**Medical Image Encryption by Content-Aware DNA Computing for Secure Healthcare**

1. **INTRODUCTION**

WITH the involvement of cloud computing and Internet of Medical Things, the healthcare system has been greatly progressed by improving clinical treatment experiment and reducing patients’ cost. However, healthcare is an attractive target for cybercrime, due to its high value and weak defences. Researchers generally define information lost, stolen, displaced, hacked, or communicated without unofficial recipients as a cybersecurity breach of healthcare information. It was reported in that about 94% of healthcare organizations have experienced at least one cyberattack and 150 million patient health records have been breached between 2009 and 2014. Among cybersecure solutions for different kinds of cyber threats, we aim to improve security under cryptographic attack, which is carried out with the intention of revealing information that has been encrypted. Cloud-based healthcare systems could transmit patients’ large size medical images at an ease of expendability and mobility. However, problems of privacy disclosure, copyright flouting, illegal redistribution, and identity theft arise even with the encryption process, since healthcare data are substantially valuable estimated as over 1000 dollars per patient. Aiming to guarantee the security of medical images in the transfer process, an efficient and reliable image encryption system is required. Famous methods, such as rivest, shamir, adleman (RSA), data encryption standard (DES), advanced encryption standard (AES), and international data encryption algorithm (IDEA), are widely used to protect text structure data by regarding images as common high-dimensional data. Since medical images own unique characteristics of strong correlation between adjacent pixels with high redundancy, we prefer special designed image encryption methods. Due to the promising properties of high speed, parallelism computation, minimal storage, and unbreakable cryptosystms, deoxyribonucleic acid (DNA) computing is adopted to encrypt medical images in the transfer process other than common methods, such as chaos, elliptic curve cryptography (ECC), etc. Following the idea of DNA computing for encryption, a content-aware DNA computing for medical image encryption is put forward in this article, designed with the following three goals for realization. 1) Consistent and High-Capacity Workflow for Medical Image Encryption: Generally, healthcare professionals prefer smooth workflow with real-time and consistent response to reach the final conclusion in clinical diagnosis. If adopting complicated and annoying image encryption workflow, professionals might resist encryption in transfer process to improve security. 2) Secure Ability Using Random Mechanism and Image Data for High Complexity: Random mechanism is widely recognized as a means to increase difficulty of cracking. Meanwhile, images themselves recognized as highdimensional data can be a natural source to bring high complexity. How to encode both source of complexity still remains an open question. 3) Content-Aware Encryption for Medical Images: Due to properties of taken devices, medical images generally own strong correlation between adjacent pixels with high redundancy. In other words, local and neighboring pixels share the characteristics of naturally and smoothly varying. How to link image content and encryption process for higher complexity becomes our focus in this article. Inspired by these ideas, highlights of the proposed work are listed as follows: 1) Building on DNA encoding and permutation, the proposed method not only involves its high-speed and parallelism computation for real-time performance, but also utilizes the minimal storage property to guarantee highcapacity ability with small cost. 2) We introduce a randomly DNA encoding module to build random mappings between image pixels and computing and a content-aware permutation and diffusion module to construct a content-related permutation sequence, where both modules greatly improve secure ability. 3) Inspired by neighboring characteristics of medical image pixels, the content-aware DNA permutation and diffusion module reorganizes the transmitting data structure by highly nonlinear functions for higher difficulty in cracking, which originate from the correlation relationship of pixels and patches in medical images.

**1.1 Objective of the project:**

There exists a rising concern on security of healthcare data and service. Even small lost, stolen, displaced, hacked, or communicated in personal health data could bring huge damage to patients. Therefore, we propose a novel content-aware deoxyribonucleic acid (DNA) computing system to encrypt medical images, thus guaranteeing privacy and promoting secure healthcare environment. The proposed system consists of sender and receiver to perform tasks of encryption and decryption, respectively, where both contain the same structure design, but perform opposite operations. In either sender or receiver, we design a randomly DNA encoding and a content-aware permutation and diffusion module. Considering introducing random mechanism to increase difficulty of cracking, the former module builds a random encryption rule selector in DNA encoding process by randomly mapping quantity of medical image pixels to outputs. Meanwhile, the latter module constructs a permutation sequence, which not only encodes information of pixel values, but also involves redundant correlation between adjacent pixels located in a patch. Such design brings awareness property of medical image content to greatly increase complexity in cracking by embedding semantically information for encryption. We demonstrate that the proposed system successfully improve cybersecurity of medical images against various attacks in robustness and effectiveness when transmitting data in wireless broadcasting scenarios.

**2. LITERATURE SURVEY:**

**“Transforming healthcare cybersecurity from reactive to proactive: Current status and future recommendations,”**

The recent rise in cybersecurity breaches in healthcare organizations has put patients' privacy at a higher risk of being exposed. Despite this threat and the additional danger posed by such incidents to patients' safety, as well as operational and financial threats to healthcare organizations, very few studies have systematically examined the cybersecurity threats in healthcare. To lay a firm foundation for healthcare organizations and policymakers in better understanding the complexity of the issue of cybersecurity, this study explores the major type of cybersecurity threats for healthcare organizations and explains the roles of the four major players (cyber attackers, cyber defenders, developers, and end-users) in cybersecurity. Finally, the paper discusses a set of recommendations for the policymakers and healthcare organizations to strengthen cybersecurity in their organization.

**“An image encryption scheme based on hybridizing digital chaos and finite state machine,”**

Image [encryption](https://www.sciencedirect.com/topics/engineering/cryptography) protects visual information by transforming images into an incomprehensible form. Chaotic systems are used to design image ciphers due to properties such as [ergodicity](https://www.sciencedirect.com/topics/engineering/ergodicity" \o "Learn more about ergodicity from ScienceDirect's AI-generated Topic Pages) and initial condition sensitivity. A chaos-based cipher derives its security strength from its underlying digital chaotic map, thus a more complex map leads to higher security. This paper introduces an enhancement to a tent map’s chaotic properties by hybridizing it with a deterministic finite state machine. We denote the resulting digital one-dimensional chaotic system as TM-DFSM. Chaotic analyses indicate that the new chaotic system has higher nonlinearity, sensitivity to initial condition, and larger chaotic parameter range than other recently proposed one-dimensional chaotic systems. We then propose a new image [encryption scheme](https://www.sciencedirect.com/topics/computer-science/encryption-scheme) based on TM-DFSM, capable of performing both confusion and diffusion operations in one pass while also having a flexible key space. The encryption operations are designed to achieve maximal confusion and diffusion properties. Changing a single bit of the plainimage or secret key will result in an entirely different cipherimage. The proposed cipher has been analyzed using [histogram analysis](https://www.sciencedirect.com/topics/computer-science/histogram-analysis), contrast analysis, local [Shannon entropy](https://www.sciencedirect.com/topics/computer-science/shannon-entropy), resistance against [differential cryptanalysis](https://www.sciencedirect.com/topics/computer-science/differential-cryptanalysis), and key security. Performance comparison with other recent schemes also depicts the proposed cipher’s superiority.

**“Robust image encryption and zero-watermarking scheme using SCA and modified logistic map,”**

In this work, we first present a modified version of the traditional logistic chaotic map. The proposed version contains an additional parameter that is used to increase the security level of the proposed digital image [copyright protection](https://www.sciencedirect.com/topics/engineering/copyright-protection) scheme. The latter merges two methods of image copyright protection, namely the image zero-watermarking and image [encryption](https://www.sciencedirect.com/topics/engineering/cryptography), which provides a high level of security when communicating images via the Internet. Next, we discuss the influence of [geometric attacks](https://www.sciencedirect.com/topics/engineering/geometric-attack) on the efficiency of the proposed scheme, and then we introduce an efficient solution that can resist such attacks. The proposed solution involves the use of Sine Cosine Algorithm (SCA) with an appropriate algorithm suitable for the correction of geometric attacks (image translation, orientation and its combination) applied to the encrypted image. On the one hand, the simulation results show that the proposed scheme provides a high level of security and can resist various attacks (differential, common image processing, geometric, etc.). On the other hand, the conducted comparison in terms of robustness against geometric attacks clearly demonstrates the superiority of our scheme over recent image encryption ones.

**“IEPSBP: A cost-efficient image encryption algorithm based on parallel chaotic system for green IoT,”**

With the fast development of the Internet of Things (IoT) technologies, more IoT devices are currently connected with the Internet, resulting in more exchange of information. However, data privacy and security threats have become emerging challenges of the IoT. In this paper, we are concerned about the security of image transmission in green IoT. Image encryption algorithms for green IoT are faced with two challenges: 1) To save the cost, devices in green Internet of Things (IoT) always have very low computing power, so they cannot support high-precision computing and 2) The algorithm deployed on the device should be efficient to guarantee the device to run for a long time. To solve the problem of precision limited, we propose a parallel chaotic system named PSBP. 1 The PSBP is composed of Piecewise Linear Chaotic Map (PWLCM), Skew Tent Map (STM) and Bernoulli map in parallel model with the limited precision of 16-bit, which can generate good-performance key matrix for encryption. Then, we introduce a cost-efficient image encryption algorithm based on PSBP (named IEPSBP) which addresses the second challenge. Instead of bits or bytes, we use one row or one column as the basic unit to realize permutation and diffusion. The security analysis and performance experiments indicate that IEPSBP is secure and cost-efficient for green IoT.

**“Supporting heterogeneity in cyber-physical systems architectures,”**

Cyber-physical systems (CPS) are heterogeneous, because they tightly couple computation, communication, and control along with physical dynamics, which are traditionally considered separately. Without a comprehensive modeling formalism, model-based development of CPS involves using a multitude of models in a variety of formalisms that capture various aspects of the system design, such as software design, networking design, physical models, and protocol design. Without a rigorous unifying framework, system integration and integration of the analysis results for various models remains ad hoc. In this paper, we propose a multi-view architecture framework that treats models as views of the underlying system structure and uses structural and semantic mappings to ensure consistency and enable system-level verification in a hierarchical and compositional manner. Throughout the paper, the theoretical concepts are illustrated using two examples: a quadrotor and an automotive intersection collision avoidance system.

