Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

A MapReduce Based Approach for Secure Batch Satellite Image Encryption

Mahmoud Ahmad Al-Khasawneh[1], M. Faheem [2] , Eman A. Aldhahri[3], Abdulrahman Alzahrani[3] , Ala Abdulsalam Alarood[3]

[1] School of Computing, Skyline University College, University City Sharjah, 1797, Sharjah, UAE
[2]School of Computing Technology and Innovations, University of Vaasa, Vaasa 65200, Finland.
[3]College of Computer Science and Engineering, University of Jeddah, 21959 Jeddah, Saudi Arabia

Corresponding author: M. Faheem (e-mail: muhammad.faheem@uwasa.fi).

ABSTRACT The overarching goal of this research was to examine the state of satellite imagery security in relation to its deteriorating form due to rising demand. The most common approaches to safeguarding satellite images during transmission across transmission networks, which are not protected by standard encryption, are the focus of this investigation. Since satellite imagery can be encrypted both in transit and while stored on a computer's hard drive, we put the suggested Image Encryption System to the test by applying it to a collection of satellite photos. Concurrently encrypting data and running MapReduce jobs is key to the study methodology employed. This will be carried out in the Hadoop ecosystem, where an innovative method of analysing random numbers for use in Image encryption will be put to the test. The encryption was processed using MapReduce in the Hadoop ecosystem. Images were saved as BMP files with added security metadata. The evaluation of experiments was based on four (4) indicators. It was found that the processing time for batch encryption calculations grew in proportion to the amount of computations. All cluster, map, and reduction processes were put to the test using encrypted images, exposing load balancing difficulties and inefficiencies. Histogram analysis, the basis of an image encryption technique, provides evidence that the encrypted pixel values are consistent. Therefore, compared to other methods, such as a histogram or information entropy, this one is superior. Because of how it was crafted, it can withstand even the most sophisticated attacks without being compromised.

INDEX TERMS Encryption, Batch-Encryption, Satellite Image, MapReduce

I. INTRODUCTION

Encryption is one of the most extensively used methods for safeguarding satellite imagery on transmission networks. This means that encryption techniques not only help secure satellite imagery during transmission but also while it is being stored [1-2]. When it comes to encrypting and storing vast volumes of satellite imagery, however, it makes it more difficult to keep the processing power of the system up and running [3]. The "Asymmetric" and "Symmetric" encryption methods both include image encryption as one of their sub-types. These two types of encryption methods are the most frequent ones that are now accessible [4]. Encrypting other forms of material, such as "Audio," "Video," or "Image," is a lot more difficult than encrypting text [5]. If the encryption key and the decryption key are different from one another, as is the case with asymmetric encryption algorithms, it is considerably more difficult to achieve this goal. As a consequence of this, it is necessary to make use of exponential operations, which makes the encryption process more time-consuming [6].

This current research is motivated and Inspired by the construction of a novel pixel-split picture encryption approach based on a 2D Salomon map [7], we created and developed a 2D hyperchaotic system for image encryption [8] based on the Schaffer function. The influence of a high-efficiency medical picture encryption approach based on a two-dimensional logistic-gaussian hyperchaotic map [9]. Creating a parallel picture encryption technique by scrambling the bits within the bitplane [10]. In addition to the creation of a meaningful picture encryption technique that is based on compressive sensing and the integer wavelet transform [11], this was also accomplished.

The challenges associated to undertaking this study lies with the fact that when dealing with huge Image, existing encryption techniques are inefficient and sluggish to function, as is generally the case [12]. However, there is a solution to this problem. Traditional image encryption techniques are unable to execute batch image encryption in an efficient manner [13]. Image encryption is a method that involves encoding an image in such a way that it can be decoded only by the sender and the recipient to whom it is intended to be sent. [14] A chaos-based image encryption technique is considered to be one of the most effective image encryption schemes.

Another challenge lies with the fact that satellite images are images of Earth that have been recorded by imaging satellites and then communicated back to the surface of the planet [15]. Satellite images are used in geospatial contexts. Satellite imaging is doing something no other method has been able to do before: it is displaying for the first time both the immense worldwide scale of fires as well as their global footprint. The mapping of coastal ecosystems has been significantly aided by the use of imagery obtained from satellites [16]. Because of this, estimates of the land coverage and alteration in the coastal environment have been produced, which has led to a better knowledge of the ecosystem [17]. Satellite images are becoming increasingly popular in the age of big data, despite the fact that modern satellites operate over a much wider range of frequencies than just three [18]. This is the case despite the fact that modern satellites operate over a much wider range of frequencies than just three. Because they do not require direct physical touch or firsthand experience, they can be captured through the use of drones, aerial pictures, and satellite sensors [19]. This makes it possible for them to be captured. When it comes to characterising locations and occurrences on the surface of the Earth and in the atmosphere, algorithms are used to improve and refine the data. This is done both manually and automatically. In order to enhance and refine the data, algorithms are utilised [20].

In the current investigation, a Chotic map combined with MapReduce image encryption was utilised. The rationale for doing that is because picture encryption is a method that requires encoding an image in such a way that it can only be decrypted by the sender and the intended recipient of the image. The picture encryption system known as chaos-based image encryption is one of the most effective image encryption schemes [14]. Several different kinds of chaotic map block chippers with secret keys of 128, 256, and 512 bits each were produced [21]. There are many different kinds of chaos maps, and they can all be used to do the same thing: improving the encryption performance of satellite photos. Examining the statistical sensitivity and performance of these maps has shown that they are both effective and secure, which has allowed the speed of the encryption algorithm to be determined [22]. This has been proved by the fact that their efficiency has been demonstrated.

The objective of this study lies with the development of the strategy characterized as a chain-based satellite image encryption system that has the potential to aid in the process of encrypting and decrypting satellite images in batch form in real time. The significance/implication of this study in this specific situation, dwells on the provision of no limitations placed on the quantity of satellite image that may be utilized. Hence the contribution of the research lies with the provision of the method that has the potential to be utilized to increase the safety of both the transmission and storage of satellite image as well as any data within the categories of image formats [23]. The technique that was used at the same time to improve encryption performance and security is comprised of several fundamental components, the most important of which is a straightforward logical XOR operation. Multiple key generation procedures were also utilised. Every part of the design of the image encryption on the satellite needs to be exposed to as much examination as is humanly possible. This must be done to the fullest extent possible. This research confirmed that the performance of the encryption method can be improved by combining the three most common encryption schemes, which are DES, AES, and 3-DES. The study considered a range of metrics to come to this conclusion. However, when compared with the three most prominent encryption algorithms—AES, 3-DES, and DES—the chaotic-enhanced MapReduce suggested approach will lead to an understanding of the efficiency of the algorithms that are used.

II. RELATED WORK

Using Chaos-based satellite imagery cryptosystem has been demonstrated by Usama et al. [1] that it is possible to create a secure satellite imagery cryptosystem with the technique; however, this study does not take into consideration the use of multiple images at the same time. A similar vein of research is being pursued by Vaseghi et al. [2], who employed finite time chaos synchronisation in time-delay channels and its application to satellite image encryption in OFDM communication systems, which is a similar approach to the work of Usama et al. [1]. A numerical simulation study is used to evaluate the proposed chaotic-based satellite image encryption/decryption system. The robustness of the proposed approach as well as the efficiency of the proposed chaotic encryption structure are demonstrated for a specific set of security parameters. Despite the fact that other image encryption algorithms are widely regarded as extremely important, a semi-symmetric image encryption scheme based on the function projective synchronization of two hyper chaotic systems was discovered to be effective in terms of robustness [4]. Despite the fact that other image encryption algorithms are widely regarded as extremely important. The result has been a hybridization of chaos-based encryption, which has led to the development of a parallel image encryption algorithm based on hybrid chaotic maps [13]. There are several other approaches that make use of small satellites [14]. This demonstrates that the batch collection of images has been completely overlooked throughout the research process to this point.

Advanced Encryption Standard (AES) was one of the popular encryption algorithm that was used for securing data. It was also applied to the image to generate a random sequence of numbers in order to distinguish characteristics by using a 256-bit key [23]. The advantages of adopting this kind of approach to image lies with improving performance, security, and complexity in order to store and transmit confidential data in a secure manner [23]. In order to encrypt satellite images with complex compositions, it was revealed that the image should converted into matrices were the row and column operations can have swapped so that the elements of the previous row is multiplied by a fixed number of times [24]. These interpolation techniques, the clustering of necessary matrix operations, and the use of simple line operations produced good results and increased interpolation efficiency when used in conjunction with the approach described above. Because the proposed cryptosystem was concise, the amount of computational work required was reduced. For a variety of file sizes and resolutions, the calculated PSNR values of the proposed technique were found to be significantly safer than those obtained from previous methods, demonstrating its superior safety. Comparing the proposed algorithm to the previous algorithm, the proposed algorithm significantly improved image quality and PSNR values while encrypting image data more efficiently. Naim et al. [25] propose a satellite image algorithm based on the linear feedback shift register generator for hyperchaotic systems, which is based on the linear feedback shift register generator.

