SECURITY ENHANCEMENT FOR IEEE 802.11 WIRELESS LANs

By

Heidi Kareh Hernández

A project report submitted in partial fulfillment of the requirements for the degree of

MASTER OF ENGINEERING

in

COMPUTER ENGINEERING

(Software Engineering)

University Of Puerto Rico

Mayagüez Campus

2005

Approved by:

_____	_____
Isidoro Couvertier, Ph.D. Member, Graduate Committee	Date
_____	_____
Manuel Rodriguez, Ph.D. Member, Graduate Committee	Date
_____	_____
Yi Qian, Ph.D. President, Graduate Committee	Date
_____	_____
Robert Acar, Ph.D. Representative of Graduate Studies	Date
_____	_____
Isidoro Couvertier, Ph.D. Chairperson of the Department	Date

ABSTRACT

SECURITY ENHANCEMENT FOR IEEE 802.11 WIRELESS LANs

By

Heidi Kareh Hernández

This project focuses on wireless network security and in particular, the possible encryption methods for IEEE 802.11 Wireless LANs. We explore the concept of encryption and the different encryption standards such as Data Encryption Standard (DES), Triple Data Encryption Standard (3DES), and Advanced Encryption Standard (AES). Through this project, we get to understand the randomness of the encryption output and the avalanche effect. We further explored triple encryption of AES using two keys, and brute force attacks with partial knowledge of the key. Finally, we conclude the project with suggestions on security enhancement using AES for IEEE 802.11 Wireless LANs.

RESUMEN

REALCE EN LA SEGURIDAD DE LAS REDES INALÁMBRICAS IEEE 802.11

Por

Heidi Kareh Hernández

.

Este proyecto se centra en la seguridad de las redes inalámbricas y en particular, a los posibles métodos de codificación para las redes inalámbricas de la IEEE 802.11. Exploramos el concepto de encripción y los diferentes estándares de codificación tales como Estándar de Codificación de Data (DES), Triple Estándar de Codificación de Data Standard (3DES), y Estándar de Codificación Avanzado (AES). A través de este proyecto, logramos entender la aleatoriedad de la codificación de la salida y el efecto avalancha. En adición, exploramos la triple codificación de AES usando dos llaves, y ataques de fuerza bruta con conocimiento parcial de la llave. Finalmente, concluimos el proyecto con sugerencias de realce en la seguridad utilizando AES para las redes inalámbricas de la IEEE 802.11.

ACKNOWLEDGEMENTS

First, I would like to thank God for giving me health, allowing me to reach this stage in my life, and taking me this far in my career. Second, I would like to thank my family, especially my father Pedro Kareh for being so caring, supportive, and understanding. In addition, I would like to thank both of my sisters, Sylvia, for being a good listener and a good friend, and Aidyl, for being so strong and persistent. I would also like to thank Jaime Ramos for being there for me, encouraging me to do my best, being a great friend, and for loving me so much.

I would like to give special thanks to my advisor, Dr. Yi Qian, for giving me the opportunity to work with him in this project, for being so supportive, and for spending so much time helping me with the development process of this project. I would like to express gratitude to Dr. Manuel Rodriguez, for encouraging me to do better each time and for believing in me. I would also like to express gratitude to Dr. Isidoro Couvertier, for being the influential point for my interest in wireless networks and wireless network security. Finally, to everyone else that has helped in the development of this project, either directly or indirectly, I am thankful.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In the present times, mobility is something everyday-people prefer. When thinking about mobility, its counterpart is the term wireless, anything that makes it easier for people to access data from any place. From mobile phones, pagers, and PDA's, to laptop computers, people want to be able to reach each other at any place and at any time. Wireless networks help them with that process. Wireless networks have come to be very popular. Almost anywhere you go has a wireless connection available. Airports, coffee houses, restaurants, libraries, colleges, even malls have wireless access connections available for the public use.

Wireless networks use electromagnetic waves to communicate from one point to another without relying on a physical connection. Wireless local area networks (WLANs) transmit and receive data over the air, combining data connectivity and user mobility. At the same time, they facilitate transmission to users by using the open air instead of wires, making networks more appealing to conventional users. The wireless networks also appeared as a possible aid to wired networks. As expressed in [6], WLANs represent a possible solution to the 'last mile' problem in the wired networks. They provide wireless

support for home and business environments where it is too expensive or it is impossible to utilize cables. For many people, wireless networks are the long awaited solution as to how far the network can extend. They are regarded as convenient and inexpensive ways to provide network access in hard-to-wire locations.

Wireless networks are evolving. They are getting to be more popular and more frequently used. Mobility is one of the reasons wireless networks have been so rapidly spreading, but it is not the only one. Other factors that contribute to the popularity of wireless networks are ease of use, interoperability, compatibility, availability, and different modes of operation, to name a few. However, wireless networks are not nearly as secure as people would want them to be. As networks expand beyond physical boundaries, operators are struggling to retain control over network usage and privacy [18]. WLANs are expanding and the amount of people wanting to be connected to them expands as well, therefore making the control and security harder on the network administrators and everyday users. Wireless network security is different from wired network security, primarily because it gives potential attackers an easy transport medium access [4]. By using the air to transfer the wireless signal, it is easier for attacker to intercept the transmitted data. Wireless networks are particularly vulnerable to intrusion, as they operate in open medium, and use cooperative strategies for network communications [4]. Vulnerability of WLANs is of common knowledge to many people. However, many choose to ignore it. Security is an important issue, but unfortunately, the current wireless networks, especially WLANs, have many security flaws.

Many research projects have been done, or are going on, trying to enhance the security mechanisms for the IEEE 802.11 WLANs. The main purpose of this project is to understand the security problems and vulnerabilities in the IEEE 802.11 WLANs, by surveying the current research on the improvement to the security of WLANs, suggesting ideas on security solutions, performing simulations to support the ideas, and proposing our own design and deployment recommendations for the WLAN security enhancement. This project further explores the concepts of wireless network security and goes into the specifics of the Advanced Encryption Standard (AES) used to secure the integrity of the transferred data.

The primary motivation for performing this project is the amount of security attacks and vulnerabilities the IEEE 802.11 Wireless LANs possess. Most of these vulnerabilities arise from using low security encryption algorithms and the fact that wireless networks transfer the data through the open air, making the data available for capture by potential attackers. Wireless networks will continue to be widely used. At the same time, they will continue to transfer data through the open air, therefore, there is a need for a security enhancement that will help protect the data from those imminent attacks.

