ISTANBUL COMMERCE UNIVERSITY

GRADUATE SCHOOL OF SCIENCES

A LIGHTWEIGHT CRYPTOGRAPHY ALGORITHM FOR SMART CITIES AND IOT

Ahmed Mohsin Abed AL-JANABI

Supervisor
Asst. Prof. Dr. Ali BOYACI

MASTER THESIS
COMPUTER ENGINEERING DEPARTMENT
ISTANBUL – 2020

ACCEPTANCE AND APPROVAL PAGE

On 30/01/2020, **Ahmed Mohsin Abed AL-JANABI** successfully defended the thesis, entitled **"A Lightweight Cryptography Algoraithm for Smart Cities and IoT",** which he prepared after fulfilling the requirements specified in the associated legislation, before the jury members whose signatures are listed below. This thesis is accepted as a **Master's Thesis** by Istanbul Commerce University, Graduate School of Natural and Applied Sciences **Computer Engineering Department.**

Supervisor	**Asst. Prof. Dr. Ali BOYACI** Istanbul Commerce University
Jury Member	**Asst. Prof. Dr. Metin TURAN** Istanbul Commerce University
Jury Member	**Asst. Prof. Dr. Zeynep TURGUT** Haliç University

Approval Date: 19.02.2020

İstanbul Ticaret Üniversitesi, Fen Bilimleri Enstitüsünün 19.02.2020 Tarih ve 2020/281 numaralı Yönetim Kurulu Kararının 12. maddesi gereği ders yüklerini ve tez yükümlülüğünü yerine getirdiği belirlenen "AHMED MOHSIN ABED AL-JANABI" (TC: 99587864174) adlı öğrencinin mezun olmasına oy birliği ile karar verilmiştir.

Prof. Dr. Necip ŞİMŞEK
Acting Head of Graduate School of Natural and Applied Sciences

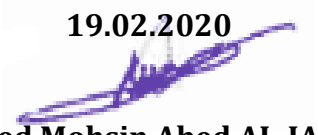
ACADEMIC AND ETHICAL RULES
DECLARATION OF CONFORMITY

In this thesis I prepared in accordance with the rules of thesis writing, Istanbul Commerce University, Institute of Science,

- I obtained all the information and documents in the thesis within the framework of academic rules.
- I present all visual, audio and written information and results in accordance with scientific moral rules.
- I refer to the related works in accordance with scientific norms in case of using others' works.
- I cited all the works I cited as a source.
- I did not make any distortions in the data used.
- and that I do not present any part of this thesis as another thesis study at this university or another university.

I declare.

19.02.2020



Ahmed Mohsin Abed AL-JANABI

CONTENTS

ABSTRACT

Master Thesis

A LIGHTWEIGHT CRYPTOGRAPHY ALGORITHM FOR SMART CITIES AND IOT

Ahmed Mohsin Abed AL-JANABI

Istanbul Commerce University
Graduate School Of Sciences And Engineering
Computer Engineering Department

Supervisor: Asst. Prof. Dr. Ali BOYACI

2020, 72 Pages

Cybersecurity is very important and is more important in smart cities, where that uses different types of electronic Internet of things (IoT) sensors to collect data from citizens, devices, and assets that are processed and analyzed to monitor and manage traffic and transportation systems, Energy, water networks, waste and environment management, crime detection, information systems, schools, libraries, hospitals, and other community services. All these services are subject to threats, risks, manipulation, and control of data through different environments such as software, networks, and devices, including sensors, especially as the sensors have limited specifications such as memory and energy used, they need protection, taking into account these specifications. One of the most important measures to protect data and information is encryption, but in the sensors, we need lightweight encryption due to its limited specifications. There are many cryptographic algorithms and despite the strength of the security level in some of them, but we need more improvements and development in the algorithms to increase the level of security and taking into account other specifications such as speed and the level of memory and energy consumption. In this thesis, we have created a new lightweight cryptography algorithm by developing and making some improvements to another algorithm to be more efficient and more sensitive to sensor features and advanced security to make Smart Cities and IoT more secure.

Key Words: Cryptography, cyber security, internet of things, IoT, lightweight cryptography, sensor, smart cities, wireless sensor network.

ÖZET

Yüksek Lisans Tezi

AKILLI ŞEHİRLER VE IOT İÇİN BİR HAFİF KRİPTOGRAFİ ALGORİTMASI

Ahmed Mohsin Abed AL-JANABI

İstanbul Ticaret Üniversitesi
Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Dr. Öğr. Üyesi Ali BOYACI

2020, 72 sayfa

Siber güvenlik çok önemlidir ve trafik ve ulaşım sistemlerini izlemek ve yönetmek için işlenen ve analiz edilen vatandaşlardan, cihazlardan ve varlıklardan veri toplamak için farklı türde elektronik İnternet (IoT) sensörlerini kullanan akıllı şehirlerde daha önemlidir, Enerji, su ağları, atık ve çevre yönetimi, suç tespiti, bilgi sistemleri, okullar, kütüphaneler, hastaneler ve diğer toplum hizmetleri. Tüm bu hizmetler, özellikle sensörler kullanılan bellek ve enerji gibi sınırlı özelliklere sahip olduğundan, korumaya ihtiyaç duyulan yazılımlar, ağlar ve cihazlar gibi farklı ortamlar aracılığıyla verilerin tehditlerine, riskleri, manipülasyonuna ve kontrolüne tabidir. Verileri ve bilgileri korumak için en önemli önlemlerden biri şifrelemedir, ancak çünkü sensörlerin özellikleri sınırlıdır nedeniyle hafif şifrelemeye ihtiyacımız vardır. Birçok şifreleme algoritması vardır ve bazılarında güvenlik düzeyinin gücüne rağmen, güvenlik düzeyini artırmak ve hız, hafıza ve enerji tüketim seviyesi gibi diğer özellikleri dikkate almak için algoritmalarda daha fazla iyileştirmeye ve geliştirmeye ihtiyacımız var. Bu tezde, Akıllı Şehirler ve IoT yi daha güvenli hale getirmek için sensör özelliklerine daha verimli ve daha duyarlı olmak için başka bir algoritmada bazı iyileştirmeler geliştirerek yapmak ve geliştirerek yeni bir hafif şifreleme algoritması oluşturduk.

Anahtar Kelimeler: Akıllı şehirler, hafif kriptografi, IoT, kablosuz sensör ağı, kriptografi, nesnelerin interneti, sensör, siber güvenlik.

ACKNOWLEDGEMENT

I thank Allah ALMIGHTY, who inspired me to aspire and who has guided my steps. I extend my sincere thanks and gratitude to Dr. Ali BOYACI who supervised this work and supported me throughout my scientific research and this thesis. I thank all those who supported me, including my family.

Ahmed Mohsin Abed AL-JANABİ
ISTANBUL, 2020

FIGURES

TABLES

SYMBOLS AND ABBREVATIONS

$\oplus$	XOR
$\odot$	XNOR
$\sim$	NOT
$\#$	Concatenation
$\gg$	Right Shift
$\ll$	Left Shift

1. INTRODUCTION

1.1. Smart Cities and Internet of Things (IoT)

1.1.1. Smart cities

The expression “smart” was used to describe the individuals that fast, spontaneous, and reliable but not devices or lifestyles. However, in the modern era, the world has changed rapidly and dramatically through the development of technology and has become a smart term referring to smart devices, services, and elements that respond quickly, reliably and flexibility, as smart devices, smart cities, and a lot of smart elements other.

There are several definitions of the smart city, and to define it simply, the smart city is an urban area that uses and integrates many information and communication technology to manage the services and resources of the city and help the citizens to build and modify them services flexibly, effectively and safely (Rassia and Pardalos, 2017).

As urban populations increase, there will be great difficulties in managing services such as transportation, health care, education, energy, etc. To facilitate the management and improvement of services, Internet services should be developed in the cities to become smarter. Smart cities need to develop digital (information technology) and physical infrastructure, and activate the Internet of Things and smart societies (including stakeholders such as governments, citizens, companies and organizations) and the interaction between them (Dustdar, 2017).

Among the most important elements in smart cities are buildings and smart homes, smart governance, smart society, smart transportation, smart healthcare, smart energy, smart industry, smart education, smart environment, smart agriculture, smart security, etc. In addition, there are sub-elements from main elements such as smart traffic systems, smart lighting, smart vehicles, smart car parks, smart irrigation, smart waste containers, smart home

appliances, and others. Figure 1.1. It illustrates some of the main elements of smart cities.

Figure 1.1. Smart city

A report of the United Nations on 16 May 2018 Indicates that 55% of the world lives in the cities and will attend to 68% in 2050, and it emphasizes that governments should concentration on integrated policies to improve the lives of both cities and rural dwellers (United Nations, 2018). In addition to the application of some Internet of things services in the cities, some can also be applied in rural areas, for example in irrigation and agriculture.

There are many benefits in the trend toward urbanization and many challenges. Consolidating a growing population in a smaller space can help conserve

resources and other benefits, and will have some Challenges, as cybersecurity threats especially in smart cities and Internet of things applications.

1.1.2. Internet of Things (IoT)

The Internet of Things (IoT) describes the network of physical objects "things" that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the internet via special protocols, to achieve intelligent recognition, location, tracking, monitoring and management (Kim, et, al., 2017). These devices range from ordinary household objects to sophisticated industrial tools.

The Internet of Things is spreading widely, and its application and services will change our lives. In 2020, it is expected that the number of Internet of Things devices will reach 50 billion devices, and the Internet of Things will have a major impact on the technology industry (Yeo, et al., 2014).

The devices of the internet of things (as sensors) are highly constrained, with memory and available energy limited, and its requirements are strict low-cost, and its essential requirements are storage, processing, and analytics of IoT data and transform them into beneficial information (Cirani, et al., 2019). IoT devices are a victim of attacks due to low memory capacity, storage and processing capacity, and therefore the measures needed to protect them are difficult, so it is important to work and research on finding better solutions to protect them (Gurunath, et al., 2018). There are a lot of sensors and other devices and technologies that support and contribute significantly to the development of the Internet of things such as wireless sensors network (WSN) and radio-frequency identification (RFID), cloud computing, devices and smart phones (Aksu and Aydin, 2019).

The components of the Internet of Things system are low energy and cost and need careful care in designing hardware and software, application of algorithms,

security and safety. The Internet of Things system is made up primarily of sensors, where the sensors are widespread (Serpanos and Wolf, 2018).

Figure 1.2. Internet of things (IoT)

IoT systems are used in a wide range of applications, where industrial sensors used to monitor industrial processes, equipment condition and product quality. In smart buildings, for example, sensors check the building's condition and temperature to control air conditioning, heating, and ventilation, and control the level of lighting and the need for them to reduce operating costs and energy consumption, as well as sensors work to monitor the condition of structures in smart buildings. In addition, in smart homes, sensors monitor and control household appliances, such as refrigerators, air-conditioners, televisions, Food warmers and more. In transportation, sensors are used to monitor vehicles, fuel consumption, emission, roads, traffic etc. In health, the sensors in medical systems work to monitor patients in hospitals, in ambulances, and in doctors 'clinics (Serpanos and Wolf, 2018). In security, smart sensors are used in smart

surveillance cameras, smart locks, and emergency monitoring. In the environment, sensors are used to monitor waste containers, environmental purity, pollutants, etc. In agriculture, the sensors control the irrigation of crops by measuring the degree of soil moisture, are used in livestock, and perform other tasks. In electric power grids, sensors monitor and control energy consumption. In commerce, sensors work to monitor materials, goods issued, imported and stored, there are many applications of Internet of things that are used in various fields and most of the tasks performed by sensors in systems and applications of Internet of Things are applied in smart cities as mentioned previously.

The new Internet of Things (IoT) applications is empowering Smart City activities around the world. It gives the capacity to remotely screen, oversees and control gadgets, and to make new bits of knowledge and noteworthy data from huge surges of constant information.

There are many difficulties in the development of Internet of Things applications that need to world keen endeavors in scholarly and industry research, to reach better ideas, innovations and solutions to improve and develop administration and services for smart cities that are highly dependent on the Internet of things, to overcome difficulties, challenges, risks and threats including cybersecurity.

1.1.3. Challenges

Despite the great benefits that smart cities offer to their citizens and the contribution of Internet of Things applications, there are also great challenges facing them.

The new networks of sensors, data analytics and makers of decision, bring many challenges. in addition, a lot of devices and systems aren't cyber resilient, posing a threat to the safety and security of the citizens, governments, services, data and information.

In 2014, researchers from the University of Michigan hacked the traffic lights of a hundred of the city's intersections, proving security flaws existed that had the potential to cause serious accidents (University of Michigan, 2014). During 2017, hackers turned on 156 severe sirens at midnight in the USA, Dallas, that causing a surge of 911 calls and distress (The Guardian, 2017). Also in the United States of America, some persons hacked the Jeep Cherokee car system away ten miles and tampered with the air conditioner, wipers of glass, radio sound, engine and brakes, and deflected it from the road (The Telegraph, 2015). Ransomware was used in the past few years to disrupt Dublin's local tram system in Ireland, and obstruct railway tickets systems and air traffic control in Stockholm, Sweden, and to shake-down plants of power from both Johannesburg in South Africa and Hyderabad in India (Muggah and Goodman, 2019).

There are more reasons that lead to the exposure of smart devices, applications and services to penetration and manipulation. Among these reasons is the use of devices vulnerable to infection in smart cities and the Internet of things due to the presence of loopholes and weak encryption, and despite the existence of a lot of instructions to make it safe, but it needs to be developed, Responsible authorities, such as companies and manufacturers, must develop devices, systems, networks, encryption, and manage and address vulnerabilities and security flaws.

Security vulnerabilities may increase with the increase in the proliferation of Internet of Things devices, and this is an opportunity for cyber criminals to carry out cyber attacks, as a research for Symantec (the American company specialized in the field of information security) found weaknesses in 50 devices from commercially available devices, including a smart door lock that can be opened online without a password (Symantec, 2018). Internet devices that are not secure can serve as entry points for cyber-attacks by allowing hackers to tamper with devices such as reprogramming, causing malfunctions and

manipulating the security and integrity of information and data, so security in the Internet of things cannot be ignored (Oracevic, et al., 2017).

Over the years, cities become smarter, but obtaining protection for the Internet of things determines the difference between a smart city and a secure city.

Despite the tremendous benefits provided by the Internet of things and smart cities applications in many areas and services such as reducing energy consumption by controlling lighting in streets and buildings and refrigeration and air conditioning devices through the work of sensors and instructing them to switch off and operate when needed, as well as reducing noise and emissions in the environment and monitoring the regulation of movement traffic and choosing the most appropriate way for emergency response cars and many other benefits. But there are serious challenges in the presence of many sensors and the Internet of things and interconnected networks where they are the target of cyber-attacks, for example when a for cyber attackers to turn off smart electric energy, there will be severe consequences and disturbances that have major effects on people's lives, all electrical powered devices, including devices used in hospitals to monitor, check and treat patients, will stop, and in food storage stores the food will be destroyed after the cooling devices have stopped, and surveillance cameras the security will stop, automated teller machines and electronic payment, or when traffic signals are compromised and tampered with, will cause chaos, terrible accidents, and many dangers.

1.2. Cyber Security and Cryptography

1.2.1. Cyber security

Cyber security is protecting the confidentiality, integrity and availability of information-whether it is personally identifiable information, email or other of communication, credit card numbers, intellectual property or government secrets and other data, information and the services that connect through the internet (Singer and Friedman, 2014).

Confidentiality is representing keeping data private, and privacy is not just some social or political goal. The protection of information in the digital world is of the utmost importance for the value of this information. Confidentiality is supported by technical tools, the most important of which is encryption, in addition to access control and legal protections. Integrity means that the system and its data have not been manipulated or changed or edited without authorization. There must be confidence that the system will be both available and behave as expected. Availability refers to being able to use the system as anticipated. It's not merely the system going down that makes availability a security concern; such as that software errors and that happen to computers in a lot of time, and when someone tries to exploit the lack of availability in some way, it becomes a security issue (Kostopoulos, 2017).

An attacker could be depriving users of a system that they depend on (such as loss of GPS that would hamper military units in conflict and other), or by merely threatening the loss of a system, as in a "ransomware" attack, there are many examples of such ransoms, where these ransoms range from hacks on individual bank accounts to global blackmail.

Threats and cyber-attacks occur by Hackers, organized criminal, commercial competitors and government intelligence agencies and others.

That threats and cyber-attacks are increasingly active on the internet and engaged in various kinds of theft, espionage, disruption and sabotage, manipulation and control, especially the smart city services and Internet of things applications are subject to serious threats. Where threaten the global financial systems, environment, traffic control systems, electrical power grids, nuclear power plants telecommunications systems, healthcare systems and more from applications of smart cities and internet of things. Cyber espionage is a pervasive epidemic; and even the smart companies and government institutions in the world have terabytes of intellectual property and financial assets being lost annually via the Internet, with cybercrime the fast growing in the world, in some segments growing 300 percent per year, cyber-crimes are

responsible for more than $1 trillion USD in stolen funds and other assets (Mowbray, 2014).

Therefore, the cybersecurity effective is a critical capability for the defense and preservation of societies and the services.

Among the most important factors that contribute to cyber security is protect data and information in all environments such as networks and devices and others. Here comes the biggest role for cryptography, that include encryption of data and information to prevent intruders and unauthorized to see or tamper with or control, because of the disclosure of data and information by people It is very dangerous because some data is very sensitive, especially with the development and spread of the Internet of things in smart cities and in many fields, but sensors for the Internet of things have limited specifications such as the size of memory and limited energy, so they require lightweight cryptography algorithms to take into account their specifications and work with high efficiency, and it needs to be lightweight algorithms with high-level of security.

1.2.2. Cryptography

Cryptology (Cryptology includes Cryptography and Cryptanalysis as in figure 1.3) was born among Arabs (Kahn, 1996). Many Arab and Muslim Scientists and philosophers who used cryptography and cryptanalysis and authored books about it for hundreds of years, Such as the famous scientist and philosopher Al-Kindi (he was born around 801 A.D. and was raised in Baghdad, Iraq) and others (Broemeling, 2012).

Cryptanalysis is the science of cryptography analysis and sometimes the art of breaking cryptosystems.

Cryptography is hiding messages and information to deny access unauthorized accessed.