**“A socio-technical approach to preventing, mitigating, and recovering from ransomware attacks,”**

Recently there have been several high-profile ransomware attacks involving hospitals around the world. Ransomware is intended to damage or disable a user’s computer unless the user makes a payment. Once the attack has been launched, users have three options: 1) try to restore their data from backup; 2) pay the ransom; or 3) lose their data. In this manuscript, we discuss a socio-technical approach to address ransomware and outline four overarching steps that organizations can undertake to secure an electronic health record (EHR) system and the underlying computing infrastructure. First, health IT professionals need to ensure adequate system protection by correctly installing and configuring computers and networks that connect them. Next, the health care organizations need to ensure more reliable system defense by implementing user-focused strategies, including simulation and training on correct and complete use of computers and network applications. Concomitantly, the organization needs to monitor computer and application use continuously in an effort to detect suspicious activities and identify and address security problems before they cause harm. Finally, organizations need to respond adequately to and recover quickly from ransomware attacks and take actions to prevent them in future. We also elaborate on recommendations from other authoritative sources, including the National Institute of Standards and Technology (NIST). Similar to approaches to address other complex socio-technical health IT challenges, the responsibility of preventing, mitigating, and recovering from these attacks is shared between health IT professionals and end-users.

**“Edge-based differential privacy computing for sensor-cloud systems,”**

In sensor–cloud systems, with more personal data being hosted in cloud, privacy leakage is becoming one of the most serious concerns. Privacy computing is emerging as a paradigm to systematically enhance privacy protection. In other words, the new paradigm requests us to improve the computing model to provide a general privacy protection service. In this paper, we propose an edge-based model for data collection, in which the raw data from wireless sensor networks (WSNs) is differentially processed by algorithms on edge servers for privacy computing. A small quantity of the core data is stored on edge and local servers while the rest is transmitted to cloud for storage. In this way, the benefits are twofold. First, the data privacy is preserved since the original data cannot be retrieved even if the data stored in the cloud is leaked. Second, implemented by a differential storage method, compared to the state of the art, the edge-based model sends less data to the cloud and reduces the cost of communication and storage. Both theoretical analyses and extensive experiments validate our proposed method.

**“Information security climate and the assessment of information security risk among healthcare employees,”**

Since 2009, over 176 million patients in the United States have been adversely impacted by data breaches affecting Health Insurance Portability and Accountability Act-covered institutions. While the popular press often attributes data breaches to external hackers, most breaches are the result of employee carelessness and/or failure to comply with information security policies and procedures. To change employee behavior, we borrow from the organizational climate literature and introduce the Information Security Climate Index, developed and validated using two pilot samples. In this study, four categories of healthcare professionals (certified nursing assistants, dentists, pharmacists, and physician assistants) were surveyed. Likert-type items were used to assess the Information Security Climate Index, information security motivation, and information security behaviors. Study results indicated that the Information Security Climate Index was related to better employee information security motivation and information security behaviors. In addition, there were observed differences between occupational groups with pharmacists reporting a more favorable climate and behaviors than physician assistants.

**“Cognitive security paradigm for cloud computing applications,”**

This article presents new paradigms of confidential data protection. New classes of paradigms are dedicated to enhancing the already‐known cryptographic solutions belonging to the group of data sharing schemes. Data sharing schemes are dedicated to executing data protection tasks by means of splitting data into parts (called shadows) and distributing those shadows among a group of secret trustees. This process is enriched by linguistic and biometric solutions to guarantee protection of the shared data by means of biometric labeling or by means of user verification with the application of meaning interpretation of individual secret parts. The novelty in this solution is in the application of cognitive algorithms to describe correctly the shared secret. Cognitive paradigms guarantee an execution of information concealment protocols and their division at various levels of knowledge held by individual protocol participants. The paradigms of cognitive description of the concealed data have been introduced into a new class of data sharing protocols, ie, into the linguistic‐biometric threshold schemes. A special feature of the proposed solution is its universality, which is a possibility to apply the here‐discussed threshold schemes in the processes of data concealment with varying importance. At the same time, this solution can be applied to execute data protection tasks against unauthorized acquisition or disclosure. This paper will present also the possibilities of application of the paradigms presented here to conceal data dedicated to the data management processes.

**“Application of cognitive computing in healthcare, cybersecurity, Big Data and IoT: A literature review,”**

Human Intelligence is considered superior compared to Artificial Intelligence (AI) because of its ability to adapt faster to changes. Due to increasing data deluge, it is cumbersome for humans to analyse the vast amount of data and hence AI systems are in demand in today's world. However, these AI systems lack self-awareness, social skills, multitasking and faster adaptability. Cognitive Computing (CC), a subset of AI, acts as an effective solution in solving these challenges by serving as an important driver for knowledge-rich automation work. Knowing the latest research and state of the art in CC is one of the initial steps needed for researchers to make progress in this front. Thus, this paper presents a comprehensive survey of prior research in the CC domain along with the challenges, solutions and future research directions. Specifically, CC-based techniques solving real-world problems in four widely-researched application areas, namely, healthcare, cybersecurity, big data and IoT, have been reviewed in detail and the open research issues are discussed.

**“Modeling, detecting, and mitigating threats against industrial healthcare systems: A combined software defined networking and reinforcement learning approach,”**

The rise of the Internet of Medical Things introduces the healthcare ecosystem in a new digital era with multiple benefits, such as remote medical assistance, real-time monitoring, and pervasive control. However, despite the valuable healthcare services, this progression raises significant cybersecurity and privacy concerns. In this article, we focus our attention on the IEC 60 870-5-104 protocol, which is widely adopted in industrial healthcare systems. First, we investigate and assess the severity of the IEC 60 870-5-104 cyberattacks by providing a quantitative threat model, which relies on Attack Defence Trees and Common Vulnerability Scoring System v3.1. Next, we introduce an intrusion detection and prevention system (IDPS), which is capable of discriminating and mitigating automatically the IEC 60 870-5-104 cyberattacks. The proposed IDPS takes full advantage of the machine learning (ML) and software defined networking (SDN) technologies. ML is used to detect the IEC 60 870-5-104 cyberattacks, utilizing 1) Transmission Control Protocol/Internet Protocol network flow statistics and 2) IEC 60 870-5-104 payload flow statistics. On the other side, the automated mitigation is transformed into a multiarmed bandit problem, which is solved through a reinforcement learning method called Thomson sampling and SDN. The evaluation analysis demonstrates the efficiency of the proposed IDPS in terms of intrusion detection accuracy and automated mitigation performance. The detection accuracy and the F1 score of the proposed IDPS reach 0.831 and 0.8258, respectively, while the mitigation accuracy is calculated at 0.923.

**3. SYSTEM ANALYSIS**

**3.1 Existing System**

In medical field all patient’s data like text or images (MRI, visual investigations) must be secure and easy to process as physicians are not technicians to apply heavy computation encryption algorithms to secure medical data. So they need more secure and less complicated algorithms for patient data security. In past many security algorithms were introduced such as AES, DES and many more but those algorithms are easy to hack and does not support random permutation or encodings. According to author images encrypted via Random number permutation are more secure as it’s difficult for the hackers to guess Random numbers to properly decode images.

**Disadvantages of Existing System:**

* Less Accuracy

**3.2 Proposed System**

In propose paper author generating PWLCM random value by applying SHA algorithm and then encoding images using DNA encoding which will encode each pixel values based on DNA rules. Each pixel will be divided between ACGT DNA encoding as each character has unique binary values like A is represented from 00 to 11 and C represents between 01 to 10 and if image pixel value falls between any DNA rules then that pixel will get replace with either A or C or G or T.After DNA encoding we will apply XOR operations on all DNA encoding with public key to encrypt images. Encrypted image will be send to user with public key and by applying reverse operations Receiver can decrypt the image.

**Advantages of Proposed System:**

* High Accuracy.

**Modules Information:**

To implement this project we have designed following modules

1. Upload Medical Image: using this module we will upload medical image to application
2. Run Random DNA Encoding Module: using this module we will apply DNA encoding on plain medical image to convert it into DNA format
3. Run Permutation & Encryption: we will apply permutation and XOR operations on DNA encoded image to encrypt the image
4. Run Decryption Algorithm: using this module we will perform reverse operation to decrypt image

**FUNCTIONAL REQUIREMENTS:**

**SOFTWARE REQIREMENTS:**

**System Atributes:**

1. Filename
2. plain\_image, public\_key
3. encrypt\_image, dna\_encoding, blue\_e,green\_e,red\_e
4. h,w
5. x0, random\_value
6. img

**Data base Requirements:**

No need

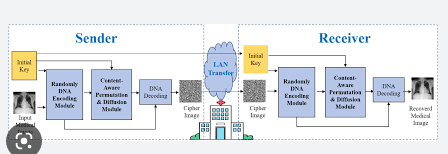
**USECASE:**

* Use cases - Use cases describe the interaction between the system and external users that leads to achieving particular goals.
* Each use case includes main elements:

1. Upload Medical Image
2. Run Random DNA Encoding Module
3. Run Permutation & Encryption
4. Run Decryption Algorithm

**User Stories:** After encrypting image we will calculate correlation between plain and encrypted images and if correlation value closer to 0 then encrypted image has no similarity to original image so more security will get. If both plain and encrypted images has correlation 1 then both images are similar and easy to hack. So by applying propose algorithm we can encrypted images with more dissimilarity so hacker cannot guess or decrypt

**Work down Structure:**

****

**Prototype:**

python 3.7.0 or 3.7.4

opencv-python==4.5.1.48

keras==2.3.1

tensorflow==1.14.0

protobuf==3.16.0

h5py==2.10.0

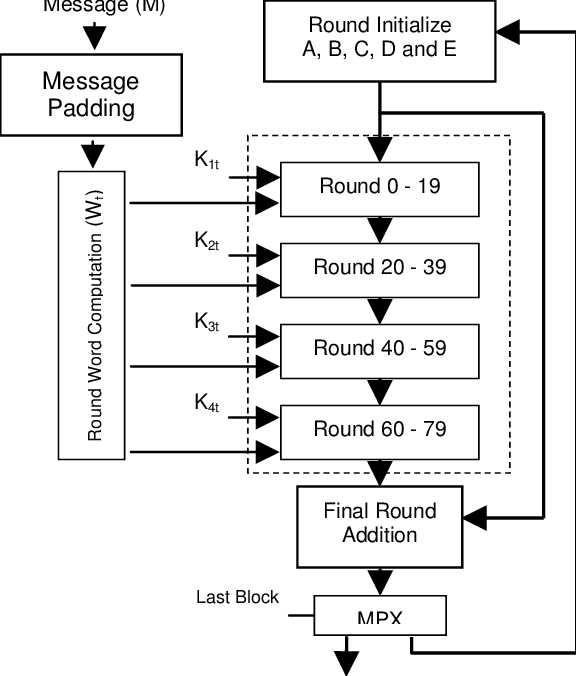
sklearn-extensions==0.0.2

scikit-learn==0.22.2.post1

Numpy

Pandas

**Models and Diagrams:**



**NON-FUNCTIONAL REQUIREMENT:**

**Usability:**  Usability is a quality attribute that assesses how easy user interfaces are to use. The word "usability" also refers to methods for improving ease-of-use during the design process.(how it was handle entire project easy)

**Security:** the quality or state of being secure: such as. a : freedom from danger : safety. b : freedom from fear or anxiety. c : freedom from the prospect of being laid off job security.