The primary goal for this project is to perform a research study that will aid us in the proposing of AES for a security enhancement to IEEE 802.11 WLANs, by providing different testing scenarios that will help us better understand the process behind the encryption. The contributions achieved by this project can be summarized in:

- Provide a survey for the current available encryption algorithms including DES, 3DES, and AES for wireless LANs.
- Perform a preliminary study of the AES algorithm to suggest ideas on wireless security by performing simulations, which include the randomness of output and avalanche effect, and the possibility of brute-force attack on AES.
- Suggest the use of Triple AES encryption in case other encryption standards become vulnerable to attacks.
- Recommend the use of AES standard for wireless network security and encryption.

The organization of this project report is as follows. Chapter 2 provides background information on wireless networks and wireless security. It contains the theoretical background to wireless security as well as the encryption concepts and the encryption algorithms of Data Encryption Standard (DES), Triple Data Encryption Standard (3DES), and AES. Chapter 3 discusses the research carried out to evaluate the performance of AES encryption. It contains the simulation study performed as well as the results obtained. Chapter 4 focuses on suggestions made towards AES and wireless network security. Chapter 5 contains conclusions drawn from the study performed on this project and future work.

CHAPTER 2

BACKGROUND AND RELATED WORK

In this chapter, we discuss the wireless networks, from the general to the specific. This chapter contains details on wireless networks, as well as details on its security aspects and flaws. On subsequent sections, we focus on the encryption part of the security, where we go into detail about the different encryption standards such as DES, 3DES and AES. We continue with the related work on the associated topics.

2.1 Wireless Networks Overview

IEEE 802.11 WLANs are currently very popular amid Internet and network users. There are many aspects to wireless networks: from modes of operation, to the attacks performed and their corresponding security solutions. A wireless network consists of a group of computers interconnected between each other through a wireless channel frequency. There are two modes for wireless connections: infrastructure (access point preferred) and ad-hoc (peer-to-peer). The infrastructure mode is characterized for having a centralized access point (AP) to which many computers can connect to and connect through to other computers in the same network.

An AP acts as an Ethernet bridge and forwards the communications onto the appropriate network [3]. In Figure 2.1, we can see a representation of an AP wireless

network. In this figure, the AP serves as a central structure that interconnects the wireless adapters to the wired Internet network. We can see the AP connected to a switch, this is usually done to get to an outside Internet connection or to interconnect to the wired network. We can also see two computers connecting to that AP through the wireless medium. An AP network is the one that is more similar to the wired network. All the computers in the AP network can be authenticated and monitored to maintain the integrity and security of the network. However, the biggest security threat derives from wireless APs being installed in 'default mode', which is the least secure mode [19]. This is a common problem when discussing wireless network security, since people want their networks to be secure, but at the same time, they do not want to spend the time or the money on setting them up in a secure manner.

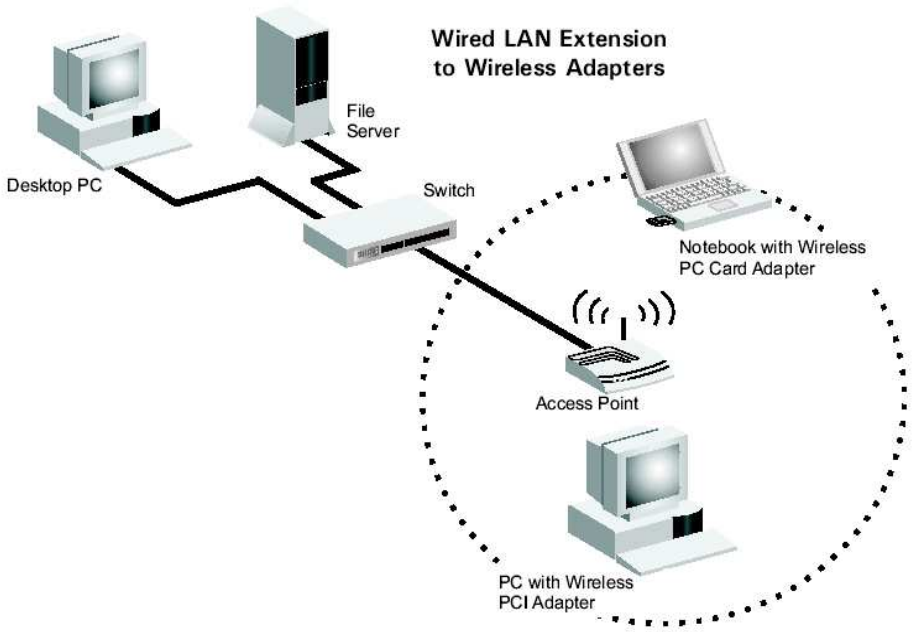

Figure 2.1 Infrastructure (AP) wireless network

Contrary to AP networks, the ad-hoc mode is designed for having computers interconnect with each other as peers, without the need of a centralized AP. As we can see in Figure 2.2, we see the lack of a centralized AP. Nonetheless, computers are interconnected to each other. This type of wireless network is less common because of many different reasons. One of the reasons is that since there is no centralized AP, authentication from one computer to the other is somewhat more complicated than AP authentication, which resembles authentication in wired networks. In ad-hoc mode, computers are interconnected to each other by performing their own authentications.

Figure 2.2 Ad hoc (peer-to-peer) wireless network

Although WLANs in general have security control problems because of weak encryption algorithms, ad-hoc networks are more susceptible to attacks because of the limitation in security solutions. Ad hoc networks are vulnerable to eavesdropping and masquerading attacks. There is no central AP and therefore, authentication cannot take place except from one computer to the next. Access point preferred networks have the

advantage of having a centralized base to which every computer connects, making security a general rule for all of the clients connected to the AP.

As WLANs get popular, different sub-standards have emerged from the IEEE 802.11 general standard. Some of the most popular sub-standards are 802.11a, 802.11b, 802.11d, 802.11e, 802.11f, 802.11g, 802.11h, and 802.11i. A brief description of each standard will follow.

802.11b is the most currently used. The products operate in the unlicensed 2.4 GHz RF band and it supports raw channel signaling rates up to 11 Mbps, depending on factors such as distance and noise. As its default security method, it uses Wired Equivalent Privacy (WEP) or no security at all, depending on the setting used. As the default setting it selects no security, therefore, it is in the hands of the user to protect their 802.11b network data transfer. 802.11g focuses on increasing data throughput of the 2.4 GHz ISM frequency band. It has a throughput that ranges from 32Mbps to 54Mbps and it is the second widely used standard. This standard uses WEP encryption algorithm or no encryption at all, just like 802.11b. 802.11a is a high-speed WLAN standard for the 5 GHz band. It has a throughput of 54Mbps and it possesses the Orthogonal Frequency Division Multiplexing (OFDM) modulation technique. 802.11e focuses on establishing Quality of Service (QoS). 802.11h focuses on power usage and transmission interference from 802.11 radio frequencies. Finally, 802.11i is known as the data security protocol. It considers two encryption algorithms, Temporal Key Integrity Protocol (TKIP) for backward compatibility with WEP and Advanced Encryption Standard (AES) to provide a stronger encryption than WEP.