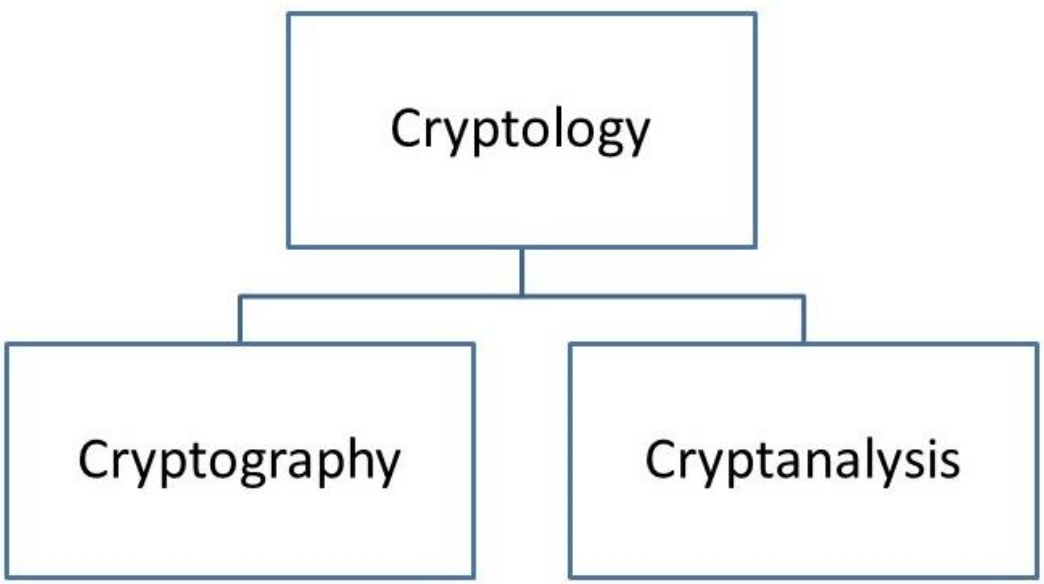


Figure 1. 3. Cryptology diagram

Cryptography includes Symmetric cryptography and asymmetric cryptography and hash function. Where hash function it is a coding method that is usually used in protocols, and it computes the message digest for short and fixed-length bits, Symmetric cryptography uses one key for encryption and decryption and it includes two types (block cipher and stream cipher), while Asymmetric cryptography uses two public and private keys (Paar and Pelzl, 2010). The figure 1.4 show cryptography diagram.

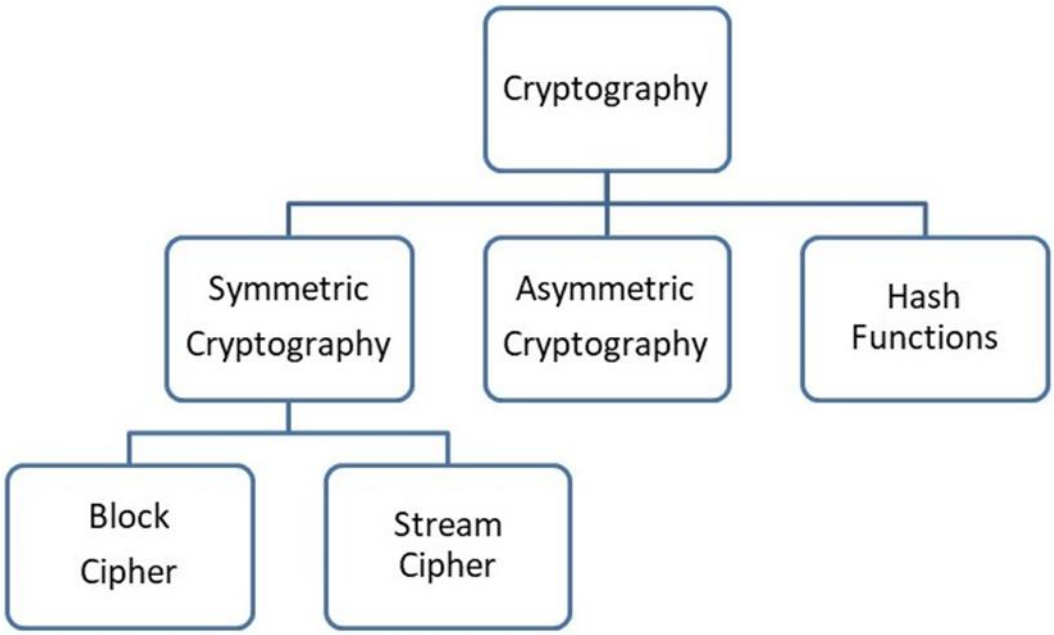


Figure 1.4. Cryptography diagram

Cryptography has been used for a long time, is very important, and has been used in World War II. Today, encryption is used in many fields, services and applications, including in communications, mobile applications, credit cards, business and government websites, banking systems and in many fields, and has become more important with Internet applications, things and smart city services where everything is working in cyberspace.

Symmetric algorithms mean using the same key in encryption and decryption, while Asymmetric algorithm means using two keys (Public key and privet key) public key for encryption and privet key for decryption, and Cryptographic Protocols are dealing with the application of cryptographic algorithms, like hash functions (Paar and Pelzl, 2010).

From symmetric algorithms, block cipher and stream cipher, a stream cipher is that encrypts a digital data stream one bit or one byte at one time and a block cipher uses a block of plaintext as a whole and used to produce an cipher text block of equal length, for example a block size 64 bits or 128 bits (Stallings, 2011), like AES and DES, and others algorithms.

In this thesis, we will talk about block cipher, and we will dedicate part about block cipher algorithms and specifically lightweight cryptographic algorithms, because it is used to secure Internet of things (secure the sensors) and smart cities.

1.2.3. Lightweight cryptography

The lightweight Cryptography algorithms designed to work in many highly restrictive environments (ex. sensor networks, healthcare, distributed control systems, Internet of things, and electronic physical systems) where they use low power and will use circuits much more limited than those on the simplest mobile phones. These devices are usually connected to each other wirelessly to work in coordination to accomplish some tasks. Because cryptographic algorithms designed for desktop environments and servers, most of them are not compatible with restricted devices (NIST, 2017).

When using a small computing power should be using lightweight cryptography algorithm to secure it, so can use lightweight block ciphers, lightweight stream ciphers or lightweight hash function. As for the lightweight block ciphers, there are many algorithms, we include many of its below according to the design of the structure.

Substitution-Permutation Network (AES, KLEIN, LED, Midori, Mysterion, SKINNY, Zorro, Fantomas, Robin, Noekeon, PRIDE, Rectangle, mCrypton, MANTIS, PRESENT, PRINCE, SPARX and others). Feistel Networks (Chaskey Cipher, HIGHT, LEA, RC5, SIMECK, SIMON, SPECK, XTEA, DESLX, GOST revisited, ITUbee, KASUMI, MISTY, LBlock, RoadRunneR, SEA, CLEFIA, Piccolo, TWINE and others), Other Designs (KTANTAN, KATAN and others) (CryptoLUX Wiki, 2017). Despite the abundance of algorithms and the efficiency of some of them, it still needs to develop, improve and search for better algorithms and solutions because of the difficulty of obtaining the required level of security in the lightweight cryptography algorithms to take into account the specifications of the restricted devices at the same time.

2. REVIEW OF THE LITERATURE

There are more cryptography algorithms and despite the strength of some of them, but they still need improvements to improve their efficiency, because of its importance and the difficulty of obtaining the maximum efficiency because many features must be considered at the same time as security and speed and energy consumption especially in lightweight cryptography algorithms.

Can increase of the security level for the cryptography algorithms by increasing the size of the block, the size of the key or the number of rounds (Stallings, 2010), but sometimes negatively affecting its speed and consumption of energy and memory. In addition to increasing the block size, key size and number of rounds that lead to increase the security level, the design of the algorithm's structure, function and generator of sub keys and operators play a very large role in the strength level of the algorithm's security.

Can improve the strength of certain algorithms by making some changes and improvements, for example, Triple DES or 3DES algorithm derived from the DES algorithm by adding some changes and improvements such as increasing the size of the key or other changes that would lead to an increase in the strength of the algorithm. As an example of lightweight algorithms is the case for the TEA (Tiny Encryption Algorithm) and XTEA (Extended Tiny Encryption Algorithm), XXTEA. TEA and XTEA is a Feistel cipher with 64-bit block, 128-bit key and 64 rounds, but there are some differences, including a more complex key-schedule, XORs, other additions and rearrangement of the shifts' Also, and XXTEA algorithm is a Feistel cipher with 64-bit block, 128-bit key and also contains some changes.

In block cipher, some algorithms use Feistel structure where the Block is split into two halves and their size is equal in the balanced algorithms like DES, Simon and others, and in two different sizes in the unbalanced algorithms as in the Skipjack algorithm, XXTEA and others. One of the major advantage of using Feistel structure is that the encryption and decryption operations are almost

same. Some other algorithms use SPN structure (Substitution-Permutation network), such as AES and present and prince and others, several alternating rounds of substitution and permutation lead to the confusion and spread of Shannon properties that require changing the encryption text in a pseudo-random way (Usman, 2017).

There is an algorithm (SIT algorithm with key size 64 bit, block size 64 bit and five rounds), it is a mixture based on Feistel and SP networks. Thus making use of the advantages of both approaches to develop a lightweight algorithm that presents more security in the IoT environment while keeping the computational complexity at a moderate level (Usman, 2017).

Although the SIT algorithm is designed to reduce power and memory consumption and increase speed, it needs to improve some processes add operations that are more complex, in the structure and sub-key generator and increase of key size, to increase of security level with taking into account the consumption of energy and memory on the same time.

To be the cryptography algorithms more secure, the key size must be at least 128 bits as studies suggest in recent years from the National Institute of Standards and Technology, NIST (Barker, 2016). In addition to when the key size larger, the key space will be larger and more resistant against brute force attack.

The proposed algorithm in this article like SIT algorithm but with the key size 128 bit and block size 64 bit and nine rounds, and added many operations for algorithm structure and sub-key generator and add other functions. And from the operations that added, operations that used in TEA XTEA XXTEA algorithms like shift operation and Delta number, Delta is the golden number or 9E3779B9 (Andem, 2003; Maitra, and Yelamarthi, 2019).

We made the improvements above to make it more complex and efficiency and more secure, and according to the analyzes, studies and comparisons in the next

chapter about the proposed algorithm, that it is more secure and more efficient compared with other algorithms.

3. PROPOSED ALGORITHM

3.1. Proposed Algorithm and Encryption

The proposed algorithm is a lightweight encryption algorithm with a block size equal 64-bit, key size equal 128-bit and number of rounds is nine. The proposed algorithm is a mixture of Feistel structure and SPN structure (Substitution-Permutation network).

Some algorithms have the size of sub-keys equal to the size of the main keys and some algorithms have the main key size different from the sub-keys sizes, in the proposed algorithm have sub-keys size is different from the master key size where the size of the sub-keys is 16 bits while the key size is 128-bit.

The main key divided into eight segments, then each segment equal 16 bit and each one divided to eight sub-segments, each sub-segment equal to two bits, this means we have 64 sub-segments of 2 bits, and through the substitution of segments, we will get on new eight segments, as shown in the following equations and figure (3.2).

Where S means a segment, s means a sub-segment.

$$S_a = s_1 \Vdash s_2 \Vdash s_{17} \Vdash s_{18} \Vdash s_{33} \Vdash s_{34} \Vdash s_{49} \Vdash s_{50} \quad (3.1)$$

$$S_b = s_9 \Vdash s_{10} \Vdash s_{25} \Vdash s_{26} \Vdash s_{41} \Vdash s_{42} \Vdash s_{57} \Vdash s_{58} \quad (3.2)$$

$$S_c = s_3 \Vdash s_{11} \Vdash s_{19} \Vdash s_{27} \Vdash s_{35} \Vdash s_{43} \Vdash s_{51} \Vdash s_{59} \quad (3.3)$$

$$S_d = s_4 \Vdash s_{12} \Vdash s_{20} \Vdash s_{28} \Vdash s_{36} \Vdash s_{44} \Vdash s_{52} \Vdash s_{60} \quad (3.4)$$

$$S_e = s_5 \Vdash s_6 \Vdash s_{21} \Vdash s_{22} \Vdash s_{37} \Vdash s_{38} \Vdash s_{53} \Vdash s_{54} \quad (3.5)$$

$$S_f = s_{13} \Vdash s_{14} \Vdash s_{29} \Vdash s_{30} \Vdash s_{45} \Vdash s_{46} \Vdash s_{61} \Vdash s_{62} \quad (3.6)$$

$$S_b = s_7 \Vdash s_{15} \Vdash s_{23} \Vdash s_{31} \Vdash s_{39} \Vdash s_{47} \Vdash s_{55} \Vdash s_{63} \quad (3.7)$$

$$S_b = s_8 \Vdash s_{16} \Vdash s_{24} \Vdash s_{32} \Vdash s_{40} \Vdash s_{48} \Vdash s_{56} \Vdash s_{64} \quad (3.8)$$

After producing eight segments the Function 1 used for the segments, the functions has special design and depends on P and Q tables show figure 3.1, the transformations made by P and Q are shown in the tables 1 and 2.

$Ka_i f = f(S_i)$ (3.9)

Where i = (a, b, c, d, e, f, g, h)

The next step is to get $Ka_i f$ by passing the 16-bits of S_i to the function1 as shown in equation (3.9).

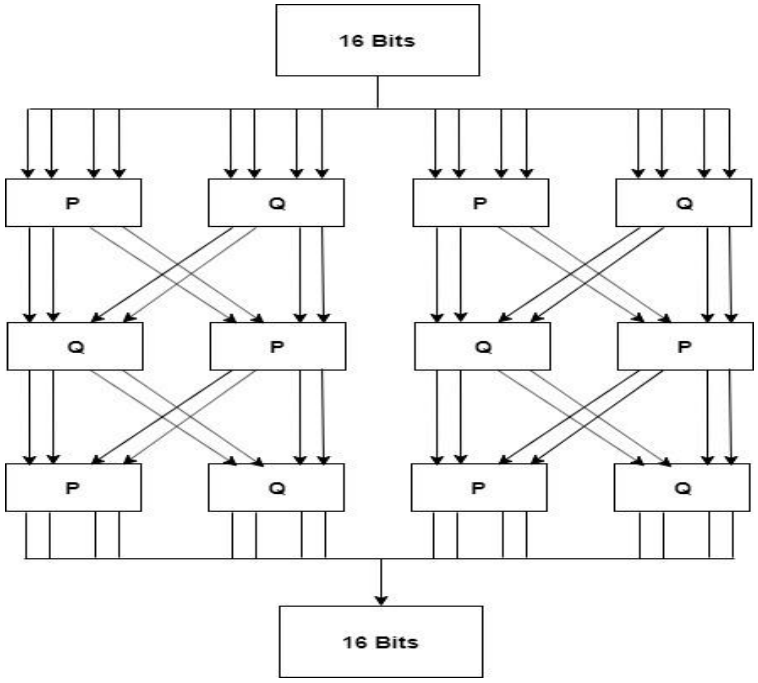


Figure 3.1. Function 1

In the function 1 (Figure 3.1), P and Q, each one represents four bits, the arrows indicate the location distribution of the bits in the sixteen bites, Where the location of the bits in the P in function 1 is according to their distribution in the job and according to the P table, and the location of the bits in the Q in function 1 is according to their distribution in the job and according to the Q table. In addition to the function 2, function 3 and function 4, it works with the same principle. For example, the first bit in function 1 is equal to 3 according to table P, then after moving to Q in the first function, it is equal to 6 according to the table Q, where the number 6 corresponds to the number 3 and after moving to P in function 1 it is equal to B according to table P, then the B corresponds to 6, and so on for the others of the bits and functions.

The Function 1 applied in sub-keys generator and in algorithm structure (Encryption structure), while the functions 2, 3 and 4 applied in Encryption structure only.

The tables and functions perform to linear and non-linear transformations for the bits to get confusion and diffusion. See table 1 and 2. The tables P and Q as S-Box (Substitution box) for the bits of sub-blocks and sub-keys through the functions.

Table 3.1. P table

S_{i}	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
P(S_{i})	3	F	E	0	5	4	B	C	D	A	9	6	7	8	2	1

Table 3.2. Q table

$\boldsymbol{S}_{\mathrm{i}}$	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Q($\boldsymbol{S}_{\mathrm{i}}$)	9	E	5	6	A	2	3	C	F	0	4	D	7	B	1	8

The output (a, b, c, d, e, f, g, h) of each function is arranged in a 4×4 matrix named M shown next, but some operations are applied to some functions before arranging them in the matrixes, as in equations (3.10, 3.11, 3.12 and 3.13).