**Readability:** Readability is the ease with which a reader can understand a written text.

**Performance**: the execution of an action. : something accomplished : deed, feat. : the fulfillment of a claim, promise, or request : implementation. 3. : the action of representing a character in a play.

**Availability**: the quality or state of being available trying to improve the availability of affordable housing. 2 : an available person or thing.

**Scalability**: Scalability is the measure of a system's ability to increase or decrease in performance and cost in response to changes in application and system processing demands.

**3.3. PROCESS MODEL USED WITH JUSTIFICATION**

**SDLC (Umbrella Model):**

**Umbrella Activity**

**Umbrella Activity**

**Umbrella Activity**

1. Feasibility Study
2. TEAM FORMATION
3. Project Specification PREPARATION

Business Requirement Documentation

ANALYSIS & DESIGN

CODE

UNIT TEST

DOCUMENT CONTROL

ASSESSMENT

TRAINING

INTEGRATION & SYSTEM TESTING

DELIVERY/INSTALLATION

ACCEPTANCE TEST

Requirements Gathering

SDLC is nothing but Software Development Life Cycle. It is a standard which is used by software industry to develop good software.

**Stages in SDLC:**

* Requirement Gathering
* Analysis
* Designing
* Coding
* Testing
* Maintenance

**Requirements Gathering** **stage:**

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements. These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.



These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

* Feasibility study is all about identification of problems in a project.
* No. of staff required to handle a project is represented as Team Formation, in this case only modules are individual tasks will be assigned to employees who are working for that project.
* Project Specifications are all about representing of various possible inputs submitting to the server and corresponding outputs along with reports maintained by administrator.

**Analysis Stage:**

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.



The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included. The outputs of the project planning stage are the configuration management plan, the quality assurance plan, and the project plan and schedule, with a detailed listing of scheduled activities for the upcoming Requirements stage, and high level estimates of effort for the out stages.

**Designing Stage:**

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.

  
When the design document is finalized and accepted, the RTM is updated to show that each design element is formally associated with a specific requirement. The outputs of the design stage are the design document, an updated RTM, and an updated project plan.

**Development (Coding) Stage:**

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.



The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

**Integration & Test Stage:**

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.



The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

* **Installation & Acceptance Test:**

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.



The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

**Maintenance:**

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation later employees will be assigned work and they will undergo training on that particular assigned category. For this life cycle there is no end, it will be continued so on like an umbrella (no ending point to umbrella sticks).

**3.4. Software Requirement Specification**

**3.4.1. Overall Description**

A Software Requirements Specification (SRS) – a [requirements specification](http://en.wikipedia.org/wiki/Requirements_specification) for a [software system](http://en.wikipedia.org/wiki/Software_system) is a complete description of the behavior of a system to be developed. It includes a set of [use cases](http://en.wikipedia.org/wiki/Use_case) that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. [Nonfunctional requirements](http://en.wikipedia.org/wiki/Non-functional_requirements) are requirements which impose constraints on the design or implementation (such as [performance engineering](http://en.wikipedia.org/wiki/Performance_engineering) requirements, [quality](http://en.wikipedia.org/wiki/Quality_%28business%29) standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A [business analyst](http://en.wikipedia.org/wiki/Business_analyst), sometimes titled [system analyst](http://en.wikipedia.org/wiki/System_analyst), is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the [systems development lifecycle](http://en.wikipedia.org/wiki/Systems_development_life_cycle) domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

* [Business requirements](http://en.wikipedia.org/wiki/Business_requirements) describe in business terms what must be delivered or accomplished to provide value.
* Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
* Process requirements describe activities performed by the developing organization. For instance, process requirements could specify .Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:
* **ECONOMIC FEASIBILITY**

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

* **Operational Feasibility**

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization’s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

* **TECHNICAL FEASIBILITY**

Earlier no system existed to cater to the needs of ‘Secure Infrastructure Implementation System’. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to .the users. The database’s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security.

**3.4.2. External Interface Requirements**

**User Interface**

The user interface of this system is a user friendly python Graphical User Interface.

**Hardware Interfaces**

The interaction between the user and the console is achieved through python capabilities.

**Software Interfaces**

The required software is python.

**SYSTEM REQUIREMENT:**

**HARDWARE REQUIREMENTS:**

# Processor - Intel i3(min)

* Speed - 1.1 GHz
* RAM - 4GB(min)
* Hard Disk - 500 GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse
* Monitor - SVGA

**SOFTWARE REQUIREMENTS:**

* Operating System - Windows10(min)
* Programming Language - Python

**4. SYSTEM DESIGN**

**CLASS DIAGRAM:**

The class diagram is the main building block of object oriented modeling. It is used both for general conceptual modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

* The upper part holds the name of the class
* The middle part contains the attributes of the class
* The bottom part gives the methods or operations the class can take or undertake



**USECASE DIAGRAM:**

A **use case diagram** at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as we



**SEQUENCE DIAGRAM**

A **sequence diagram** is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.



**COLLABORATION DIAGRAM:**

A collaboration diagram describes interactions among objects in terms of sequenced messages. Collaboration diagrams represent a combination of information taken from class, sequence, and use case diagrams describing both the static structure and dynamic behaviour of a system.



**COMPONENT DIAGRAM:**

In the Unified Modelling Language, a component diagram depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.



**DEPLOYMENT DIAGRAM:**

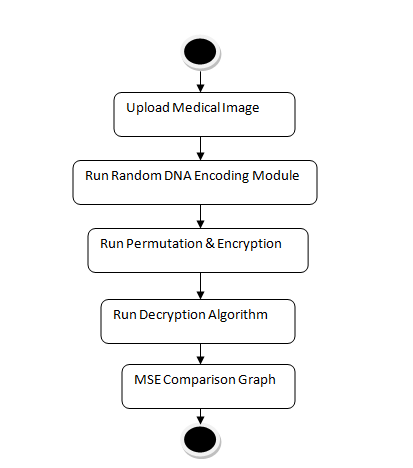
A **deployment diagram** in the Unified Modeling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

****

**ACTIVITY DIAGRAM:**

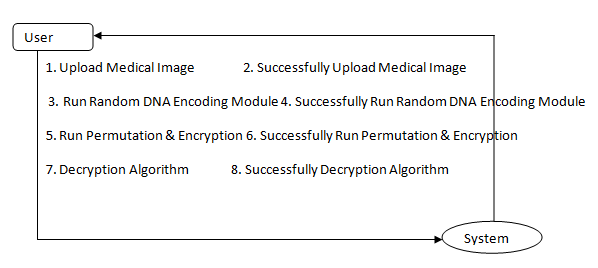
Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent



**Data flow :**

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. The technique starts with an overall picture of the business and continues by analyzing each of the functional areas of interest. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy-to-use, free downloadable diagramming tools. A DFD is a model for constructing and analyzing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.



**5. IMPLEMETATION**

**5.1 Python**

Python is a general-purpose language. It has wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean and length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

**History of Python:**

Python is a fairly old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

**Why Python was created?**

In late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to-understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to design of a new language which was later named Python.

**Why the name Python?**

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

**Features of Python:**

**A simple language which is easier to learn**

Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like: C++, Java, C#. Python makes programming fun and allows you to focus on the solution rather than syntax.

If you are a newbie, it's a great choice to start your journey with Python.

**Free and open-source**

You can freely use and distribute Python, even for commercial use. Not only can you use and distribute software’s written in it, you can even make changes to the Python's source code.

Python has a large community constantly improving it in each iteration.

**Portability**

You can move Python programs from one platform to another, and run it without any changes.

It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.

**Extensible and Embeddable**

Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code.

This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

**A high-level, interpreted language**

Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on.

Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

**Large standard libraries to solve common tasks**

Python has a number of standard libraries which makes life of a programmer much easier since you don't have to write all the code yourself. For example: Need to connect MySQL database on a Web server? You can use MySQLdb library using import MySQLdb .

Standard libraries in Python are well tested and used by hundreds of people. So you can be sure that it won't break your application.

**Object-oriented**

Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively.

With OOP, you are able to divide these complex problems into smaller sets by creating objects.

**Applications of Python:**

**1. Simple Elegant Syntax**

Programming in Python is fun. It's easier to understand and write Python code. Why? The syntax feels natural. Take this source code for an example:

a = 2

b = 3

sum = a + b

print(sum)

**2. Not overly strict**

You don't need to define the type of a variable in Python. Also, it's not necessary to add semicolon at the end of the statement.

Python enforces you to follow good practices (like proper indentation). These small things can make learning much easier for beginners.

**3. Expressiveness of the language**

Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

**4. Great Community and Support**

Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck.

**5.2 Sample Code:**

**Medicalimage.py**

from tkinter import messagebox

from tkinter import \*

from tkinter import simpledialog

import tkinter

from tkinter import filedialog

import matplotlib.pyplot as plt

from tkinter.filedialog import askopenfilename

import numpy as np

import os

import cv2

from hashlib import sha1

from dna import dna\_decode,dna\_encode, decompose\_matrix

import random

from skimage.metrics import structural\_similarity as ssim

main = tkinter.Tk()

main.title("Medical Image Encryption by Content-Aware DNA Computing for Secure Healthcare")

main.geometry("1300x1200")

global filename

global plain\_image, public\_key

global encrypt\_image, dna\_encoding, blue\_e,green\_e,red\_e

global h,w

global x0, random\_value

global img

def upload():

global filename, plain\_image

filename = filedialog.askopenfilename(initialdir="testImages")

text.delete('1.0', END)

text.insert(END,filename+" loaded\n");

plain\_image = cv2.imread(filename)

plain\_image = cv2.resize(plain\_image, (300, 300))

cv2.imwrite("test.png", plain\_image)

plain\_image = cv2.imread("test.png")

filename = "test.png"

cv2.imshow("Image Loaded", plain\_image)

cv2.waitKey(0)