As time goes by, more IEEE 802.11 standards will keep emerging, as well as updates to the ones that are available at this time. In addition, new security solutions and encryption algorithms are being proposed for use with the wireless networks. At the same time, new attacks and threats will have to be taken into account and new security solutions for these will have to be developed. We will discuss security threats and attacks in the next section.

2.2 Wireless Network Security

It is important to make a distinction between the terms threat and attack. A *threat* is a potential violation of security. In other words, it is a possible danger that might exploit a vulnerability of the system [29]. An *attack* is an actual violation of security. In other words, it is a deliberate attempt to evade security services and to violate the security policy of a system [29]. Attacks on wireless networks fall into four basic categories: passive attacks, active attacks, man-in-the-middle attacks, and jamming attacks [32]. Passive attacks attempt to learn or make use of system information but do not affect system resources [29]. Active attacks attempt to alter system resources or affect their operation.

Passive attacks on wireless networks are extremely common, and at the same time, very difficult to detect. Common passive attacks are those that involve eavesdropping or listening to the network traffic. A passive attack on a wireless network may not be malicious in nature. They help an attacker prepare for an active attack, by providing sufficient information from the system. Once an attacker has gained sufficient

information from the passive attack, he can launch an active attack against the network.

There are different types of active attacks that an adversary can launch against a wireless network. For the most part, these attacks are identical to the kinds of active attacks that are encountered on wired networks. These include, but are not limited to, unauthorized access, spoofing or masquerading, and Denial of Service (DoS) and Flooding attacks [32]. Because of the nature of wireless networks and the weaknesses of WEP, unauthorized access and spoofing are the most common threats to a wireless networks.

Spoofing or masquerading attacks occurs when an adversary is able to use an unauthorized station to impersonate an authorized station on a wireless network. Once the attacker has authenticated and associated with the wireless network, he or she can run port scans, use special tools to dump user lists and passwords, impersonate users, connect to shares, and in general, create disruption on the network through DoS and Flooding attacks. These DoS attacks can be traditional in nature, such as a ping flood, or they can be specific to wireless networks through the placement and use of rogue access points to prevent wireless traffic from being forwarded properly.

Placing a rogue access point within range of wireless stations is wireless-specific variation of a man-in-the-middle attack. Jamming is a special kind of DoS attack specific to wireless networks. Jamming occurs when spurious RF frequencies interfere with the operation of the wireless network. In some cases, the jamming is not malicious and is caused by the presence of other devices, such as a cordless phone, that operate in the same frequency as the wireless network. Intentional and malicious jamming occurs when

an attacker analyzes the spectrum being used by wireless networks and then transmits a powerful signal to interfere with communication on the discovered frequencies. Fortunately, this kind of attack is not very common because of the high expense of acquiring hardware capable of launching jamming attacks.

Wireless networks provide security in different ways. Security is defined as the operations undertaken to protect and defend information and information systems by ensuring their integrity, authentication, confidentiality, and non-repudiation [29]. The IEEE 802.11 standard provides network security through two methods: authentication and encryption [29]. Authentication is the means by which one station is verified to have authorization to communicate with a second station in given coverage area. In the infrastructure mode, authentication is established between an Access Point and each one of the stations. Authentication applies to passwords and keys used to validate the users in the network. Authentication itself can subdivide into two subsets: Open System and Shared Key System. In an Open System, any wireless station may request authentication to the AP. On the other hand, in a Shared Key System, only the wireless stations that possess the secret encrypted key are the authenticated ones.

Encryption is most widely used to ensure data privacy. It should provide a level of security comparable to that of a wired LAN. Commonly used after user authentication has taken place, its primal function is to maintain the integrity of the transferred data. Encryption algorithms have been used for many years and some of them have been made into encryption standards [26]. When talking about wireless network security encryption is one of the most important mechanisms used to protect the data.

2.3 Encryption Algorithms

Encryption algorithms are of great importance in the process of securing a network and the data transferred though it. In the following subsections, we will explain some important definitions, and then we will do a theoretical background on the encryption algorithms. After this part, we will go into the details of three of the most important encryption algorithms DES, 3DES, and AES. Finalizing this section, we will talk about some related work on the associated topics.

2.3.1 Definitions

Before continuing to the next session, terms such as cryptography, encryption, decryption, plaintext, cipher, ciphertext, avalanche effect, and brute-force attack, to name a few, will be defined. *Cryptography* is the study of means of converting information from its normal, comprehensible forms, into an incomprehensible format, rendering it unreadable without secret knowledge. *Cipher* is an algorithm for putting a message into code by transposition and/or substitution of symbols. *Encryption* is the process of obscuring information to make it unreadable without secret knowledge. The opposite of encryption is *decryption*, which is the process of making information readable again by restoring it to its original form.

Plaintext is the text sent as data, without the encryption applied to it. In other words, it is the input obtained to perform the encryption on. *Ciphertext* is the outcome of the plaintext after encoding it with the predefined encryption key. A desirable property of any encryption algorithm is the avalanche effect. The *Avalanche Effect* consists in the

change in the outcome of at least half of the bits in the ciphertext, when changing one bit in the plaintext. It is evident if, with a slight change in the input, the output changes significantly. If the change were small, it might provide a way to reduce the size of the plaintext or key space to be searched [29].

There are two general approaches to attacking an encryption standard. The first approach is *cryptanalysis* or cryptanalytic attack. Cryptanalytic attacks rely on the nature of the algorithm plus some knowledge of the general characteristics of the plaintext or some sample plaintext-ciphertext pairs. This type of attack exploits the characteristics of the algorithm to attempt to deduce a specific plaintext or the key being used [29]. If the attack succeeds, all future and past messages, with that key, could be compromised. The second approach is a *brute-force attack.* In this case, the attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained. This is a method of defeating a cryptographic scheme by trying a large number of possibilities, for example, exhaustively working through all possible keys in order to decrypt a message. One definition of "breaking" a cryptographic scheme is to find a method faster than a brute force attack.