$$\mathrm{bs} = b \ll 4 \tag{3.10}$$

$$\mathrm{ds} = d \gg 8 \tag{3.11}$$

$$\mathrm{fs} = f \ll 4 \tag{3.12}$$

$$\mathrm{hs} = h \gg 8 \tag{3.13}$$

$$M_{\mathrm{a}} = \begin{bmatrix} K\mathrm{a}_1\mathrm{f}_1 & K\mathrm{a}_1\mathrm{f}_2 & K\mathrm{a}_1\mathrm{f}_3 & K\mathrm{a}_1\mathrm{f}_4 \\ K\mathrm{a}_1\mathrm{f}_5 & K\mathrm{a}_1\mathrm{f}_6 & K\mathrm{a}_1\mathrm{f}_7 & K\mathrm{a}_1\mathrm{f}_8 \\ K\mathrm{a}_1\mathrm{f}_9 & K\mathrm{a}_1\mathrm{f}_{10} & K\mathrm{a}_1\mathrm{f}_{11} & K\mathrm{a}_1\mathrm{f}_{12} \\ K\mathrm{a}_1\mathrm{f}_{13} & K\mathrm{a}_1\mathrm{f}_{14} & K\mathrm{a}_1\mathrm{f}_{15} & K\mathrm{a}_1\mathrm{f}_{16} \end{bmatrix} \tag{3.14}$$

$$M_{\mathrm{b}} = \begin{bmatrix} K\mathrm{a}_2\,\mathrm{f}_1 & K\mathrm{a}_2\mathrm{f}_2 & K\mathrm{a}_2\mathrm{f}_3 & K\mathrm{a}_2\mathrm{f}_4 \\ K\mathrm{a}_2\mathrm{f}_5 & K\mathrm{a}_2\mathrm{f}_6 & K\mathrm{a}_2\mathrm{f}_7 & K\mathrm{a}_2\mathrm{f}_8 \\ K\mathrm{a}_2\mathrm{f}_9 & K\mathrm{a}_2\mathrm{f}_{10} & K\mathrm{a}_2\mathrm{f}_{11} & K\mathrm{a}_2\mathrm{f}_{12} \\ K\mathrm{a}_2\mathrm{f}_{13} & K\mathrm{a}_2\mathrm{f}_{14} & K\mathrm{a}_2\mathrm{f}_{15} & K\mathrm{a}_2\mathrm{f}_{16} \end{bmatrix} \tag{3.15}$$

$$M_{\mathrm{c}} = \begin{bmatrix} Ka_3 f_1 & Ka_3 f_2 & Ka_3 f_3 & Ka_3 f_4 \\ Ka_3 f_5 & Ka_3 f_6 & Ka_3 f_7 & Ka_3 f_8 \\ Ka_3 f_9 & Ka_3 f_{10} & Ka_3 f_{11} & Ka_3 f_{12} \\ Ka_3 f_{13} & Ka_3 f_{14} & Ka_3 f_{15} & Ka_3 f_{16} \end{bmatrix} \quad (3.16)$$

$$M_{\mathrm{d}} = \begin{bmatrix} Ka_4 f_1 & Ka_4 f_2 & Ka_4 f_3 & Ka_4 f_4 \\ Ka_4 f_5 & Ka_4 f_6 & Ka_4 f_7 & Ka_4 f_8 \\ Ka_4 f_9 & Ka_4 f_{10} & Ka_4 f_{11} & Ka_4 f_{12} \\ Ka_4 f_{13} & Ka_4 f_{14} & Ka_4 f_{15} & Ka_4 f_{16} \end{bmatrix} \quad (3.17)$$

$$M_{\mathrm{e}} = \begin{bmatrix} Ka_5 f_1 & Ka_5 f_2 & Ka_5 f_3 & Ka_5 f_4 \\ Ka_5 f_5 & Ka_5 f_6 & Ka_5 f_7 & Ka_5 f_8 \\ Ka_5 f_9 & Ka_5 f_{10} & Ka_5 f_{11} & Ka_5 f_{12} \\ Ka_5 f_{13} & Ka_5 f_{14} & Ka_5 f_{15} & Ka_5 f_{16} \end{bmatrix} \quad (3.18)$$

$$M_{\mathrm{f}} = \begin{bmatrix} Ka_6 f_1 & Ka_6 f_2 & Ka_6 f_3 & Ka_6 f_4 \\ Ka_6 f_5 & Ka_6 f_6 & Ka_6 f_7 & Ka_6 f_8 \\ Ka_6 f_9 & Ka_6 f_{10} & Ka_6 f_{11} & Ka_6 f_{12} \\ Ka_6 f_{13} & Ka_6 f_{14} & Ka_6 f_{15} & Ka_6 f_{16} \end{bmatrix} \quad (3.19)$$

$$M_{\mathrm{g}} = \begin{bmatrix} Ka_7 f_1 & Ka_7 f_2 & Ka_7 f_3 & Ka_7 f_4 \\ Ka_7 f_5 & Ka_7 f_6 & Ka_7 f_7 & Ka_7 f_8 \\ Ka_7 f_9 & Ka_7 f_{10} & Ka_7 f_{11} & Ka_7 f_{12} \\ Ka_7 f_{13} & Ka_7 f_{14} & Ka_7 f_{15} & Ka_7 f_{16} \end{bmatrix} \quad (3.20)$$

$$M_{\mathrm{h}} = \begin{bmatrix} Ka_8 f_1 & Ka_8 f_2 & Ka_8 f_3 & Ka_8 f_4 \\ Ka_8 f_5 & Ka_8 f_6 & Ka_8 f_7 & Ka_8 f_8 \\ Ka_8 f_9 & Ka_8 f_{10} & Ka_8 f_{11} & Ka_8 f_{12} \\ Ka_8 f_{13} & Ka_8 f_{14} & Ka_8 f_{15} & Ka_8 f_{16} \end{bmatrix} \quad (3.21)$$

Then apply on XNOR on matrix M_{a} and opposite it, and apply XOR on matrix M_{h} and opposite it as follow equations. That to make the sub-keys more complexity and more independent and from this sub-keys, K9 because those sub-keys shared by creating sub-key nine.

$$M_{\mathrm{a2}} = \left(\overline{M_{\mathrm{a}} \oplus \overline{M_{\mathrm{a}}}}\right) \quad (3.22)$$

$$M_{\mathrm{h2}} = M_{\mathrm{h}} \oplus \overline{M_{\mathrm{h}}} \quad (3.23)$$

To obtain the matrices are transformed into eight arrays of 16 bits, the arrangement of these bits are shown in next equations.

$$\mathrm{A} = a_4 \shuffle a_3 \shuffle a_2 \shuffle a_1 \shuffle a_5 \shuffle a_6 \shuffle a_7 \shuffle a_8 \shuffle a_{12} \shuffle a_{11} \shuffle a_{10} \shuffle a_9 \shuffle a_{13} \shuffle a_{14} \shuffle a_{15} \shuffle a_{16} \quad (3.24)$$

$$B = b_1 \# b_5 \# b_9 \# b_{13} \# b_{14} \# b_{10} \# b_6 \# b_2 \# b_3 \# b_7 \# b_{11} \# b_{15} \# b_{16} \# b_{12} \# b_8 \# b_4 \quad (3.25)$$

$$C = c_1 \# c_2 \# c_3 \# c_4 \# c_8 \# c_7 \# c_6 \# c_5 \# c_9 \# c_{10} \# c_{11} \# c_{12} \# c_{16} \# c_{15} \# c_{14} \# c_{13} \quad (3.26)$$

$$D = d_{13} \# d_9 \# d_5 \# d_1 \# d_2 \# d_6 \# d_{10} \# d_{14} \# d_{15} \# d_{11} \# d_7 \# d_3 \# d_4 \# d_8 \# d_{12} \# d_{16} \quad (3.27)$$

$$E = e_4 \# e_3 \# e_2 \# e_1 \# e_5 \# e_6 \# e_7 \# e_8 \# e_{12} \# e_{11} \# e_{10} \# e_9 \# e_{13} \# e_{14} \# e_{15} \# e_{16} \quad (3.28)$$

$$F = f_1 \# f_5 \# f_9 \# f_{13} \# f_{14} \# f_{10} \# f_6 \# f_2 \# f_3 \# f_7 \# f_{11} \# f_{15} \# f_{16} \# f_{12} \# f_8 \# f_4 \quad (3.29)$$

$$G = g_1 \# g_2 \# g_3 \# g_4 \# g_8 \# g_7 \# g_6 \# g_5 \# g_9 \# g_{10} \# g_{11} \# g_{12} \# g_{16} \# g_{15} \# g_{14} \# g_{13} \quad (3.30)$$

$$H = h_{13} \# h_9 \# h_5 \# h_1 \# h_2 \# h_6 \# h_{10} \# h_{14} \# h_{15} \# h_{11} \# h_7 \# h_3 \# h_4 \# h_8 \# h_{12} \# h_{16} \quad (3.31)$$

From next equations, we will get on sub-keys (K1, K3, K5, K7), through shift five bits to the right for previous arrays (A, C, E, G), we will get on sub-keys (K2, K4, K6, K8) through shift nine bits to the left for previous arrays (B, D, F, H). Then we will get on sub key nine (K9), by use XOR of shift five bits to the left for T1 (Equation 3.40) and shift nine bits to the right for T2 (Equation 3.41), where T1 equal XOR of K1 opposite (NOT) and D, T2 equal XOR of K5 and NOT of H. where K1 means sub-key 1 and K2 means sub-key 2 and so on with the others sub-keys.

$$K1 = A \gg 5 \quad (3.32)$$

$$K2 = B \ll 9 \quad (3.33)$$

$$K3 = C \gg 5 \quad (3.34)$$

$$K4 = D \ll 9 \quad (3.35)$$

$$K5 = E \gg 5 \quad (3.36)$$

$$K6 = F \ll 9 \quad (3.37)$$

$$K7 = G \gg 5 \quad (3.38)$$

$$K8 = H \ll 9 \quad (3.39)$$

The ninth sub-key or K9 is obtained through some operations as shown in the following equations:

$$\mathrm{T1} = \overline{K1} \oplus D \tag{3.40}$$

$$\mathrm{T2} = K5 \oplus \overline{H} \tag{3.41}$$

$$\mathrm{K9} = (T1 \ll 5) \oplus (T2 \gg 9) \tag{3.42}$$

So we get the ninth sub-key (K9) through the equations 3.40, 3.41 and 3.42, to make the sub-key 9 more independent from other sub-keys. In the next figure (Figure 3.2) we will see the details how getting of sub-keys and the operations that share to produce it through the sub-keys generator.

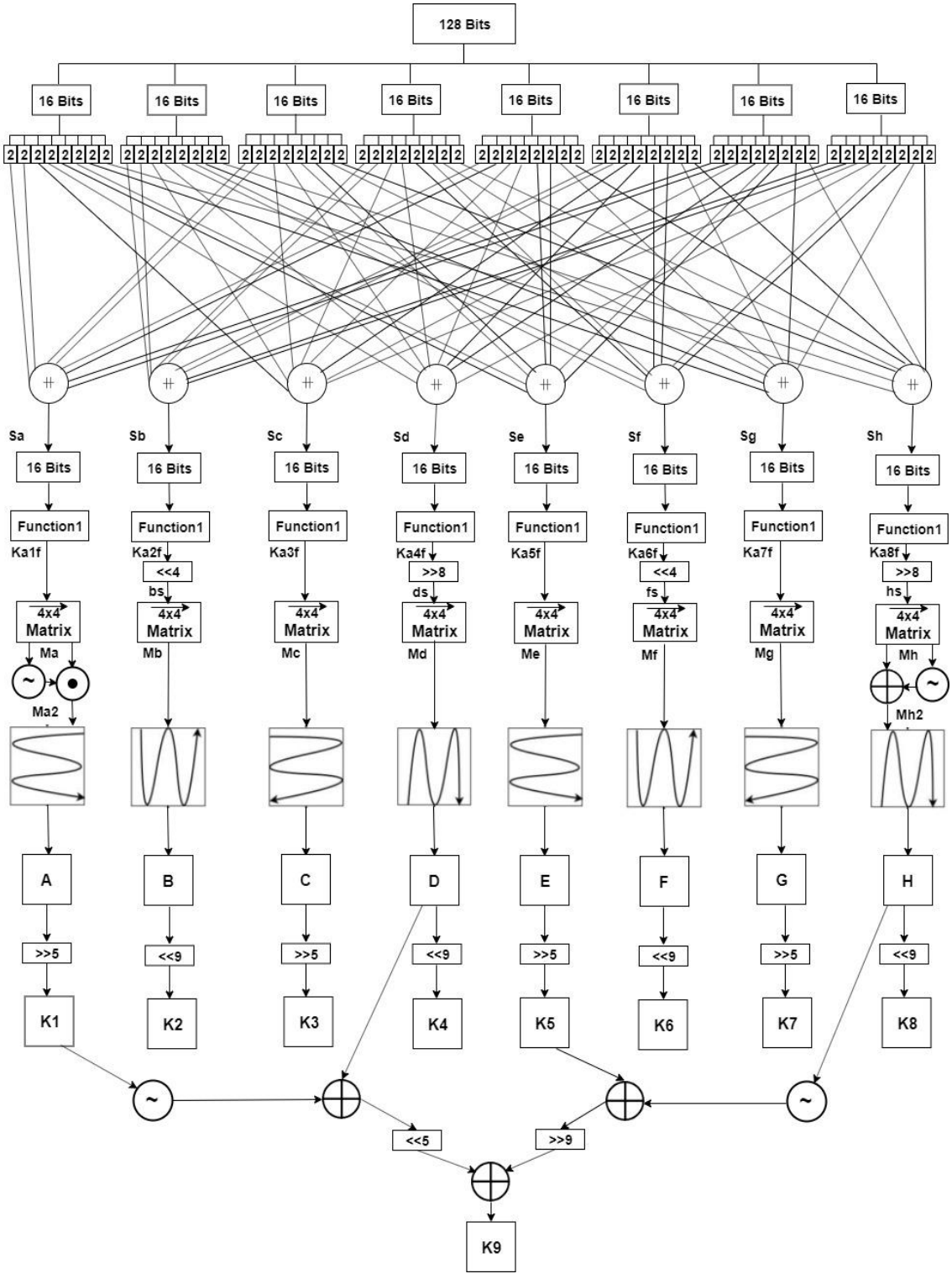


Figure 3.2. Sub-Keys generator

From the above, we saw how creating nine sub-keys from the main key. In the next equations, we will explain the algorithm structure and encryption.

In the begin, the main block divided to four blocks (each block equal 16 bits) and each block divided to eight sub-blocks (each block equal two bits), this means we have 32 sub-blocks of two bits, then produce six blocks through next equations.

Where B means Block, b means sub-block.

$$Ba_1 = b_{26} \# b_{18} \# b_{10} \# b_2 \# b_{32} \# b_{24} \# b_{16} \# b_8 \quad (3.43)$$

$$Bb_1 = b_{30} \# b_{22} \# b_{14} \# b_6 \# \overline{b_{30}} \# \overline{b_{22}} \# \overline{b_{14}} \# \overline{b_6} \quad (3.44)$$

$$Bc_1 = b_{29} \# b_{21} \# b_{13} \# b_5 \# \overline{b_{29}} \# \overline{b_{21}} \# \overline{b_{13}} \# \overline{b_5} \quad (3.45)$$

$$Bd_1 = b_{28} \# b_{20} \# b_{12} \# b_4 \# \overline{b_{28}} \# \overline{b_{20}} \# \overline{b_{12}} \# \overline{b_4} \quad (3.46)$$

$$Be_1 = b_{27} \# b_{19} \# b_{11} \# b_3 \# \overline{b_{27}} \# \overline{b_{19}} \# \overline{b_{11}} \# \overline{b_3} \quad (3.47)$$

$$Bf_1 = b_{25} \# b_{17} \# b_9 \# b_1 \# b_{31} \# b_{23} \# b_{15} \# b_7 \quad (3.48)$$

Then in every round, will be obtained new blocks through the following equations.

Where F_1= Function1.

$$Ba_2 = (Ba_1 \gg 5) \oplus K1 \quad (3.49)$$

$$Bb_2 = \left(\overline{Bb_1 \oplus K1}\right) \oplus Delta \quad (3.50)$$

$$Bc_2 = F_1\left(\overline{Bb_1 \oplus K1}\right) \oplus Bd_1 \quad (3.51)$$

$$Bd_2 = \left(F_1\left(\overline{Be_1 \oplus K1}\right) \oplus Bc_2\right) \oplus Delta \quad (3.52)$$

$$Be_2 = \left(\overline{Be_1 \oplus K1}\right) \quad (3.53)$$

$$Bf_2 = (Bf_1 \gg 5) \oplus (\mathrm{K1} \oplus Delta) \quad (3.54)$$

After nine rounds, there are some operations to get on the block cipher from this operations, three functions (Function 1, Function 2, Function 3, Function 4) , show the next equations and figure 3.6.

Where BC = block cipher, F_1= Function1, F_2 = Function2, F_3= Function3 and F_4= Function4.

$$B_1 = F_1(Bf_{10}) \quad (3.55)$$

$$B_2 = F_4(Bb_{10}) \oplus F_2(Bc_{10}) \quad (3.56)$$

$$B_3 = F_1(Bd_{10}) \oplus F_3(Be_{10}) \quad (3.57)$$

$$B_4 = F_1(Ba_{10}) \quad (3.58)$$

$$BC = B_2 \shuffle B_3 \shuffle B_1 \shuffle B_4 \quad (3.59)$$

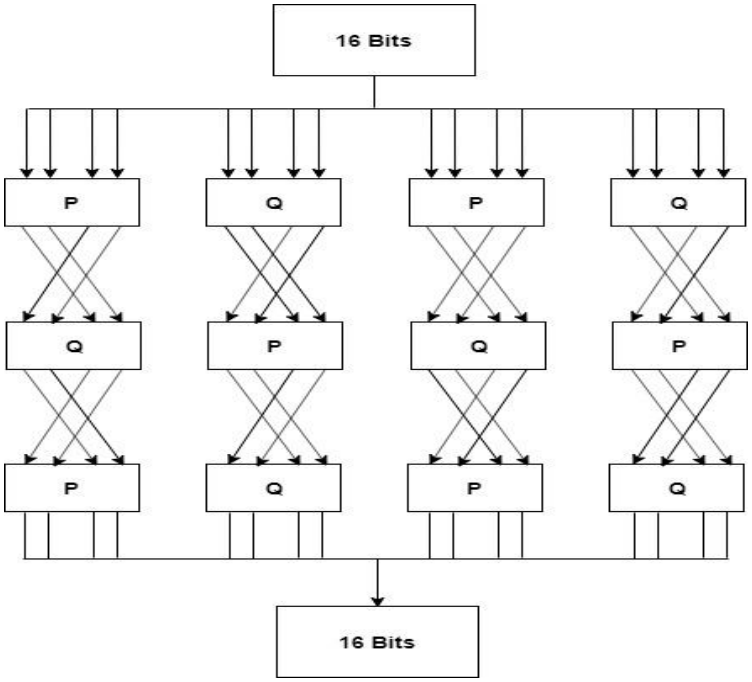


Figure 3.3. Function 2

Function 2 (Figure 3.3), applied in the last steps in encryption structure for proposed algorithm, where it is applied on the third block in the result of the ninth round, and function 1 (Figure 3.1) applied on the fourth block and on the sixth block in the result of the ninth round, a see figure 3.6.