#get Random value for pixel encoding using SHA256

def generateRandomValue():

global x0, random\_value

global plain\_image

global encrypt\_image

sha = sha1(plain\_image).hexdigest()

x0 = ord(sha[0]) ^ ord(sha[1]) ^ ord(sha[2]) ^ ord(sha[3]) ^ ord(sha[4]) ^ ord(sha[5]) ^ ord(sha[6]) ^ ord(sha[7]) ^ ord(sha[8]) ^ ord(sha[9]) ^ ord(sha[10]) ^ ord(sha[11]) ^ord(sha[12]) ^ ord(sha[13]) ^ord(sha[14]) ^ ord(sha[15])

random\_value = ord(sha[16]) ^ ord(sha[17]) ^ ord(sha[18]) ^ ord(sha[19]) ^ ord(sha[20]) ^ ord(sha[21]) ^ ord(sha[22]) ^ ord(sha[23]) ^ ord(sha[24]) ^ ord(sha[25]) ^ ord(sha[26]) ^ ord(sha[27]) ^ ord(sha[28]) ^ ord(sha[29]) ^ ord(sha[30]) ^ ord(sha[31])

x0 = x0 / random.randrange(150, 255) #generate random encoding for each pixel

random\_value = int(random\_value / 510)

text.insert(END,"PWLCM Random Pixel Encoding Value : "+str(random\_value + x0)+"\n\n")

def dnaEncoding(): #function to apply dna encoding on image

text.delete('1.0', END)

global filename, plain\_image, dna\_encoding, blue\_e,green\_e,red\_e

blue,green,red = decompose\_matrix(filename) #get al colour pixels from given file image

blue\_e,green\_e,red\_e = dna\_encode(blue,green,red) #now apply DNA encoding on all pixels

dna\_encoding = np.dstack((red\_e,green\_e,blue\_e))#convert DNA encoded pixels into image

text.insert(END,"DNA Encoded pixels\n")

text.insert(END,str(dna\_encoding)+"\n\n")

print(dna\_encoding.shape)

def correlation(original, encrypted):

original = cv2.cvtColor(original, cv2.COLOR\_BGR2GRAY)

encrypted = cv2.cvtColor(encrypted, cv2.COLOR\_BGR2GRAY)

original = cv2.resize(original, (100, 100))

encrypted = cv2.resize(encrypted, (100, 100))

print(original.shape)

print(encrypted.shape)

corr = ssim(original, encrypted, data\_range = encrypted.max() - encrypted.min())

return corr

def runEncryption():#function to encrypt DNA encoded image

text.delete('1.0', END)

global dna\_encoding, h, w, random\_value, encrypt\_image, plain\_image, public\_key

generateRandomValue()

h = dna\_encoding.shape[0]

w = dna\_encoding.shape[1]

public\_key = random.randrange(29, 31)

for y in range(0, h):

for x in range(0, w):

img1 = dna\_encoding[y,x,0]

img2 = dna\_encoding[y,x,1]

img3 = dna\_encoding[y,x,2]

dna\_encoding[y,x,0] = ord(img1) ^ random\_value #XOR operations

dna\_encoding[y,x,1] = ord(img2) ^ random\_value

dna\_encoding[y,x,2] = ord(img3) ^ random\_value

encrypt\_image = dna\_encoding

encrypt\_image = encrypt\_image.astype(int) \* public\_key

print(encrypt\_image)

cv2.imwrite("test.png", encrypt\_image)

corr = correlation(plain\_image, cv2.imread("test.png"))

text.insert(END,"Propose Algorithm Image Correlation : "+str(corr)+"\n\n")

figure, axis = plt.subplots(nrows=1, ncols=3,figsize=(10,10))

axis[0].set\_title("Original Image")

axis[1].set\_title("Encrypted Image")

axis[2].set\_title("Histogram")

axis[0].imshow(cv2.cvtColor(plain\_image, cv2.COLOR\_BGR2RGB))

axis[1].imshow(encrypt\_image/255)

axis[2].hist(encrypt\_image.ravel(),256,[0,256])

figure.tight\_layout()

plt.show()

def runDecryption():

global encrypt\_image, public\_key, random\_value, dna\_encoding

global blue\_e,green\_e,red\_e

enc = dna\_encoding

h = enc.shape[0]

w = enc.shape[1]

for y in range(0, h):

for x in range(0, w):

img1 = int(enc[y,x,0])#XOR operations

img2 = int(enc[y,x,1])

img3 = int(enc[y,x,2])

e1 = img1 ^ random\_value

e2 = img2 ^ random\_value

e3 = img3 ^ random\_value

enc[y,x,0] = chr(e1)

enc[y,x,1] = chr(e2)

enc[y,x,2] = chr(e3)#reverse decryption process

b,g,r = dna\_decode(blue\_e,green\_e,red\_e)#DNA decoding to get normal image

decrypt\_img = np.dstack((r,g,b))

figure, axis = plt.subplots(nrows=1, ncols=3,figsize=(10,10))

axis[0].set\_title("Original Image")

axis[1].set\_title("Encrypted Image")

axis[2].set\_title("Decrypted Image")

axis[0].imshow(cv2.cvtColor(plain\_image, cv2.COLOR\_BGR2RGB))

axis[1].imshow(encrypt\_image/255)

axis[2].imshow(decrypt\_img)

figure.tight\_layout()

plt.show()

def close():

main.destroy()

font = ('times', 16, 'bold')

title = Label(main, text='Medical Image Encryption by Content-Aware DNA Computing for Secure Healthcare')

title.config(bg='firebrick4', fg='dodger blue')

title.config(font=font)

title.config(height=3, width=120)

title.place(x=0,y=5)

font1 = ('times', 12, 'bold')

text=Text(main,height=20,width=150)

scroll=Scrollbar(text)

text.configure(yscrollcommand=scroll.set)

text.place(x=50,y=120)

text.config(font=font1)

font1 = ('times', 13, 'bold')

uploadButton = Button(main, text="Upload Medical Image", command=upload, bg='#ffb3fe')

uploadButton.place(x=50,y=550)

uploadButton.config(font=font1)

randomDNAButton = Button(main, text="Run Random DNA Encoding Module", command=dnaEncoding, bg='#ffb3fe')

randomDNAButton.place(x=350,y=550)

randomDNAButton.config(font=font1)

encButton1 = Button(main, text="Run Permutation & Encryption", command=runEncryption, bg='#ffb3fe')

encButton1.place(x=690,y=550)

encButton1.config(font=font1)

decButton = Button(main, text="Run Decryption Algorithm", command=runDecryption, bg='#ffb3fe')

decButton.place(x=50,y=600)

decButton.config(font=font1)

exitButton = Button(main, text="Exit", command=close, bg='#ffb3fe')

exitButton.place(x=350,y=600)

exitButton.config(font=font1)

main.config(bg='LightSalmon3')

main.mainloop()

**dna.py**

import cv2

import numpy as np

#defining rules

dna={}

dna["00"]="A"

dna["01"]="T"

dna["10"]="G"

dna["11"]="C"

dna["A"]=[0,0]

dna["T"]=[0,1]

dna["G"]=[1,0]

dna["C"]=[1,1]

#DNA xor

dna["AA"]=dna["TT"]=dna["GG"]=dna["CC"]="A"

dna["AG"]=dna["GA"]=dna["TC"]=dna["CT"]="G"

dna["AC"]=dna["CA"]=dna["GT"]=dna["TG"]="C"

dna["AT"]=dna["TA"]=dna["CG"]=dna["GC"]="T"

# Maximum time point and total number of time points

tmax, N = 100, 10000

def dna\_decode(b,g,r):#call this function to decode DNA value to pixel values back

m,n = b.shape

r\_dec= np.ndarray((m,int(n\*2)),dtype=np.uint8)

g\_dec= np.ndarray((m,int(n\*2)),dtype=np.uint8)

b\_dec= np.ndarray((m,int(n\*2)),dtype=np.uint8)

for color,dec in zip((b,g,r),(b\_dec,g\_dec,r\_dec)):

for j in range(0,m):

for i in range(0,n):

dec[j,2\*i]=dna["{0}".format(color[j,i])][0]

dec[j,2\*i+1]=dna["{0}".format(color[j,i])][1]

b\_dec=(np.packbits(b\_dec,axis=-1))

g\_dec=(np.packbits(g\_dec,axis=-1))

r\_dec=(np.packbits(r\_dec,axis=-1))

return b\_dec,g\_dec,r\_dec

def dna\_encode(b,g,r): #function to encode images as per DNA rules

b = np.unpackbits(b,axis=1)

g = np.unpackbits(g,axis=1)

r = np.unpackbits(r,axis=1)

m,n = b.shape

r\_enc= np.chararray((m,int(n/2)))

g\_enc= np.chararray((m,int(n/2)))

b\_enc= np.chararray((m,int(n/2)))

for color,enc in zip((b,g,r),(b\_enc,g\_enc,r\_enc)):

idx=0

for j in range(0,m):

for i in range(0,n,2):

enc[j,idx]=dna["{0}{1}".format(color[j,i],color[j,i+1])]

idx+=1

if (i==n-2):

idx=0

break

b\_enc=b\_enc.astype(str)

g\_enc=g\_enc.astype(str)

r\_enc=r\_enc.astype(str)

return b\_enc,g\_enc,r\_enc

def split\_into\_rgb\_channels(image):

red = image[:,:,2]

green = image[:,:,1]

blue = image[:,:,0]

return red, green, blue

def decompose\_matrix(iname):

image = cv2.imread(iname)

blue,green,red = split\_into\_rgb\_channels(image)

for values, channel in zip((red, green, blue), (2,1,0)):

img = np.zeros((values.shape[0], values.shape[1]), dtype = np.uint8)

img[:,:] = (values)

if channel == 0:

B = np.asmatrix(img)

elif channel == 1:

G = np.asmatrix(img)

else:

R = np.asmatrix(img)

return B,G,R

def recover\_image(b,g,r,in\_image,out):

'''

img = cv2.imread(in\_image)

img[:,:,2] = r

img[:,:,1] = g

img[:,:,0] = b

'''

img = np.dstack((r,g,b))

cv2.imwrite(out, img)

print("saved ecrypted image as enc.jpg")

return img

'''

blue,green,red=decompose\_matrix('caller.jpg')

blue\_e,green\_e,red\_e=dna\_encode(blue,green,red)

img = np.dstack((red\_e,green\_e,blue\_e))

print(img)

print(type(img))

cv2.imshow("aa",img)

cv2.waitKey(0)

#img=recover\_image(blue\_e,green\_e,red\_e,'caller.jpg',"enc.png")

b,g,r=dna\_decode(blue\_e,green\_e,red\_e)

img=recover\_image(b,g,r,'enc.jpg',"dec.png")

'''

**6. TESTING:**

**Implementation and Testing:**

Implementation is one of the most important tasks in project is the phase in which one has to be cautions because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

**Implementation**

The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modifies as a result of a programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

## Testing

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible condition. Actually testing is the state of implementation which aimed at ensuring that the system works accurately and efficiently before the actual operation commence. The following is the description of the testing strategies, which were carried out during the testing period.

