarks (Murugan and Gounder; 2016).

The Image scrambling techniques and improved Logistic Mapping help the encryption algorithm to be more secure and efficient. The results of this approach is ideal as the Entropy value is close to 8 and the Gray value is fairly uniform which enhances the efficacy of the encryption algorithm (Bing and Die; 2017).

Stream-based and Block-based encryption is evaluated against one another where the pixel correlation of the Encrypted image and the Original image is studied. The image data is read using a technique called as 'Spiral Wave Scan'. The resulting Histogram is not as varied when compared to the original image also the Adjacent pixel correlation is higher, thus resulting in the encryption technique to be less efficient compared to the other techniques (Singar et al.; 2017).

2.3.2 Confusion and Diffusion techniques

The use of Two step iterated logistic map implements confusion and diffusion, which is designed for Gray-scale images. The NPCR and UACI values estimate that the proposed algorithm doesn't get compromised to Differential attacks. The encryption technique is superior compared to others as this performs in a single scan instead of iterative reads (Sharma and Bhargava; 2016).

Pixel Transposition is utilized to encrypt the image using the RGB histogram of the original image. The compression ratio is very less, hence the time to transmit is reduced. The compression ratio is around 65% which enables quicker transmission of images to the receiver. This approach also depicts a good Avalanche Effect which is desirable for any Cryptographic algorithm. The drawback of the proposed approach is that it results in improper decryption due to bit loss of data from logarithmic operation (Ranjan et al.; 2017).

Coupled Map Lattice is utilized for the process of Diffusion in order to encrypt the image. The experimental results suggest that the NPCR and UACI values are appropriate for the encrypted image to be random completely and not resemble the original image (Oravec et al.; 2018).

The asymmetric encryption scheme using Arnold Transform, Binary Matrix and Complex function is simple and efficient for the effort that is adopted during encryption. The image is encrypted using the Arnold transform which divides into 'n' QR codes which are processed using the Binary Matrix, then image is processed and manipulated using a complex function known as Fresnel Domain algorithm (Kumar and Nishchal; 2018).

The use of DWT (Discrete Wavelet Transform) where the image is sub-sampled then it is passed through the Quantizer and then through the Selective Encryption where the image data is sampled and encrypted, this is then passed through the Entropy coder which produces the Encrypted image. The decryption follows the similar process in reverse manner to obtain the decrypted image. This method achieves speed and efficiency. This methodology supports JPEG2000 standard images also. The major drawback of this approach is that the encryption technique is only suited for X-Ray and scans and thus suffers limitation of encryption any image (Abdel-Nabi and Al-Haj; 2017).

The image encryption process by using digital signal transform is where the RGB (Red, Green, Blue) values are extracted and sent as input in parallel and then DCST (Discrete Cosine Singular Transform) is used to transform the image using the Fourier

and wavelet transform on time-frequency decomposition, this is then sent through SVD (Singular Value Decomposition) which is used to diagonalize the matrix. The algorithm although follows a series of iterations and processing, the image transformed and reconstructed is higher in quality and color when compared to other algorithms (Vaish and Kumar; 2018).

A method to Compress and Encrypt simultaneously is achievable and feasible. This approach performs 8 x 8 DCT (Discrete Cosine Transform) where the input image is transformed over Four Stages. The Joint Compression consists of two steps: Generation of the secret key and Alternating transform using the secret key. The R-D (Rate Distortion) and compression ratio are closer to the Original image when compared to the other approach. The approach doesn't aim at sustaining the confidentiality aspect of the image, thus is not a suitable approach for the Medical organizations (Li and Lo; 2015).

2.3.3 Phase Encoding

Compressive sensing and Random Phase Encoding technique is an advanced approach where the encryption is done using Runge-Kutta algorithm. The results show that there is distortion in the decrypted image when the original image is compressed and retrieved in a 2:1 ratio, whereas the decrypted image looks like the original image in 4:3 ratio compression. The PSNR (Peak Signal-Noise Ratio) is higher in the decrypted images. Also, the MSE (Mean Squared Error) is comparatively higher for the correct key, which make the encryption process a visually challenging approach, leaving alone UACI and NPCR as benchmarks to evaluate the encryption (Huang and Yang; 2018).

The Hybrid approach using Arnold Cat Map, Logistic Map at each stage of the encryption process is not that efficient. The results of the NCPR and UACI are close to random image, but the process of pixel shuffling doesn't affect the image as much as the effort that is deployed in the shuffling process (Abdullah and Abdullah; 2017).

2.3.4 Magic Rectangle

Magic rectangle is a method to shuffle the image contents proposes an approach to encrypt the images using MR and reconstruct the cipher and to encrypt the image. This method is fast and reliable as the magic rectangle generates four series of ciphers and the suitable cipher should be chosen to decrypt the portion of the image. It is advanced when compared to the Stenographic method of using a cipher image to encrypt the original image. The evaluation of UACI and NPCR results is void, which doesn't standardize the test results. The encryption and decryption is done at a faster rate as the image is compressed. The encryption algorithm doesn't scale well as the time to encrypt and decrypt is directly dependent on the size of the image (Amalarethinam and Geetha; 2015).

2.3.5 Discrete Fractional Angular Transform

A hybrid approach to encrypt the image data using Discrete Fractional Angular Transform and Arnold Transform provide an ideal result. The Arnold Transform is a scrambling method, whereas Discrete Fractional Transform generates eight different bit plane transforms. The image first undergoes Bit plane transform, followed by Arnold Transform and conclusively using Bit plane transform. The decryption process follows the same set of operations in a recursive reverse order. The variance of the encrypted image is uniform for any kind of image taken as input. The image cannot be retrieved with one

parameter right (out of 'a' and 'n'). This secures the image in a better manner, but the decrypted image differs from the original image although the structure is regained, the color is distorted and is visually identifiable. The algorithm is not benchmarked for the UACI and NPCR values which are considered as standard to assess the randomness of Encrypted Image when compared to the Original Image (ZHOU et al.; 2018).

2.4 Evaluation of the Encrypted Image

An Encryption and decryption based on pixel transposition of row vector and column vector on the binary image data. The encryption can be tested against the NPCR (Number of Pixel Change Rate) and UACI (Unified Average Change Intensity) . It is also suggested that the UACI value should be around 33% and NPCR should be as large as possible in order to determine the efficacy of the encryption (Balouch et al.; 2017).

CIA (Confidentiality, Integrity and Authenticity) triad of Cloud computing is to be ensured in the data perspective where the Confidentiality must be ensured so that only authorized users have access to the relevant data, Integrity ensures that the data that is transmitted is not altered or tampered and is the same from source to destination, and Authenticity ensures that the receiver is verified to view or modify the data. The method of authorizing is limited to the user, it can be illegally circulated to unauthorized users. Textual data encryption algorithms have been constantly upgrading and getting better, but in case of the images, the decimal or hexadecimal representation needs to be encrypted, where the encryption algorithm is not efficient for reasons like iterations, low NPCR rates (Abdel-Nabi and Al-Haj; 2017). The confidentiality of medical images and data is very high as the loss or leak of such data can lead to a disastrous result for both patient and the host. Most health organizations store the images as it is uploaded without any proper encryption mechanisms, which can allow back-door access which can put down the integrity of a medical organization (Kester et al.; 2015).