There have been a great number of studies done in the past that involve the utilization of satellite image encryption. The research on "Preserving privacy of classified legitimate satellite lane images utilizing proxy re-encryption and UAV technology" [26] is extremely important to this endeavor and plays a pivotal role in it. The difficulty in the study is related with the computational complexity that is involved in the encryption channels. This technology is the state of the art, and its virtues lay with "proxy re-encryption." However, this technique has its limitations. As a result, in order to make up for this deficiency, the current research proceeded with the batch method. An end-to-end colour image encryption system that makes use of a deep generative model has been suggested using Encipher GAN by Panwar et al. [27], and its effective implementation has been demonstrated. This is another state-of-the-art technique, and its strengths lay in its implementation of a "deep generative model," while the problem in the study is associated with the training function related to errors functions in the deep generative model that are involved in the Encipher GAN function. As a result, in order to close this knowledge gap, the current research proceeded with both the batch process and the chaotic map processing.

A successful development and implementation of a privacy-preserving image encryption system that is based on an optimal deep transfer learning-based accident severity classification model has been accomplished in Sirisha and Chandana,[28]. The strengths of this technique lie in "deep transfer learning," while the issue at hand in the study is associated with the classification model's capacity to be involved in the encryption channels. This technique represents the state of the art in terms of its capabilities. As a result, in order to make up for this deficiency, the current research proceeded with the batch method [29]. Recent discoveries have shed light on previously unknown aspects of chaos-based picture encryption and their applications. A good success that encouraged this research was accomplished by an enhanced chaotic picture encryption technique that used a hadoop-based mapReduce framework for huge distant sensed images in parallel IoT applications [30]. This research was driven by the success of this algorithm. A method of image encryption that uses a combination of many chaotic maps was suggested in [31]. A successful hybrid privacy-preserving deep learning strategy for object classification in very high-resolution satellite pictures was demonstrated by Boulila et al. [32]. This approach was utilised for the classification of objects in the images. It has been demonstrated that the use of a hybrid hyper chaotic map with LSB for the purpose of image encryption and decryption is successful in Shankar and Nandini [33].

All of these studies place a greater emphasis on the raw encryptions standard, but they fall short of delivering on their promise to take into consideration a wide variety of different raw image data file combinations. This is one of the most significant research voids related with the current state of the art. As a result, the implementation of a MapReduce-based strategy for secure batch satellite picture encryption would result in an improvement in their quality as a result of this ongoing research.

In light of the critical information associated to encryption, a high level of security, a sufficiently large key space, Single Event Upset tolerance, and low complexity are among the characteristics revealed by the research. Kumar and Dua [34] proposed approaches for encrypting satellite images with pseudo random keys and cosine transformed chaotic maps that were based on cosine transformed chaotic maps. With various parameters, the proposed pseudo random key and CTLSM combination were tested, and the results showed that they were effective in resisting all types of attacks against the encryption technique. Bensikaddour et al. [35] use a chaos-based block cypher to encrypt multispectral satellite images, which they describe as follows: The findings of the study demonstrated that the proposed method has a high level of security while requiring little hardware complexity and power, and that it is capable of reaching a throughput of 120 Mbit/sec.

Using the MapReduce model, it is possible to encrypt large images. This enabled the resolution of large amounts of data that would otherwise have been impossible to fit into the memory [36]. In recent years, the use of image encryption has gained widespread acceptance in numerous studies. Image encryption for large amounts of data in a variety of environments [37]. Cryptosystems that break down a given image into smaller components that are then processed by different processors before being combined to produce the final output are available. There are countless procedures for encrypting images, and it has been proven to be a successful method of communicating confidential information in the past.

II. ARCHITECTURAL FRAMEWORK

Image Encryption continues to attract researchers despite this, owing to the phenomenal increase in the use of images in all forms of digital communication that has occurred in the last few years [38]. As a result, the implementation of the MapReduce programmer was proposed because it provided several advantages over other distributed computing technologies such as MPI, shared-memory models, and so on. Most of the time, distributed programming is driven by nodes of computing that are connected by a shared file system, and the data from each job must be transferred to the nodes in order to continue the job execution process [39]. As a result, the MapReduce framework was built on top of a distributed file system, as data transfer over the network would otherwise be prohibitively expensive. As a result, massive images were processed while the location remained unchanged.

Comparatively to other equivalent programming models, the MapReduce programming model provides the application developer with an interface that is significantly simplified to the point where it can serve as the only map. Apart from that, reduce functions are required for the entire distributed processing, as well as for the handling of input and output data, when using MapReduce (see Figure 1). According to some accounts, the use of this method eliminated the possibility of exerting control over issues relating to the location and timing of an executable mapper or reducer, whereby chunks of input data were processed via the function of a specific node or mapper, and which intermediate data is processed by the mapper or reducer [40].

The Hadoop environment was chosen as the testing platform for the newly developed Image encryption scheme in the current study. Among the factors influencing this decision is that the method is the primary open-source implementation of the simple MapReduce programming of Hadoop-based systems and applications [41]. Although several issues have been addressed in relation to the application of MapReduce-based algorithms in the context of the Hadoop environment, there are still some issues to be resolved. These include the issue of the format of the input and output files for processing, which is important when considering that the primary purpose of Hadoop was to process large amounts of unstructured text. Another point of contention concerned the design of the proposed Image encryption algorithm, which made use of the map- and reduce-functions to accomplish its goal. As a result, in addition to the Hadoop implementation for image processing, the importance of design ideas for addressing these issues was emphasized.

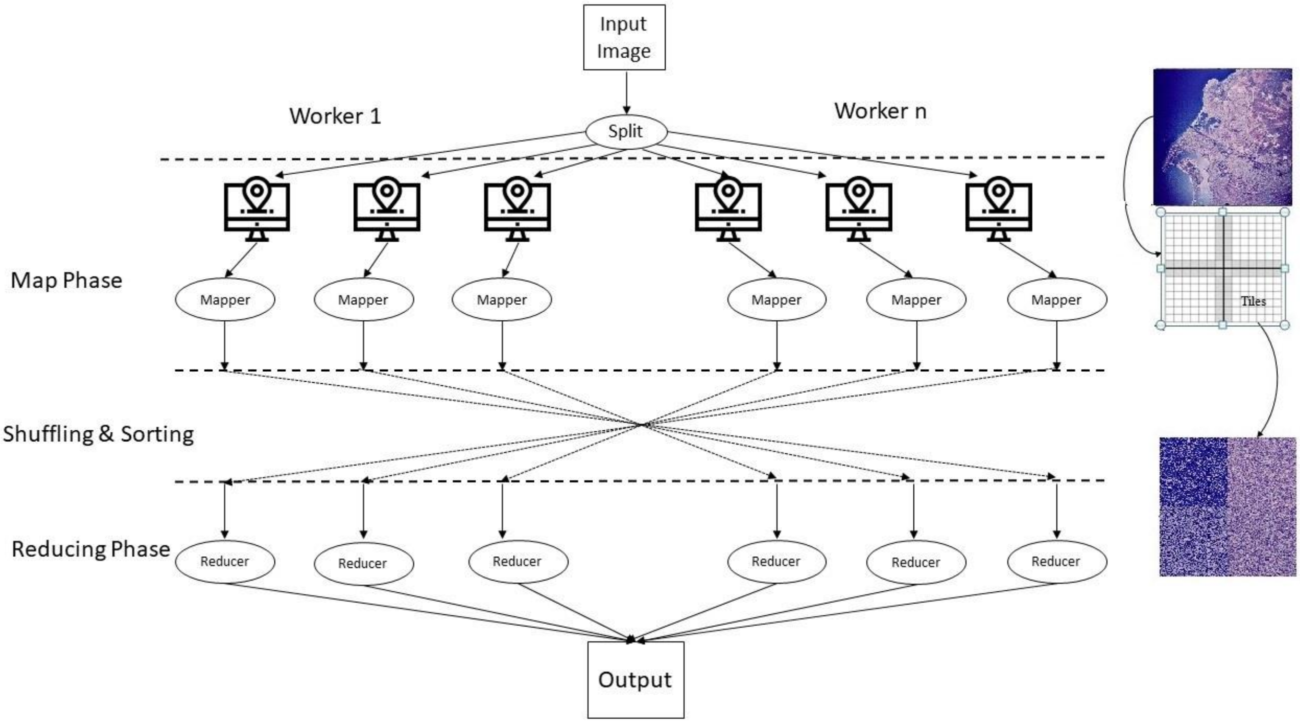

FIGURE 1. **The MapReduce process for the proposed IE scheme**

The MapReduce framework is being used in the proposed work to process large image files that have been stored in the Hadoop Distributed File System, as described in greater depth below (HDFS). These images were saved in the BMP format, with additional metadata attached to the entire image file as an additional layer of security. The Hadoop system was able to accept and output data in a variety of formats crucial to the implementation of the InputFormat and OutputFormat interfaces. These formats included text, binary, database, and others. Large-scale text processing tasks such as word counting, sorting, TF-IDF and other similar tasks are typically performed using Hadoop. In order to accommodate text processing, the vast majority of its built-in input formats were designed with this in mind. The basic Hadoop InputFormat interfaces have also been extended to allow for the execution of non-text distributed processing operations in addition to text-based distributed processing operations. As a result of the use of such extendable interfaces, the authors were able to design their own file formats on top of the Hadoop's interface, which was built on the Hadoop's interface.