### System Testing

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to be check if one is capable to with stand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to user to use the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and pattern of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

**Module Testing**

To locate errors, each module is tested individually. This enables us to detect error and correct it without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different job and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

**Integration Testing**

After the module testing, the integration testing is applied. When linking the modules there may be chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

**Acceptance Testing**

When that user fined no major problems with its accuracy, the system passers through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which elimination wastage of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case Id** | **Test Case Name** | **Test Case Desc.** | **Test Steps** | | | | **Test Case Status** | **Test Priority** |
| **Step** | **Expected** | | **Actual** |
| 01 | Upload Medical Image | Verify Your Upload Medical Image or not | If Upload Medical Image may not upload | we cannot do any further operations | we can do further operations | | High | High |
| 02 | Run Random DNA Encoding Module | Verify Run Random DNA Encoding Module or not | If Dataset may not Preprocess | we cannot do any further operations | we can do further operations | | High | High |
| 03 | Run Permutation & Encryption | Verify Run Permutation & Encryption or not | If Run Permutation & Encryption not be | we cannot do any further operations | we can do further operations | | High | High |
| 04 | Run Decryption Algorithm | Verify Run Decryption Algorithm or not | If Run Decryption Algorithm not Run | We cannot run  operation | We can Run the Operation | | High | High |

**7. SCREENSHOTS:**

Medical Image Encryption by Content-Aware DNA Computing for Secure Healthcare

In medical field all patient’s data like text or images (MRI, visual investigations) must be secure and easy to process as physicians are not technicians to apply heavy computation encryption algorithms to secure medical data. So they need more secure and less complicated algorithms for patient data security. In past many security algorithms were introduced such as AES, DES and many more but those algorithms are easy to hack and does not support random permutation or encodings. According to author images encrypted via Random number permutation are more secure as it’s difficult for the hackers to guess Random numbers to properly decode images.

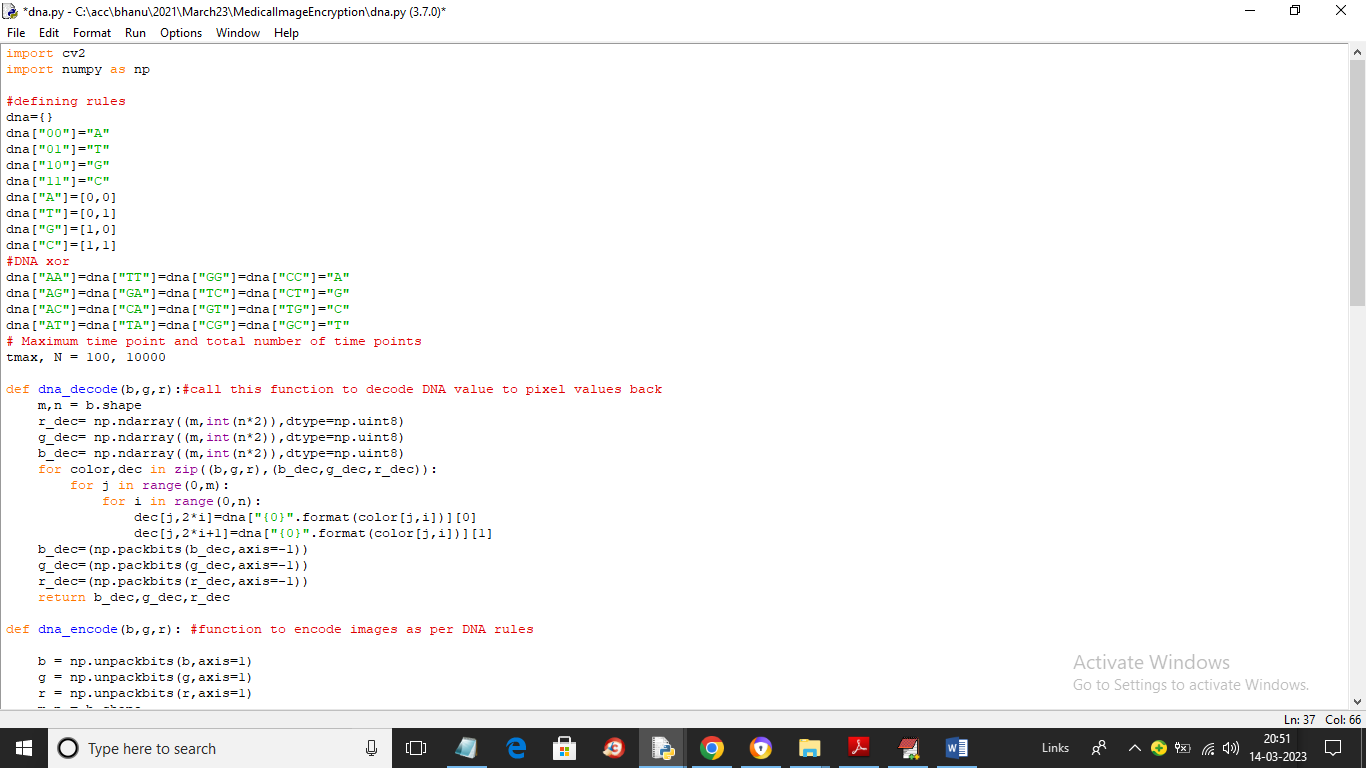
In propose paper author generating PWLCM random value by applying SHA algorithm and then encoding images using DNA encoding which will encode each pixel values based on DNA rules. Each pixel will be divided between ACGT DNA encoding as each character has unique binary values like A is represented from 00 to 11 and C represents between 01 to 10 and if image pixel value falls between any DNA rules then that pixel will get replace with either A or C or G or T.

After DNA encoding we will apply XOR operations on all DNA encoding with public key to encrypt images. Encrypted image will be send to user with public key and by applying reverse operations Receiver can decrypt the image.

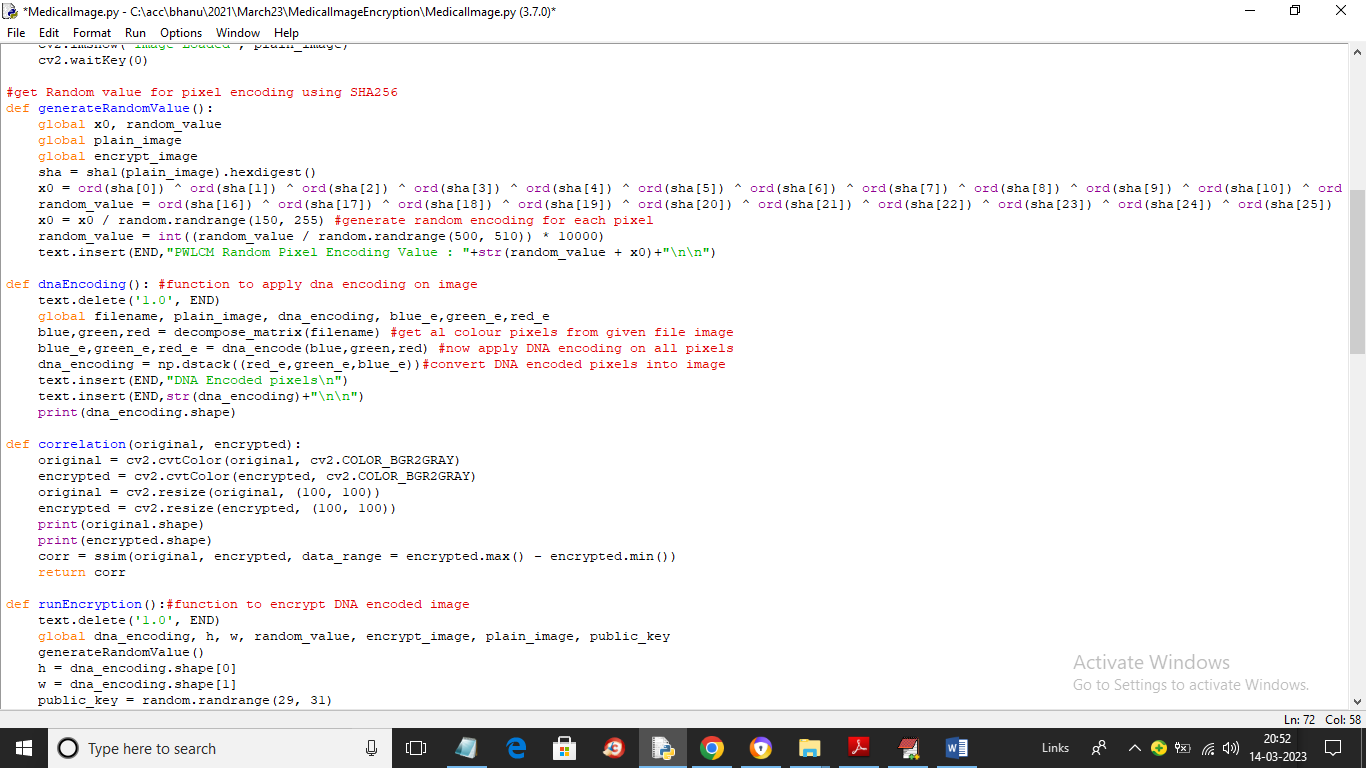
After encrypting image we will calculate correlation between plain and encrypted images and if correlation value closer to 0 then encrypted image has no similarity to original image so more security will get. If both plain and encrypted images has correlation 1 then both images are similar and easy to hack. So by applying propose algorithm we can encrypted images with more dissimilarity so hacker cannot guess or decrypt.

This paper proposes a novel content-aware deoxyribonucleic acid (DNA)computing system to encrypt medical images, thus guaranteeingprivacy and promoting secure healthcare environment.The proposed system consists of sender andreceiver to perform tasks of encryption and decryption,respectively, where both contain the same structure design,but perform opposite operations. In either senderor receiver, we design a randomly DNA encoding anda content-aware permutation and diffusion module. Consideringintroducing random mechanism to increase difficultyof cracking, the former module builds a randomencryption rule selector in DNA encoding process by randomlymapping quantity of medical image pixels to outputs.Meanwhile, the latter module constructs a permutation sequence,which not only encodes information of pixel values,but also involves redundant correlation between adjacentpixels located in a patch. Such design brings awarenessproperty of medical image content to greatly increasecomplexity in cracking by embedding semantically informationfor encryption.