Approaches	**Methodology**	**NCPR**	**Entropy**
(Abdullah and Abdullah; 2017)	Hybrid chaotic map	99.63	7.9975
(Oravec et al.; 2018)	Coupled Map Lattice	99.30	7.9994
(Murugan and Gounder; 2016)	Confusion and Diffusion	99.62	7.9994
(Singar et al.; 2017)	Cell shuffling	99.60	7.81
(Hazra and Bhattacharyya; 2016)	Blockwise Fisher Yates	N/A	7.46
(Sharma and Bhargava; 2016)	Two step logistic map	99.61	N/A

Table 1: NPCR and Entropy values observed in the past work

2.5 Conclusion

The study of the past approaches suggests the need for the image encryption in domain like healthcare and SaaS solutions which emphasize on the importance of the image security. Although the concept of image stenography has been around for a while, the need to encrypt faster and efficiently with layered approach is most expected. Hence it would be better if a faster and efficient algorithm can achieve the same or better results. The summary of Evaluation metrics in the past work are represented in Table 1 above.

3 Methodology

The level of security of the Encryption algorithm is not only decided by the effort employed in the process of encryption, but also in the level of randomness and level of rigidity. In this research, a dual encryption of image is employed with colour distortion mechanism in order to ensure security, performance and reliability. In order to encrypt the image in a quick and efficient manner, we utilize Java programming language for this purpose. The encryption follows a sequence of three step procedure, as below:

1. Fast Image extraction - obtaining 2D array from RGB pixel values
2. Pixel shuffling using Modified Yates Fisher Method.
3. Binary transformation - to improve randomness.

3.1 Fast Image extraction

The image extraction can take a lot of time as the image is buffered and then read in pixel array. This pixel array is then translated into a Two-dimensional matrix, which takes a lot of iterations and thus reduces the performance. In order to improve the speed of extraction, the Color class is avoided and the bitwise shift is utilized. The image is imported using the ImageIO class in Java. The image that is obtained as a BufferedImage is translated into byte array of pixels. This byte array is then parsed to convert into a Two-dimensional array.

3.2 Yates Fisher Method

The Yates Fisher Method (named after Ronald Fisher and Frank Yates), also known as Knuth shuffle (named after Donald Knuth) is one of the most prominent shuffling to generate randomness. In this method of shuffling, each element of the One-Dimensional (1D) array is picked and swapped with an element at random until all the elements have been replaced. The random key to perform through this encryption is the same key which is used to decrypt the image. (Alam et al; 2014).

The Yates Fisher method is modified to suit the need of a Two-Dimensional (2D) array for the image encryption in this scenario. The random element is chosen from a 2D array position instead of the 1D implementation of the Yates Fisher Method. The modified Yates Fisher method consumes the key to generate the random position to perform the pixel shuffling.

3.3 Binary transformation

The image pixels can be transformed using the XOR operations in order to confuse the pixels and to increase the Entropy and NPCR(Number of Pixel Change Ratio) values. This transformation strengthens the image encryption process as there is no additional data or noise that is added to achieve the level of encryption.

4 Design Specification

The Figure 2 below is the process flow of Encryption which helps to understand the Encryption method proposed in detail.

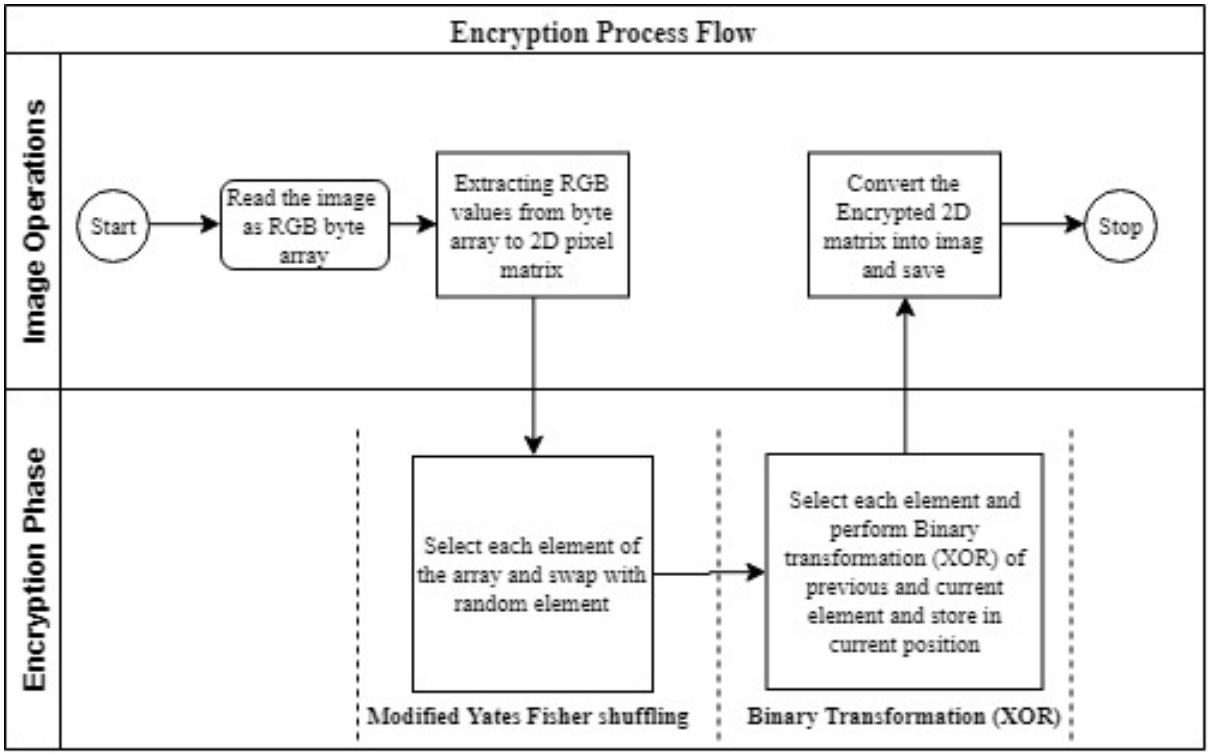


Figure 2: Process Flow diagram of Proposed Encryption

Algorithm 1 Pixel extraction from RGB image

Input: Original image

Output: Original image 2D matrix

$pixelCounter \leftarrow 0$
$column \leftarrow 0$
$row \leftarrow 0$
while pixelCounter < pixelCount **do**
  $blue \leftarrow (pixel[pixelCounter] \& 0xFF) << 16$
  $green \leftarrow (pixel[pixelCounter + 1] \& 0xFF) << 8$
  $red \leftarrow (pixel[pixelCounter + 2] \& 0xFF)$
  $pixelValue \leftarrow red + green + blue$
  $pixelCounter \leftarrow pixelCounter + 3$
end while