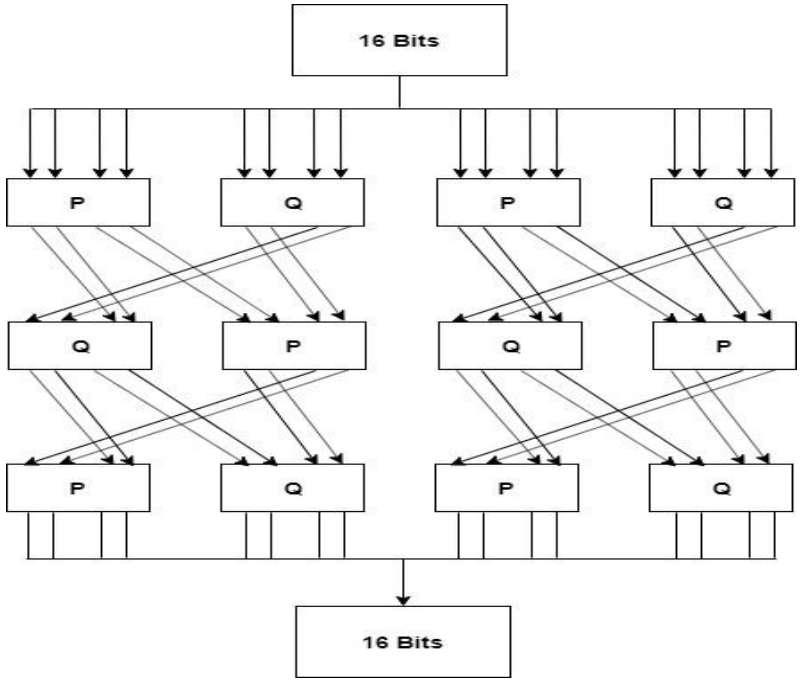


Figure 3.4. Function 3

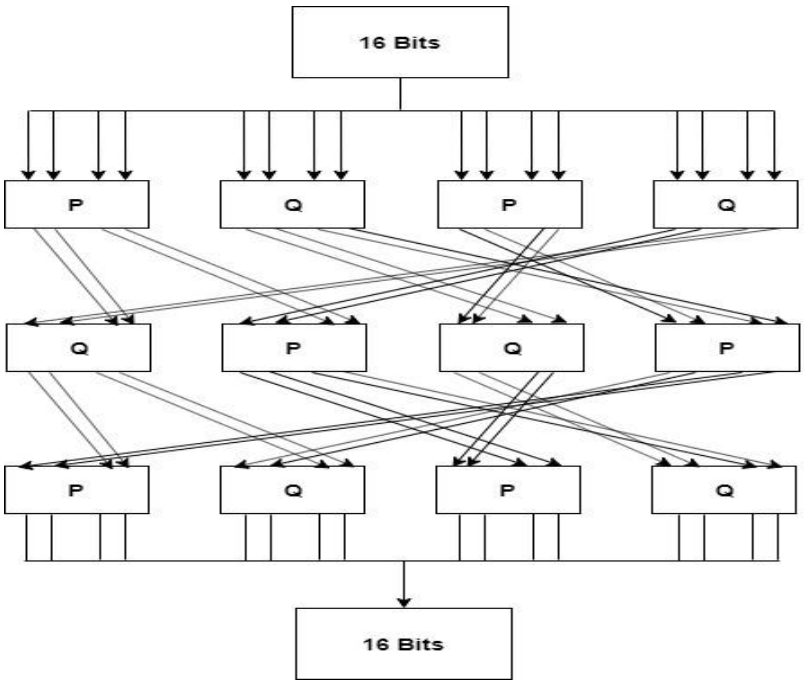


Figure 3.5. Function 4

Function 3 (Figure 3.4) applied on the fifth block in result of the ninth round, and Function 4 (Figure 3.5) applied on the first block and on the second block in result of the ninth round.

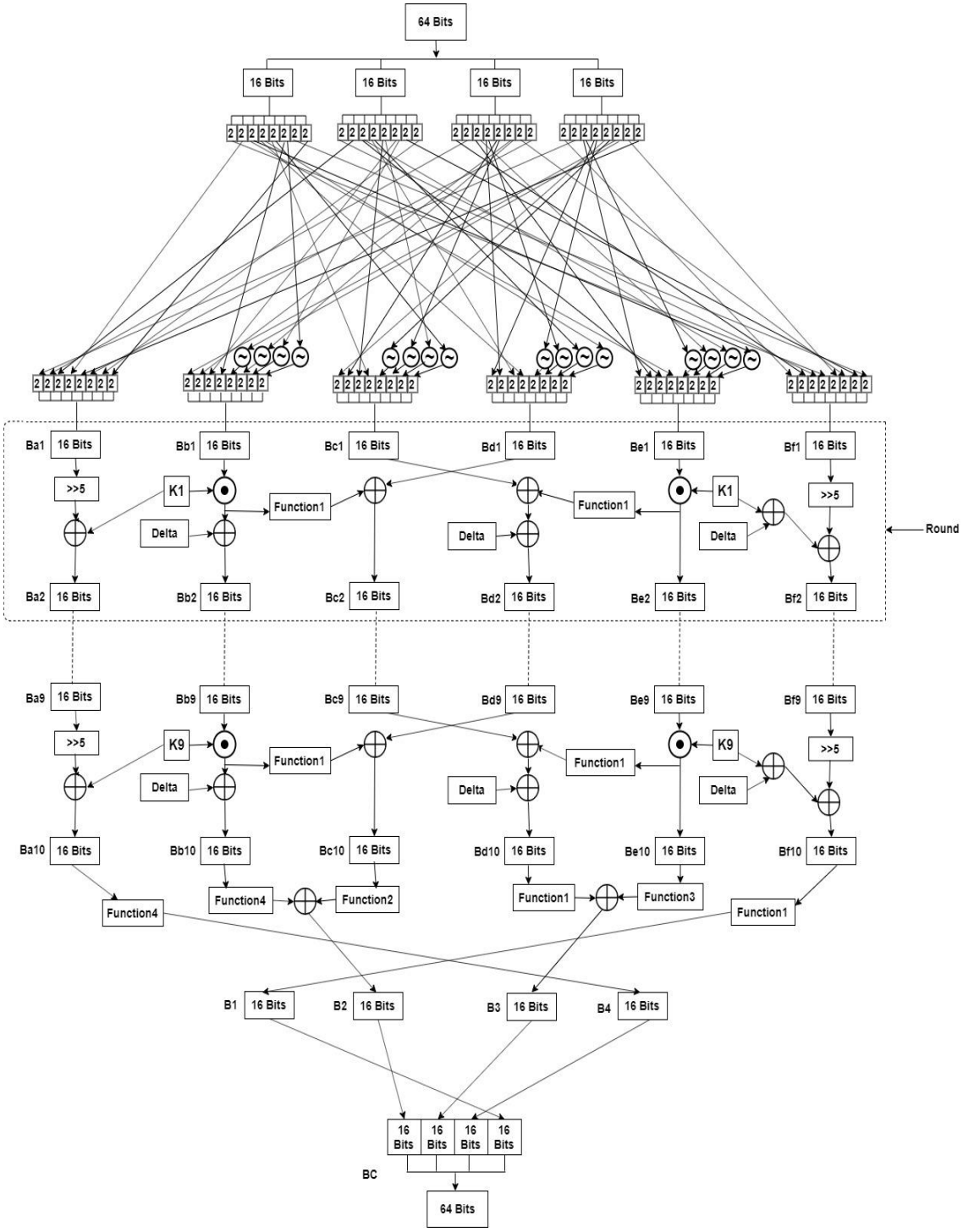


Figure 3.6. Structure of the proposed algorithm (Encryption)

In the encryption structure for the proposed algorithm in above figure, show the rounds and the steps before the rounds and after the rounds.

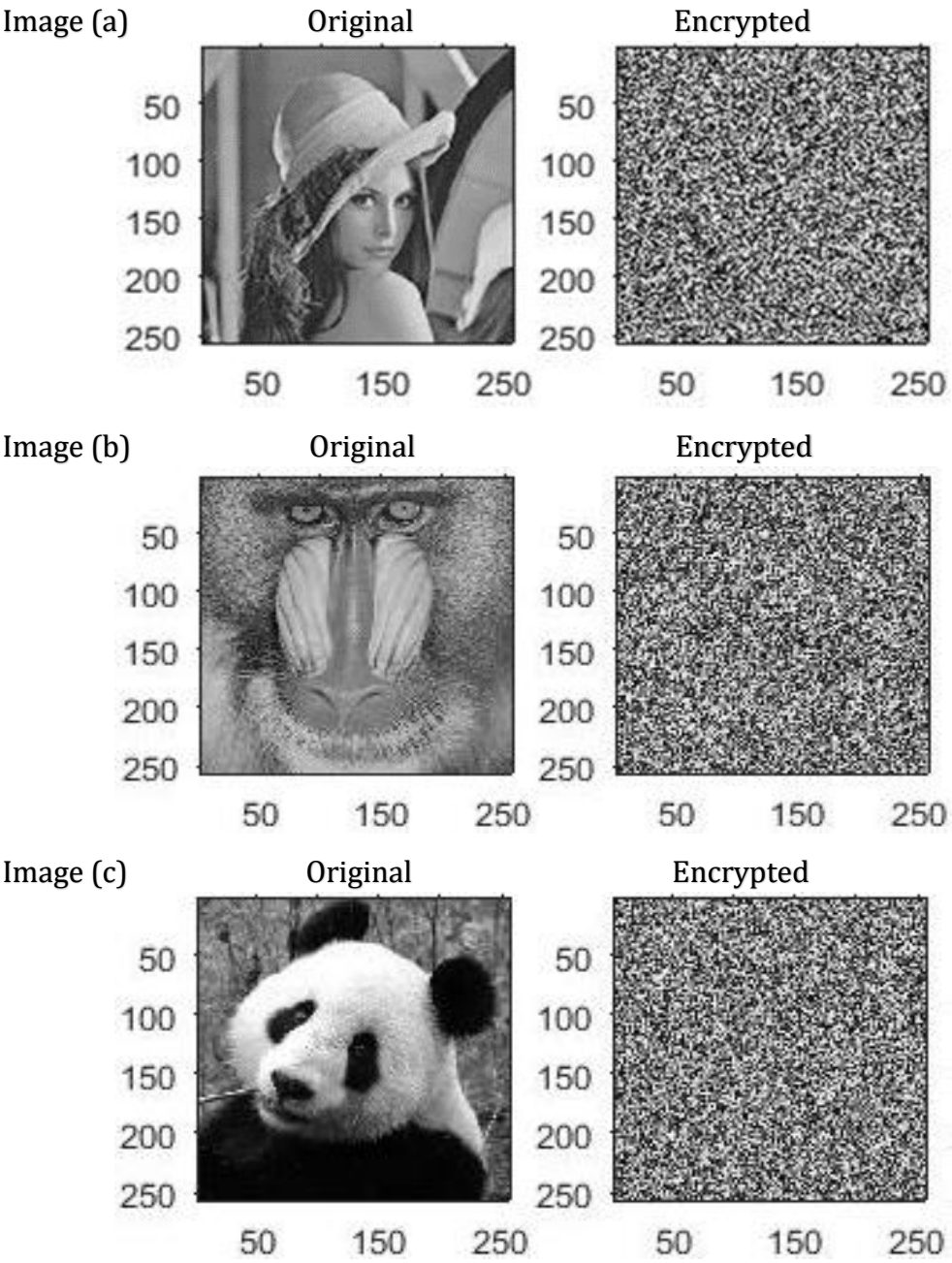


Figure 3.7. Encryption images

In figure 3.7 show three images (a, b and c) before encryption (The original images) and after encryption (The encrypted images) by use the proposed algorithm.

3.2. Analysis of Proposed Algorithm and Encryption

3.2.1 Linear and differential cryptanalysis

In the proposed algorithm, the tables 1 and 2 (P and Q), and the functions 1, 2, 3 and 4 are inspired by (Barreto P. S. L. M., Rijmen V., 2000) whose cryptanalysis shows in the complete cipher that differential and linear attacks do not have the success. where the input and output correlation is so large if the linear approximation is done for two rounds, Also, the round transformation is preserved uniform, that treats every bit in a similar way and provides resistance against differential attacks.

3.2.2. Correlation coefficient analysis

A correlation is a statistical measure of the relationship between two variables, and it ranges between +1 and −1. positive one which that means that variables move in the one direction along which means an ideal correlation, whereas zero implies that there's no relationship between the variables and if negative one refers to an ideal indirect correlation, which means that one variable goes up, the other goes down.

In the encryption of images means the result of the correlation that the closer to zero (if positive or Negative), that relationship between the encrypted image and the original image is weaker and this means that the level of encryption stronger. If that the result of the correlation farther away from zero, whether negative or positive means that the relationship between the original image and encrypted image stronger this means that the encryption level is weaker. If that result of correlation equal zero this means there is no relationship between the original and encrypted image and refers to the best level of encryption (Ramasamy, 2019).

$$\mathrm{CR} = \frac{\mathrm{cov(X,Y)}}{\sqrt{D(X)}\sqrt{D(Y)}} \tag{3.60}$$

$$\mathrm{cov(X,Y)} = \frac{1}{256}\sum_{1}^{256} (X - E(X))(Yi - E(Y)) \tag{3.61}$$

Where X and Y are the pixels and neighboring pixels of the original and encrypted image, cov(X,Y) is the covariance between X and Y, D(X) is the variance of X, and E(X) is the expected value of X.

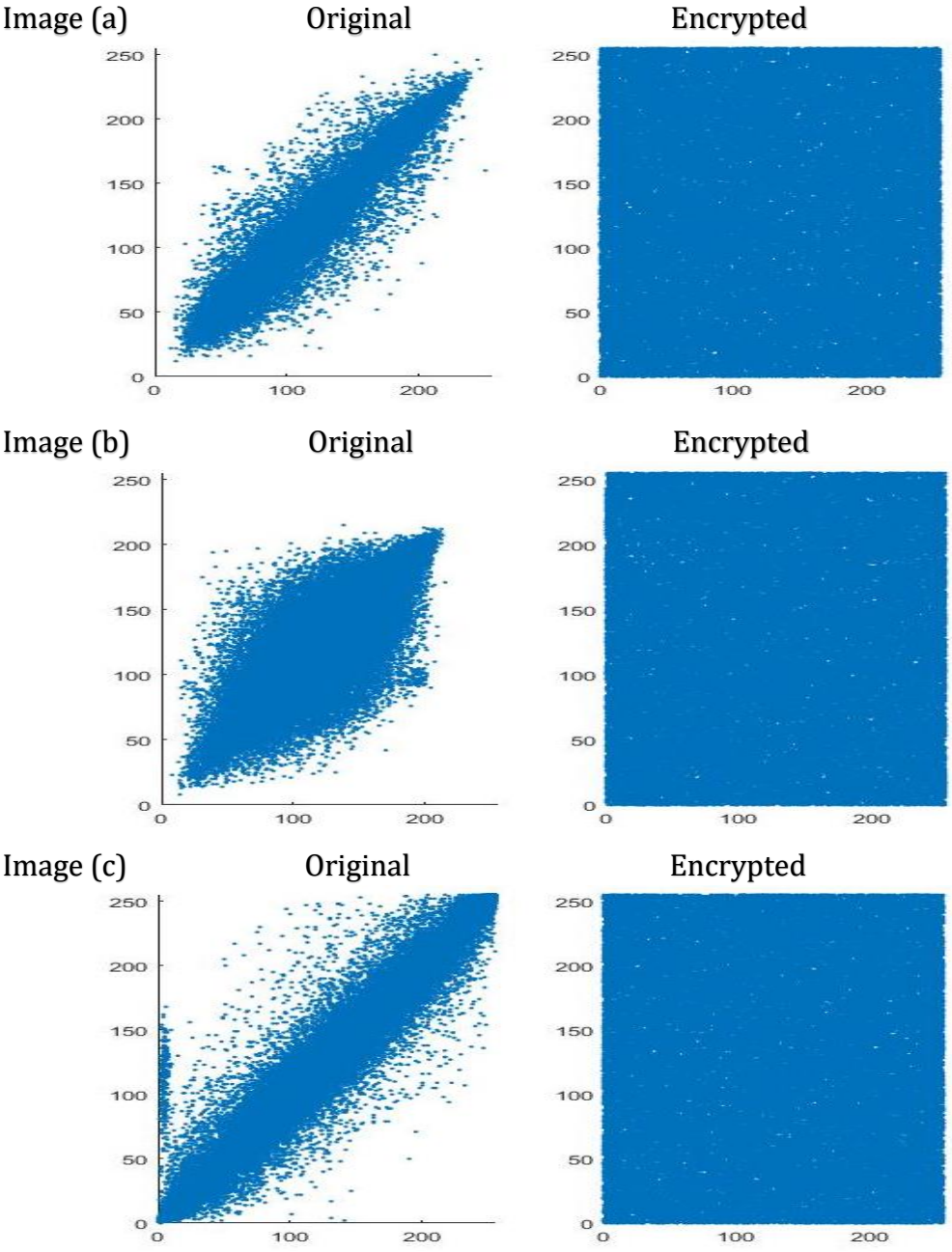


Figure 3.8. Correlation of images

The relationship between the encoded and original images in the figure above is very weak, this is inversely proportional to the strength of the encryption.

3.2.3. Information entropy analysis

The entropy of information estimates the uncertainty of a random variable. When the entropy applied to evaluate image encryption, the larger value of entropy refers to a greater level of security, and it is secure from a brute force attack when an entropy value that is very close to a perfect value of 8 (Ramasamy, 2019).

$$E = \sum_{i=0}^{255} P(i)\log\left(\frac{1}{P(i)}\right) \tag{3.62}$$

Where E is Entropy, P(i) is the probability of the presence of pixel i.

3.2.4. Histogram analysis

The histogram of the image is a graphical and statistical representation for the distribution of pixel values information. The histogram of the perfect encrypted image must be a uniformly distributed and fully different in comparison with the original image, to prevent extract any information from the histogram for the encrypted image (Abdullah and Abdullah, 2017).

In the histogram, the image is highly random and highly resistant against the statistical attacks if the intensity of the pixels is uniformly distributed. (Maddodi, et al., 2018).