The first Hadoop operation was to extend the “ImageInputFormat” class to include the “FileInputFormat” class, which was then used to read the image information from the input file. This was the first Hadoop operation. The image file could not be divided, in contrast to a text file, which could be safely divided and processed on different mappers. This was due to the fact that the image file contained essential format specific metadata for decoding the image. As a result, the “ImageInputFormat” class informed Hadoop that the file should not be split across workers. This class also informed Hadoop to use the “ImageRecordReader” class to read in file data into the system, which was previously not done. The operation is carried out by extending the “ImageOutputFormat” to include. It is the output writer's responsibility to collect output from the reducer and write it to HDFS. In a similar vein to the Input Reader, Hadoop provided expendable classes that could be used to write custom data back to the HDFS. The “FileOutputFormat” and “RecordWriter” classes have been further developed to allow for the writing of the final encrypted/decrypted image to HDFS as an output. Afterwards, the “ImageOutputFormat” class extended Hadoop's “FileOutputFormat” class to notify Hadoop to use the “ImageRecordWriter” class to write the image data to Hadoop's Distributed File System (HDFS).

The “ImageRecordWriter,” which extended Hadoop's RecordWriter class, made use of the ImageWritable class functions to obtain image data and then convert it to an image format for storage in HDFS using the ImageWritable class functions. Following that, the “ImageRecordReader” class extended the Hadoop's “RecordReader” class to allow for the loading of image data and the conversion of that data into useful Key-value objects that could be read by the Mappers. This class was responsible for the actual image data reading, which was accomplished through the use of Java Image libraries. Following the reading of the image data, the output of the “ImageWritable” Hadoop compatible image class was written.

This was followed by the addition of the “ImageRecordWriter” class, which extended the Hadoop's RecordWriter class and was responsible for the actual image data writing. Once again, the Java Image libraries were used to write in the image data and output it to the Hadoop file management system, as was the case with the previous class. After that, the "ImageWritable" class was created to extend the Hadoop's Writable class. Having the ability to copy and read information from and to other workers over a network, Hadoop is a distributed data processing platform. Hadoop's "ImageWritable" class contained all of the image information required for the image to be processed in the cluster. For example, it contained the filename, “BufferedImage” data, and the file format in one package. All of this information was used to read or even create an actual intermediate or final image, depending on the situation.

Following the production and representation of the image splits in key-value format in key-value format via the proposed RecordReader, the map function was applied to each pair of key-value pairs through the mapper class. The key-value pairs served as the foundation of the data structure. These pairs were used as the primary component of the map function in order to initiate the processing. The key contained the file name of the image as well as the ID of the sub-image. In the meantime, the value was created by the sub-image itself. As an intermediary value, the map function generated an Image encryption value, which was then combined with the new keys of the output. The map function also performed the comprehensive task of Image encryption on the image chunks and assigned the outputs of encrypted image chunks to the outputs of the map function. These outputs were merged together using the reduce function. Detailed implementation of the map and reduce functions, along with the involved method, must be discussed in detail. This is particularly important.

It is composed of several components in the system, and these components work together to enable parallel processing in a distributed environment. Hadoop performs the job on behalf of the user. Hadoop's characteristic master slave architecture divides responsibilities between distributed components in a straightforward manner (see Figure 2). Jobs are submitted to the master, who then distributes the tasks and resources to the slave workers who are spread throughout the world. Slaves carry out their tasks and return data to their master, who then assigns them additional tasks if necessary.

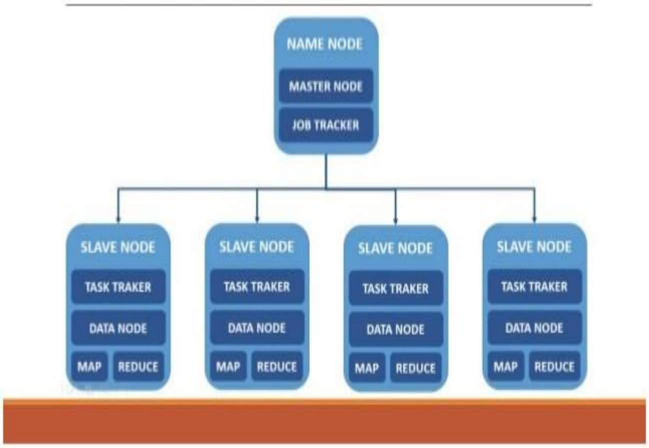

FIGURE 2. **Schematic of Hadoop elements in typical master slave architecture**

These responsibilities can include a number of functions that represent a variety of roles at different times. For example, when the job begins, the master schedules the map tasks on the available workers based on their availability. It is also possible, if specified by the user, for the combined tasks to be executed on mappers immediately following completion of the mapping tasks themselves. Figure 3 illustrates how master schedules the reduce tasks on these workers after all mappers have completed their work. This model was used in the current research for large-scale Image encryption and decryption. The job was submitted to the master node with a directory containing the input images included in the job submission. The map tasks were scheduled by the master on available workers in such a way that each mapper could read in a single image from the specified input directory when the tasks were scheduled.

This was accomplished by partitioning and encrypting/decrypting each chunk of the image according to the input provided by the user through the mapper. After finishing the mapping, the master scheduled all of the reduce jobs on the available workers. Each reducer then reads all of the image partitions that belong to a single image, iterates over them, and merges all of the image pieces to produce the final output image. The utility played a significant role in the overall flow of events. For the most part, the proposed method has the potential to efficiently process a large number of images in parallel on a distributed cluster environment.

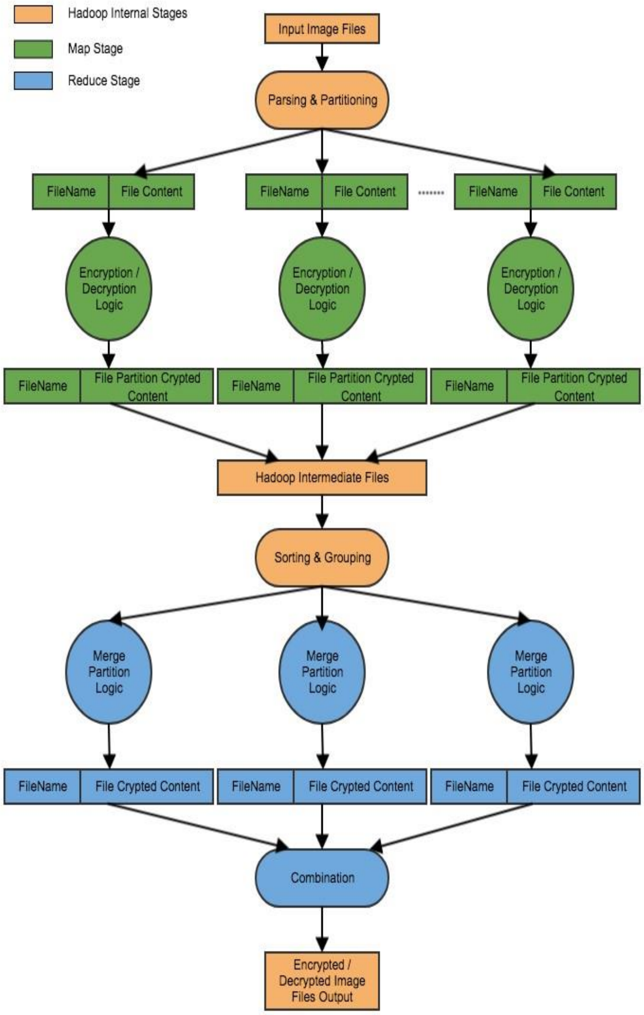

FIGURE 3. **Hadoop Image Processing Encryption Work Flow**

II. ALGORITHM DEVELOPMENT

It is necessary to develop the algorithms in two phases: the mapper and the reducer (see algorithms 1 and 2 for examples). Because of this, the map method is used exclusively for processing the data partitioning represented as pairs of key-values in the parallel approach when the other map processes are not communicated with each other. Aside from the key and value, the map method also returns two additional parameters, which are the Output Collector and the Reporter, which are both optional. The output images were created by the Output Collector using a different key-value format than the input images. Following that, such output images were forwarded to the reduce tasks to be processed. In the meantime, the Reporter provided information about the map-making assignment. The map function proposed in this study first grabbed the key before employing the java image reader classes to like the value for decompression and reading purposes, as demonstrated in the following video. Following that, the image is subjected to the execution of the encryption algorithm. It is explicitly executed from the beginning to the end during the implementation of image encryption algorithms. Using only a few minor adjustments, the chronological execution of the Image encryption algorithms could be easily plugged into the map function and used. After that, the images that were created were compressed and stored in the bytes assortment. Images were first classified as key-value pairs and then forwarded to the reducers for further processing. Following the completion of a map task, the key-value pairs of the completed map task were sent to the reduce workers via the master.