Algorithm 2 Modified Fisher Yates Shuffle

Input: Original image 2D matrix
Output: Stage I encrypted image 2D matrix

$i \leftarrow 0, j \leftarrow 0$
while i < 2d Matrix height **do**
  while j < 2d Matrix width **do**
    Generate and select random element position (m,n)
    $temp \leftarrow array[i][j]$
    $array[i][j] \leftarrow array[m][n]$
    $array[m][n] \leftarrow array[m][n]$
    $j \leftarrow j + 1$ {Swap elements in (i,j) with (m,n)}
  end while
  $i \leftarrow i + 1$
end while

Algorithm 3 Binary Transformation

Input: Stage I encrypted image 2D matrix
Output: Final encrypted image 2D matrix

$i \leftarrow 0$
$j \leftarrow 0$
while i < arrayHeight **do**
  while j < arrayWidth **do**
    $inext \leftarrow i$
    $jnext \leftarrow j + 1$
    array[inext][jnext] $\leftarrow$ array[inext][jnext] $\bigoplus array[i][j]$
if j == width **then**
      $inext \leftarrow i + 1$
      $jnext \leftarrow 0$
      {If last element of the row is encountered ,
      Next Element is next row and first column element}
end if
$j \leftarrow j + 1$
  end while
  $i \leftarrow i + 1$
end while

The Section 5 discusses the implementation of the algorithm and an overview about the flow.

5 Implementation

The prototype is built using a Java application using Eclipse IDE. This approach supports colour image encryption also, unlike the grayscale implementations as reviewed in the past work.

The image is read from the file and then converted into a stream of bytes (byte array). This array is a One-dimensional array which contains the pixel values of the image which is of the **size = (width * height * no. of components)**. The number of components here is the ARGB (Alpha, Red, Blue, Green - 4 components) or RGB(Red, Blue, Green - 3 components). The two-dimensional matrix is then constructed by iterating and shifting the components bitwise to form a pixel value.

The bits are shifted right during the conversion of RGB to 2D, whereas it is left shifted during the conversion of 2D to RGB. Each component takes up 8 bits which represent the 256 bit color.The shifting of bits is as follows for each component.

1. Alpha (24-31) - 24 bit shift
2. Red (16-23) - 16 bit shift
3. Green (8-15) - 8 bit shift
4. Blue (0-7) - no bit shift

5.1 Phase I encryption

The pixels are then parsed to perform the Modified Fisher Yates shuffle (Novelist approach). At this stage the input is the original image in two-dimensional matrix format. The output is pixel transpositioned 2D array. The random position is selected for swap at each element, where all the elements are traversed. To improve the performance, the translated array is not created, rather it is done on the same array by swapping, in order to improve the efficiency.

At this stage, the NPCR value was observed to be 98.46% whereas the expected value is around >99%. The Entropy value is around 7.09, whereas to be considered as a randomized and encrypted image, the value should be over 7.20. To achieve this, the binary transformation is to be implemented.

5.2 Final phase encryption

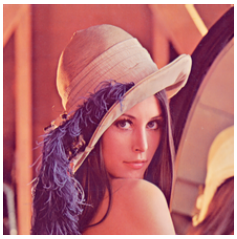
At the next stage, the pixel transpositioned array (output of Modified Fisher Yates Encryption) is the input for the binary transformation. At this stage the pixels are selected pairwise and XORed to increase the Entropy and the randomness. The output of this stage is converted into an image using the pixel to RGB conversion and then stored to memory as the encrypted image.

5.3 Decryption phase

The image is decrypted using the reverse approach by re-iteration of the algorithm. At the first stage, the encrypted image is XORed to obtain the Stage I decrypted image as XOR's compliment is XOR. If C = $A \oplus B$, in order to obtain A, A = $B \oplus C$. At the next stage, the random element choice array is constructed and iterated in reverse to obtain

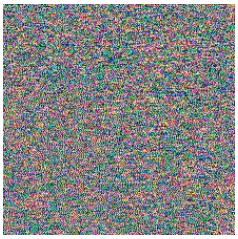
the decrypted 2D array. The image is then stored after translating the 2D decrypted array to image.

The Figure 3,4(a) is the original image, Figure 3,4(b) is the image after Phase I encryption (Modified Fisher Yates shuffling), Figure 3,4(c) is the final encrypted image and Figure 3,4(d) is the decrypted image.



(a) Original Image

(b) After Stage I

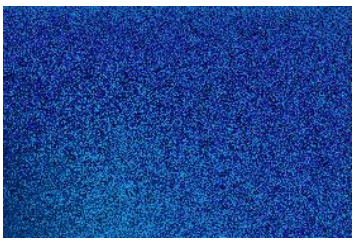


(c) Encrypted Image

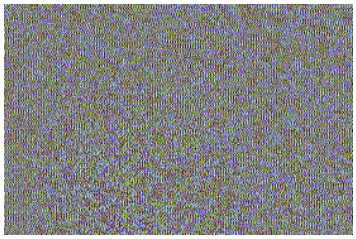
(d) Decrypted Image

Figure 3: Lena Image

(a) Original Image



(b) After Stage I



(c) Encrypted Image

(d) Decrypted Image

Figure 4: Blue Rose Image

6 Evaluation

The extent of encryption can be determined by the quantitatively with the help of NPCR(Number of Pixel Change Ratio) and Entropy, in order to evaluate the randomness and the relevance to the original image. A series of experiments were carried out to evaluate this prototype, out of which four experiments are discussed in this section.

There are three sets of images in each experiment:

(a) Original image

(b) Image after Modified Yates Fisher Encryption

(c) Image after Binary Translation

6.1 Case Study 1

In this approach, the pixel values are using the inbuilt getRGB method of the BufferedImage object. The average time for reading the pixel values of six different images were recorded to be 25.33 milliseconds.

6.2 Case Study 2

In this approach, the extraction of pixel values is by bit-shifting as explained in the Section 3. The average time for reading the pixel values of six different images were recorded to be 13.167 milliseconds.

6.3 Experiment 1

The original image is a benchmark image that is used in most encryption approaches where the encryption is performed over a 256 bit coloured image, called as Lena. The image sizes 220 wide and 220 pixels high. The process is depicted in Figure 5.

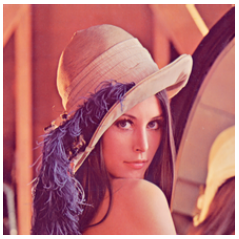
- Entropy value of Stage I image: 7.12
- Entropy value of Encrypted image: 7.63
- NPCR value of Stage I image: 99.86%
- NPCR value of Encrypted image: 100%

The Entropy value is good, and the NPCR value is ideal. At the Stage I, the NPCR value is better than most approaches as 99% is the recommended value for NPCR.