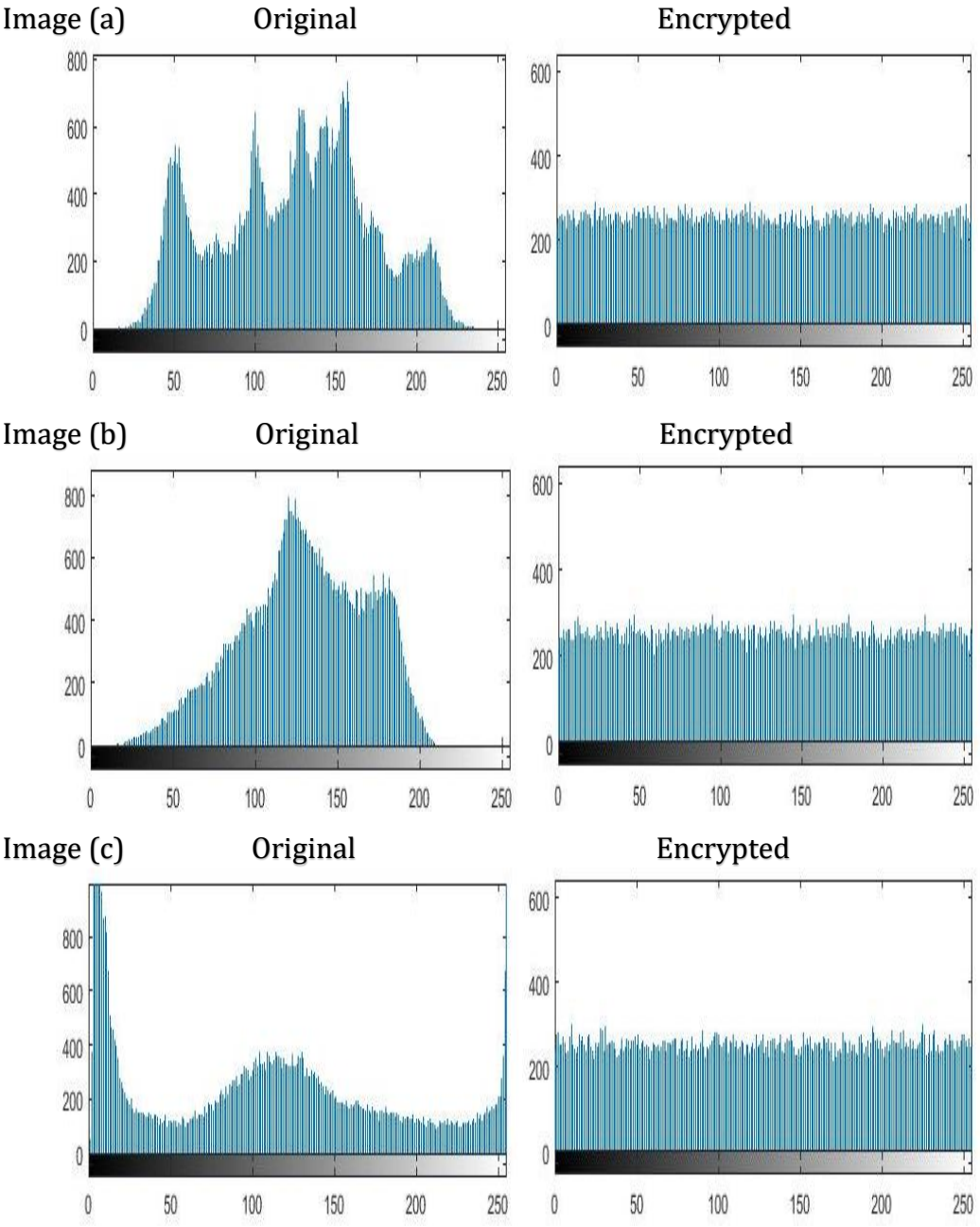


Figure 3.9. Histogram comparison

In the histogram comparison, we notice a large difference in the distribution of data between the original and encrypted images, and this indicates the encryption strength.

3.2.5. Key space analysis

Key space means number of bits that used to encrypt image, and for good encryption, that key space should be as large as possible to repulse brute force attack. Key space size means the total number of different keys of the same number of bits used for encryption (Kaur and Sharma, 2013). For high security, the key space should be greater than 2^{100} (Askar, et al., 2018). In proposed algorithm key size = 128-bit and according to equation (3.63) the key space is = 2^{128} this mean key space size is good to repulse brute force attack. Where Ks= Key space, Kz= Key size.

$$\text{Ks} = 2^{Kz} \tag{3.63}$$

3.2.6. Related keys

Through performing cipher operations and by using unknown or partially known keys can be made an attack. The related key attack predominately depends on having symmetry in key expansion block or upon either slow diffusion in the proposed algorithm, the sub-keys process is designed for fast and non-linear diffusion to the difference of the main key and sub-keys.

3.2.7. Interpolation attacks

Dependent these attacks on the simple structures for the cipher components that may yield a rationalistic expression with a handy intricacy. In the proposed algorithm, the S-box expression for along with the diffusion layer makes this type of attack impractical.

3.2.8. Other analyzes

There are important things that must be taken into account in the lightweight cryptography algorithms (the speed or processing time, the amount of memory and energy consumption of the algorithm during data encryption operations) as

the proposed algorithm was designed to suit Internet of things environments. The memory and energy consumption are directly proportional as well as with the time and speed.

4. RESEARCH FINDINGS AND DISCUSSION

Through some analysis for the proposed algorithm, where the simulations were performed on a desktop computer with Intel(R) Core(TM) i7 CPU L 620 @2.00 GHz., 4GB RAM., and Windows 10 Professional operating system, and by using MATLAB R2015a. In addition to some other algorithms on FELICS (Rana et al., 2018). We get the results in next tables (4.1, 4.2 and 4.3) and figures (4.1 and 4.2).

Table 4.1. Correlation and entropy analysis

Image	Size	Correlation		Entropy	
		Original image	Encrypted image	Original image	Encrypted image
a	256 x 256	0.9744	0.0001	7.4509	7.9976
b	256 x 256	0.8198	0.0013	7.2316	7.9972
c	256 x 256	0.9811	-0.0005	7.4938	7.9971

In table 4.1 correlation of encrypted image for image (a) is it reaches the closest level to zero, as well as the correlation of other encrypted images. It is very close to zero, meaning that the relationship between the original and encrypted images is almost non-existent. This indicates that the level of encryption and security is very high. The results of entropy analysis for the encrypted images for (a, b and c) in the above table, also refer to the highest level for encryption and security, where the values of entropy closest to eight.

In the table 4.2, the Block size and key size are in bits, while the code size and RAM are in bytes. We note that the level of memory consumption of the proposed algorithm is less than other algorithms that has a key size of 128 and less than some other algorithms that have a key size less than 128.

Table 4.2. Comparison between the proposed algorithm and other algorithms

The Algorithm	Block Size	Key Size	Rounds	Code Size	RAM
PRESENT	64	80	32	1738	274
Simon	64	96	42	1370	188
Speck	64	96	26	2552	124
SIT	64	64	5	826	22
AES	128	128	10	23090	720
LEA	128	128	24	3700	432
RC5	64	128	20	20044	360
HIGHT	64	128	32	13476	288
Proposed	64	128	9	823	144

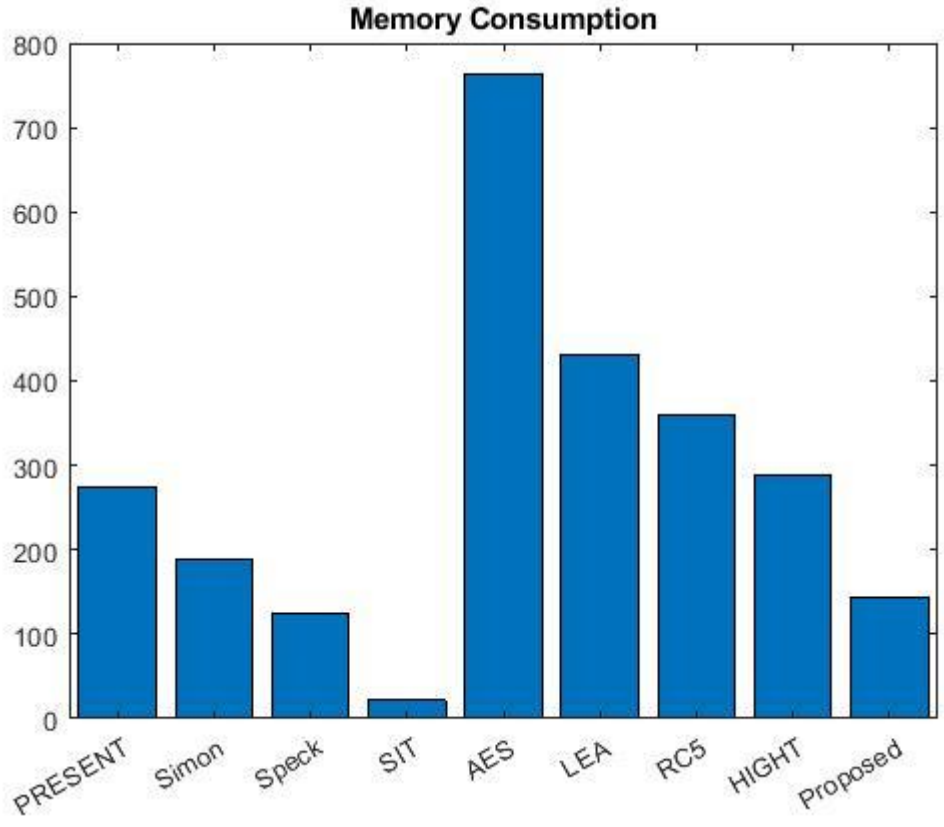


Figure 4.1. Bar diagram for algorithms memory consumption

From the bar diagram for algorithms memory consumption, we see that the proposed algorithm come in the third level of memory consumption after the SIT and Speck algorithms, and the algorithms that have a 128-bit key size in addition to the PRESENT algorithm.

The time required for the encryption process for the proposed algorithm is 0.063691437 seconds. This refers that the proposed algorithm is fast in the encryption process, and the execution time it is good for a lightweight cryptography algorithm.

Table 4.3. Average time required for exhaustive key search

The Algorithm	Key Size	Number of Alternative Keys (Key Space)	Time Required at 1 Decryption/μs	Time Required at 10^6 Decryptions/μs
PRESENT	80	$2^{80}=1.2*10^{24}$	2^{79}μs = $1.91542738*2^{13}$ years	19154273.8 years
Simon	96	$2^{96}=7.92281625*10^{28}$	2^{95}μs = $1.25529448*10^{18}$ years	$1.25529448*10^{12}$ years
Speck	96	$2^{96}=7.92281625*10^{28}$	2^{95}μs = $1.25529448*10^{18}$ years	$1.25529448*10^{12}$ years
SIT	64	$2^{64}=1.84467441*10^{19}$	2^{63}μs = 292271021 years	292.271021 years
AES	128	$2^{128}=3.4*10^{38}$	2^{127}μs = $5.4*10^{24}$ years	$5.4*10^{18}$ years
LEA	128	$2^{128}=3.4*10^{38}$	2^{127}μs = $5.4*10^{24}$ years	$5.4*10^{18}$ years
RC5	128	$2^{128}=3.4*10^{38}$	2^{127}μs = $5.4*10^{24}$ years	$5.4*10^{18}$ years
HIGHT	128	$2^{128}=3.4*10^{38}$	2^{127}μs = $5.4*10^{24}$ years	$5.4*10^{18}$ years
Proposed	128	$2^{128}=3.4*10^{38}$	2^{127}μs = $5.4*10^{24}$ years	$5.4*10^{18}$ years

We see in the above those algorithms (AES, LEA, RC5, HIGHT and Proposed) Average Time Required for Exhaustive Key Search is so long time, because the key size of these algorithms is large and contributes to increasing the level of security of the algorithms and is sufficient to repel the brute force attack.

In the following figure, the key space for those algorithms that have a 128-bit key size (AES, LEA, RC5, HIGHT, and Proposed) are highly valued. Whereas, algorithms that have a key size less than 128 bits appear at a low value for the key space.

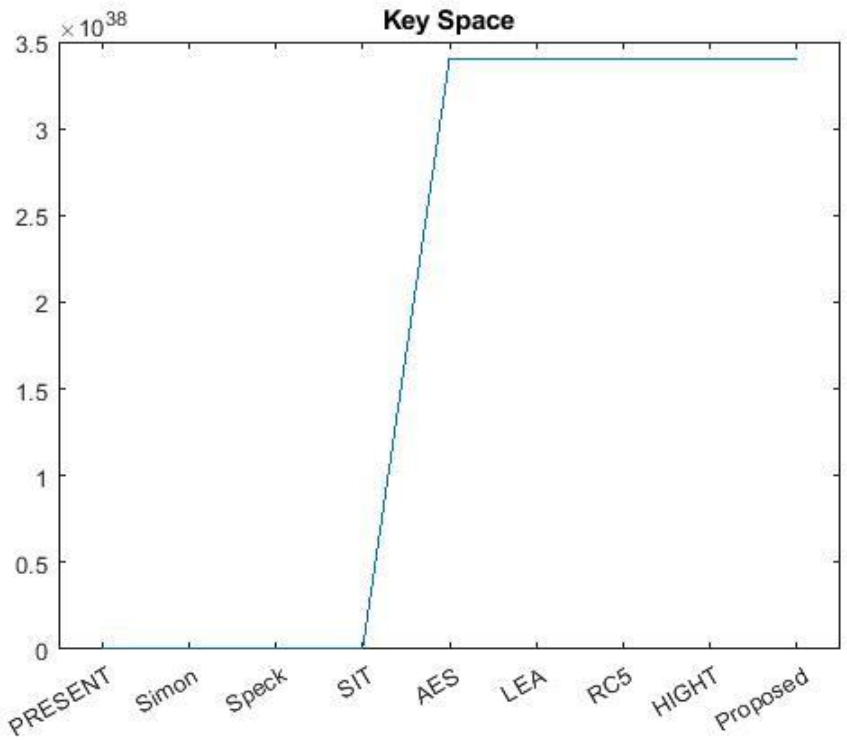


Figure 4.2 Plot diagram for algorthms key space

From the results show the proposed algorithm high efficient in comparison with another algorithm especially with those algorithms of 128 bit.

Can testing the work and performance of the algorithm in different environments, for example in FPGA and other devices with limited specifications, and further improvements can be made to the security level and performance of the algorithm through changes and improvements to the function.

5. CONCLUSION AND IMPLICATIONS

Due to the need for smart cities for IoT applications, as well as the spread of Internet applications things in several areas, and with the increase of applications increases the cyber security threats, which is more dangerous in the Internet of things applications at level of smart cities because it is related to many fields and the management of many services and control and impact on Security, economic, environmental and other aspects. Here, improving cyber security is very important, including securing Internet of things sensors by developing and improving the lightweight cryptographic algorithms (improving security, taking into account other features that improve the performance Such as speed, power consumption and memory). In addition to the security is the most important factor in encryption algorithms, there are other required and important factors in the lightweight encryption such as the size of memory consumption, energy consumption and processing speed. One of the main benefits of the Internet of things is the ability to deploy sensors in traditionally inaccessible places, and here the extreme necessity of using batteries is required, but their energy consumption must be taken into account. where in the Internet of things environments, like sensors, taking into account energy consumption is very important, because these devices operate with limited energy, as they operate on the batteries, and the power consumption is dependent on the processing speed. As well as limited memory, where some of the small IoT resource constraint devices contain 8 KB of Random Access Memory (RAM) and the size of RAM is the factor that determines the possibility of implementation in the devices. Because the power is greatly dependent on the hardware and processing, the memory size becomes important for the lightness of the encryption method and for the power. In this thesis, we proposed a lightweight cryptography algorithm, where the results of the analysis show the proposed algorithm It has a high level of security, high processing speed, a low level of memory consumption and that is directly proportional to the energy consumption, that means the power consumption for the proposed algorithm is low. This is done by designing the structure of the algorithm and designing the sub-key generator in addition to the functions and

processes that in turn lead to increase the level of security by increasing the complexity of the encryption, and the same time consuming a small amount of memory where the best processes used in the other algorithms were used, and some other operations were added to the proposed algorithm. And that is through a lot of analyzes such as analyzing the relationship between encrypted images and original images, entropy analysis for encrypted image and testing the level of memory consumption in addition to measuring the key area of the algorithm.

It also made the algorithm more sophisticated and more resistant to attack by using the substitution box, the functions used in the key generator, and the structure of the algorithm.

The time required to implement the encryption process in the proposed algorithm indicates that the algorithm is of good speed and the low level of memory consumption with the use of a 128-bit key, its suitable for Internet of Things environments.

Although difficult of getting on high security and low level of memory and energy consumption at the same time in the lightweight encryption algorithms because of the limited specification in the hardware and environments that need the lightweight encryption algorithms, the proposed algorithm and through studies, analyzes and results, appear that it highly efficient, and good to protect the Internet of Things applications and smart city services from those cyber attacks, manipulation, and tampering with services, information and data through sensors and overcoming the weakness due to its limited specifications.

In the future, we are interested in the analysis and testing of security and performance of this algorithm in various environments related to smart cities and Internet of things.

REFERENCES

Abdullah, H. N., Abdullah, H. A., 2017. Image Encryption Using Hybrid Chaotic Map. International Conference on Current Research in Computer Science and Information Technology (ICCIT), IEEE, Slemani – Iraq, 26-27 April 2017, 121-125.

Aksu D., Aydin M. A., 2019. A Survey of IoT Architectural Reference Models. International Multi-Conference on Systems, Signals & Devices (SSD'19), IEEE, Istanbul, Turkey, 21-24 March 2019, 413-417.

Andem, V. R., 2003. A Cryptanalysis of the Tiny Encryption Algorithm, University of Alabama, The Graduate School, M.Sc. Thesis, 68p, Alabama.

Askar, S. S., Karawia, A. A., Al-Khedhairi, A., Al-Ammar, F. S., 1, 2018. An Algorithm of Image Encryption Using Logistic and Two-Dimensional Chaotic Economic Maps. Entropy, 21 (1), 1-17.

Barker, E., 2016. National Institute of Standards and Technology. Special Publication, 800-57, Part 1, Revision 4.

Barreto, P. S. L. M., Rijmen, V., 2000. The KHAZAD Legacy-Level Block Cipher. New European Schemes for Signature, Integrity and Encryption (NESSIE), 97, 1-20.

Broemeling, L. D., 2012. An Account of Early Statistical Inference in Arab Cryptology. The American Statistician, 65 (4), 255-257.

Cirani, S., Ferrari, G., Picone, M., Veltri, L., 2018. Internet of Things, Architectures, Protocols and Standards. Wiley, 408p, Hoboken, New Jersey, USA.

CryptoLUX Wiki, 2017. Lightweight Block Ciphers, Accessed: 11.04.2017. https://www.cryptolux.org/index.php/Lightweight_Block_Ciphers

Dustdar, S., Nastic, S., Scekic, O., 2017. Smart Cities, The Internet of Things, People and Systems. Springer, 268p, Cham, Switzerland.

Gurunath, R., Agarwal, M., Nandi, A., Samanta, D., 2019. An Overview: Security Issue in IoT Network. Proceedings of the Second International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud, IEEE, 30-31 Aug. 2018, Palladam, India, 104-107.

Kahn d., 1996. The Codebreakers. Scribner, 1200p, USA.

Kaur T., Sharma R., 2013. Security Definitive Parameters for Image Encryption Techniques. International Journal of Emerging Technology and Advanced Engineering, 3, 109-112.

Kim, T., Ramos, C., Mohammed, S., 2017. Smart City and IoT. Elsevier, 76, 159–162.

Kostopoulos, G., 2017. Cyberspace and Cybersecurity. Auerbach, CRC Press, Taylor & Francis Group, 316p, New York, USA.

Maddodi, G., Awad, A., Awad, D., Awad, M., Lee, B., 2018. A new image encryption algorithm based on heterogeneous chaotic neural network generator and dna encoding. Springer, 77 (19), 24701–24725.

Maitra, S., Yelamarthi, K., 2019. Rapidly Deployable IoT Architecture with Data Security: Implementation and Experimental Evaluation. Sensors, 19 (2484), 1-22.

Mowbray T.J., 2014. Cybersecurity: Managing Systems, Conducting Testing, and Investigating Intrusions. Wiley, 360p, Indiana, USA.