The map function accepted a succession of key-value pairs for processing and generating zero or more such pairs. In this study, the map function was implemented by CryptMapper class. CryptMapper class received ImageWritable object as input. As explained before, the impossibility of input image dividing enforced to consider the entire image as input by the single map function. Algorithm 1 provides a pseudo-code involving the map procedure. The map function performed the following operations on the input image:

- Divided the input image into parts (partition width and height was specified by user)
- Generated the secret key and XOR matrix
- Used the secret key and XORmatrix to encrypt/decrypt all partitions (encryption or decryption was specified by user input)
- Released all partitions with same key i.e. filename. This ensured that all partitions belonging to a single image could end up on same reducer and eventually get merged into a final single image.

Algorithm 1 Mapper

```
procedure CryptMapper(image)
  if image height and width smaller than partition height and
      /* No need to partition */
      img = EncryptChunk(image)
      emit (filename, img) as key/value pair
      else
/* divide image into smaller chunks */
chunkList = splitImageFile(image)
for each chunk in chunkList do
img = EncryptChunk(chunk)
emit (filename, img) as key/value pair
end for
end if
end procedure
procedure EncryptChunk(imgChunk)
Generate SecretKey matrix
Generate Xor matrix
Rearrange imgChunk matrix based on secret key matrix
encryptedChunk = Xor rearranged matrix with Xor matrix keys
return encryptedChunk
end procedure
procedure DecryptChunk(imgChunk)
Generate SecretKey matrix
Generate Xor matrix
Xor imgChunk with Xor matrix keys (0-255)
encryptedChunk = Rearrange xor'ed matrix based on secret key
return decryptedChunk
```

```
28:   end procedure
```

The proposed reducer had a straightforward task, requiring only that the resultant images obtained from the map tasks be merged to produce the final image, which was very straightforward. Using the initialization and termination options in the Hadoop API, the reduce task was started immediately after the completion of each map task, and this was repeated for each map task. As a result, the first step was to determine the final destination for the final aggregated image. It is necessary to apply the information obtained from the original input image in order to produce an output image that is identical in size to the original one in this step. The key-value pairs generated by the mappers were then captured and arranged in a hierarchy based on their key identifier. As a result of this information, the original location of the image chunks within the full image could be determined. Following the verification of the boundary conditions, the value of the image was read, decompressed, and streamed to the position that had been determined. The reduce tasks were run in parallel in order to facilitate the writing of images to the distributed file system. This procedure ensured that all chunks of the image were piled into their corresponding locations within the refined image. As a result, Algorithm 5.2 was able to successfully complete the tasks of the reduce function, including the reading and writing of image files.

Hadoop called the specified reduce function once for each distinct key in the intermediate list of key-value pairs produced by the map function, resulting in a total of ten calls to the reduce function. The reduce function iterated through all of the values associated with the unique key until it arrived at the final output. Later on, the CryptReducer implemented the reduce logic using the Algorithm 5.2, which was later modified. Each reducer was given all of the chunks that were associated with a specific image. The function was then iterated through all of the chunks in order to calculate the appropriate index in the final image, and all of the chunks were merged into the encrypted/decrypted image at the end of the iteration.

Algorithm 2 Reducer

```
procedure CryptReducer(filename, chunksList)
        finalImage placeholder
    for each chunk in chunksList do
        x,y = Calculate index of this chunk in final
        Place chunk at x,y in finalImage
    end for
emit finalImage as output
end procedure
```

The driver program was designed after the MapReduce read the image data format. Eventually, the driver program provided the details on the manner in which the job was executed within the Hadoop environment. Additionally, the program specified the data input/output formats and the exact location of data application for processing. The use of Hadoop API allowed easy setting of the necessitated parameters and other configurations for the execution of a MapReduce job via the application of the JobConf object [40-41]. The job was performed using such driver that in turn allowed input and output paths as well as their formats besides those parameters linked to the map job to be specified. Furthermore, this driver also determined the amount of job needs to be executed by MapReduce. The JAR compression was used to pack the codes wherein this class became the key driver responsible for the job submission, running, and progress reporting.

III. EXPERIMENTAL ANALYSIS

The initial experiments were conducted on the UTM Big Data Center using single node server to verify the capacity of the proposed Image encryption method for running in the Hadoop environment. These experiments were performed on Google Cloud Platform due to its notable advantages including “Google Compute Engine”. The “Hadoop on cloud” symbolized the virtual instances within cloud to set up of Hadoop Clusters to perform the MapReduce jobs. Thus, MapReduce jobs could run with remote services as well as deployment scripts. Google Cloud Dataproc provided a manageable Spark and Hadoop in a speedy, effortless, and cost-effective way. It automatically managed the deployment, logging, and monitoring of the cluster. The cluster was formed rapidly with nodes of various types of virtual machines, large disk spaces, and wide network selections. Thus, it could be resized and pulled down as quickly. Cloud Dataproc being a complete and robust data platform with built-in integration and Cloud Storage added extra benefits to the numerical calculations and data processing.

The initial experiment displays the initial experimental results obtained by running the proposed IE method on Hadoop with successful job submission. These experiments were performed on a server using Cloudera QuickStart CDH 5.12. The program illustrates the successful completion of the jobs using the dataset of 7 BMP images with total size of 8 GB.

Three Hadoop clusters were created on Goggle Cloud equipped with n1-standard-2 Machine type configured to use 2 vCPU and 7.5 GB memory (RAM) for testing the developed Image encryption algorithm. Same machine type was used for each NameNode and DataNode. Table 1 provides the details of cluster hardware configuration. NameNode was utilized for both masternode and datanode with jobtracker and namenode on it for controlling all the respective MapReduce jobs and datanodes (Figure 5.6). Other machines operated as worker nodes with datanode and tasktracker on them. Table 2 summarizes the Hadoop clusters configurations. The three clusters used for testing. Figure 4 depicts the details of the cluster configurations.

TABLE 1
CLUSTER HARDWARE CONFIGURATIONS

Name	Machine type	Details
NameNode	n1-standard-2	2 vCPU, 7.5 GB memory
DataNode	n1-standard-2	2 vCPU, 7.5 GB memory

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2023.3279719

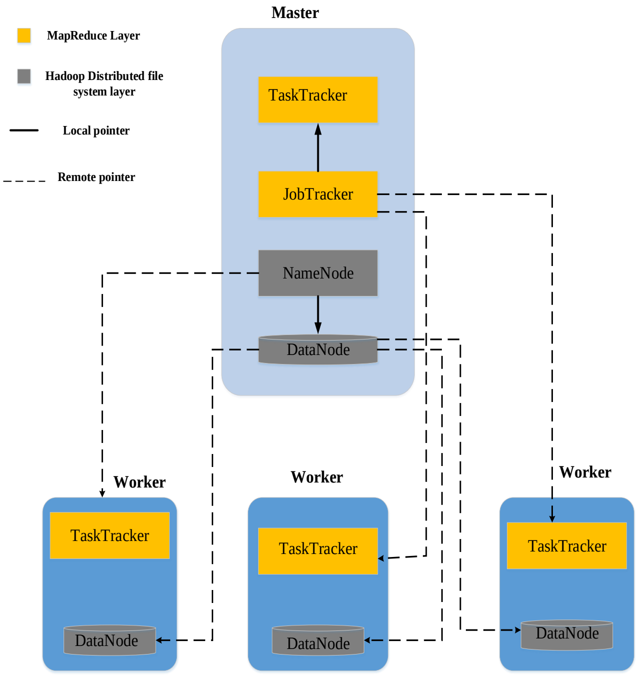

FIGURE 4. **Test environment setup**

TABLE 1
DETAILS OF TEST DATASET USED IN THE EXPERIMENT

Image Name	Resolution	Size
Sample1	3257 ⨯ 2873	26.7 MB
Sample2	3257 ⨯ 5747	53.5 MB
Sample3	6515 ⨯ 5747	107 MB
Sample4	6515 ⨯ 5747	107 MB
Sample5	10859 ⨯ 9579	297 MB
Sample6	10980 ⨯ 10980	344 MB
Sample7	10980 ⨯ 10980	344 MB
Sample8	16288 ⨯ 14368	669 MB
Sample9	16288 ⨯ 14368	669 MB
Sample10	16288 ⨯ 14368	669 MB
Sample11	12128 ⨯ 22400	777 MB
Sample12	24256 ⨯ 14933	1.01 GB
Sample13	19968 ⨯ 18368	1.02 GB
Sample14	19968 ⨯ 18368	1.02 GB
Sample15	18362 ⨯ 20482	1.05 GB