6.4 Experiment 2

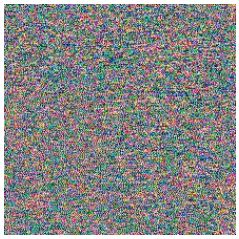
The original image is used as a benchmark in black & white encryption approaches, called as Cameraman. The image sizes 225 pixels wide and 225 pixels high. The process is depicted in Figure 6.

- Entropy value of Stage I image: 7.09
- Entropy value of Encrypted image: 7.76



(a) Original Image

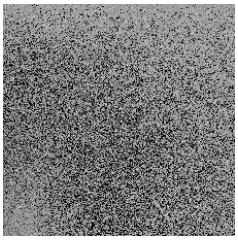
(b) After Stage I



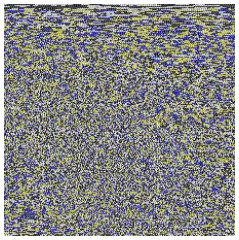
(c) Encrypted Image

Figure 5: Lena Image

(a) Original Image



(b) After Stage I



(c) Encrypted Image

Figure 6: Cameraman

- NPCR value of Stage I image: 98.66%
- NPCR value of Encrypted image: 100%

The Entropy value is high in this experiment and there is a drastic improvement from that of Initial to Encrypted image. The NPCR value is at the ideal level of 100%.

6.5 Experiment 3

The original image in this experiment is the Blue rose which has an overall blue color in the colour spectrum. This image sizes 276 pixels wide and 183 pixels high. The process is depicted in Figure 7.

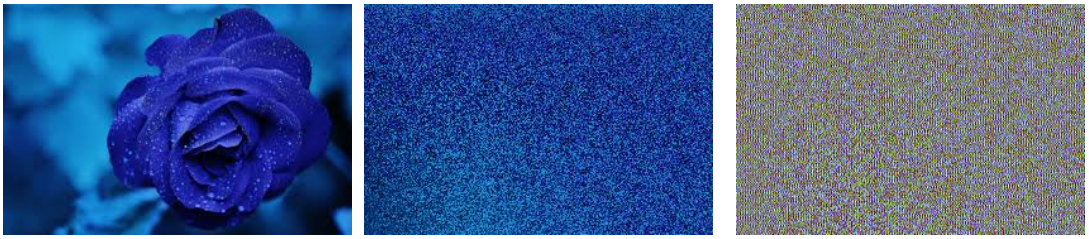
- Entropy value of Stage I image: 7.18
- Entropy value of Encrypted image: 7.63
- NPCR value of Stage I image: 99.81%
- NPCR value of Encrypted image: 100%

The NPCR value is ideal and the Entropy value is good.

6.6 Experiment 4

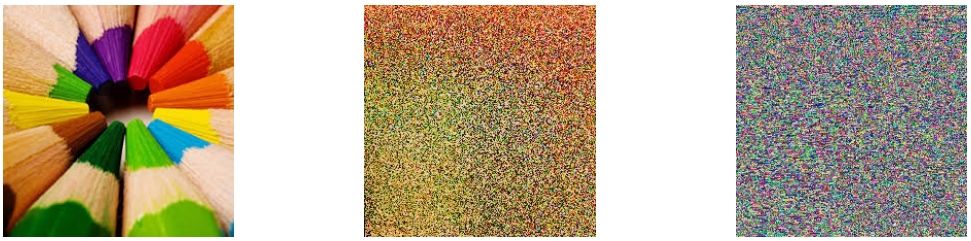
The original image is of a assortment of colour pencils, this is considered to be a good input as the colour distribution of the image pixels will vary. The image sizes 225 pixels wide and 225 pixels high. The process is depicted in Figure 8.

- Entropy value of Stage I image: 7.23



(a) Original Image (b) After Stage I (c) Encrypted Image

Figure 7: Blue rose



(a) Original Image (b) After Stage I (c) Encrypted Image

Figure 8: Colour Pencils

- Entropy value of Encrypted image: 7.63
- NPCR value of Stage I image: 99.84%
- NPCR value of Encrypted image: 100%

The NPCR is a ideal value and the Entropy is commendable.

6.7 Discussion

In the case studies, the Case Study 2 resulted in an efficient pixel value extraction, hence we utilize that approach to extract pixels from the image for performance.

The NPCR value achieved in all the experiments is 100% whereas the maximum value achieved in the past approaches is 99.62%. The Entropy value is not at par when compared to the other approaches, but not distinctively low.

7 Conclusion and Future Work

The sensitivity of the image that is stored can only be defined by the owner or the user. There are times when the value of the image is only known when misused or hacked. In domains like health-care, the image stored can be a X-Ray scan, or a receipt or a DNA image. The level of sensitivity can be determined in the event of a breach in this case.

This paper discuses the implementation of a Dual Pixel shuffling algorithm using modified Yates Fisher method and Binary Transposition, which is a loss-less method of encryption and there is no compression which results in pixel loss during decryption. The entropy value is slightly lesser when compared to other approaches which needs to be considered to improve in the future. The future scope of the research would be: (a) To improve the Entropy value, (b) Analyze the level of security, (c) improve the colour recovery during decryption.

Acknowledgement

I am thankful to my thesis supervisor Dr. Sachin Sharma for constantly encouraging and motivating me throughout the course of the project which helped me to thrive towards the successful implementation. The critics about my approaches helped me to direct myself towards a more efficient and novelist research implementation, without which this wouldn't have been possible.

References

Abdel-Nabi, H. and Al-Haj, A. (2017). Medical imaging security using partial encryption and histogram shifting watermarking, *Information Technology (ICIT), 2017 8th International Conference on*, IEEE, pp. 802–807.

Abdullah, H. N. and Abdullah, H. A. (2017). Image encryption using hybrid chaotic map, *Current Research in Computer Science and Information Technology (ICCIT), 2017 International Conference on*, IEEE, pp. 121–125.

Amalarethinam, D. G. and Geetha, J. S. (2015). Image encryption and decryption in public key cryptography based on mr, *Computing and Communications Technologies (ICCCT), 2015 International Conference on*, IEEE, pp. 133–138.

Balouch, Z. A., Aslam, M. I. and Ahmed, I. (2017). Energy efficient image encryption algorithm, *Innovations in Electrical Engineering and Computational Technologies (ICIEECT), 2017 International Conference on*, IEEE, pp. 1–6.

Bing, L. and Die, F. (2017). An image encryption algorithm of scrambling binary sequences by improved logistic mapping, *Communication Technology (ICCT), 2017 IEEE 17th International Conference on*, IEEE, pp. 1747–1751.

Brindha, M. (2017). Multiple stage image encryption using chaotic logistic map, *2017 International Conference on Intelligent Sustainable Systems (ICISS)*, IEEE, pp. 1239–1243.