There are various types of cryptanalytic attacks, based on the amount of information known to the cryptanalyst or attacker. The most difficult problem is presented when all that is available is the ciphertext only. In some cases, not even the encryption algorithm is known. Cryptanalytic attacks require the attacker to rely on an analysis of the ciphertext itself, generally applying various statistical tests to it. To use the previous approach, the opponent must have some general idea of the type of plaintext that

is concealed [29]. The other possible approach is the brute-force approach of trying all possible keys. However, if the key space is very large, this becomes impractical. Being that performing a brute-force attack is the less complex way of attacking an encryption, if this fails, a cryptanalytic attack would have to be performed and this would involve carrying out various statistical tests.

2.3.2 Theoretical Background

Network encryption is a security process that applies cryptographic services at the network transfer layer. Its purpose is to ensure data integrity and authentication. The main reason for the use of encryption is that only authorized users will be able to decrypt the data, as well as have the access to read it and understand it. Thus, encryption is used to make it harder for possible attackers to be able to read the data.

The two basic building blocks of all encryption techniques are substitution and transposition [29]. A substitution technique is the one in which the letters in the plaintext are substituted with other letters, numbers, or symbols. Transposition is achieved by performing some sort of permutation on the plaintext letters [29]. A transposition cipher can be made more secure by performing more than one stage of transposition to the plaintext.

Different encryption standards have been used for the last 30 years. Since the 1970s, different standards have been developed for encryption purposes for different applications. There are encryption algorithms for un-classified but sensitive data, and there are encryption algorithms for classified and extremely important data. At the same

time, each person can develop their own encryption algorithm and share their keys with only trusted sources.

All encryption algorithms before 1976 were known as secret key algorithms, private key algorithms, or symmetric key algorithms. The earliest known use of the substitution cipher, and the simplest, was by Julius Caesar [29]. The Caesar cipher involved replacing each letter of the alphabet with the letter standing three places further down the alphabet. In 1470, Leon Battista Alberti invented a mechanical cipher disk [29]. A class of systems known as rotor machines was the most important application of the principle of multiple stages of encryption before the 1970s. Machines based on the rotor principle were widely used during World War II (WWII). The significance of the rotor machine is that it gave way to the DES algorithm.

Public key algorithms were introduced in 1976 by Whitfield Diffie and Martin Hellman. The first public key algorithm to be accepted as a standard was the Data Encryption Standard. We will go into the detail of the specifics for the DES algorithm in the following sub-section.

2.3.3 Data Encryption Standard (DES)

The Data Encryption Standard (DES) was the first formal standard accepted by the National Institute of Standards and Technology (NIST), back in the year 1977. It has been widely used as the preferred encrypting solution for data integrity. This standard has also been the focus of many research studies. Some of these studies have criticized the standard while some have admired it.

Before its acceptance as a standard, the proposed DES was subjected to intense criticism. Two areas drew critics' attention: First, the key length of the proposed system was only 56 bits long. Critics feared that this key length was too short to withstand brute-force attacks. Second, the design criteria for the internal structure of DES, the s-boxes, were classified. Thus, users could not be sure that the internal structure of DES was free of any hidden weak points that would enable the National Security Agency (NSA) to decipher messages without the benefit of the key [29].

IBM, outside consultants, and NSA cryptanalysts developed the algorithm for this standard, basing its design on IBM's LUCIFER algorithm. The original LUCIFER algorithm operated on blocks of 64 bits, using a key size of 128 bits. The outcome of the DES project was an algorithm with a reduced key size of 56 bits. The main purpose for this reduction was to be able to contain the algorithm on a single chip. The DES was denoted as the Federal Information Processing Standard 46 (FIPS PUB 46). Data are encrypted in 64-bit blocks using a 56-bit key. Figure 2.3 has the design for the DES encryption process. The algorithm transforms a 64-bit input in a series of steps into a 64-bit output. The same steps, with the same key, are used to reverse the encryption [29]. Like other encryption standards, the process for decryption is the same as the process for encryption.

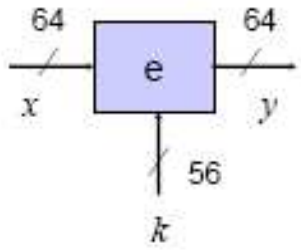

Figure 2.3 DES encryption

The DES algorithm consists in having a plaintext that we want to encrypt and encrypting it with a predefined key to obtain the ciphertext. The formula for encryption is $C = E_{k1}(P)$ where P is the plaintext being encrypted, k_1 is the predefined key, E_k is the DES encryption with key k, and C is the ciphertext.

Although the DES has a small key size, it has taken many years for researchers to be able to have serious attacks towards the encryption. The DES has proven itself over the years. Even though it has been around for more than 30 years, researchers still consider DES when proposing new encryption standards. DES is the block cipher that all new ciphers are compared to when being proposed. Researchers even found a way to "extend" the length of the DES key and increase the security of the algorithm. This was done by adding a second and, in some cases, a third key to the encryption, and performing the encryption three consecutive times. After adding the extra keys, encryption was to be performed using the two or three keys at a time, instead of using only one 56-bit key. Thus, the Triple DES standard was developed. We will go into the details of this standard in the next sub-section.

2.3.4 Triple Data Encryption Standard (3DES)

The Triple Data Encryption Standard (3DES) is one of the interim solutions created to recover the security and avoid attacks. The 3DES uses the same process the DES uses, but it performs the encryption three times with three or two different keys. In this case, making the complexity of the encryption somewhat more difficult to understand or decrypt. Triple DES is a block cipher formed from the DES cipher. Walter Tuchman (the leader of the DES development team at IBM) developed it. It is specified in FIPS Pub 46-3 [25].

Triple DES can be defined using three keys, where the algorithm would consist in $C = E_{k3} (E_{k2} (E_{k1} (P)))$. It is known as 3DES-EEE since it involves performing three DES encryptions one after the other. Figure 2.4 has a representation of the encryption process. It would receive a plaintext of 64 bits, and perform the encryption with the 56-bit key #1. After that, the encrypted plaintext is encrypted with another 56-bit key #2, and then the double-encrypted plaintext is encrypted a third time with another 56-bit key #3. After three encryptions are performed on the plaintext, the ciphertext is obtained.