TABLE **2**
HADOOP CLUSTERS CONFIGURATION

Cluster Name	Machine type	Number of DataNodes
Cluster 1	n1-standard-2	0
Cluster 2	n1-standard-2	2
Cluster 3	n1-standard-2	3

This study used the satellite images obtained from Earth Explorer portal of United States Geological Survey (USGS) (URL:http://earthexplorer.usgs.gov/). Figure 5 displays the dataset sample image. Table 3 presents the employed test dataset for conducting the experiment. These images were forwarded to the HDFS together with corresponding metadata file to successfully run the MapReduce applications. Since the image database was presumed to be kept utilizing the HDFS, the input file transfer was not included for the performance evaluation.

FIGURE 5. **Typical dataset Sample7 image (taken from USGS database)**

.

IV. PRESENTATION OF RESULTS AND DISCUSSION

The performance of the proposed Image encryption method was evaluated in the context of the MapReduce programming framework model in order to determine the feasibility of effective encryption for large images in the future. It is important to note that the presence of Image encryption on the MapReduce framework has no effect on the quality of the final product. This section is divided into four broad sub-sections for convenience. On the first page, you will find a detailed description of the hardware and software configurations for the test setting, which will be used to run the prototype framework and applications. Following that, the characteristics of the test datasets that were used are highlighted. The third section included a description of the qualitative evaluation of the output images, which was divided into two parts. Finally, in the concluding section, we discussed the performance evaluation of the proposed Image Encryption scheme as well as the benchmarking of the results of the tests.

These included the size of the satellite image, the number of worker nodes who participated, and the number of map and reduce jobs that were completed. The execution time of a job that was ordered by the master node was the primary metric used in the context of evaluating the computational performance of the system. The performance of the Image Encryption method was experimentally evaluated in terms of the frequency with which jobs were completed. The parameters used to evaluate the performance were as follows:

It was possible to track job history and progression, as well as job completion time, as well as unsuccessful tasks and other job statistics, thanks to a Hadoop-provided integrated multi-paged user interface. Figure 6 illustrates an example of a typical Jobtracker page within the Hadoop web-based interface. The job accomplishment time indicated the total amount of time spent by the worker nodes performing their tasks, which included processing the input images and transferring the results to the master node, as well as the total amount of time spent by the master node processing the results. Aside from that, and taking into account the fact that the nodes are connected via WAN during the experiment, the day that the job was performed to examine the effect on computation time was taken into account. Therefore, each job order was executed multiple times, for a total of more than ten times, in order to mitigate the impact of network latency and traffic on the overall performance. In order to capture the overall performance of the algorithms, the average completion time of the algorithms was determined for each algorithm. Furthermore, in order to confirm the findings, the exploration of peak performance with these extremely heterogeneous and computing limited nodes was carried out. It was discovered that the range of sound performance was quite broad in this case.

The sequential execution of applications of the Sobel, Laplacian, and Canny algorithms was accomplished through the use of Java programming techniques. A control case is created in order to evaluate the performance of MapReduce implementations. This is done in order to obtain a control case that will be used in the evaluation. A single-node performance demonstration provided evidence of the sequential application of the algorithms. The MapReduce programming model was evaluated using four (4) performance metrics, which were determined in the following way: The number of nodes and the size of the data were the metrics that were used for the evaluation, which will be explained in greater detail in the subsequent sections.

.

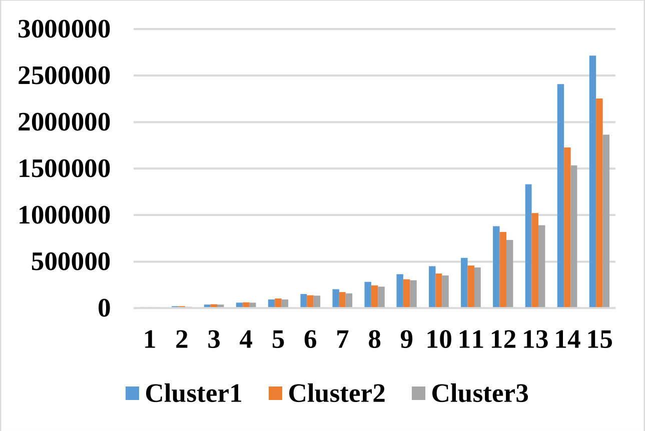

FIGURE 6 **CPU time spent by different cluster to process the image**

In order to evaluate the efficiency of the MapReduce processes in terms of the amount of time required for calculation, the total image size was maintained at its original resolution while the number of cluster nodes was increased. After determining the amount of time required for the computation on a single computer, the number of computers in the system was increased until it reached a maximum of 4 nodes.

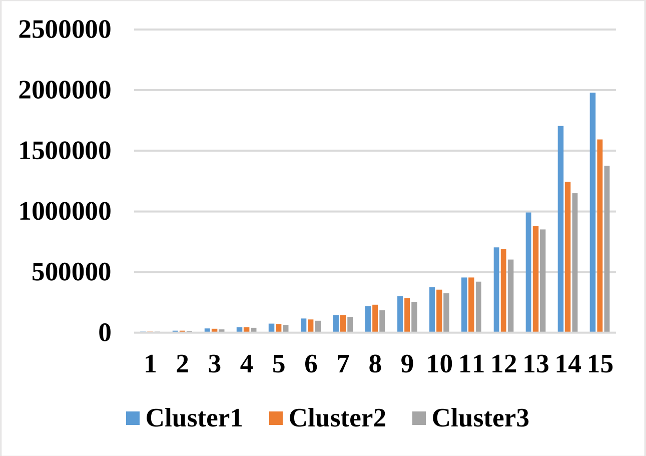

FIGURE 7 Total time spent by all map tasks for image processing

For the purpose of determining the average time, the experiment was repeated ten times in each cluster. Figures 7 and 8 show, respectively, the amount of time that was spent by all of the map tasks (ms), and the amount of time that was spent by all of the reduce jobs (ms). The processing of those photos was done in parallel using a single MapReduce task. The findings made it abundantly evident that the amount of processing time required by the MapReduce implementation was drastically cut down as the number of processors increased.

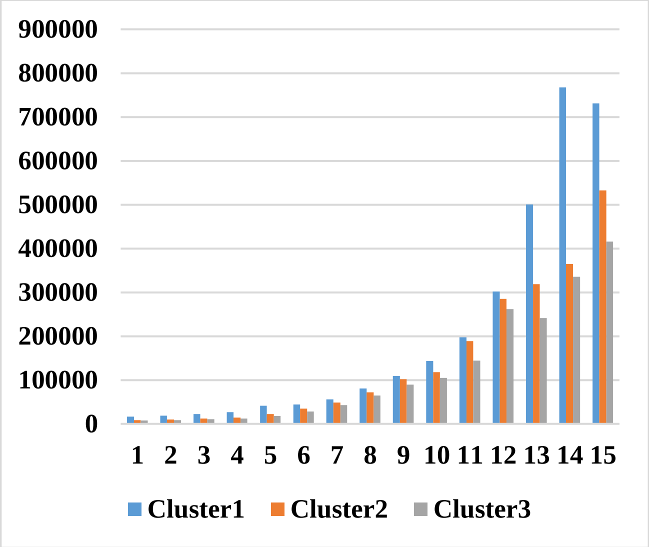

FIGURE 8 Total time spent by all reduce tasks to process the image

Calculating the speedup and efficiency of the suggested picture encryption system allowed for the determination of the impact that all cluster, map task, and reduction task had on the amount of time required to process the image. The term "speedup" refers to the reduction in the ratio of the amount of time required for the sequential execution on a single computer to the amount of time required for the distributed execution on several machines. Utilization of computers within the system was the yardstick by which efficiency was measured. The expression for and are given by:

$$Speedup\ (n) = \frac{Time\ on\ Single\ Computer}{Time\ on\ n\ Computers} \qquad (1)$$

$$Efficiency = \frac{Speedup\ (n)}{n} \qquad (2)$$

In order to define the speedup parameter, the total number of computational nodes that were utilised in the Hadoop implementation was taken into consideration. The change in speedup and efficiency as a function of the number of nodes used by the Image encryption method is shown in Figure 9 and Figure 10, respectively. It was discovered that increasing the number of nodes in the Hadoop Cluster might produce a significant speedup.