Daman, R., Tripathi, M. M. and Mishra, S. K. (2016). Security issues in cloud computing for healthcare, *Computing for Sustainable Global Development (INDIACom), 2016 3rd International Conference on*, IEEE, pp. 1231–1236.

El Makkaoui, K., Ezzati, A., Beni-Hssane, A. and Motamed, C. (2016). Cloud security and privacy model for providing secure cloud services, *Cloud Computing Technologies and Applications (CloudTech), 2016 2nd International Conference on*, IEEE, pp. 81–86.

Hazra, T. K. and Bhattacharyya, S. (2016). Image encryption by blockwise pixel shuffling using modified fisher yates shuffle and pseudorandom permutations, *Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2016 IEEE 7th Annual*, IEEE, pp. 1–6.

Hendre, A. and Joshi, K. P. (2015). A semantic approach to cloud security and compliance, *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on*, IEEE, pp. 1081–1084.

Huang, H. and Yang, S. (2018). Image encryption technique combining compressive sensing with double random-phase encoding, *Mathematical Problems in Engineering* **2018**.

Kaur, R. and Kaur, J. (2015). Cloud computing security issues and its solution: A review, *Computing for Sustainable Global Development (INDIACom), 2015 2nd International Conference on*, IEEE, pp. 1198–1200.

Kester, Q.-A., Nana, L., Pascu, A. C., Gire, S., Eghan, J. M. and Quaynor, N. N. (2015). A security technique for authentication and security of medical images in health information systems, *Computational Science and Its Applications (ICCSA), 2015 15th International Conference on*, IEEE, pp. 8–13.

Kumar, A. and Nishchal, N. K. (2018). An image encryption scheme employing quick response codes, *Microwave and Photonics (ICMAP), 2018 3rd International Conference on*, IEEE, pp. 1–2.

Li, P. and Lo, K.-T. (2015). Joint image compression and encryption based on alternating transforms with quality control, *Visual Communications and Image Processing (VCIP), 2015*, IEEE, pp. 1–4.

Murugan, B. and Gounder, A. G. N. (2016). Image encryption scheme based on block-based confusion and multiple levels of diffusion, *IET Computer Vision* **10**(6): 593–602.

Nayak, A. A., Sridhar, N., Poornima, G. et al. (2017). Security issues in cloud computing and its counter measure, *Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2017 2nd IEEE International Conference on*, IEEE, pp. 35–41.

Oravec, J., Turan, J. and Ovsenik, L. (2018). Image encryption technique with key diffused by coupled map lattice, *Radioelektronika (RADIOELEKTRONIKA), 2018 28th International Conference*, IEEE, pp. 1–6.

Ranjan, K. H., Fathimath, S. S., Shetty, S. and Aithal, G. (2017). Image encryption based on pixel transposition and lehmer pseudo random number generation, *Recent Trends in Electronics, Information & Communication Technology (RTEICT), 2017 2nd IEEE International Conference on*, IEEE, pp. 1188–1193.

Safi, H. W. and Maghari, A. Y. (2017). Image encryption using double chaotic logistic map, *Promising Electronic Technologies (ICPET), 2017 International Conference on*, IEEE, pp. 66–70.

Sharma, M. and Bhargava, A. (2016). Chaos based image encryption using two step iterated logistic map, *Recent Advances and Innovations in Engineering (ICRAIE), 2016 International Conference on*, IEEE, pp. 1–5.

Singar, C. P., Bharti, J. and Pateriya, R. (2017). Image encryption based on cell shuffling and scanning techniques, *Recent Innovations in Signal processing and Embedded Systems (RISE), 2017 International Conference on*, IEEE, pp. 257–263.

Vaish, A. and Kumar, M. (2018). Color image encryption using singular value decomposition in discrete cosine stockwell transform domain, *Optica Applicata* **48**(1).

Zhe, D., Qinghong, W., Naizheng, S. and Yuhan, Z. (2017). Study on data security policy based on cloud storage, *Big Data Security on Cloud (BigDataSecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), 2017 IEEE 3rd International Conference on*, IEEE, pp. 145–149.

ZHOU, Z., YU, J., LIAO, Q. and GONG, L. (2018). Image encryption combining discrete fractional angular transform with arnold transform in image bit planes, *Optica Applicata* **48**(2).

National College of Ireland

Configuration Manual

MSc Research Project
MSc Cloud Computing

Mukul Gopinath
Student ID: x17123739

School of Computing
National College of Ireland

Supervisor: Dr. Sachin Sharma

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Mukul Gopinath
Student ID:	x17123739
Programme:	MSc Cloud Computing
Year:	2018
Module:	MSc Research Project
Supervisor:	Dr. Sachin Sharma
Submission Due Date:	20/12/2018
Project Title:	Configuration Manual
Word Count:	635
Page Count:	6

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	18th December 2018

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Mukul Gopinath
x17123739

1 Eclipse IDE

Eclipse IDE is a Integrated Development Environment which enables the Developers to create, deploy and test code hassle-free. Eclipse provides suitable plugins for deployment over cloud providers like Azure by Microsoft, AWS from Amazon. Eclipse IDE is suitable for Java development.

2 Downloading Eclipse IDE

Click on Eclipse Downloads to access the Eclipse Foundation website. Choose the Eclipse IDE for JavaEE as shown in Figure 1. In order to choose the suitable for the system, check