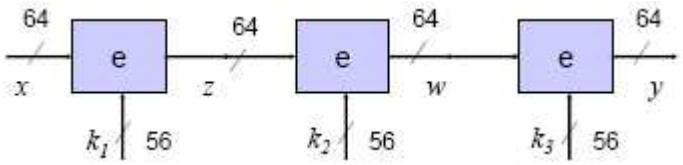

Figure 2.4 Triple DES (case #1 Encryption-Encryption-Encryption)

Triple DES can also be defined using two keys, where the algorithm would consist in $C = E_{k1} (D_{k2} (E_{k1} (P)))$. This is knows as 3DES-EDE, since it involves

performing a DES encryption follows by a DES decryption followed by a DES encryption. Figure 2.5 has the design for the 3DES process. This algorithm consists in performing an encryption on the 64-bit plaintext with 56-bit key #1. This is followed by a decryption of the encrypted plaintext using a different 56-bit key #2. After this is done, the decrypted plaintext is encrypted a second time with the 56-bit key #1, thus, creating the final ciphertext.

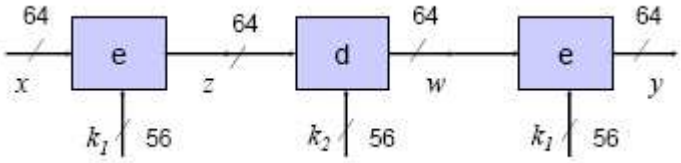

Figure 2.5 Triple DES (case #2 Encryption-Decryption-Encryption)

The most commonly used 3DES encryption is the one where only two keys are used, since 112 bits is adequate for routine commercial applications for the time being and it is backwards compatible with DES. Going to 168 bits would just add the unnecessary overhead of managing and transporting another key for little real gain [31]. Adding a third encryption key makes 3DES have an effective key length of 168 bits.

The 3DES-EEE and 3DES-EDE can be used indifferently and they both will produce an encrypted ciphertext. About the middle decryption, there is no cryptographic significance to the use of decryption for the second stage. Its only advantage is that is allows users of 3DES to decrypt the data encrypted by users of the older single DES [29].

The decryption process for the 3DES algorithm consists in performing the inverse process to the ciphertext, to obtain the plaintext. The following equation describes the process better: $P = D_{k1} (E_{k2} (D_{k1} (C)))$, where P is plaintext, C is ciphertext, k_1 and k_2 are

56-bit DES keys, D_{kn} is DES decryption with key n, and E_{kn} is DES encryption with key n.

The decryption process for 3DES with three keys consists in performing a triple decryption with the same three keys in the inverse order. $P = D_{k1} (D_{k2} (D_{k3} (C)))$ where P is plaintext, C is ciphertext, k_1, k_2, and k_3 are 56-bit DES keys, D_{kn} is DES decryption with key n, and E_{kn} is DES encryption with key n.

Although 3DES has not had any cryptanalytic or encryption attacks performed to it, it was only regarded as an interim solution while a new encryption standard was selected. The main concern for not using it as the default encryption algorithm was the slow execution, since it took a long time to perform DES three times. Its successor was the Advanced Encryption Standard.

.

2.3.5 Advanced Encryption Standard (AES)

The primary motivation for a new standard was the fact that DES has a relatively small 56-bit key that was becoming vulnerable to brute force attacks. In addition, the DES was designed primarily for hardware and it is relatively slow when implemented in software. While Triple-DES avoids the problem of a small key size, it is very slow in software, and unsuitable for limited-resource platforms. Therefore, the search for a new encryption algorithm took place.

On January 1997, the National Institute of Standards and Technology (NIST) called for international cryptographers to propose a new standard block cipher for United

States Government's use in non-classified, but sensitive applications. The main purpose of AES was to replace DES and Triple DES.

One of the requirements of the AES was a block size of 128 bits, and the ability to support key sizes of 128, 192, and 256 bits. The evaluation committee verified that the cipher was secure and that it would not take a long time to execute. In addition, it had to be capable of running in small restricted-space environments such as smart cards, which have limited amounts of RAM and ROM. In 1998, NIST announced the 15 candidate algorithms that were going to be evaluated and analyzed as potential contenders. In 1999, they narrowed the field down to five algorithms that carried the names: MARS, Twofish, Serpent, RC-6, and Rijndael. By October 2, 2000, NIST had selected Rijndael algorithms as the proposed AES. NIST announced the decision and proceeded to make AES the official standard. Finally, on November 26, 2001, NIST announced that AES was approved as an official standard when they published it as FIPS-197 [23].

AES has multiple advantages over DES/3DES. AES adds additional security through the multiple length keys, with a maximum of 256-bit keys supported by this standard [23]. The algorithm selected was the Rijndael algorithm, submitted by Dr. Vincent Rijmen and Dr. Joan Daemen, two Belgian researchers. Rijndael is a block cipher algorithm that encrypts blocks of 128, 192, or 256 bits using symmetric keys of 128, 192, or 256 bits respectively. The number of rounds corresponding to key size is 10, 12, and 14 respectively.

AES encryption involves the process of performing an encryption on a 128-bit plaintext using a 128-bit, 192-bit, or 256-bit encryption key. The encryption consists of

only one plaintext and one key, which will produce a ciphertext as the output. The formula representation for AES encryption can be $C = E_{k1}(P)$, where C is the ciphertext, E_k is an AES encryption with key k, and P is the plaintext. Figure 2.6 shows a better representation of the AES encryption algorithm

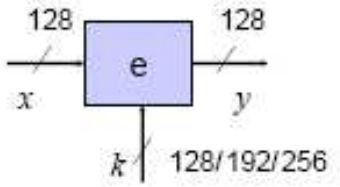

Figure 2.6 AES Encryption

Table 2.1 shows AES parameters for key size, plaintext block, number of rounds, round key size, and expanded key size. It shows the different values for the encryption blocks of 128, 192, and 256 bits on each column respectively. From this table we can see that even though they have different key sizes, number of round and expanded keys, the plaintext block size and the round key size is the same for the three encryption blocks of different sizes.

Key size (words/bytes/bits)	4/16/128	6/24/192	8/32/256
Plaintext block size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Number of rounds	10	12	14
Round key size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Expanded key size (words/bytes)	44/176	52/208	60/240

Table 2.1 AES Parameters [29].

As mentioned before the AES algorithm selected was the Rijndael algorithm. This algorithm consists in performing four transformations on the plaintext in *r* number of

rounds, depending on the key size, to obtain the encrypted ciphertext. The four transformations are Bytes Substitution (ByteSub), Shift Rows (ShiftRow), Multiply Columns (MixColumn), and AddRoundKey. We will explain the four transformations in the subsequent paragraphs. For simplicity reasons we will explain the transformations for a block of 128 bits.

The ByteSub transformation consists of grouping the input into blocks of 16 bytes and arranging them into a 4x4 matrix denoted as "state matrix". Figure 2.7 has an example of the byte order in the state matrix. As shown, the order for the bytes goes top-to-bottom and left-to-right. This is the default order for the matrices shown in this project.