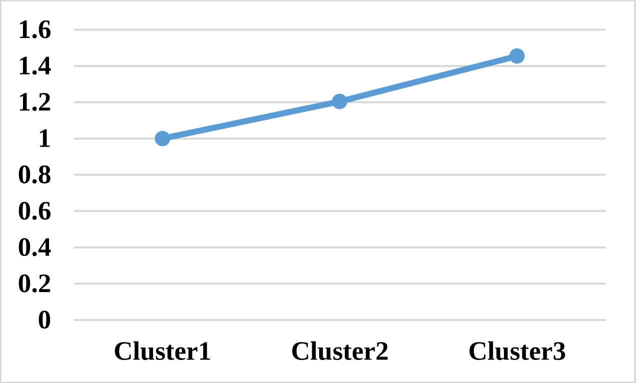

FIGURE 9. Variation of speedup as a function of number of nodes

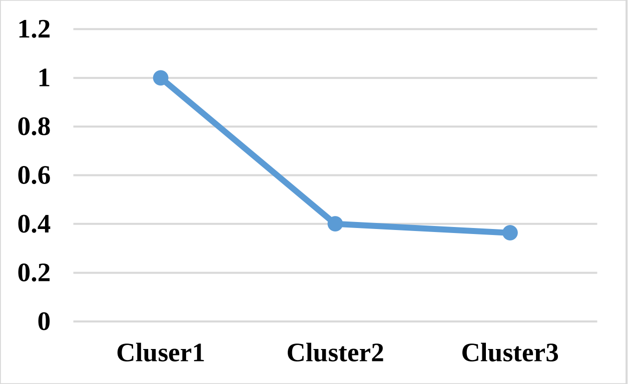

FIGURE 10. Variation of performance efficiency as a function of number of nodes

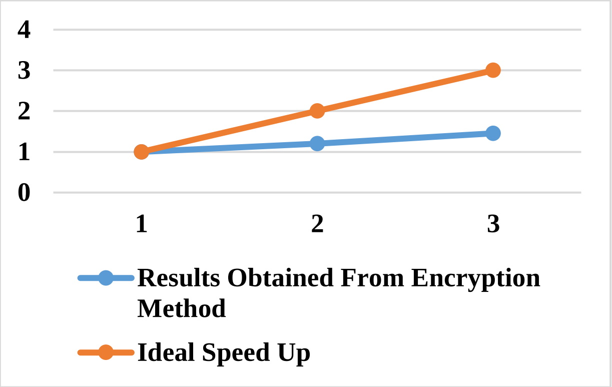

FIGURE 11Comparison of obtained performance speedup with ideal one

In the ideal case the speedup was found to enhance linearly with the increase in computing processors number (Figure 11). However, it was hard to attain the same speedup for the practical one because the cost was higher for larger number of processors, processes with efficiency above 0.5 (50%) are regarded to achieve good performance. Briefly, to attain improved performance the number of computing nodes must be higher.

To examine the effects of data size on the MapReduce programming model very large satellite images dataset were implement. The scale-up properties was evaluated in term of computational time. The data size and number of clusters were enhanced by the same fold. The idea of scale-up was to maintain the amount of job on every computing cluster fixed. Following the scale-up approach, the blockages of the distributed system (HDFS) was inspected. In this approach each cluster processed the same amount of data, where the influence of load balancing problem as well as inefficiencies caused by the distributed method was circumvented. This metric provided a good evaluation measure of the MapReduce implementation to manage data of varying sizes. Ideally, the graph of scale-up exhibits a linear behavior without any impact of varying data sizes on the computation time. Present experiment began with the data size of 669 MB megabytes and the size was further increased by adding 669 MB at each time to test another cluster. Figure 12 shows the performance results of the proposed Image encryption method under scale-up approach.

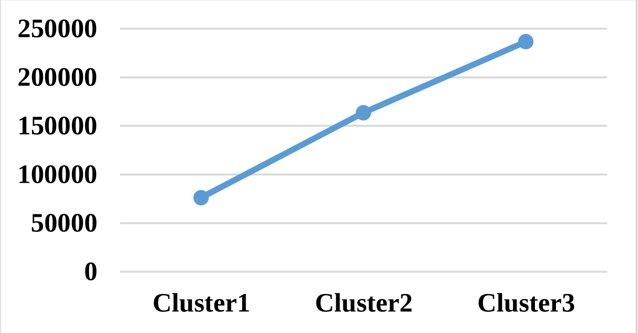

FIGURE 12. Data size dependent variation of performance of the proposed IE method under scale-up approach

Figure 12 depicts the variance in the amount of computational time that is required by the suggested IE approach when it is applied to a large dataset. This variation is depending on the size of the data. It was possible to determine the size up effect on the suggested IE system by maintaining the same number of computing nodes but increasing the data size by some factor. The computing performance of every cluster was evaluated by repeatedly doubling the dataset, starting with a value of 669 MB and ending with a value of 2007 MB. The formula for sizeup is given by:

$$Sizeup = \frac{Time\ of\ Increased\ Data\ Size}{Time\ of\ Original\ Data\ Size} \qquad (3)$$

Figure 13 demonstrates very clearly that the amount of time required to compute the proposed IE increased in a linear fashion with the amount of data that was being processed. In addition, it was fair to increase the size of all of the nodes in order to increase the communication cost.

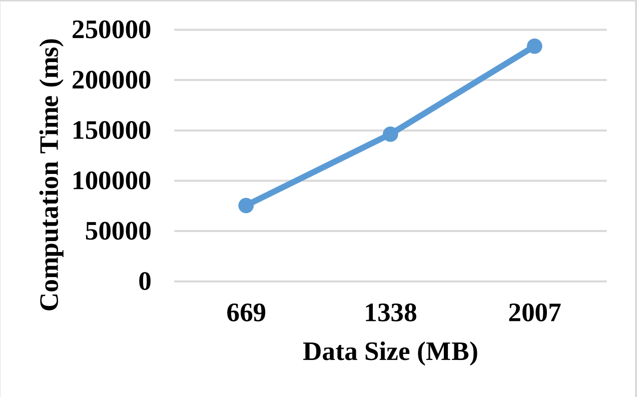

FIGURE 13. Variation of computational time as a function of data size

The output efficiency of the created image encryption method was used to evaluate the quality of the final product that was produced by the method. The following metrics, which were determined based on a number of different analyses, are presented below. The graphical depiction of the intensity frequency found in an image is referred to as a histogram. Histograms can be thought of as either an image property or information associated with the image when dealing with plain images. Histograms present an opportunity for cryptanalysis to launch an assault on the encrypted image. Therefore, the encryption approach was utilised to change the pixel values in order to alter the histogram in a manner that would prevent statistical analysis from being performed. After the encryption procedure, the image histograms should have the same shape in order to prevent cryptanalysis from taking advantage of any differences.

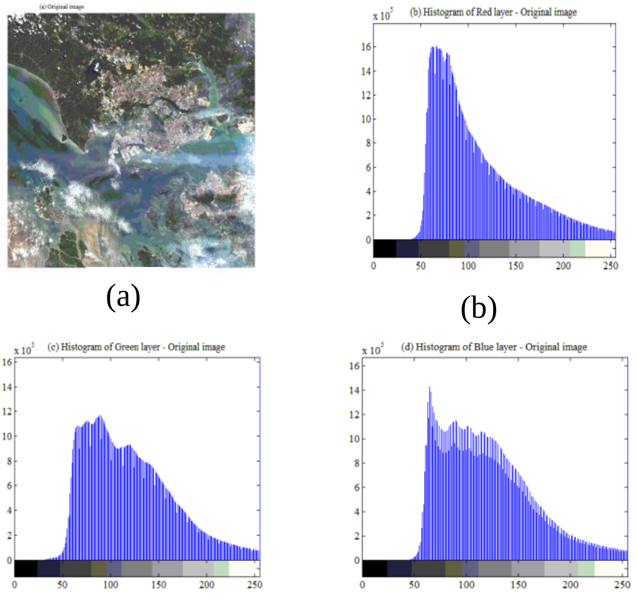

FIGURE 14. (a) Original image, and the histograms of (b) red, (c) green, and (d) blue color components

(c) (d)

Following the use of the suggested approach for encrypting photographs, the histograms of the unencrypted and encrypted versions of the images are presented in Figures 14 and 15, respectively. It was discovered that the histograms for the encrypted image that were generated were of a linear-like consistency and had a uniform appearance.