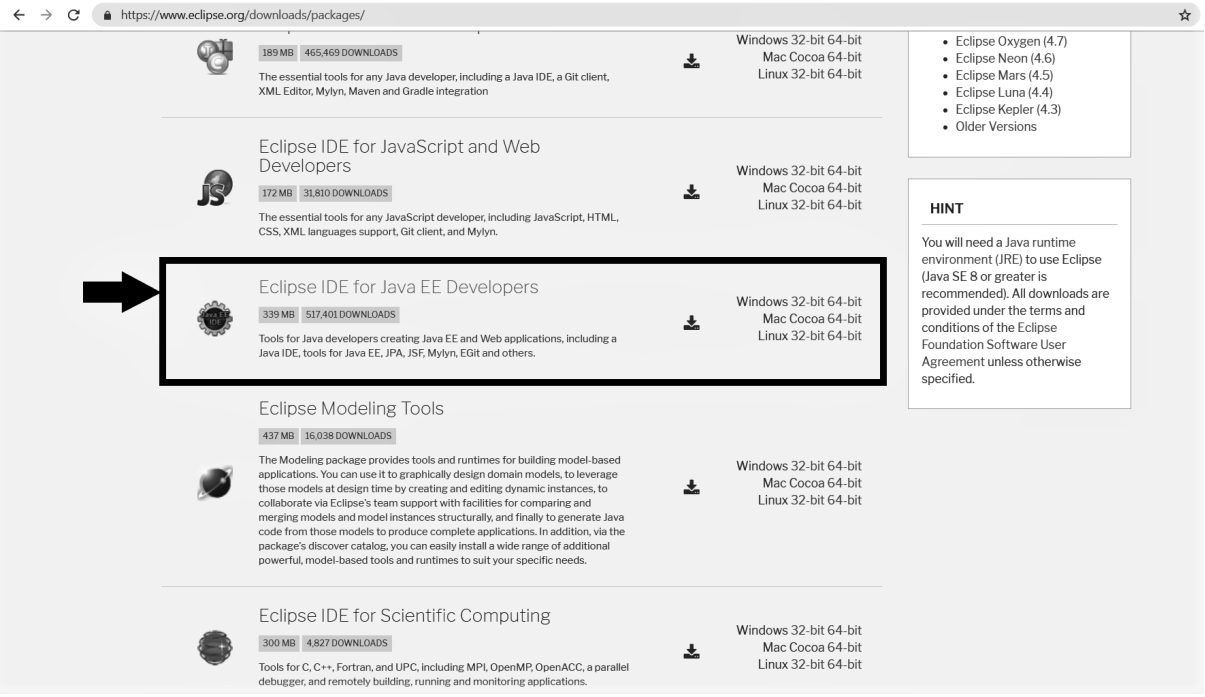


Figure 1: Eclipse Website

the system specification like Processor 32-bit or 64-bit, Windows or Linux or MacOS. Click on the appropriate option to download the Eclipse setup as shown in Figure 2.

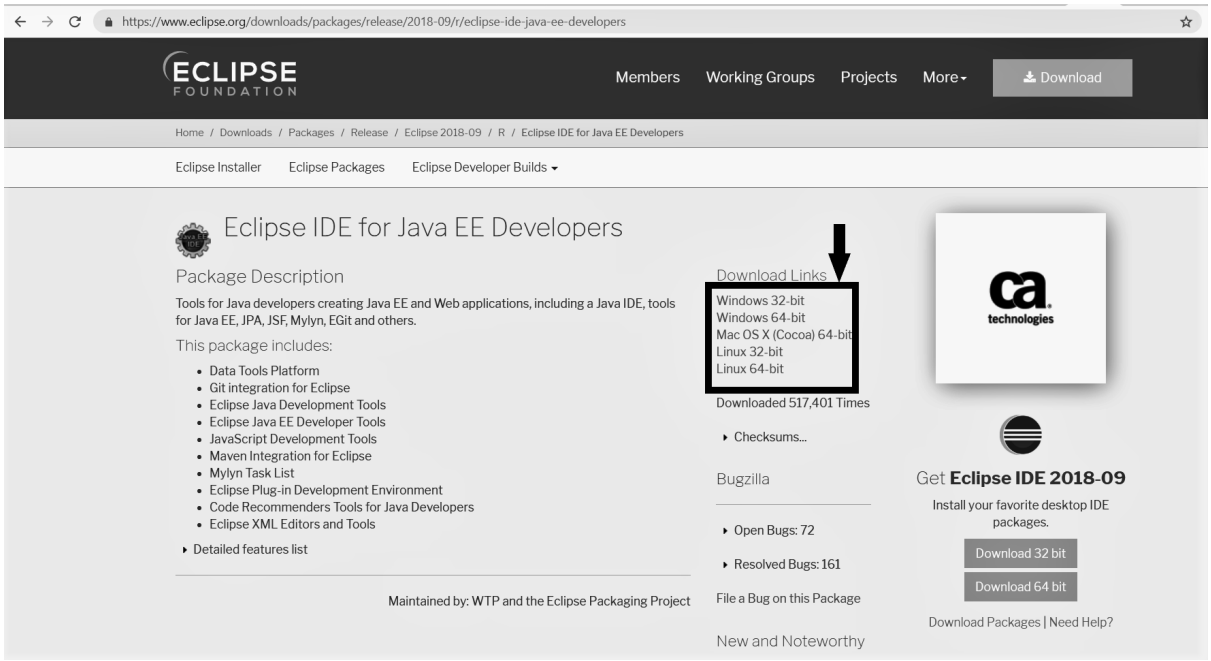


Figure 2: Eclipse Website

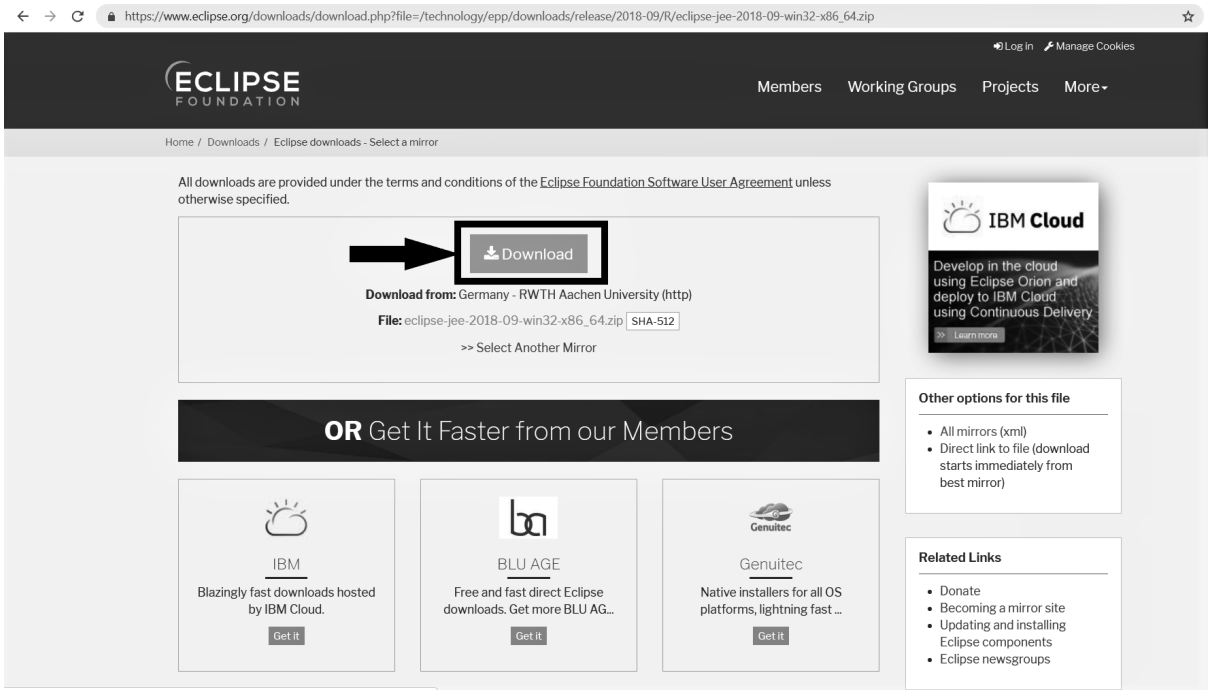


Figure 3: Eclipse Website

Click on download to confirm the download as shown in Figure 3. Then you will see the file being downloaded as shown in Figure 4.