Muggah, R., Goodman, M., 2019. The World Economic Forum. Cities are easy prey for cybercriminals, Here's how they can fight back. Date Accessed: 30.09.2019.
https://www.weforum.org/agenda/2019/09/our-cities-are-increasingly-vulnerable-to-cyberattacks-heres-how-they-can-fight-back/

Oracevic, A., Dilek, S., Ozdemir, S., 2017. Security in Internet of Things: A Survey. International Symposium on Networks, Computers and Communications (ISNCC), IEEE Marrakech, Morocco, 16-18 May 2017, 1-6.

Paar C., Pelzl J., 2010. Understanding Cryptography. Springer, 392p, Berlin, Germany.

Ramasamy, P., Ranganathan, V., Kadry, S., Damasevicius, R., Blazauskas, T., 2019. An Image Encryption Scheme Based on Block Scrambling, Modified Zigzag Transformation and Key Generation Using Enhanced Logistic—Tent Map. Entropy, 21 (7), 1-17.

Rassia, S. TH., Pardalos, P. M. (ED), 2017. Smart City Networks: Through the Internet of Things. Springer, 227p, Cham, Switzerland.

Rana, S., Hossain, S., Imam, H., Abul Kashem, M., 2018. An Effective Lightweight Cryptographic Algorithm to Secure Resource-Constrained Devices. (IJACSA) International Journal of Advanced Computer Science and Applications, 9 (11), 267-275.

Serpanos, D., Wolf, M., 2018. Internet-of-Things (IoT) Systems: Architectures, Algorithms, Methodologies. Springer, 95p, Cham, Switzerland.

Singer and Friedman, 2014. Cybersecurity and Cyberwar. Oxford University Press, 320p, New York.

Stallings M., 2011. Cryptography and Network Security: Principles and Practice. Pearson, 900p, New York.

Symantec, 2018. 10 cyber security facts and statistics for 2018. January 2018. https://us.norton.com/internetsecurity-emerging-threats-10-facts-about-todays-cybersecurity-landscape-that-you-should-know.html

The Guardian, 2017. False alarms: hackers take over Dallas's 156 sirens before system deactivated. Date Accessed: 09.04.2017. https://www.theguardian.com/us-news/2017/apr/09/dallas-hackers-sirens-alarms-emergency-system#top

The National Institute of Standards and Technology (NIST), 2017. Lightweight Cryptography. Date Accessed: 03.01.2017. https://csrc.nist.gov/projects/lightweight-cryptography

The Telegraph, 2015. Hacker remotely crashes Jeep from 10 miles away. Date Accessed:21.07.2015. https://www.telegraph.co.uk/news/worldnews/northamerica/usa/11754089/Hacker-remotely-crashes-Jeep-from-10-miles-away.html

United Nations, Department of Economic and Social Affairs, 2018. 2018 Revision of World Urbanization Prospects. Date Accessed: 16.05.2018. https://www.un.org/development/desa/publications/2018-revision-of-world urbanization-prospects.html

University of Michigan, College of Engineering, 2014. Researchers Demo Hack to Seize Control of Municipal Traffic Signal Systems. Date Accessed: 21.08.2014. https://www.eecs.umich.edu/eecs/about/articles/2014/Green-Lights-Forever.html

Usman M., Ahmed, I., Aslam, M. I., Khan, S., Shah U. A., 2017. SIT: A Lightweight Encryption Algorithm for Secure Internet of Things, (IJACSA) International Journal of Advanced Computer Science and Applications, 8 (1), 402-411.

Yeo, K. S., Chian, M. C., Ng, T. C W., and Tuan, D. A., 2014. Internet of Things: Trends, Challenges and Applications. International Symposium on Integrated Circuits (ISIC), IEEE, Singapore, 10-12 December 2014, 568-571.

APPENDICES

Appendix A. Matlab Code for Main Function for Proposed Algorithm

```
clc
clear all
close all

%% Initialization
Test=0; % for right key encryption
%Test=1; % for key sensitivity test
addpath functions
Images_Path='Images\Original\';
 fname={'Lena','panda','baboon'}; % filename
 ext='.jpg';
 fid=3; % file ID 1 for lena
 IS =256; % Image size
 Data=imread(strcat(Images_Path,fname{fid},ext));
 if (size(Data,3)==3)
     Data=rgb2gray(Data);
 end
 Data=imresize(Data,[IS IS]); % Image Size

 [row,col]=size(Data);
 [Data,padding]=Scalling(Data,8);
 Data_binary=convert2bin(Data);

 hex_key = 'AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA';
 [bin_key] = Hex2Bin( hex_key );
 [K1,K2,K3,K4,K5,K6,K7,K8,K9]=SF_Key_Gen(bin_key);
 Original_msg=[];
 encrypt_msg=[];
 decrypt_msg=[];

 %% Encryption Process
 for kk=1:2
   for i=1:size(Data_binary,1)
     Original=Data_binary(i,:);
     tic
      [cipher]=SF_Encrypt
 (Original,K1,K2,K3,K4,K5,K6,K7,K8,K9);
     encryption_time(i)=toc;
     tK1=[K1(1:8),Original(1:8)];tK2=
 [Original(9:16),K2(9:16)];
 tK3=[Original(17:24),K3(9:16)];
 tK4=[Original(25:32),K4(9:16)];
 tK5=[Original(33:40),K5(9:16)];
 tK6=[Original(41:48),K6(9:16)];
 tK7=[Original(49:56),K7(9:16)];
 tK8=[Original(57:60),Original(57:60),K8(9:16)];
 tK9=[Original(61:64),Original(61:64),K9(9:16)];
     K1=tK1;K2=tK2;K3=tK3;K4=tK4;K5=tK5
 ;K6=tK6;K7=tK7;K8=tK8;K9=tK9;
      encrypt_msg(:,i)=binary2dec(cipher);
     cipher_data(i,:)=double(cipher);
     if(kk<2)
     Data_binary(i,:)=cipher_data(i,:);
     end
   end
 if (kk==1)
```

```
D=reshape(encrypt_msg,[row,col]);
D=D';
[row,col]=size(D);
[D,padding]=Scalling(D,8);
% Data=[Data Data];
Data_binary=double(convert2bin(D));
    TT=[K1,K2,K3,K4,K5,K6,K7,K8,K9];
encrypt_msg=[];
end
end

TTT=TT;
%% Decryption
if (Test==1)
 K1(end)=~K1(end);
 TT=[K1,K2,K3,K4,K5,K6,K7,K8,K9];
end

for kk=kk:-1:1
if(kk==2)
    K11=TT(1:16);K12=TT(17:32);K13=TT(33:48)
;K14=TT(49:64);K15=TT(65:80);K16=TT(81:96)
;K17=TT(97:112);K18=TT(113:128);K19=TT(129:144);
else
    [K11,K12,K13,K14,K15,K16,K17,K18,K19]=
SF_Key_Gen(bin_key);
   D=reshape(decrypt_msg,[row,col]);
D=D';
[row,col]=size(D);
[D,padding]=Scalling(D,8);
cipher_data=double(convert2bin(D));

decrypt_msg=[];
end
for i=1:size(Data_binary,1)
    cipher=cipher_data(i,:);
    [plaintext]=SF_Decryption(cipher,K11,K12,K13,K14,
K15,K16,K17,K18,K19);
    K11=[K11(1:8),plaintext(1:8)];K12=
[plaintext(9:16),K12(9:16)];K13=
[plaintext(17:24),K13(9:16)];K14=
[plaintext(25:32),K14(9:16)];K15=
[plaintext(33:40),K15(9:16)];K16=
 [plaintext(41:48),K16(9:16)];K17=
 [plaintext(49:56),K17(9:16)];K18=
 [plaintext(57:60),plaintext(57:60),K18(9:16)]
 ;K19=[plaintext(61:64),plaintext(61:64),K19(9:16)];
     decrypt_msg(:,i)=binary2dec(plaintext);
     cipher_data(i,:)=double(plaintext);
 end
 end

 % %% Results
 % 5 Original Image
 Original=uint8(reshape(Data,[row,col]));
 % 6 Encrypted Image
 Encrypted=uint8(reshape(encrypt_msg,[row,col]));
 % 7(Wrong Key Decyption (Key Sensitivity))
 % %% 5 6 7
 figure
 subplot(1,3,1)
```

```
imshow(Original)
title('Original')
subplot(1,3,2)
imshow(Encrypted)
title('Encrypted')
%
% 8 Histogram
figure
subplot(2,1,1)
imhist(Original);
subplot(2,1,2)
imhist(Encrypted);

% 14 Image Entropy
Y=(imhist(Encrypted)+0.00001)/(row*col)
;%(length(Data)-padding);
Y=-sum(Y.*log2(Y));
X=(imhist(Original)+0.00001)/(row*col)
;%(length(Data)-padding);
X=-sum(X.*log2(X));
Re=[X Y]
% 9 Correlation
figure
subplot(1,2,1)
scatter(Original(1:end-1),Original(2:end),'.')
axis([0 255 0 255])
subplot(1,2,2)
scatter(Encrypted(1:end-1),Encrypted(2:end),'.')
axis([0 255 0 255])
%
display(sprintf('Total encryption time:
%f',sum(encryption_time)))
%
display('correlation coefficient of Original image')
corrcoef(double(Original(1:end-1))
,double(Original(2:end)))
display('correlation coefficient of encrypted image')
corrcoef(double(Encrypted(1:end-1))
,double(Encrypted(2:end)))
%
save(strcat('Images\Results\',fname{fid},'.mat'))
```

Appendix B. Matlab Code for Subkey Generator Function

```
function [K1,K2,K3,K4,K5,K6,K7,K8,K9]=
SF_Key_Gen(bin_key)

%making a cell
key3= cell(9,4);
%First for matrix of 16bit each

key3_1{1,1}=(1:16);
key3_11{1,1}=(1:2);
 key3_12{1,1}=(3:4);
 key3_13{1,1}=(5:6);
 key3_14{1,1}=(7:8);
 key3_15{1,1}=(9:10);
 key3_16{1,1}=(11:12);
 key3_17{1,1}=(13:14);
 key3_18{1,1}=(15:16);

 key3_1{1,2}=(17:32);
 key3_11{1,2}=(17:18);
 key3_12{1,2}=(19:20);
 key3_13{1,2}=(21:22);
 key3_14{1,2}=(23:24);
 key3_15{1,2}=(25:26);
 key3_16{1,2}=(27:28);
 key3_17{1,2}=(29:30);
 key3_18{1,2}=(31:32);

 key3_1{1,3}=(33:48);
 key3_11{1,3}=(33:34);
 key3_12{1,3}=(35:36);
 key3_13{1,3}=(37:38);
 key3_14{1,3}=(39:40);
 key3_15{1,3}=(41:42);
 key3_16{1,3}=(43:44);
 key3_17{1,3}=(45:46);
 key3_18{1,3}=(47:48);

 key3_1{1,4}=(49:64);
 key3_11{1,4}=(49:50);
 key3_12{1,4}=(51:52);
 key3_13{1,4}=(53:54);
 key3_14{1,4}=(55:56);
 key3_15{1,4}=(57:58);
 key3_16{1,4}=(59:60);
 key3_17{1,4}=(61:62);
 key3_18{1,4}=(63:64);

 key3_1{1,5}=(65:80);
 key3_11{1,5}=(65:66);
 key3_12{1,5}=(67:68);
 key3_13{1,5}=(69:70);
 key3_14{1,5}=(71:72);
 key3_15{1,5}=(73:74);
 key3_16{1,5}=(75:76);
 key3_17{1,5}=(77:78);
 key3_18{1,5}=(79:80);

 key3_1{1,6}=(81:96);
```

```
key3_11{1,6}=(81:82);
key3_12{1,6}=(83:84);
key3_13{1,6}=(85:86);
key3_14{1,6}=(87:88);
key3_15{1,6}=(89:90);
key3_16{1,6}=(91:92);
key3_17{1,6}=(93:94);
key3_18{1,6}=(95:96);

key3_1{1,7}=(97:112);
key3_11{1,7}=(97:98);
key3_12{1,7}=(99:100);
key3_13{1,7}=(101:102);
key3_14{1,7}=(103:104);
key3_15{1,7}=(105:106);
key3_16{1,7}=(107:108);
key3_17{1,7}=(109:110);
key3_18{1,7}=(111:112);

key3_1{1,8}=(113:128);
key3_11{1,8}=(113:114);
key3_12{1,8}=(115:116);
key3_13{1,8}=(117:118);
key3_14{1,8}=(119:120);
key3_15{1,8}=(121:122);
key3_16{1,8}=(123:124);
key3_17{1,8}=(125:126);
key3_18{1,8}=(127:128);

key3_2{1,1}=['key3_11{1,1}' 'key3_12{1,1}'
'key3_11{1,3}' 'key3_12{1,3}' 'key3_11{1,5}'
 'key3_12{1,5}' 'key3_11{1,7}' 'key3_12{1,7}'];
key3{1,1}=f_fun(bin_key(key3_2{1,1}));

key3_2{1,2}=['key3_11{1,2}' 'key3_12{1,2}'
 'key3_11{1,4}' 'key3_12{1,4}' 'key3_11{1,6}'
'key3_12{1,6}' 'key3_11{1,8}' 'key3_12{1,8}'];
key31{1,2}=f_fun(bin_key(key3_2{1,2}));
key32{1,2}=uint16(key31{1,2});
key3{1,2}=bitshift(key32{1,2},-4);

 key3_2{1,3}=['key3_13{1,1}' 'key3_13{1,2}'
 'key3_13{1,3}' 'key3_13{1,4}' 'key3_13{1,5}'
 'key3_13{1,6}' 'key3_13{1,7}' 'key3_13{1,8}'];
 key3{1,3}=f_fun(bin_key(key3_2{1,3}));

 key3_2{1,4}=['key3_14{1,1}' 'key3_14{1,2}'
 'key3_14{1,3}' 'key3_14{1,4}' 'key3_14{1,5}'
  'key3_14{1,6}' 'key3_14{1,7}' 'key3_14{1,8}'];
 key31{1,4}=f_fun(bin_key(key3_2{1,4}));
 key32{1,4}=uint16(key31{1,4});
 key3{1,4}=bitshift(key32{1,4},8);

 key3_2{1,5}=['key3_15{1,1}' 'key3_16{1,1}'
 'key3_15{1,3}' 'key3_16{1,3}' 'key3_15{1,5}'
  'key3_16{1,5}' 'key3_15{1,7}' 'key3_16{1,7}'];
 key3{1,5}=f_fun(bin_key(key3_2{1,5}));

 key3_2{1,6}=['key3_15{1,2}' 'key3_16{1,2}'
 'key3_15{1,4}' 'key3_16{1,4}' 'key3_15{1,6}'
 'key3_16{1,6}' 'key3_15{1,7}' 'key3_16{1,8}'];
```

```
key31{1,6}=f_fun(bin_key(key3_2{1,6}));
key32{1,6}=uint16(key31{1,6});
key3{1,6}=bitshift(key32{1,6},-4);

key3_2{1,7}=['key3_17{1,1}' 'key3_17{1,2}'
 'key3_17{1,3}' 'key3_17{1,4}' 'key3_17{1,5}'
'key3_17{1,6}' 'key3_17{1,7}' 'key3_17{1,8}'];
key3{1,7}=f_fun(bin_key(key3_2{1,7}));

key3_2{1,8}=['key3_18{1,1}' 'key3_18{1,2}'
'key3_18{1,3}' 'key3_18{1,4}' 'key3_18{1,5}'
 'key3_18{1,6}' 'key3_18{1,7}' 'key3_18{1,8}'];
key31{1,8}=f_fun(bin_key(key3_2{1,8}));
key32{1,8}=uint16(key31{1,8});
key3{1,8}=bitshift(key32{1,8},8);

% generating a matrix of each 16bit int 4*4
key3{2,1}(1,:)=key3{1,1}(1:4);
key3{2,1}(2,:)=key3{1,1}(5:8);
key3{2,1}(3,:)=key3{1,1}(9:12);
key3{2,1}(4,:)=key3{1,1}(13:16);
key3{2,2}(1,:)=key3{1,2}(1:4);
key3{2,2}(2,:)=key3{1,2}(5:8);
key3{2,2}(3,:)=key3{1,2}(9:12);
key3{2,2}(4,:)=key3{1,2}(13:16);
key3{2,3}(1,:)=key3{1,3}(1:4);
key3{2,3}(2,:)=key3{1,3}(5:8);
key3{2,3}(3,:)=key3{1,3}(9:12);
key3{2,3}(4,:)=key3{1,3}(13:16);
key3{2,4}(1,:)=key3{1,4}(1:4);
key3{2,4}(2,:)=key3{1,4}(5:8);
key3{2,4}(3,:)=key3{1,4}(9:12);
key3{2,4}(4,:)=key3{1,4}(13:16);

key3{2,5}(1,:)=key3{1,5}(1:4);
key3{2,5}(2,:)=key3{1,5}(5:8);
key3{2,5}(3,:)=key3{1,5}(9:12);
key3{2,5}(4,:)=key3{1,5}(13:16);
key3{2,6}(1,:)=key3{1,6}(1:4);
key3{2,6}(2,:)=key3{1,6}(5:8);
key3{2,6}(3,:)=key3{1,6}(9:12);
key3{2,6}(4,:)=key3{1,6}(13:16);
key3{2,7}(1,:)=key3{1,7}(1:4);
key3{2,7}(2,:)=key3{1,7}(5:8);
key3{2,7}(3,:)=key3{1,7}(9:12);
key3{2,7}(4,:)=key3{1,7}(13:16);
key3{2,8}(1,:)=key3{1,8}(1:4);
key3{2,8}(2,:)=key3{1,8}(5:8);
key3{2,8}(3,:)=key3{1,8}(9:12);
key3{2,8}(4,:)=key3{1,8}(13:16);
%shifting
key3{3,1}(1,:)=key3{2,1}(1,:);
key3{3,1}(2,:)=circshift(key3{2,1}(2,:),[1 1]);
key3{3,1}(3,:)=circshift(key3{2,1}(3,:),[1 2]);
key3{3,1}(4,:)=circshift(key3{2,1}(4,:),[1 3]);

key3{3,2}(1,:)=key3{2,2}(1,:);
key3{3,2}(2,:)=circshift(key3{2,2}(2,:),[1 1]);
key3{3,2}(3,:)=circshift(key3{2,2}(3,:),[1 2]);
key3{3,2}(4,:)=circshift(key3{2,2}(4,:),[1 3]);