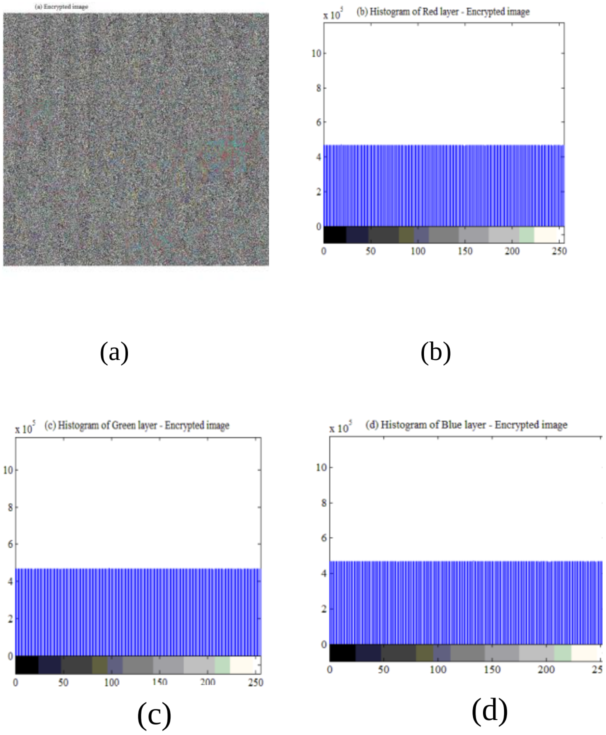

FIGURE 15. (a) Encrypted image, and the histograms of (b) red, (c) green, and (d) blue color components

With the help of the information entropy parameter, it is possible to express how much uncertainty there is in a system, which can be measured by the degree of uncertainty in the system. When it comes to some systems, entropy is defined as the degree to which the system is subject to randomness or disorder. Therefore, when evaluating image encryption systems, information entropy is an important parameter to take into consideration. In ciphered pictures, the information entropy (or the distribution of colours) indicates how evenly the colours are distributed throughout the image. A good ciphered picture is one in which the colour values (or grey values in grayscale pictures) are evenly distributed throughout the image. In this particular instance, the ideal entropy value is equal to 8. This table provides a summary of the ideal entropy values for all images in the dataset that were obtained by employing the proposed Image encryption method.

TABLE 3
ENTROPY RESULTS ACHIEVED USING THE PROPOSED IMAGE ENCRYPTION METHOD

Image Name	Resolution	Size	Entropy
Sample1	3257 × 2873	26.7 MB	8.0000
Sample2	3257 × 5747	53.5 MB	8.0000
Sample3	6515 × 5747	107 MB	8.0000
Sample4	6515 × 5747	107 MB	8.0000
Sample5	10859 × 9579	297 MB	8.0000
Sample6	10980 × 10980	344 MB	8.0000
Sample7	10980 × 10980	344 MB	8.0000
Sample8	16288 × 14368	669 MB	8.0000
Sample9	16288 × 14368	669 MB	8.0000
Sample10	16288 × 14368	669 MB	8.0000
Sample11	12128 × 22400	777 MB	8.0000
Sample12	24256 × 14933	1.01 GB	8.0000
Sample13	19968 × 18368	1.02 GB	8.0000
Sample14	19968 × 18368	1.02 GB	8.0000
Sample15	18362 × 20482	1.05 GB	8.0000

.

X. CONCLUSION

Encryption is one of the most extensively utilised methods now accessible, and it can be found in a wide variety of applications. This is because encryption may safeguard satellite imagery while it is being transported across transmission networks. By utilising various forms of encryption, it is possible to safeguard satellite imagery both while it is being transmitted and while it is being kept on the hard drive of a computer. As a result, the performance of the suggested Image Encryption system is analysed and contrasted with that of a variety of other recently developed systems in this work. The evaluation of the newly designed Image Encryption system was complicated and required the use of several calculations in order to evaluate whether or not it was effective. This was due to the fact that there is no clear definition or aspect for randomness. Multiple approaches were utilised to ensure that the Image Encryption system that was proposed excelled its rivals in terms of the numerous qualities that define an effective cryptosystem. The proposed Image Encryption method was evaluated with regard to random number keys, key security, statistical features, the quality of the ciphered image, and the level of resistance it possessed against differential attacks. The performance of the suggested Image Encryption method was compared to that of several previous approaches, and the results demonstrated that the created method is a significantly better cryptosystem with a large number of differentiating characteristics. Following the completion of an investigation, it was found that the Image Encryption system that was provided is superior in all aspects of the evaluation and attains perfect outcomes in terms of both the histogram and the information entropy. In a nutshell, a strong Internet Explorer system was designed that offered comprehensive protection and resistance against a variety of different assaults that were known at the time. The findings indicate that it was difficult to get the same speedup for the practical encryption since the cost increased with the number of processors. Procedures whose efficiency is greater than 0.5 (50%) are considered to achieve good performance.

REFERENCES

[1] Alkhelaiwi, M., Boulila, W., Ahmad, J., Koubaa, A., & Driss, M. (2021). An efficient approach based on privacy-preserving deep learning for satellite image classification. Remote Sensing, 13(11), 2221.

[2] Patra, A., Saha, A., Chakraborty, D., & Bhattacharya, K. (2021). Compression of high-resolution satellite images using optical image processing. In Satellite Systems—Design, Modeling, Simulation and Analysis . IntechOpen.

[3] B. Vaseghi, S.S. Hashemi, S. Mobayen, and A. Fekih. Finite time chaos synchronization in time-delay channel and its application to satellite image encryption in OFDM communication systems. IEEE Access, 9, pp.21332-21344, 2021.

[4] I.A.T. Hashem, I. Yaqoob, N.B. Anuar, S. Mokhtar, A. Gani, and S.U. Khan. The rise of "big data" on cloud computing: Review and open research issues. Information systems, 47, pp.98-115, 2015.

[5] X. Di, J. Li, H. Qi, L. Cong, and H. Yang. "A semi-symmetric image encryption scheme based on the function projective synchronization of two hyperchaotic systems." PloS one 12, no. 9 (2017): e0184586.

[6] E.S. Hureib, and A.A. Gutub. "Enhancing medical data security via combining elliptic curve cryptography and image steganography." Int. J. Comput. Sci. Netw. Secur.(IJCSNS) 20, no. 8 (2020): 1-8.

[7] J.N. Gaithuru, M. Bakhtiari, M. Salleh, and A.M. Muteb. A comprehensive literature review of asymmetric key cryptography algorithms for establishment of the existing gap. In 2015 9th Malaysian Software Engineering Conference (MySEC) (pp. 236-244), 2015.

[8] Lai, Q., Hu, G., Erkan, U., & Toktas, A. "A novel pixel-split image encryption scheme based on 2D Salomon map." Expert Systems with Applications 213 (2023): 118845.

[9] Erkan, U., Toktas, A., & Lai, Q. "2D hyperchaotic system based on Schaffer function for image encryption." Expert Systems with Applications 213 (2023): 119076.

[10] Lai, Q., Hu, G., Erkan, U., & Toktas, A. "High-efficiency medical image encryption method based on 2D Logistic-Gaussian hyperchaotic map." Applied Mathematics and Computation 442 (2023): 127738.

[11] Song, W., Fu, C., Zheng, Y., Tie, M., Liu, J., & Chen, J. "A parallel image encryption algorithm using intra bitplane scrambling." Mathematics and Computers in Simulation 204 (2023): 71-88.

[12] Huang, X., Dong, Y., Ye, G., & Shi, Y. "Meaningful image encryption algorithm based on compressive sensing and integer wavelet transform." Frontiers of Computer Science 17.3 (2023): 173804.

[13] A. Vizitiu, C.I. Niţă, A. Puiu, C. Suciu, & L.M. Itu. Applying deep neural networks over homomorphic encrypted medical data. Computational and mathematical methods in medicine, 2020.

[14] L. You, E. Yang, & G. Wang. A novel parallel image encryption algorithm based on hybrid chaotic maps with OpenCL implementation. Soft Computing, 1-15, 2020.

[15] M. Jarrold, & D. Meltzer. Small Satellites and Innovations in Terminal and Teleport Design, Deployment, and Operation. Handbook of Small Satellites: Technology, Design, Manufacture, Applications, Economics and Regulation, pp. 599-618, 2020.

[16] M. Rana, Q. Mamun, & R. Islam. Current Lightweight Cryptography Protocols in Smart City IoT Networks: A Survey. arXiv preprint arXiv:2010.00852, 2020.

[17] E. Bensikaddour, & Y. Bentoutou. Satellite Image Encryption Based on AES and Discretised Chaotic Maps. Automatic Control and Computer Sciences, vol. 54, 5, pp. 446-455, 2020.