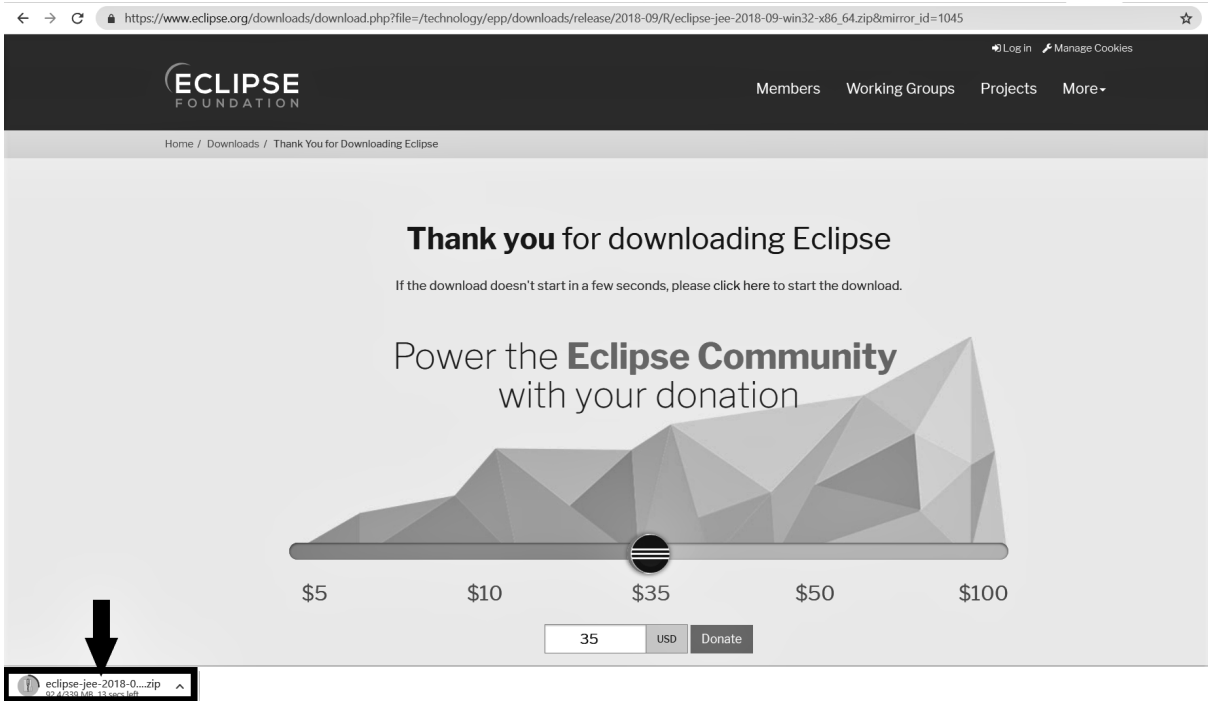


Figure 4: Eclipse Website

Run the installation by clicking on the file after download.Follow the steps and specify the location and Finish installation. After successful installation, start Eclipse. (*Eclipse Installer 2018-09 Ra*; n.d.)

3 Checking Java version

Open command prompt or terminal and check for java version by typing the syntax as in Figure 5.

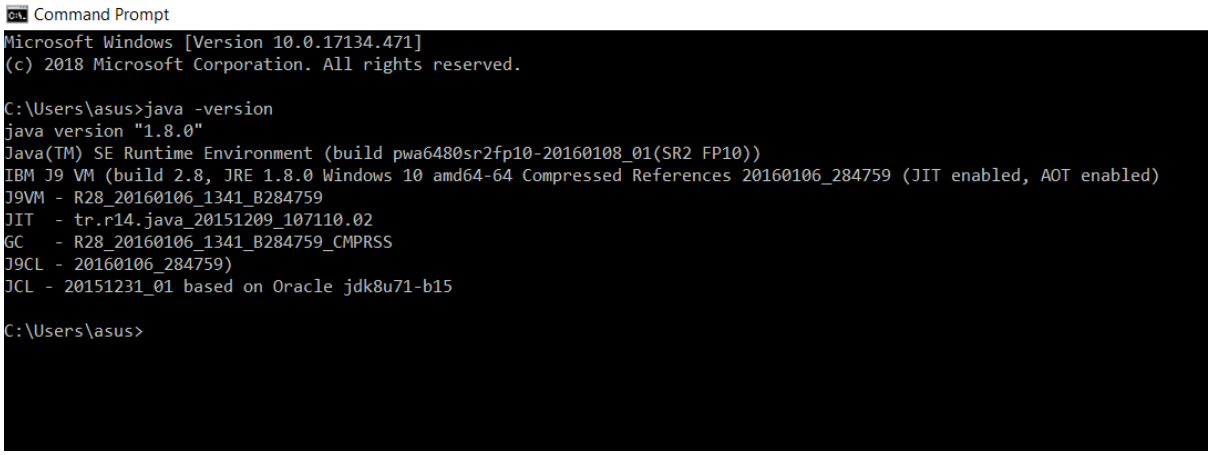


Figure 5: Java Version

If the version is lesser than Java 8, follow the steps below to update to the newest version. Click on Oracle Link to download the Java Standard Edition package as shown

in Figure 6.

Figure 6: Download Java SE from Oracle

After successful download, click on the setup and install Java. After installation, verify the Java version as in the previous step (*How to Install JDK 10 (on Windows, Mac OS Ubuntu) and Get Started with Java Programming*; n.d.).

4 Running a Java Web Application to Encrypt

Use the below code in Java to perform Encryption.

```
public static void encrypt_the_image(int image_id) {
    try {
        System.out.println("Original Image Entropy ------------------->");
        EvaluationTests.getShannonEntropy_Image(originalImage);
        EncryptProcess.checksum = 0;
        for(int i=0;i<EncryptProcess.height; i++) {
            for(int j=0; j<EncryptProcess.width; j++) {
                EncryptProcess.checksum += Math.abs(EncryptProcess.encryptedImage[i][j]);
            }
        }
        System.out.println("CHECKSUM    : "+EncryptProcess.checksum);
        shufflePixelsUsingYatesModel(1002); //arr,42);


        System.out.println("Encrypted Image NPCR Stage 1 ------------------->");
        EvaluationTests.calculateNPCR();
        System.out.println("Encrypted Image UACI Stage 1 ------------------->");
        EvaluationTests.calculate_UACI();
        System.out.println("Encrypted Image Entropy Stage 1 ------------------->");
        EvaluationTests.getShannonEntropy_Image(encryptedImage);

        EncryptProcess.checksum = 0;
        for(int i=0;i<EncryptProcess.height; i++) {
            for(int j=0; j<EncryptProcess.width; j++) {
                EncryptProcess.checksum += Math.abs(EncryptProcess.encryptedImage[i][j]);
            }
        }
        System.out.println("CHECKSUM    : "+EncryptProcess.checksum);
        IMGOperation.convert_2D_to_RGB("C:\\Users\\asus\\Desktop\\Encrypt\\encrypt\\q"+image_id+"i1.jpg");
        shuffleUsingXOROperation();

        System.out.println("Encrypted Image NPCR Stage 2 ------------------->");
        EvaluationTests.calculateNPCR();
        System.out.println("Encrypted Image UACI Stage 2 ------------------->");
        EvaluationTests.calculate_UACI();
        System.out.println("Encrypted Image Entropy Stage 2 ------------------->");
        EvaluationTests.getShannonEntropy_Image(encryptedImage);
        IMGOperation.convert_2D_to_RGB("C:\\Users\\asus\\Desktop\\Encrypt\\encrypt\\q"+image_id+"1.jpg");

    }catch (Exception e) {
        e.printStackTrace();
    }
}
```