In below screen we are showing code which will convert images to DNA encoding rules



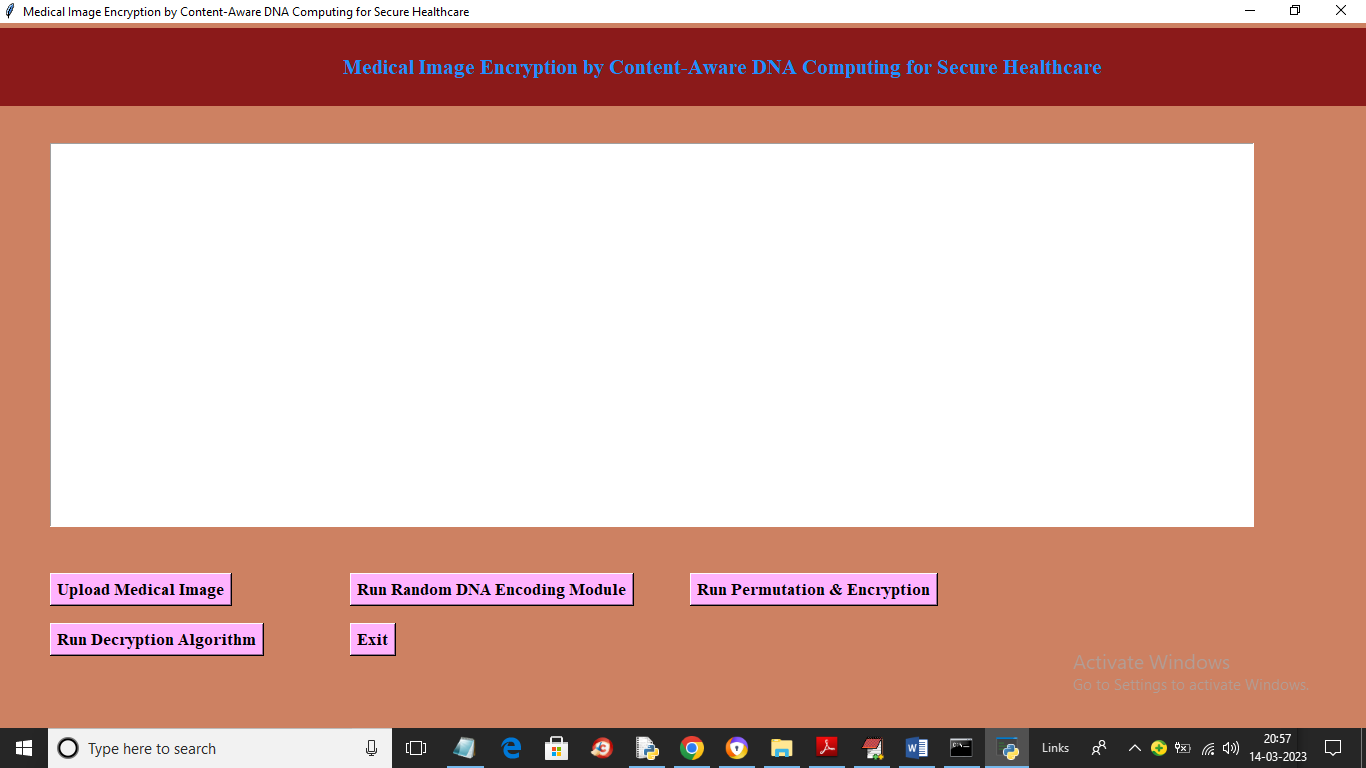
In above screen read red colour comments to know about DNA encoding and decoding



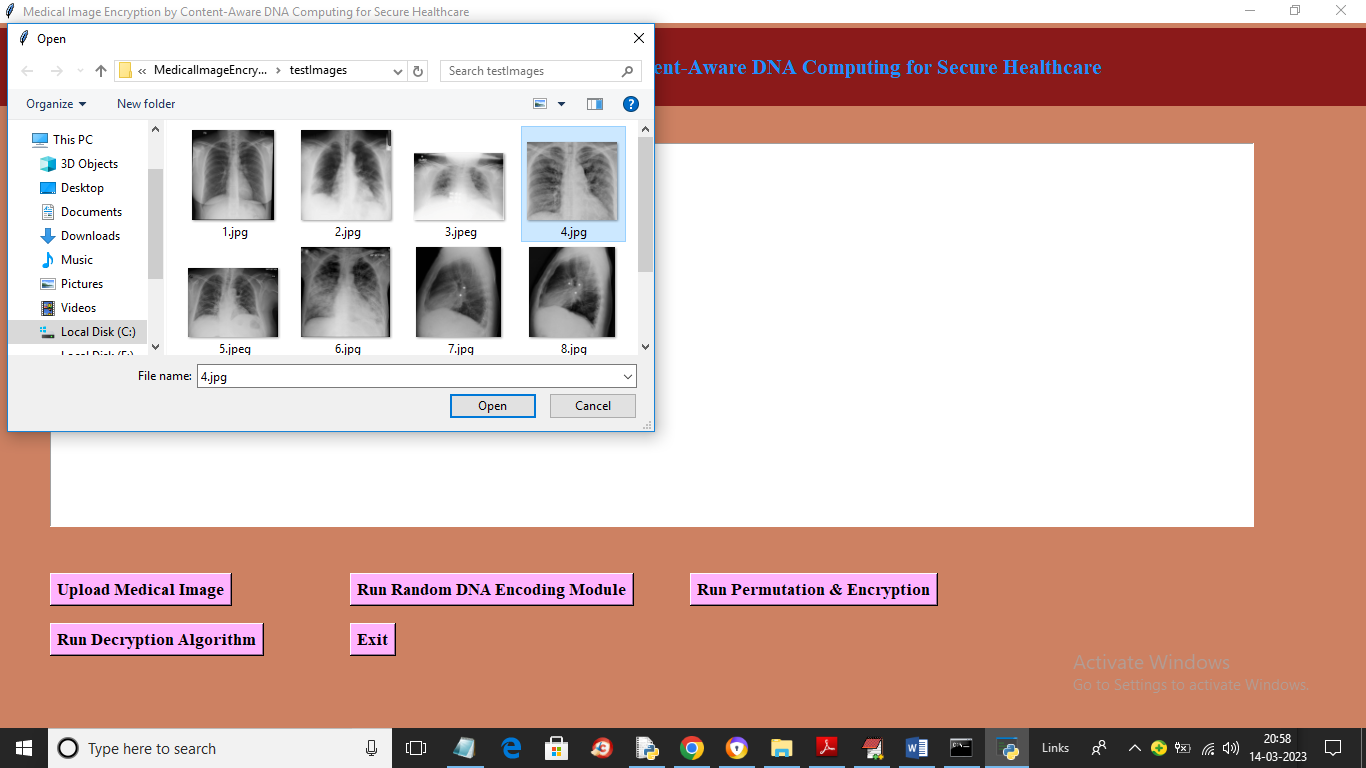
In above screen we are applying DNA encoding on image and then encrypting DNA encoded image using public key and XOR operations

SCREEN SHOTS

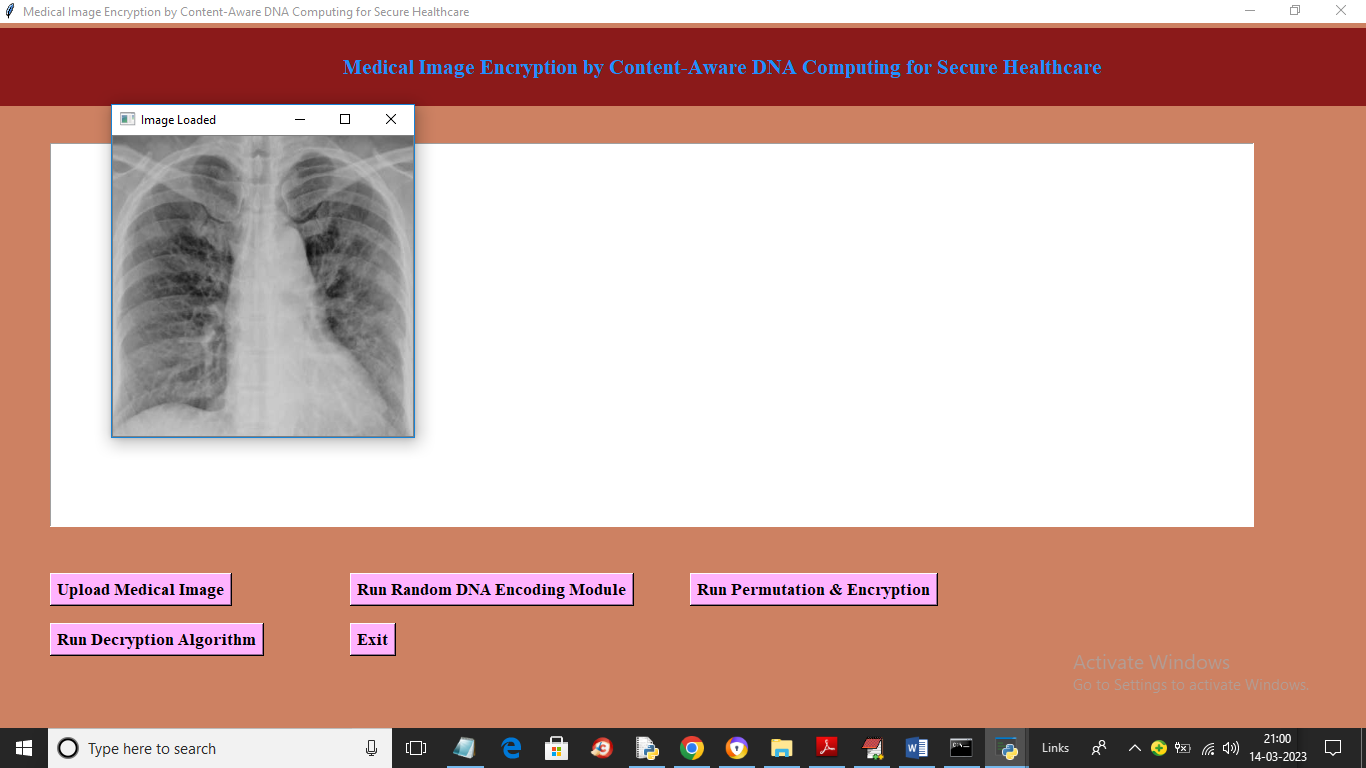
To run project double click on ‘run.bat’ file to get below screen