A_0	A_4	A_8	A_{12}
A_1	A_5	A_9	A_{13}
A_2	A_6	A_{10}	A_{14}
A_3	A_7	A_{11}	A_{15}

Figure 2.7 State Matrix

After grouping the bytes into the state matrix, each byte $s_{1,1}$='*xy*' in the matrix is substituted with another byte by looking for the entry in the *x*-row and the *y*-column of the S-box (Substitution Box) table. The new value that is in the (*x*-row, *y*-column) in the S-box, will be assigned to the position where value $s_{1,1}$='*xy*' was placed before. For better understanding, Figure 2.8 represents the byte substitution from the S-box. This figure helps us better understand the concept behind the ByteSub transformation. In other words, the value inside cell $s_{1,1,}$ will be represented as '*xy*'. Then, the ByteSub process

involves obtaining the value from the cell in the S-box corresponding to $s_{x,y}$. Lets assume that the value in the S-box $s_{x,y}$ = '*ab*'. Then the value of $s'_{1,1}$ would be 'ab'. This is the final value in the 'substituted' matrix for the position $s'_{1,1}$.

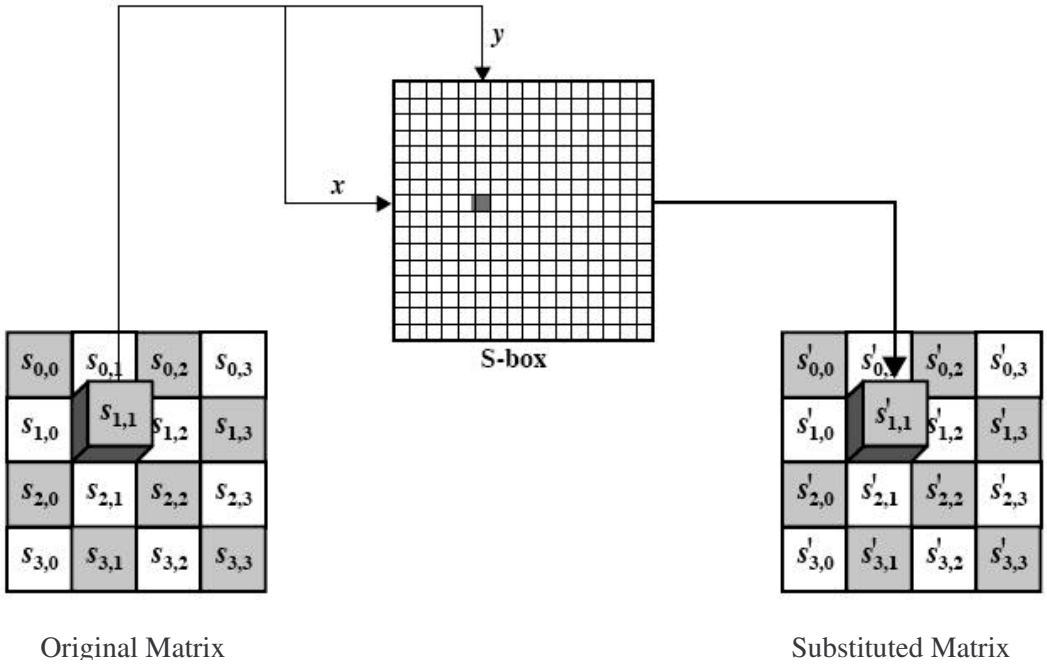

Figure 2.8 S-box byte substitutions [29].

The ShiftRow transformation consists of shifting the rows in the matrix to the left by a different quantity depending on the row. The row 0 in the matrix is not shifted, row 1 is shifted left by one byte, row 2 is shifted left by two bytes, and row 3 is shifted left by three bytes. Figure 2.9 has a representation of the ShiftRow transformation of the matrix. We can see the original order of the matrix and then the final order of the matrix after the ShiftRow transformation has taken place. This transformation takes place in different or various steps.

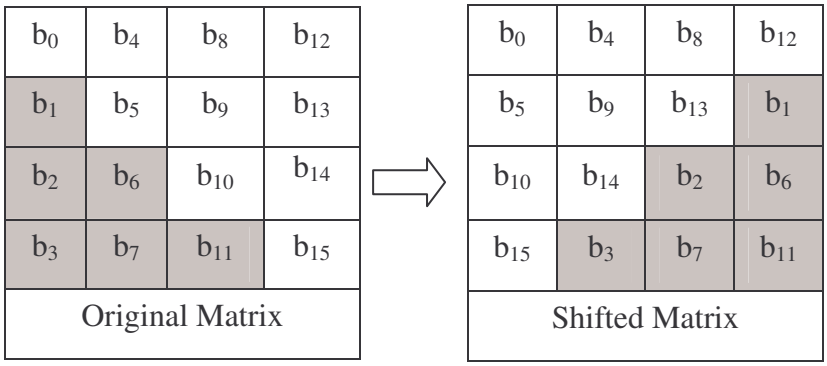

Figure 2.9 ShiftRow Transformation

The third transformation is the MixColumn or Multiply Columns. In this step, the matrix obtained from the last transformation is multiplied by a standard matrix. Figure 2.10 shows a standard matrix. This matrix has the values 1, 2, and 3 placed throughout the matrix in specific positions.

2	3	1	1
1	2	3	1
1	1	2	3
3	1	1	2

Figure 2.10 Standard Matrix

Figure 2.11 illustrates the MixColumn transformation. It shows the input matrix, the standard matrix, and the output matrix of the multiplication of the original matrix by the standard matrix. This transformation operates column by column treating each column as a four-term polynomial [23].

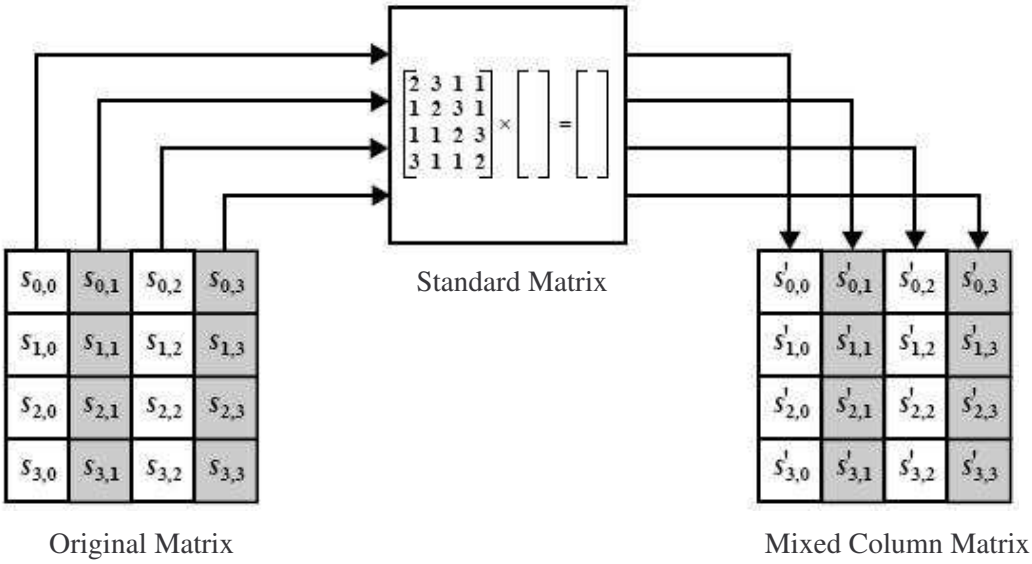

Figure 2.11 MixColumn Transformation [29]