[18] L. Wang, C. Diao, G. Xian, D. Yin, Y. Lu, S. Zou, and T.A. Erickson. "A summary of the special issue on remote sensing of land change science with Google earth engine." (2020): 112002.

[19] P. Chinnasamy, & A. Parikh. Remote sensing-based assessment of Coastal Regulation Zones in India: a case study of Mumbai, India. Environment, Development and Sustainability, vol.23, no.5, pp.7931-7950, 2021.

[20] M.E. O'Hanlon. 4. Space, Missile Defense, and Nuclear Weapons: Three Case Studies in the Science of War. In Defense 101 (pp. 162-200). Cornell University Press, 2021.

[21] S.Y. Lee, C. Du, Z. Chen, H. Wu, K. Guan, Y. Liu, W. Liao. Assessing safety and suitability of old trails for hiking using ground and drone surveys. ISPRS International Journal of Geo-Information, vol. 9, no. 4, pp. 221, 2020.

[22] S. Salcedo-Sanz, P. Ghamisi, M. Piles, M. Werner, L. Cuadra, A. Moreno-Martínez, & G. Camps-Valls. Machine learning information fusion in Earth observation: A comprehensive review of methods, applications and data sources. Information Fusion, vol. 63, pp.256-272, 2020.

[23] T. Li, L. Wang, R. Chen, W. Fu, B. Xu, P. Jiang, & Y. Han. Refining the empirical global pressure and temperature model with the ERA5 reanalysis and radiosonde data. Journal of Geodesy, 95(3), 1-17, 2021.

[24] F. T. B. Muhaya, "Chaotic and AES cryptosystem for satellite imagery", Telecommunication Systems, vol. 52, no. 2, pp. 573-581, 2013.

[25] A. Padmapriya and M. S. Benazir, "Elementary Matrix operation based Satellite image encryption", International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), vol. 4, no. 7, pp. 367-371, 2015.

[26] M. Naim, A.A. Pacha, & C. Serief. A novel satellite image encryption algorithm based on hyperchaotic systems and Josephus problem. Advances in Space Research, vol. 67, 7, pp. 2077-2103, 2021.

[27] Nagasree, Y., Rupa, C., Akshitha, P., Srivastava, G., Gadekallu, T. R., & Lakshmanna, K. "Preserving Privacy of Classified Authentic Satellite Lane Imagery Using Proxy Re-Encryption and UAV Technologies." Drones 7.1 (2023): 53.

[28] Panwar, K., Singh, A., Kukreja, S., Singh, K. K., Shakhovska, N., & Boichuk, A. "Encipher GAN: An End-to-End Color Image Encryption System Using a Deep Generative Model." Systems 11.1 (2023): 36.

[29] Sirisha, U., & Chandana, B. S."Privacy preserving image encryption with optimal deep transfer learning based accident severity classification model." Sensors 23.1 (2023): 519.

[30] Mukherjee, S. "New Insights Into Chaos Based Image Encryption & Its Application." Journal of Mathematical Sciences & Computational Mathematics 4.2 (2023): 241-264.

[31] Al-Khasawneh, M. A., Uddin, I., Shah, S. A. A., Khasawneh, A. M., Abualigah, L., & Mahmoud, M. "An improved chaotic image encryption algorithm using Hadoop-based MapReduce framework for massive remote sensed images in parallel IoT applications." Cluster Computing (2022): 1-15.

[32] Elkandoz, M. T., & Alexan, W. "Image encryption based on a combination of multiple chaotic maps." Multimedia Tools and Applications 81.18 (2022): 25497-25518.

[33] Boulila, W., Khlifi, M. K., Ammar, A., Koubaa, A., Benjdira, B., & Farah, I. R. "A Hybrid Privacy-Preserving Deep Learning Approach for Object Classification in Very High-Resolution Satellite Images." Remote Sensing 14.18 (2022): 4631.

[34] Shankar, J., & Nandini, C. "Hybrid Hyper Chaotic Map with LSB for Image Encryption and Decryption." Scalable Computing: Practice and Experience 23.4 (2022): 181-192.

[35] A. Kumar, & M. Dua. Novel pseudo random key & cosine transformed chaotic maps based satellite image encryption. Multimedia Tools and Applications, 1-21, 2021.

[36] E.H. Bensikaddour, Y. Bentoutou, & N. Taleb. Embedded implementation of multispectral satellite image encryption using a chaos-based block cipher. Journal of King Saud University-Computer and Information Sciences, vol. 32, 1, pp. 50-56, 2020.

[37] A. Alabdulatif, I. Khalil, & X. Yi. Towards secure big data analytic for cloud-enabled applications with fully homomorphic encryption. Journal of Parallel and Distributed Computing, vol. 137, pp. 192-204, 2020.

[38] G. George, & A.J. Bock. The business model in practice and its implications for entrepreneurship research. Entrepreneurship theory and practice, vol. 35, no. 1, pp. 83-111, 2011.

[39] Xian, Y., & Wang, X. (2021). Fractal sorting matrix and its application on chaotic image encryption. Information Sciences, 547, 1154-1169.

[40] R.A. McLaren & T.J. Kennie. Visualisation of digital terrain models: techniques and applications. Three Dimensional Applications in Geographical Information Systems, pp. 79-98, 2020.

[41] M.L. Larsen & C.K. Blouin. Refinements to Data Acquired by 2-Dimensional Video Disdrometers. Atmosphere, vol. 11, 8, pp. 855, 2020.

[42] W. Shafik, M. Matinkhah, M. Asadi, Z. Ahmadi, & Z. Hadiyan. A study on internet of things performance evaluation. Journal of Communications Technology, Electronics and Computer Science, vol. 28, pp. 1-19, 2020.

ACKNOWLEDGEMENT

This research was supported in part by the University of Vaasa, and in part by the Academy of Finland.

AUTHORS INFORMATION

Mahmoud A. S. Al-Khasawneh received the B.Sc. degree in computer science from Yarmouk University, Jordan, in 2003, and the M.Sc. and PhD degrees in computer science from (UTM) Universiti Teknologi Malaysia, Johor, Malaysia, in 2013, and 2018 respectively. He is currently an Assistant Professor in the Faculty of Computing Skyline University College, Sharjah UAE. His research interests include Security, image encryption, Wireless Networks, Blockchain, the Internet of things, and Big Data.

M. Faheem received the B.Sc. Computer Engineering degree in 2010 from the Department of Computer Engineering at the University College of Engineering & Technology, Bahauddin Zakariya University Multan, Pakistan. In 2012, he received an MS degree in Computer Science from the Faculty of Computer Science and Information Systems at Universiti Teknologi Malaysia. He received his Ph.D. in Computer Science from the Faculty of Engineering, Universiti Teknologi Malaysia, Johor Bahru, Malaysia in 2021. Previously, he served as a lecturer at Comsats Institute of Information & Technology from 2012 to 2014, in Pakistan. Also, he worked as an Assistant Professor in the Department of Computer Engineering at Abdullah Gul University from 2014 to 2022, in Turkey. Currently, he is a Sr. researcher in the School of Computing (Innovations and Technology) at the University of Vaasa, Vaasa, Finland. His research interest includes the areas of Cybersecurity, Blockchain, Smart Grid, Smart Cities, and Industry 4.0. Dr. Faheem has authored several papers in refereed journals and conferences, and served as a reviewer for numerous journals in IEEE, Elsevier, Springer, Willey, Hindawi, and MDPI.

Dr. Eman A. Aldhahri is currently an assistant professor in the Department of Software Engineering, College of Computer Science and Engineering, University of Jeddah, Jeddah, Saudi Arabia. Her research interest includes the areas of Cybersecurity and Blockchain. Dr. Eman has authored several papers in refereed journals and conferences, and served as a reviewer for numerous journals in IEEE and Willey.

Abdulrahman Alzahrani is an assistance professor in the College of Computer Science and Engineering at university of Jeddah, Kingdom of Saudi Arabia. Alzahrani’s research is focused on the field of information technology and innovations in the area of Data Science, Machine Learning, and Health Informatics. He dissertation titled “Exposome Factors: published his Exploratory Study Approach and the Role of Persuasive Technology to Raise Awareness About the Exposome Concept” in 2020. Abdulrahman teaches several courses to bachelor and master students. He teaches E Commerce, Data Visualization classes. He is currently holding the head of academic and exam affair unit within the college.

Ala Abdusalam Alarood received the bachelor’s and master’s degrees from Yarmok University, Jordan, and the Ph.D. degree from the University of Technology, Malaysia, all in computer science. He is currently an Assistant Professor of computer science with the Faculty of Computer and Information Technology, University of Jed-dah. His current research interest is in the area of information security, networks security, steganalysis, machine learning, and neural network.