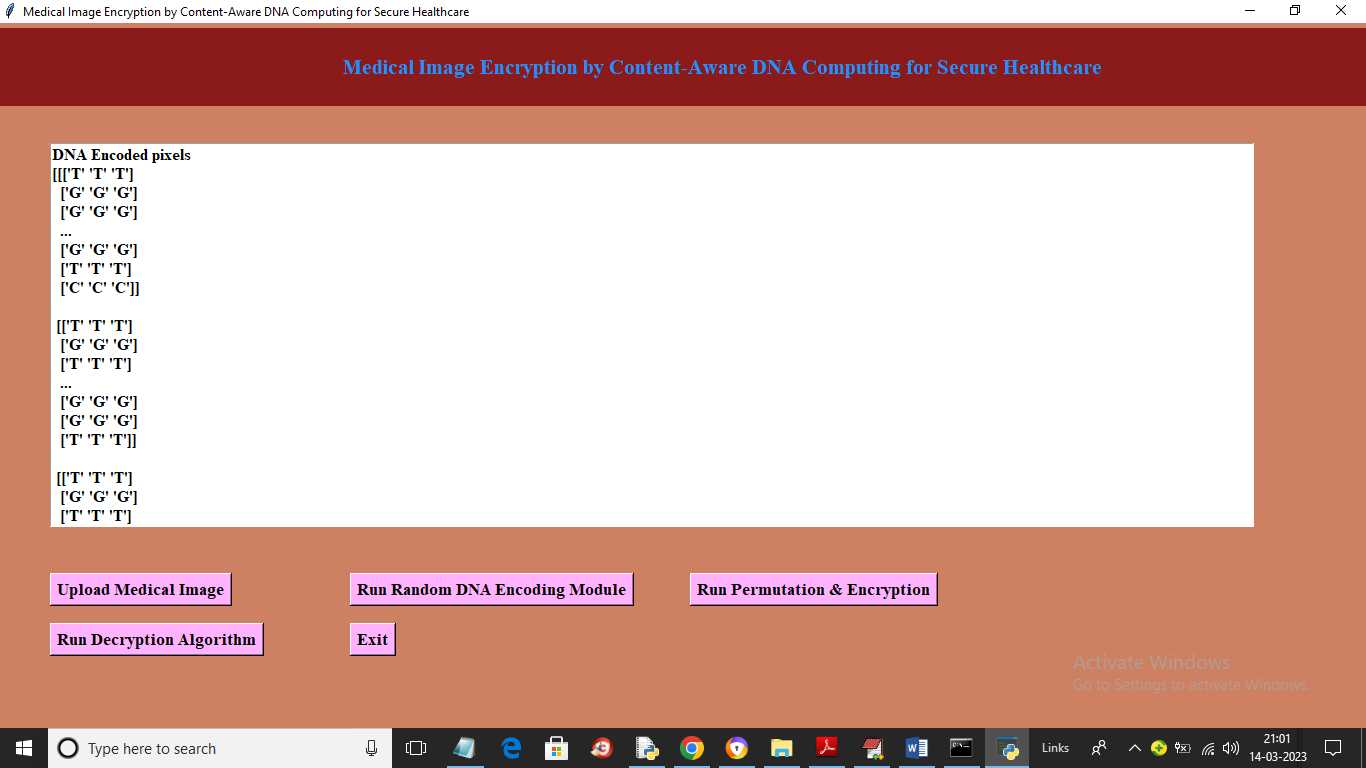
In above screen click on ‘Upload Medical Image’ button to upload image and get below output



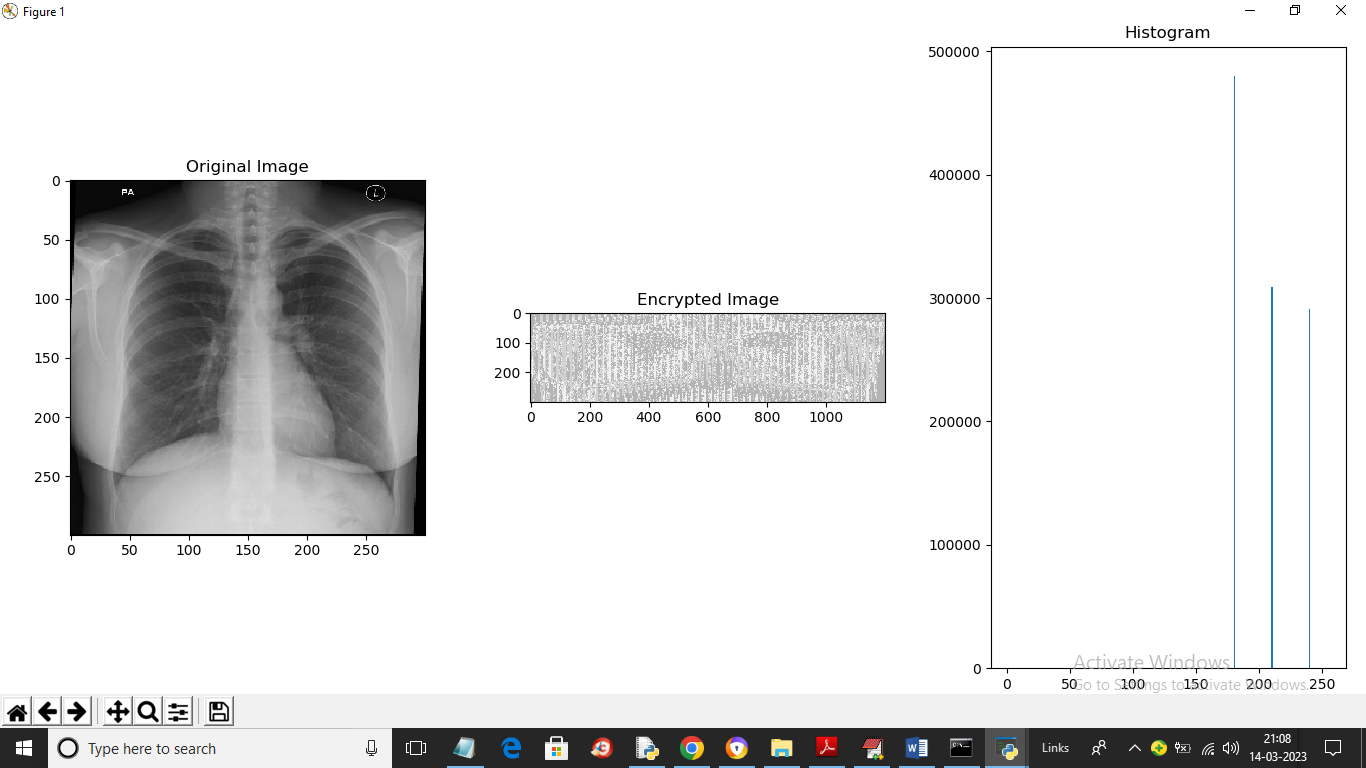
In above screen selecting and uploading medical image and then click on ‘Open’ button to get below output



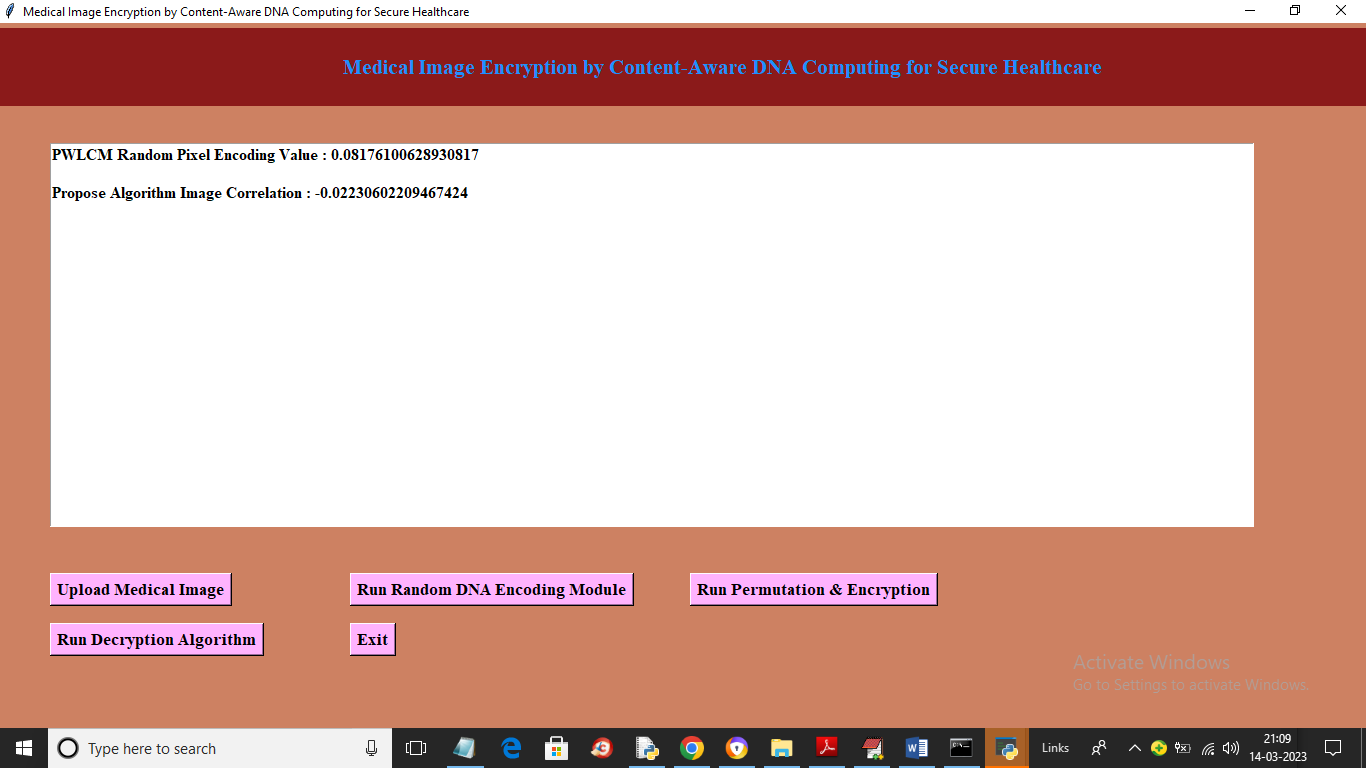
In above screen image is loaded and now click on ‘Run Random DNA Encoding Module’ button to apply DNA encoding on image and get below output