Before performing the AddRoundKey transformation, a process called key expansion takes place, since the expanded key is used in this transformation. The AES performs the key expansion routine to generate a key schedule. The process for obtaining the expanded key involves the input of the 16 bytes key, which generates a new array of 176 bytes from the original key. This process creates many new keys (round keys) with a length of 16 bytes. These keys are used later on in the encryption rounds in the AddRoundKey transformation. It is important to note the fact that each of the round keys are never reused, in other words, they are disposable. By espanding the key into 176 bytes, it creates sufficient keys for the initial round and the following 10 rounds for the 128bits AES encryption.

Each time the AddRoundKey transformation is called, a different part of the expanded key is XORed against the state matrix. The size of the expanded key is different for each key size. For the 128-bit or 16-byte key size, it is defined as

16*(NumberOfRounds+1) = 16(10+1) = 176 bytes. Table 2.1 seen before, has the different key expansion sizes for the different AES key sizes. AES expansion takes as input the 16-byte key and produces a 176-byte linear array. This provides a 16-byte round key for the initial AddRoundKey stage and each of the 10 rounds of the cipher [29].

Finally, the last transformation is the AddRoundKey. In this step, each of the 16 bytes of the original matrix are XORed against each of the 16 bytes of a portion of the expanded key for the current round. Figure 2.12 shows a representation of the AddRoundKey transformation. The w_i in Figure 2.12 refers to the notation in words (4 words = 16 bytes).

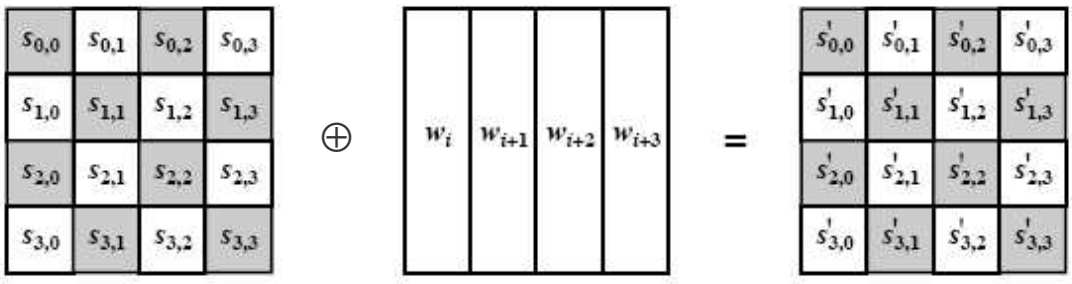

Figure 2.12 AddRoundKey Transformation [29]

To understand the AES encryption algorithm in general, Figure 2.13 shows the flowchart for the Rijndael Algorithm. The Rijndael algorithm receives a plaintext and processes it through different rounds to obtain the ciphertext. The algorithm consists on an initial round (AddRoundKey) and *r* number of rounds (10, 12, or 14). The first *r-1* rounds are similar and consist of four transformations: ByteSub, ShiftRow, MixColumn, and AddRoundKey. During the final round, only three of the four transformations take

place. These are ByteSub, ShiftRow, and AddRoundKey. Finally, after this has taken place, the result is the encrypted ciphertext.

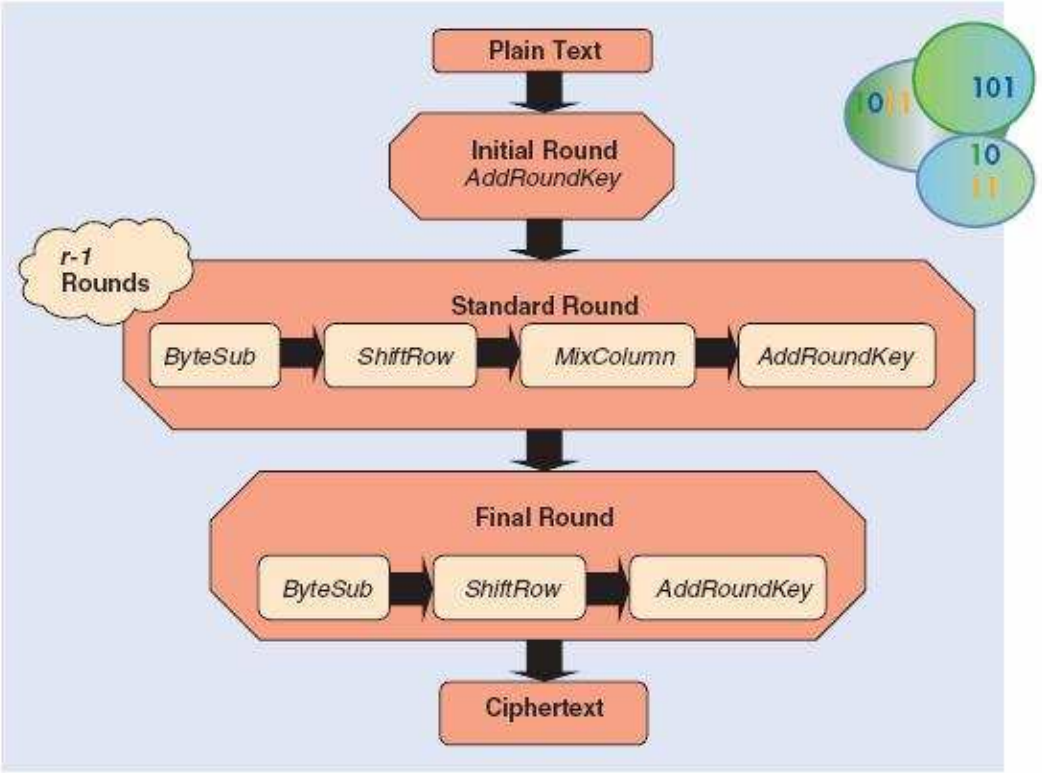

Figure 2.13 Flowchart for Rijndael Algorithm [22].