```

```
key3{3,3}(1,:)=key3{2,3}(1,:);
key3{3,3}(2,:)=circshift(key3{2,3}(2,:),[1 1]);
key3{3,3}(3,:)=circshift(key3{2,3}(3,:),[1 2]);
key3{3,3}(4,:)=circshift(key3{2,3}(4,:),[1 3]);

key3{3,4}(1,:)=key3{2,4}(1,:);
key3{3,4}(2,:)=circshift(key3{2,4}(2,:),[1 1]);
key3{3,4}(3,:)=circshift(key3{2,4}(3,:),[1 2]);
key3{3,4}(4,:)=circshift(key3{2,4}(4,:),[1 3]);

key3{3,5}(1,:)=key3{2,5}(1,:);
key3{3,5}(2,:)=circshift(key3{2,5}(2,:),[1 1]);
key3{3,5}(3,:)=circshift(key3{2,5}(3,:),[1 2]);
key3{3,5}(4,:)=circshift(key3{2,5}(4,:),[1 3]);

key3{3,6}(1,:)=key3{2,6}(1,:);
key3{3,6}(2,:)=circshift(key3{2,6}(2,:),[1 1]);
key3{3,6}(3,:)=circshift(key3{2,6}(3,:),[1 2]);
key3{3,6}(4,:)=circshift(key3{2,6}(4,:),[1 3]);

key3{3,7}(1,:)=key3{2,7}(1,:);
key3{3,7}(2,:)=circshift(key3{2,7}(2,:),[1 1]);
key3{3,7}(3,:)=circshift(key3{2,7}(3,:),[1 2]);
key3{3,7}(4,:)=circshift(key3{2,7}(4,:),[1 3]);

key3{3,8}(1,:)=key3{2,8}(1,:);
key3{3,8}(2,:)=circshift(key3{2,8}(2,:),[1 1]);
key3{3,8}(3,:)=circshift(key3{2,8}(3,:),[1 2]);
key3{3,8}(4,:)=circshift(key3{2,8}(4,:),[1 3]);

x1=key3{3,1}; %Modified
w1=~key3{3,1};
key3_1{8,1}=not(xor(x1,w1));
key3_1{8,2}=key3{3,2}; %Modified
key3_1{8,3}=key3{3,3}; %Modified
key3_1{8,4}=key3{3,4}; %Modified

key3_1{8,5}=key3{3,5};%Modified
key3_1{8,6}=key3{3,6}; %Modified
key3_1{8,7}=key3{3,7}; %Modified
x8=key3{3,8}; %Modified
w8=~key3{3,8};
key3_1{8,8}=xor(w8,x8);

key3_1{9,1}=[key3_1{8,1}(1,:),key3_1{8,1}(2,:)
,key3_1{8,1}(3,:),key3_1{8,1}(4,:)];
key3_1{9,2}=[key3_1{8,2}(:,1)',
flipdim(key3_1{8,2}(:,2)',2),key3_1{8,2}(:,3)',...
    flipdim(key3_1{8,2}(:,4)',2)];
key3_1{9,3}=[key3_1{8,3}(:,1)',
flipdim(key3_1{8,3}(:,2)',2),key3_1{8,3}(:,3)',...
    flipdim(key3_1{8,3}(:,4)',2)];
key3_1{9,4}=[key3_1{8,4}(1,:),key3_1{8,4}(2,:),
key3_1{8,4}(3,:),key3_1{8,4}(4,:)];

key3_1{9,5}=[key3_1{8,5}(1,:),key3_1{8,5}(2,:),
key3_1{8,5}(3,:),key3_1{8,5}(4,:)];
key3_1{9,6}=[key3_1{8,6}(:,1)',
flipdim(key3_1{8,6}(:,2)',2),key3_1{8,6}(:,3)',...
    flipdim(key3_1{8,6}(:,4)',2)];
key3_1{9,7}=[key3_1{8,7}(:,1)',
```

```
242   flipdim(key3_1{8,7}(:,2)',2),key3_1{8,7}(:,3)',...
243       flipdim(key3_1{8,7}(:,4)',2)];
244   key3_1{9,8}=[key3_1{8,8}(1,:),key3_1{8,8}(2,:),
245   key3_1{8,8}(3,:),key3_1{8,8}(4,:)];
246
247   a=key3_1{9,1};
248   b=key3_1{9,2};
249   c=key3_1{9,3};
250   d=key3_1{9,4};
251   e=key3_1{9,5};
252   f=key3_1{9,6};
253   g=key3_1{9,7};
254   h=key3_1{9,8};
255   %K1
256   A=uint16(a);
257   K1=bitshift(A,5);
258   %K2
259   B=uint16(b);
260   K2=bitshift(B,-9);
261   %K3
262   C=uint16(c);
263   K3=bitshift(C,5);
264   %K4
265   D=uint16(d);
266   K4=bitshift(D,-9);
267   %K5
268   E=uint16(e);
269   K5=bitshift(E,5);
270   %K6
271   F=uint16(f);
272   K6=bitshift(F,-9);
273   %K7
274   G=uint16(g);
275   K7=bitshift(G,5);
276   %K8
277   H=uint16(h);
278   K8=bitshift(H,-9);
279   %K9
280   t3=xor(~K1,d);
281   t1=uint16(t3);
282   t2=bitshift(t1,-5);
283   t6=xor(K5,~h);
284   t4=uint16(t6);
285   t5=bitshift(t4,9);
286   K9=xor(t2,t5);
```

Appendix C. Matlab Code for Encryption Function

```
1   Function[cipher]=SF_Encrypt
2   (bin_msg,K1,K2,K3,K4,K5,K6,K7,K8,K9);
3   enc=cell(12,4);
4   % Arranging 64 bit into 16 bit block
5   enc1{1,1}=bin_msg(1:16);
6   enc11{1,1}=bin_msg(1:2);
7   enc12{1,1}=bin_msg(3:4);
8   enc13{1,1}=bin_msg(5:6);
9   enc14{1,1}=bin_msg(7:8);
10   enc15{1,1}=bin_msg(9:10);
11   enc16{1,1}=bin_msg(11:12);
12   enc17{1,1}=bin_msg(13:14);
13   enc18{1,1}=bin_msg(15:16);
14
15   enc2{1,2}=bin_msg(17:32);
16   enc21{1,2}=bin_msg(17:18);
17   enc22{1,2}=bin_msg(19:20);
18   enc23{1,2}=bin_msg(21:22);
19   enc24{1,2}=bin_msg(23:24);
20   enc25{1,2}=bin_msg(25:26);
21   enc26{1,2}=bin_msg(27:28);
22   enc27{1,2}=bin_msg(29:30);
23   enc28{1,2}=bin_msg(31:32);
24
25   enc3{1,3}=bin_msg(33:48);
26   enc31{1,3}=bin_msg(33:34);
27   enc32{1,3}=bin_msg(35:36);
28   enc33{1,3}=bin_msg(37:38);
29   enc34{1,3}=bin_msg(39:40);
30   enc35{1,3}=bin_msg(41:42);
31   enc36{1,3}=bin_msg(43:44);
32   enc37{1,3}=bin_msg(45:46);
33   enc38{1,3}=bin_msg(47:48);
34
35   enc4{1,4}=bin_msg(49:64);
36   enc41{1,4}=bin_msg(49:50);
37   enc42{1,4}=bin_msg(51:52);
38   enc43{1,4}=bin_msg(53:54);
39   enc44{1,4}=bin_msg(55:56);
40   enc45{1,4}=bin_msg(57:58);
41   enc46{1,4}=bin_msg(59:60);
42   enc47{1,4}=bin_msg(61:62);
43   enc48{1,4}=bin_msg(63:64);
44
45   enc{1,1}=[enc42{1,4},enc32{1,3},enc22{1,2},enc12{1,1}
46   ,enc48{1,4},enc38{1,3},enc28{1,2},enc18{1,1}];
47   enc{1,2}=[enc46{1,4},enc36{1,3},enc26{1,2},enc16{1,1}
48   ,~enc46{1,4},~enc36{1,3},~enc26{1,2},~enc16{1,1}];
49   enc{1,3}=[enc45{1,4},enc35{1,3},enc25{1,2},enc15{1,1}
50   ,~enc45{1,4},~enc35{1,3},~enc25{1,2},~enc15{1,1}];
51   enc{1,4}=[enc44{1,4},enc34{1,3},enc24{1,2},enc14{1,1}
52   ,~enc44{1,4},~enc34{1,3},~enc24{1,2},~enc14{1,1}];
53   enc{1,5}=[enc43{1,4},enc33{1,3},enc23{1,2},enc13{1,1}
54   ,~enc43{1,4},~enc33{1,3},~enc23{1,2},~enc13{1,1}];
55   enc{1,6}=[enc41{1,4},enc31{1,3},enc21{1,2},enc11{1,1}
56   ,enc47{1,4},enc37{1,3},enc27{1,2},enc17{1,1}];
57
58   d=hex2dec('9e3779b9');
59   delta=uint16(d);
```

```
% For First round
%Step no 01
A1=uint16(enc{1,1});
enc{2,1}=xor(K1,bitshift(A1,5));
%performing xnor operation in first 16 bits with K1
enc2{2,2}=not(xor(K1,enc{1,2}));
enc{2,2}=xor(enc2{2,2},delta);
%
enc{2,3}=xor(f_fun(enc{2,2}),enc{1,4});
%performing xnor operation in last 16 bits with K1
enc{2,5}=not(xor(K1,enc{1,5}));
%
enc2{2,4}=xor(f_fun(enc{2,5}),enc{1,3});
enc{2,4}=xor(enc2{2,4},delta);
%
B1=uint16(enc{1,6});
C1=xor(K1,delta);
enc{2,6}=xor(C1,bitshift(B1,5));

% For 2 Round
A2=uint16(enc{2,1});
enc{3,1}=xor(K2,bitshift(A2,5));
%performing Xnor operation with K2
enc2{3,2}=not(xor(K2,enc{2,2}));
enc{3,2}=xor(enc2{3,2},delta);
%
enc{3,3}=xor(f_fun(enc{3,2}),enc{2,3});
%Performing xnor operation on last 16 bit
enc{3,5}=not(xor(K2,enc{2,5}));
%
enc2{3,4}=xor(f_fun(enc{3,5}),enc{2,4});
enc{3,4}=xor(enc2{3,4},delta);
%
B2=uint16(enc{2,6});
C2=xor(K2,delta);
enc{3,6}=xor(C2,bitshift(B2,5));

% For 3 Round
A3=uint16(enc{3,1});
enc{4,1}=xor(K3,bitshift(A3,1));
 %performing Xnor operation with K3
 enc2{4,2}=not(xor(K3,enc{3,2}));
 enc{4,2}=xor(enc2{4,2},delta);
 %
 enc{4,3}=xor(f_fun(enc{4,2}),enc{3,3});
 %Performing xnor operation on last 16 bit
 enc{4,5}=not(xor(K3,enc{3,5}));
 %
 enc2{4,4}=xor(f_fun(enc{4,5}),enc{3,4});
 enc{4,4}=xor(enc2{4,4},delta);
 %
 B3=uint16(enc{3,6});
 C3=xor(K3,delta);
 enc{4,6}=xor(C3,bitshift(B3,5));

 % For 4 Round
 A4=uint16(enc{4,1});
 enc{5,1}=xor(K4,bitshift(A4,5));
 %performing xnor operation in first 16 bits with K4
 enc2{5,2}=not(xor(K4,enc{4,2}));
 enc{5,2}=xor(enc2{5,2},delta);
```

```
%
enc{5,3}=xor(f_fun(enc{5,2}),enc{4,3});
%performing xnor operation in last 16 bits with K1
enc{5,5}=not(xor(K4,enc{4,5}));
%
enc2{5,4}=xor(f_fun(enc{5,5}),enc{4,4});
enc{5,4}=xor(enc2{5,4},delta);
%
B4=uint16(enc{4,6});
C4=xor(K4,delta);
enc{5,6}=xor(C4,bitshift(B4,5));

%Round no 5
A5=uint16(enc{5,1});
enc{6,1}=xor(K5,bitshift(A5,5));
%performing Xnor operation with K5
enc2{6,2}=not(xor(K5,enc{5,2}));
enc{6,2}=xor(enc2{6,2},delta);
%
enc{6,3}=xor(f_fun(enc{6,2}),enc{5,3});
%Performing xnor operation on last 16 bit
enc{6,5}=not(xor(K5,enc{5,5}));
%
enc2{6,4}=xor(f_fun(enc{6,5}),enc{5,4});
enc{6,4}=xor(enc2{6,4},delta);
%
B5=uint16(enc{5,6});
C5=xor(K5,delta);
enc{6,6}=xor(C5,bitshift(B5,5));

%Round no 6
A6=uint16(enc{6,1});
enc{7,1}=xor(K6,bitshift(A6,5));
%performing Xnor operation with K6
enc2{7,2}=not(xor(K6,enc{6,2}));
enc{7,2}=xor(enc2{7,2},delta);
%
enc{7,3}=xor(f_fun(enc{7,2}),enc{6,3});
%Performing xnor operation on last 16 bit
enc{7,5}=not(xor(K6,enc{6,5}));
%
enc2{7,4}=xor(f_fun(enc{7,5}),enc{6,4});
enc{7,4}=xor(enc2{7,4},delta);
%
B6=uint16(enc{6,6});
C6=xor(K6,delta);
enc{7,6}=xor(C6,bitshift(B6,5));

%Round no 7
A7=uint16(enc{7,1});
enc{8,1}=xor(K7,bitshift(A7,5));
%performing Xnor operation with K7
enc2{8,2}=not(xor(K7,enc{7,2}));
enc{8,2}=xor(enc2{8,2},delta);
%
enc{8,3}=xor(f_fun(enc{8,2}),enc{7,3});
%Performing xnor operation on last 16 bit
enc{8,5}=not(xor(K7,enc{7,5}));
%
enc2{8,4}=xor(f_fun(enc{8,5}),enc{7,4});
enc{8,4}=xor(enc2{8,4},delta);
```

```
%
B7=uint16(enc{7,6});
C7=xor(K7,delta);
enc{8,6}=xor(C7,bitshift(B7,5));

%Round no 8
A8=uint16(enc{8,1});
enc{9,1}=xor(K8,bitshift(A8,5));
%performing Xnor operation with K8
enc2{9,2}=not(xor(K8,enc{8,2}));
enc{9,2}=xor(enc2{9,2},delta);
%
enc{9,3}=xor(f_fun(enc{9,2}),enc{8,3});
%Performing xnor operation on last 16 bit
enc{9,5}=not(xor(K8,enc{8,5}));
%
enc2{9,4}=xor(f_fun(enc{9,5}),enc{8,4});
enc{9,4}=xor(enc2{9,4},delta);
%
B8=uint16(enc{8,6});
C8=xor(K8,delta);
enc{9,6}=xor(C8,bitshift(B8,5));

%Round no 9
A9=uint16(enc{9,1});
enc2{10,1}=bitshift(A9,5);
enc{10,1}=xor(K9,enc2{10,1});
%performing Xnor operation with K9
enc2{10,2}=not(xor(K9,enc{9,2}));
enc{10,2}=xor(enc2{10,2},delta);
%
enc{10,3}=xor(f_fun(enc{10,2}),enc{9,3});
%Performing xnor operation on last 16 bit
enc{10,5}=not(xor(K9,enc{9,5}));
%
enc2{10,4}=xor(f_fun(enc{10,5}),enc{9,4});
enc{10,4}=xor(enc2{10,4},delta);
%
B9=uint16(enc{9,6});
enc2{10,6}=bitshift(B9,5);
C9=xor(K9,delta);
enc{10,6}=xor(C9,enc2{10,6});

%Swaping Blocks
enc{11,1}=f_fun(enc{10,6});
enc{11,2}=xor(f_fun2(enc{10,3}),f_fun4(enc{10,2}));
enc{11,3}=xor(f_fun3(enc{10,5}),f_fun(enc{10,4}));
enc{11,4}=f_fun4(enc{10,1});
%
enc{12,1}=enc{11,2};
enc{12,2}=enc{11,3};
enc{12,3}=enc{11,4};
enc{12,4}=enc{11,1};

cipher=[enc{12,1},enc{12,2},enc{12,3},enc{12,4}];
```