Figure 7: Main code flow

```
public static void decrypt_the_image(int image_id) {
    try {
        EncryptProcess.checksum = 0;
        for(int i=0;i<EncryptProcess.height; i++) {
            for(int j=0; j<EncryptProcess.width; j++) {
                EncryptProcess.checksum += Math.abs(EncryptProcess.encryptedImage[i][j]);
            }
        }
        deShufflePixelUsingYatesModel(1002);
        IMGOperation.convert_2D_to_RGB("C:\\Users\\asus\\Desktop\\Encrypt\\decrypt\\q"+image_id+".jpg");
    }catch (Exception e) {
        e.printStackTrace();
    }
}
public static void shufflePixelsUsingYatesModel(long seed) {
    Random random = new Random(seed);
    for (int i = EncryptProcess.height - 1; i > 0; i--) {
        for (int j = EncryptProcess.width - 1; j > 0; j--) {
            int m = random.nextInt(i + 1);
            int n = random.nextInt(j + 1);
            int temp = EncryptProcess.encryptedImage[i][j];
            EncryptProcess.encryptedImage[i][j] = EncryptProcess.encryptedImage[m][n];
            EncryptProcess.encryptedImage[m][n] = temp;
        }
    }
}
public static void shuffleUsingXOROperation() {
    try {
        for(int i = 0; i < EncryptProcess.height; i++) {
            for(int j = 0; j < EncryptProcess.width; j++) {
                if(i == EncryptProcess.height - 1 && j == EncryptProcess.width - 1) {
                    continue;
                }
                int next_iloc = i, next_jloc = j + 1;
                if(j == EncryptProcess.width - 1) {
                    next_jloc = 0;
                    next_iloc = i + 1;
                }
                EncryptProcess.encryptedImage[next_iloc][next_jloc] = -1 * (Math.abs(EncryptProcess.encryptedImage[i][j]) ^ Math.abs(EncryptProcess.encryptedImage[next_iloc][next_jloc]));
            }
        }
     }catch(Exception e) {
        e.printStackTrace();
     }
}
```

Figure 8: Encryption and decryption code

```
public static void convert_RGB_to_2D(String originalImagepath) {
    try {
        File imgFile = new File(originalImagepath);
        BufferedImage image = ImageIO.read(imgFile);
        final byte[] pixels = ((DataBufferByte) image.getRaster().getDataBuffer()).getData();
        final int width = image.getWidth();
        final int height = image.getHeight();
        EncryptProcess.encryptedImage = new int[height][width];//new int[width][height];
        EncryptProcess.originalImage =  new int[height][width]; //new int[width][height];
        final boolean hasAlphaChannel = image.getAlphaRaster() != null;
        if (hasAlphaChannel) {
            final int pixelLength = 4;
            for (int pixel = 0, row = 0, col = 0; pixel < pixels.length; pixel += pixelLength) {
                int argb = 0;
                argb += (((int) pixels[pixel] & 0xff) << 24); // alpha
                argb += ((int) pixels[pixel + 1] & 0xff); // blue
                argb += (((int) pixels[pixel + 2] & 0xff) << 8); // green
                argb += (((int) pixels[pixel + 3] & 0xff) << 16); // red
                EncryptProcess.originalImage[row][col] = argb;
                EncryptProcess.encryptedImage[row][col] = argb;

                if (col == width) {
                   col = 0;
                   row++;
                }
            }
        } else {
            final int pixelLength = 3;
            for (int pixel = 0, row = 0, col = 0; pixel < pixels.length; pixel += pixelLength) {
                int argb = 0;
                argb += -16777216; // 255 alpha
                argb += ((int) pixels[pixel] & 0xff); // blue
                argb += (((int) pixels[pixel + 1] & 0xff) << 8); // green
                argb += (((int) pixels[pixel + 2] & 0xff) << 16); // red
                EncryptProcess.originalImage[row][col] = argb;
                EncryptProcess.encryptedImage[row][col] = argb;

                col++;
                if (col == width) {
                    col = 0;
                    row++;
                }
            }
            EncryptProcess.width = EncryptProcess.encryptedImage[0].length;
            EncryptProcess.height = EncryptProcess.encryptedImage.length;
        }
```

Figure 9: Image to 2D array conversion

```
    public static void convert_2D_to_RGB(String filename) {
        try {
            BufferedImage convertedImage = new BufferedImage(EncryptProcess.width,EncryptProcess.height, BufferedImage.TYPE_3BYTE_BGR);
            int pixelSize = (EncryptProcess.height * EncryptProcess.width * 3);
            final int[] pixelReconstructInt = new int[pixelSize];
            int cnt = 0;
            for(int i=0;i<EncryptProcess.height;i++) {
                for(int j=0;j<EncryptProcess.width;j++) {
                    pixelReconstructInt[cnt] = EncryptProcess.encryptedImage[i][j];
                    cnt++;
                }
            }
//          System.out.println(pixelReconstructInt[0]+" "+pixelReconstructInt[1]+" "+pixelReconstructInt[2]);
            convertedImage.setRGB(0,0,EncryptProcess.width,EncryptProcess.height,pixelReconstructInt,0,EncryptProcess.width);
            System.out.println(convertedImage.getWidth() +" "+ convertedImage.getHeight());
            if(convertedImage != null) {
                ImageIO.write(convertedImage , "jpg", new File(filename));
                System.out.println("Success");
            }
        }catch(Exception ex) {
            ex.printStackTrace();
        }
    }
```

Figure 10: 2D array to Image conversion

References

Eclipse Installer 2018-09 Ra (n.d.). *Eclipse Public License - Version 1.0* .
URL: *https://www.eclipse.org/downloads/packages/installer*

How to Install JDK 10 (on Windows, Mac OS Ubuntu) and Get Started with Java Programming (n.d.). *Elements inthe Literature Review* .
URL: *https://www3.ntu.edu.sg/home/ehchua/programming/howto/JDK$_H$owto.html*