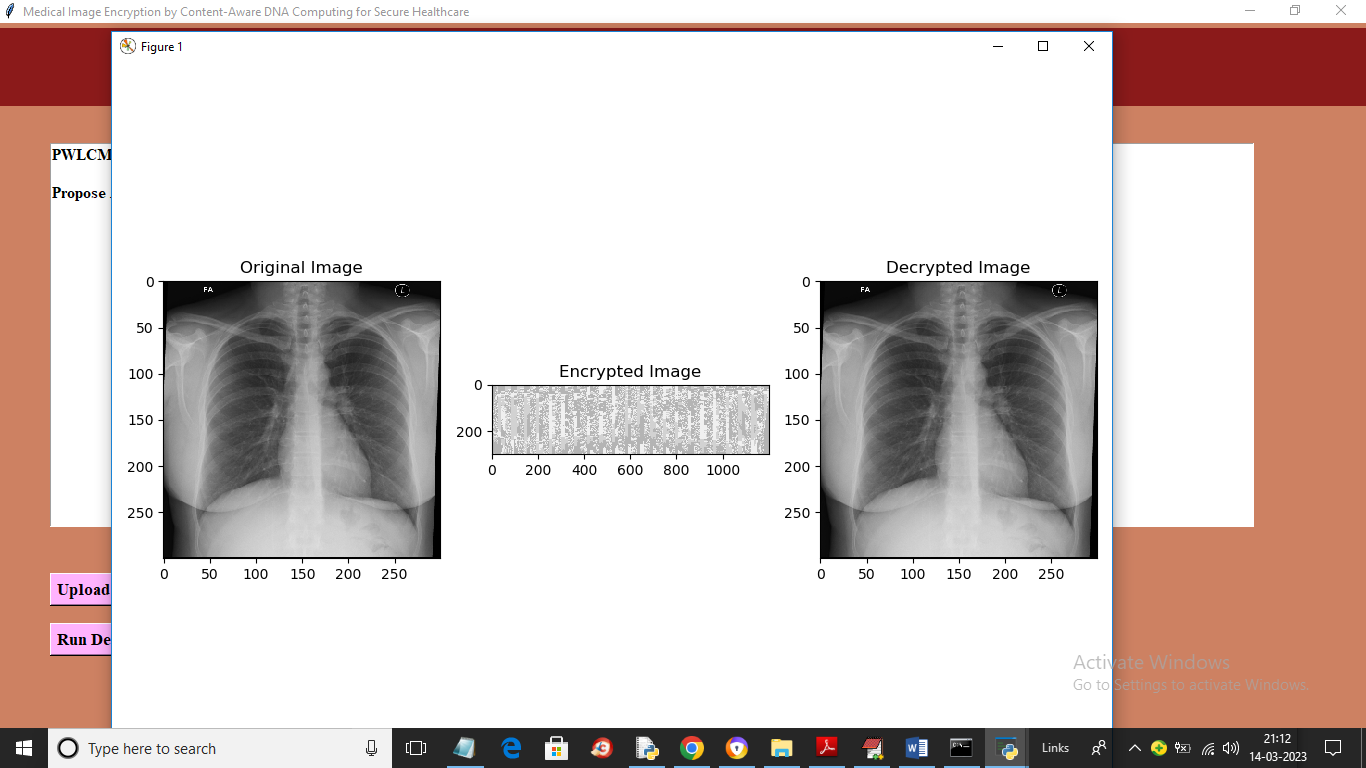
In above screen entire image pixels are converted to DNA encoding and now click on ‘Run Permutation & Encryption’ button to encrypt image and get below output



In above screen first image is the original image and second is thee encrypted image and 3rd is the histogram of encrypted image and in histogram we can see more big lines so we can say encrypted image has lots of noise and cannot be understand or hack by the attacker and now close above image to get blow output



In above screen we can see PWLCM calculated value to encrypt image and in next line we can see correlation between encrypted and plain image as -0.022 so we can say there is not even 1 percent of similarity we can find between original plain image and encrypted image so we can say propose algorithm is more secure. Now click on ‘Run Decryption Algorithm’ button to decrypt image and get below output



In above screen first is the original image and second is the encrypted image and 3rd is the decrypted image. Similarly you can upload and test other images

**8. CONCLUSION:**

To ensure the security of cipher images, this article proposed a novel cryptosystem for secure healthcare with two effective modules: 1) randomly DNA encoding module; and 2) contentaware permutation and diffusion module. The former one builds a random encryption rule selector in DNA encoding process, which increases security by building quantity of random mappings from image pixels to computations and greatly improves key sensitivity. The latter module constructs a permutation sequence, which not only encodes information of pixel values, but also breaks the strong correlation between adjacent pixels located in a patch.

**9. REFERENCES:**

[1] S. S. Bhuyan et al., “Transforming healthcare cybersecurity from reactive to proactive: Current status and future recommendations,” J. Med. Syst., vol. 44, no. 5, 2020, Art. no. 98.

[2] L. Coventry and D. Branley, “Cybersecurity in healthcare: A narrative review of trends, threats and ways forward,” Maturitas, vol. 113, pp. 48–52, 2018.

[3] M. Alawida, J. S. Teh, A. Samsudin, and W. H. Alshoura, “An image encryption scheme based on hybridizing digital chaos and finite state machine,” Signal Process., vol. 164, pp. 249–266, 2019. [4] A. Abusukhon, Z. Mohammad, and A. Al-Thaher, “An authenticated, secure, and mutable multiple-session-keys protocol based on elliptic curve cryptography and text-to-image encryption algorithm,”Concurrency Comput., Pract. Experience, vol. 34, no. 4, 2022, Art. no. e6649.

[5] A. Daoui, H. Karmouni, O. E. Ogri, M. Sayyouri, and H. Qjidaa, “Robust image encryption and zero-watermarking scheme using SCA and modified logistic map,” Expert Syst. Appl., vol. 190, 2022, Art. no. 116193.

[6] Z. Gu et al., “IEPSBP: A cost-efficient image encryption algorithm based on parallel chaotic system for green IoT,” IEEE Trans. Green Commun. Netw., vol. 6, no. 1, pp. 89–106, Mar. 2022.

[7] A. Rajhans et al., “Supporting heterogeneity in cyber-physical systems architectures,” IEEE Trans. Autom. Control., vol. 59, no. 12, pp. 3178–3193, Dec. 2014.

[8] H. Singh and D. F. Sittig, “A socio-technical approach to preventing, mitigating, and recovering from ransomware attacks,” Appl. Clin. Inform., vol. 7, no. 2, pp. 624–632, 2016.

[9] T. Wang, Y. Mei, W. Jia, X. Zheng, G. Wang, and M. Xie, “Edge-based differential privacy computing for sensor-cloud systems,” J. Parallel Distrib. Comput., vol. 136, pp. 75–85, 2020.

[10] S. R. Kessler, S. Pindek, G. Kleinman, S. Andel, and P. E. Spector, “Information security climate and the assessment of information security risk among healthcare employees,” Health Informat. J., vol. 26, no. 1, pp. 461–473, 2020.

[11] L. Ogiela and M. R. Ogiela, “Cognitive security paradigm for cloud computing applications,” Concurrency Comput. Pract. Exp., vol. 32, no. 8, 2020, Art. no. e5316.

[12] A. G. Sreedevi, T. N. Harshitha, V. Sugumaran, and P. Shankar, “Application of cognitive computing in healthcare, cybersecurity, Big Data and IoT: A literature review,” Inf. Process. Manage., vol. 59, no. 2, 2022, Art. no. 102888.

[13] P. Radoglou-Grammatikis et al., “Modeling, detecting, and mitigating threats against industrial healthcare systems: A combined software defined networking and reinforcement learning approach,” IEEE Trans. Ind. Informat., vol. 18, no. 3, pp. 2041–2052, Mar. 2022.

[14] G. N. Nguyen, N. H. L. Viet, M. Elhoseny, K. Shankar, B. B. Gupta, and A. A. A. El-Latif, “Secure blockchain enabled cyber-physical systems in healthcare using deep belief network with resnet model,” J. Parallel Distrib. Comput., vol. 153, pp. 150–160, 2021.

[15] S. Nifakos et al., “Influence of human factors on cyber security within healthcare organisations: A systematic review,” Sensors, vol. 21, no. 15, 2021, Art. no. 5119.

[16] A. Acar et al., “A lightweight privacy-aware continuous authentication protocol-paca,” ACM Trans. Privacy Secur., vol. 24, no. 4, pp. 1–28, 2021.

[17] P. Soni, J. Pradhan, A. K. Pal, and S. H. Islam, “Cybersecurity attack-resilience authentication mechanism for intelligent healthcare system,” IEEE Trans. Ind. Informat., early access, Jun. 1, 2022, doi: 10.1109/TII.2022.3179429.

[18] P. A. Apostolopoulos, E. E. Tsiropoulou, and S. Papavassiliou, “Riskaware data offloading in multi-server multi-access edge computing environment,” IEEE/ACM Trans. Netw., vol. 28, no. 3, pp. 1405–1418, Jun. 2020.

[19] K. Shankar and P. Eswaran, “RGB based multiple share creation in visual cryptography with aid of elliptic curve cryptography,” China Commun., vol. 14, no. 2, pp. 118–130, Feb. 2017. [20] A. Anand and A. K. Singh, “SDH: Secure data hiding in fused medical image for smart healthcare,” IEEE Trans. Computat. Social Syst., vol. 9, no. 4, pp. 1265–1273, Aug. 2022.

[21] Z. Li, C. Peng, W. Tan, and L. Li, “A novel chaos-based image encryption scheme by using randomly DNA encode and plaintext related permutation,” Appl. Sci., vol. 10, no. 21, 2020, Art. no. 7469.

[22] J. Chen, Z. Zhu, L. Zhang, Y. Zhang, and B. Yang, “Exploiting self-adaptive permutation-diffusion and DNA random encoding for secure and efficient image encryption,” Signal Process., vol. 142, pp. 340–353, 2018.

[23] Q. Zhang, J. Han, and Y. Ye, “Multi-image encryption algorithm based on image hash, bit-plane decomposition and dynamic DNA coding,” IET Image Process., vol. 15, no. 4, pp. 885–896, 2021.

[24] W. Xingyuan, W. Yu, Z. Xiaoqiang, and L. Chao, “A novel chaotic algorithm for image encryption utilizing one-time pad based on pixel level and dna level,” Opt. Lasers Eng., vol. 125, 2020, Art. no. 105851.

[25] E. Z. Zefreh, “An image encryption scheme based on a hybrid model of DNA computing, chaotic systems and hash functions,” Multimedia Tools Appl., vol. 79, no. 33-34, pp. 24993–25022, 2020. [26] K. Zhan, D. Wei, J. Shi, and J. Yu, “Cross-utilizing hyperchaotic and DNA sequences for image encryption,” J. Electron. Imag., vol. 26, no. 1, 2017, Art. no. 13021.

[27] X. Chai, Z. Gan, K. Yuan, Y. Chen, and X. Liu, “A novel image encryption scheme based on DNA sequence operations and chaotic systems,” Neural Comput. Appl., vol. 31, no. 1, pp. 219–237, 2019

[28] X. Yan, X. Wang, and Y. Xian, “Chaotic image encryption algorithm based on arithmetic sequence scrambling model and DNA encoding operation,” Multimedia Tools Appl., vol. 80, no. 7, pp. 10949–10983, 2021.

[29] I. Aouissaoui, T. Bakir, and A. Sakly, “Robustly correlated key-medical image for dna-chaos based encryption,” IET Image Process., vol. 15, no. 12, pp. 2770–2786, 2021.