Appendix D. Matlab Code for Decryption Function

```
function[bin_msg]=SF_Decryption
(binary_cipher,K1,K2,K3,K4,K5,K6,K7,K8,K9);
%Decryption
dec=cell(12,4);
% Arranging 64 bit into 16 bit block

dec1{1,1}=binary_cipher(1:16);
dec11{1,1}=binary_cipher(1:2);
dec12{1,1}=binary_cipher(3:4);
 dec13{1,1}=binary_cipher(5:6);
 dec14{1,1}=binary_cipher(7:8);
 dec15{1,1}=binary_cipher(9:10);
 dec16{1,1}=binary_cipher(11:12);
 dec17{1,1}=binary_cipher(13:14);
 dec18{1,1}=binary_cipher(15:16);

 dec2{1,2}=binary_cipher(17:32);
 dec21{1,2}=binary_cipher(17:18);
 dec22{1,2}=binary_cipher(19:20);
 dec23{1,2}=binary_cipher(21:22);
 dec24{1,2}=binary_cipher(23:24);
 dec25{1,2}=binary_cipher(25:26);
 dec26{1,2}=binary_cipher(27:28);
 dec27{1,2}=binary_cipher(29:30);
 dec28{1,2}=binary_cipher(31:32);

 dec3{1,3}=binary_cipher(33:48);
 dec31{1,3}=binary_cipher(33:34);
 dec32{1,3}=binary_cipher(35:36);
 dec33{1,3}=binary_cipher(37:38);
 dec34{1,3}=binary_cipher(39:40);
 dec35{1,3}=binary_cipher(41:42);
 dec36{1,3}=binary_cipher(43:44);
 dec37{1,3}=binary_cipher(45:46);
 dec38{1,3}=binary_cipher(47:48);

 dec4{1,4}=binary_cipher(49:64);
 dec41{1,4}=binary_cipher(49:50);
 dec42{1,4}=binary_cipher(51:52);
 dec43{1,4}=binary_cipher(53:54);
 dec44{1,4}=binary_cipher(55:56);
 dec45{1,4}=binary_cipher(57:58);
 dec46{1,4}=binary_cipher(59:60);
 dec47{1,4}=binary_cipher(61:62);
 dec48{1,4}=binary_cipher(63:64);

 dec{1,1}=[dec42{1,4},dec32{1,3},dec22{1,2},dec12{1,1}
 ,dec48{1,4},dec38{1,3},dec28{1,2},dec18{1,1}];
 dec{1,2}=[dec46{1,4},dec36{1,3},dec26{1,2},dec16{1,1}
 ,~dec46{1,4},~dec36{1,3},~dec26{1,2},~dec16{1,1}];
 dec{1,3}=[dec45{1,4},dec35{1,3},dec25{1,2},dec15{1,1}
 ,~dec45{1,4},~dec35{1,3},~dec25{1,2},~dec15{1,1}];
 dec{1,4}=[dec44{1,4},dec34{1,3},dec24{1,2},dec14{1,1}
 ,~dec44{1,4},~dec34{1,3},~dec24{1,2},~dec14{1,1}];
 dec{1,5}=[dec43{1,4},dec33{1,3},dec23{1,2},dec13{1,1}
 ,~dec43{1,4},~dec33{1,3},~dec23{1,2},~dec13{1,1}];
 dec{1,6}=[dec41{1,4},dec31{1,3},dec21{1,2},dec11{1,1}
 ,dec47{1,4},dec37{1,3},dec27{1,2},dec17{1,1}];
```

```
%
d=hex2dec('9e3779b9');
delta=uint16(d);

% For round 1
C1=xor(K9,delta);
dec1{2,1}=xor(C1,dec{1,1});
B1=uint16(dec1{2,1});
dec2{2,1}=bitshift(B1,5);
dec{2,1}=xor(C1,dec2{2,1});
%
dec{2,2}=not(xor(K9,dec{1,2}));
%
dec2{2,4}=xor(dec{1,3},delta);
dec{2,4}=xor(f_fun(dec{1,2}),dec2{2,4});
%
dec2{2,5}=xor(dec{1,5},delta);
dec{2,5}=not(xor(K9,dec2{2,5}));
%
dec{2,3}=xor(f_fun(dec2{2,5}),dec{1,4});
%
A1=uint16(dec{1,6});
dec2{2,6}=bitshift(A1,5);
dec{2,6}=xor(K9,dec2{2,6});
%

% For 2 Round
C2=xor(K8,delta);
dec1{3,1}=xor(C2,dec{2,1});
B2=uint16(dec1{3,1});
dec2{3,1}=bitshift(B2,5);
dec{3,1}=xor(C2,dec2{3,1});
%
dec{3,2}=not(xor(K8,dec{2,2}));
%
dec2{3,4}=xor(dec{2,3},delta);
dec{3,4}=xor(f_fun(dec{2,2}),dec2{3,4});
%
dec2{3,5}=xor(dec{2,3},delta);
dec{3,5}=not(xor(K8,dec2{3,5}));
%
 dec{3,3}=xor(f_fun(dec2{3,5}),dec{2,4});
 %
 A2=uint16(dec{2,6});
 dec2{3,6}=bitshift(A2,5);
 dec{3,6}=xor(K8,dec2{3,6});
 %Step no 02

 %Round no 3
 C3=xor(K7,delta);
 dec1{4,1}=xor(C3,dec{3,1});
 B3=uint16(dec1{4,1});
 dec2{4,1}=bitshift(B3,5);
 dec{4,1}=xor(C3,dec2{4,1});
 %
 dec{4,2}=not(xor(K7,dec{3,2}));
 %
 dec2{4,4}=xor(dec{3,3},delta);
 dec{4,4}=xor(f_fun(dec{3,2}),dec2{4,4});
 %
 dec2{4,5}=xor(dec{3,5},delta);
```

```
dec{4,5}=not(xor(K7,dec2{4,5}));
%
dec{4,3}=xor(f_fun(dec2{4,5}),dec{3,4});
%
A3=uint16(dec{3,6});
dec2{4,6}=bitshift(A3,5);
dec{4,6}=xor(K7,dec2{4,6});

%Round no 4
C4=xor(K6,delta);
dec1{5,1}=xor(C4,dec{4,1});
B4=uint16(dec1{5,1});
dec2{5,1}=bitshift(B4,5);
dec{5,1}=xor(C4,dec2{5,1});
%
dec{5,2}=not(xor(K6,dec{4,2}));
%
dec2{5,4}=xor(dec{4,3},delta);
dec{5,4}=xor(f_fun(dec{4,2}),dec2{5,4});
%
dec2{5,5}=xor(dec{4,5},delta);
dec{5,5}=not(xor(K6,dec2{5,5}));
%
dec{5,3}=xor(f_fun(dec2{5,5}),dec{4,4});
%
A4=uint16(dec{4,6});
dec2{5,6}=bitshift(A4,5);
dec{5,6}=xor(K6,dec2{5,6});

%Round no 5
C5=xor(K5,delta);
dec1{6,1}=xor(C5,dec{5,1});
B5=uint16(dec1{6,1});
dec2{6,1}=bitshift(B5,5);
dec{6,1}=xor(C5,dec2{6,1});
%
dec{6,2}=not(xor(K5,dec{5,2}));
%
dec2{6,4}=xor(dec{5,3},delta);
dec{6,4}=xor(f_fun(dec{5,2}),dec2{6,4});
%
dec2{6,5}=xor(dec{5,5},delta);
dec{6,5}=not(xor(K5,dec2{6,5}));
%
dec{6,3}=xor(f_fun(dec2{6,5}),dec{5,4});
%
A5=uint16(dec{5,6});
dec2{6,6}=bitshift(A5,5);
dec{6,6}=xor(K5,dec2{6,6});
%
%Round no 6
C6=xor(K4,delta);
dec1{7,1}=xor(C6,dec{6,1});
B6=uint16(dec1{7,1});
dec2{7,1}=bitshift(B6,5);
dec{7,1}=xor(C6,dec2{7,1});
%
dec{7,2}=not(xor(K4,dec{6,2}));
%
dec2{7,4}=xor(dec{6,3},delta);
dec{7,4}=xor(f_fun(dec{6,2}),dec2{7,4});
```

```
%
dec2{7,5}=xor(dec{6,5},delta);
dec{7,5}=not(xor(K4,dec2{7,5}));
%
dec{7,3}=xor(f_fun(dec2{7,5}),dec{6,4});
%
A6=uint16(dec{6,6});
dec2{7,6}=bitshift(A6,5);
dec{7,6}=xor(K4,dec2{7,6});

%Round no 7
C7=xor(K3,delta);
dec1{8,1}=xor(C7,dec{7,1});
B7=uint16(dec1{8,1});
dec2{8,1}=bitshift(B7,5);
dec{8,1}=xor(C7,dec2{8,1});
%
dec{8,2}=not(xor(K3,dec{7,2}));
%
dec2{8,4}=xor(dec{7,3},delta);
dec{8,4}=xor(f_fun(dec{7,2}),dec2{8,4});
%
dec2{8,5}=xor(dec{7,5},delta);
dec{8,5}=not(xor(K3,dec2{8,5}));
%
dec{8,3}=xor(f_fun(dec2{8,5}),dec{7,4});
%
A7=uint16(dec{7,6});
dec2{8,6}=bitshift(A7,5);
dec{8,6}=xor(K4,dec2{8,6});

%Round no 8
C8=xor(K2,delta);
dec1{9,1}=xor(C8,dec{8,1});
B8=uint16(dec1{9,1});
dec2{9,1}=bitshift(B8,5);
dec{9,1}=xor(C8,dec2{9,1});
%
dec{9,2}=not(xor(K2,dec{8,2}));
%
dec2{9,4}=xor(dec{8,3},delta);
dec{9,4}=xor(f_fun(dec{8,2}),dec2{9,4});
%
dec2{9,5}=xor(dec{8,5},delta);
dec{9,5}=not(xor(K2,dec2{9,5}));
%
dec{9,3}=xor(f_fun(dec2{9,5}),dec{8,4});
%
A8=uint16(dec{8,6});
dec2{9,6}=bitshift(A8,5);
dec{9,6}=xor(K3,dec2{9,6});

%Round no 9
C9=xor(K1,delta);
dec1{10,1}=xor(C9,dec{9,1});
B9=uint16(dec1{10,1});
dec2{10,1}=bitshift(B9,5);
dec{10,1}=xor(C9,dec2{10,1});
%
dec{10,2}=not(xor(K1,dec{9,2}));
%
```

```
242   dec2{10,4}=xor(dec{9,3},delta);
243   dec{10,4}=xor(f_fun(dec{9,2}),dec2{10,4});
244   %
245   dec2{10,5}=xor(dec{9,5},delta);
246   dec{10,5}=not(xor(K1,dec2{10,5}));
247   %
248   dec{10,3}=xor(f_fun(dec2{10,5}),dec{9,4});
249   %
250   A9=uint16(dec{9,6});
251   dec2{10,6}=bitshift(A9,5);
252   dec{10,6}=xor(K3,dec2{10,6});
253   %
254   dec{11,1}=f_fun(dec{10,6});
255   dec{11,2}=xor(f_fun2(dec{10,3}),f_fun4(dec{10,2}));
256   dec{11,3}=xor(f_fun3(dec{10,5}),f_fun(dec{10,4}));
257   dec{11,4}=f_fun4(dec{10,1});
258   %
259   dec{12,1}=dec{11,2};
260   dec{12,2}=dec{11,3};
261   dec{12,3}=dec{11,4};
262   dec{12,4}=dec{11,1};
263
264   bin_msg=[dec{12,1},dec{12,2},dec{12,3},dec{12,4}];
```

Appendix E. Matlab Code for Scalling Function

```
1   function [Data,padding] = Scalling (Data,sf);
2   % Data=imread('cameraman.tif');
3   Data=reshape(Data,[size(Data,1)*size(Data,2) 1]);
4   Data=double(Data);
5   padding=mod(length(Data),sf);
6   if (padding~=0)
7       padding=sf-padding;
8   end
9   Data=[Data;zeros(padding,1)];
10   Data=reshape(Data,[sf,length(Data)/sf]);
```

Appendix F. Matlab Code for P Function

```
1  function O = P(I)
2  if(I==logical([0 0 0 0]))
3           O = logical([0 0 1 1]); end
4  if(I==logical([0 0 0 1]))
5           O = logical([1 1 1 1]); end
6  if(I==logical([0 0 1 0]))
7           O = logical([1 1 1 0]); end
8  if(I==logical([0 0 1 1]))
9           O = logical([0 0 0 0]); end
10  if(I==logical([0 1 0 0]))
11           O = logical([0 1 0 1]); end
12  if(I==logical([0 1 0 1]))
13           O = logical([0 1 0 0]); end
14  if(I==logical([0 1 1 0]))
15           O = logical([1 0 1 1]); end
16  if(I==logical([0 1 1 1]))
17           O = logical([1 1 0 0]); end
18  if(I==logical([1 0 0 0]))
19           O = logical([1 1 0 1]); end
20  if(I==logical([1 0 0 1]))
21           O = logical([1 0 1 0]); end
22  if(I==logical([1 0 1 0]))
23           O = logical([1 0 0 1]); end
24  if(I==logical([1 0 1 1]))
25           O = logical([0 1 1 0]); end
26  if(I==logical([1 1 0 0]))
27           O = logical([0 1 1 1]); end
28  if(I==logical([1 1 0 1]))
29           O = logical([1 0 0 0]); end
30  if(I==logical([1 1 1 0]))
31           O = logical([0 0 1 0]); end
32  if(I==logical([1 1 1 1]))
33           O = logical([0 0 0 1]); end
34  End
```

Appendix G. Matlab Code for Q Function

```
function O = Q_fun(I)
if(I==logical([0 0 0 0]))
        O = logical([1 0 0 1]); end
if(I==logical([0 0 0 1]))
        O = logical([1 1 1 0]); end
if(I==logical([0 0 1 0]))
        O = logical([0 1 0 1]); end
if(I==logical([0 0 1 1]))
        O = logical([0 1 1 0]); end
 if(I==logical([0 1 0 0]))
         O = logical([1 0 1 0]); end
 if(I==logical([0 1 0 1]))
         O = logical([0 0 1 0]); end
 if(I==logical([0 1 1 0]))
         O = logical([0 0 1 1]); end
 if(I==logical([0 1 1 1]))
         O = logical([1 1 0 0]); end
 if(I==logical([1 0 0 0]))
         O = logical([1 1 1 1]); end
 if(I==logical([1 0 0 1]))
         O = logical([0 0 0 0]); end
 if(I==logical([1 0 1 0]))
         O = logical([0 1 0 0]); end
 if(I==logical([1 0 1 1]))
         O = logical([1 1 0 1]); end
 if(I==logical([1 1 0 0]))
         O = logical([0 1 1 1]); end
 if(I==logical([1 1 0 1]))
         O = logical([1 0 1 1]); end
 if(I==logical([1 1 1 0]))
         O = logical([0 0 0 1]); end
 if(I==logical([1 1 1 1]))
         O = logical([1 0 0 0]); end
 End
```

Appendix H. Matlab Code for Function 1

```
function a = f_fun(a)
a=[P_fun(a(1:4)),Q_fun(a(5:8)),P_fun(a(9:12))
,Q_fun(a(13:16))];
a=[Q_fun([a(1:2),a(5:6)]),P_fun([a(3:4),a(7:8)])
,Q_fun([a(9:10),a(13:14)]),P_fun([a(11:12),a(15:16)])];
a=[P_fun([a(1:2),a(5:6)]),Q_fun([a(3:4),a(7:8)])
,P_fun([a(9:10),a(13:14)]),Q_fun([a(11:12),a(15:16)])];
end
```

Appendix I. Matlab Code for Function 2

```
function a = f_fun2(a)
a=[P_fun(a(1:4)),Q_fun(a(5:8)),P_fun(a(9:12))
,Q_fun(a(13:16))];
a=[Q_fun([a(3:4),a(1:2)]),P_fun([a(7:8),a(5:6)])
,Q_fun([a(11:12),a(9:10)]),P_fun([a(15:16),a(13:14)])];
a=[P_fun([a(3:4),a(1:2)]),Q_fun([a(7:8),a(5:6)])
,P_fun([a(11:12),a(9:10)]),Q_fun([a(15:16),a(13:14)])];
end
```

Appendix J. Matlab Code for Function 3

```
function a = f_fun3(a)
a=[P_fun(a(1:4)),Q_fun(a(5:8)),P_fun(a(9:12))
,Q_fun(a(13:16))];
a=[Q_fun([a(7:8),a(1:2)]),P_fun([a(3:4),a(5:6)])
,Q_fun([a(15:16),a(9:10)]),P_fun([a(11:12),a(13:14)])];
a=[P_fun([a(7:8),a(1:2)]),Q_fun([a(3:4),a(5:6)])
,P_fun([a(15:16),a(9:10)]),Q_fun([a(11:12),a(13:14)])];
end
```

Appendix K. Matlab Code for Function 4

```
function a = f_fun4(a)
a=[P_fun(a(1:4)),Q_fun(a(5:8)),P_fun(a(9:12))
,Q_fun(a(13:16))];
a=[Q_fun([a(15:16),a(1:2)]),P_fun([a(13:14)
,a(3:4)]),Q_fun([a(11:12),a(5:6)]),P_fun([a(9:10)
,a(7:8)])];
a=[P_fun([a(15:16),a(1:2)]),Q_fun([a(13:14),a(3:4)])
,P_fun([a(11:12),a(5:6)]),Q_fun([a(9:10),a(7:8)])];
end
```

Appendix L. Matlab Code for Function of Convert Binary to Decimal

```
function [ dec ] = binary2dec( binary )
%binaryB2H Summary of this function goes here
%   Detailed explanation goes here
E1=bin2dec(num2str(binary(1:8)));
 E2=bin2dec(num2str(binary(9:16)));
 E3=bin2dec(num2str(binary(17:24)));
 E4=bin2dec(num2str(binary(25:32)));
E5=bin2dec(num2str(binary(33:40)));
 E6=bin2dec(num2str(binary(41:48)));
 E7=bin2dec(num2str(binary(49:56)));
  E8=bin2dec(num2str(binary(57:64)));
  dec=cell2mat({E1 E2 E3 E4 E5 E6 E7 E8});
 end
```

Appendix M. Matlab Code for Function of Convert to Binary

```
function [ Data_binary ] = convert2bin( Data )
%CONVERT2BIN Summary of this function goes here
%   Detailed explanation goes here
Data_binary=[];
for j=1:size(Data,2)
temp=[];
for i=1:size(Data,1)
temp = [temp Dec2Bin(Data(i,j))];
end
 Data_binary=[Data_binary;temp];
 end
 end
```

Appendix N. Matlab Code for Function of Convert Decimal to Binary

```
1   function [ bin ] = Dec2Bin( dec )
2   %DEC2BIN Summary of this function goes here
3   %   Detailed explanation goes here
4   % bin=logical([]);
5   hex=[];
6   for b=1:length(dec)
7       hex=[hex  dec2hex(dec(b),2)];
8   end
9   [bin] = Hex2Bin(hex);
10   End
```

Appendix O. Matlab Code for Function of Convert Hexadecimal to Binary

```
function [ bin ] = Hex2Bin( hex )
%HEX2BIN Summary of this function goes here
%   Detailed explanation goes here
bin=logical([]);
for b=1:length(hex)
    bin=[bin  logical(h2b(hex(b)))];
end
end
```

Appendix P. Matlab Code for Function 2 of Convert Hexadecimal to Binary

```
function b = h2b(h)
switch h
    case {'0'}
        b = logical([0 0 0 0]);
    case {'1'}
        b = logical([0 0 0 1]);
    case {'2'}
        b = logical([0 0 1 0]);
    case {'3'}
         b = logical([0 0 1 1]);
     case {'4'}
         b = logical([0 1 0 0]);
     case {'5'}
         b = logical([0 1 0 1]);
     case {'6'}
         b = logical([0 1 1 0]);
     case {'7'}
         b = logical([0 1 1 1]);
     case {'8'}
         b = logical([1 0 0 0]);
     case {'9'}
         b = logical([1 0 0 1]);
     case {'A', 'a'}
         b = logical([1 0 1 0]);
     case {'B', 'b'}
         b = logical([1 0 1 1]);
     case {'C', 'c'}
         b = logical([1 1 0 0]);
     case {'D', 'd'}
         b = logical([1 1 0 1]);
     case {'E', 'e'}
         b = logical([1 1 1 0]);
     case {'F', 'f'}
         b = logical([1 1 1 1]);
end
```

BIBLIOGRAPHY

Name Surname : Ahmed Mohsin Abed AL-JANABI

Place and Year of Birth : Iraq, 19/02/1985

Marital Status : Married

Foreign Language : Arabic

Email : amabed.janabi@istanbulticaret.edu.tr

Education Status

High School : Sanaa for Boys, 2003.

Undergraduate : Baghdad College for Economic Sciences University, Software Engineering, 2008.

Master : Istanbul Commerce University, Graduate school of Science and Engineering, Computer Engineering, 2020.

Professional experiences

Ministry of Construction, and Housing and Municipalities and Public Works, Information Systems Center, Communications and Networking Department, Internet section, 2009-2017.

Publications

Al-Janabi, A.M.A., Boyacı, A., 2020. A Lightweight Cryptography Algorithm for Secure Smart Cities and IOT, Electrica, 20(2), 168-176.