To decrypt the ciphertext, the procedure followed is the exact opposite of the encryption process. That is, a standard round consists of Inverse Byte Substitution (InvByteSub), Inverse Shift Row (InvShiftRow), Inverse Mix Column (InvMixColumn), and AddRoundKey operations while the final round consists of the same operations excluding the InvMixColumn operation [22]. As with most block ciphers, the decryption algorithm makes use of the expanded key in reverse order [29].

After exploring the AES encryption algorithm, we can see that this algorithm has many improvements over DES and 3DES. In the next section, we will discuss some related work on wireless network security and encryption.

2.4 Related Work

Wireless networks have been the focus of some security concerns regarding the insecurity of its use. For this and other reasons wireless security solutions have been proposed and provided by the IEEE 802.11 working group. The first security standard proposed was the Wired Equivalent Privacy (WEP) protocol. This standard was developed to provide access control, data confidentiality, and data integrity. The purpose for proposing this standard was to simulate the wired network security in a wireless network. WEP was defined as a means of protecting (using a 40-bit key) the confidentiality of data exchanged among authorized users of a WLAN from casual eavesdropping [30]. This standard, however, is vulnerable to various attacks that reveal the shared key used to encrypt, thus leaving the network open to some possible attacks mentioned before. The implementation of WEP is optional. In other words, security is not turned on by default. WEP vulnerabilities arise from design flaws [2, 5, 11, 17]. Some of the vulnerabilities found are the lack of specific mechanism for key management, the lack of mechanism for generating initialization vectors (IV), the fact that it offers no replay protection, and the fact that CRC32 is not a cryptographic integrity checker.

Another method of increasing security in a wireless network is the Media Access Control (MAC) address filtering. By using this, clients are authorized by their unique

MAC address. The MAC addresses of client devices are added to a registry or access list, thus, only registered users have access to the wireless network connection. This method is not used much with ad hoc networks since there is a need for an administrator to authorize the MAC addresses by adding them to a list of addresses. However, it is very popular for the access point connections, since it filters out unregistered users.

To improve security, the IEEE formed the 802.11i task group. This task group proposed a new security architecture known as Robust Security Network (RSN). This architecture offers centralized authentication of users and tones down some of the weaknesses of WEP. It uses a pair-wise key exchange protocol utilizing 802.1x for mutual authentication [1].

The actual protocol used in RSN communications is called Extensive Authentication Protocol (EAP) [19]. There are several types of EAP implementations: EAP-TLS, EAP-TTLS, LEAP, PEAP, and EAP-MD5. These implementations have been made by different companies with different purposes in mind. EAP-TLS is highly secure, since it requires asymmetric public and private keys on the client and sender sides. EAP-TTLS requires a certificate only on the authentication server, making it easy to deploy and almost as secure as EAP-TLS. LEAP is Cisco's version of EAP. It initially worked only with Cisco APs, but now it is being widely supported by other manufacturers. PEAP is the more secure version of LEAP. Finally, EAP-MD5 is the least secure version of EAP, since it does not support dynamic WEP key rotation [19].

Michael's Message Integrity Code (MIC) is proposed to improve the integrity checking of IEEE 802.11 networks [9]. By means of a 64-bit MIC computed with

Michael's algorithm, alterations in the content of the transmitted data can be detected. In other words, this MIC keeps messages from being replayed or modified. This MIC is part of the interim security effort while 802.11i standard is completed.

Another security effort is Wi-Fi Protected Access (WPA). WPA is an interim release pending the final standards recommendation of the 802.11i wireless security task group [10]. It uses 802.1x authentication combined with a Temporary Key Integrity Protocol (TKIP) encryption to make it more secure against attacks. TKIP fixes some of the problems that WEP has, for example, the small initialization vectors and the short encryption keys. TKIP uses the same encryption algorithm as WEP, which is RC4. However, to ensure that encryption keys are not reused, the length of the initialization vectors (IVs) increased to 48-bit (instead of 24-bit which is WEP's actual IVs' length) and the MAC address of the client is used in the key stream generator [9].

The IEEE 802.11i standard is supplementary to the MAC sub-layer to improve security. The purpose is to provide an alternative to WEP with new encryption methods and authentication procedures. IEEE 802.1x forms a key part of 802.11i. The current draft of the IEEE 802.11i addresses numerous security problems and specifies two different AES-based encryption modes, AES-OCB and AES-CCMP [10]. Both of them have been submitted to NIST as new modes of operation. One of these two modes, the Counter with CBC MAC mode (AES-CCMP), has been selected as the "mandatory to implement" mode for IEEE 802.11i [20].

Current studies and researches have focused on the new AES standard. Some of the studies took place while the Rijndael standard was being selected as the AES

encryption algorithm. Other researchers have done studies after the algorithm was selected, as means of trying to find different ways of “breaking” it. Other studies have been done to have better understanding of the process behind the encryption algorithm. One of these studies was done by Dr. Buchholz [21], where he developed a Matlab implementation for the AES standard. Other AES studies have been made for the purpose of the IEEE 802.11i standard [23, 36]. In the next chapter, we use [21] as the starting point and examine the performance of the AES standard. We make recommendations on security enhancement using AES for IEEE 802.11 Wireless LANs. These recommendations are based on the survey performed and rules of thumb. The performed simulations on the recommendations can be seen as starting points for further analyses and studies that would be needed to develop stronger suggestions.

CHAPTER 3

SIMULATION STUDY

We use the Matlab code developed by [21] as the start point to simulate the AES encryption process. After the analysis, we simulate different environments to use AES in and to obtain the result for further analysis of the standard. We then modify the Matlab code in [21], and develop our own Matlab code to simulate a brute force attack with partial knowledge of the key. The simulation can be subdivided into three different parts. The first part takes into account the randomness of the output as well as the avalanche effect produced in the output when changing a small part of the input. The second part takes into account a simulation of a Triple AES encryption by inserting a second and a third encryption key and performing encryption and decryption on the plaintext and ciphertext. Finally, the third part involves performing a brute-force attack to try to obtain the last bit of the encryption key out of a partial key. We will explain the simulations in the next subsections.

3.1 Randomness of output and avalanche effect

The setup, for this part of the experiment, involved modifying the Matlab source code, and altering the input plaintext and the encryption key used. The plaintext for this experiment is a list of 16 bytes, which is represented by 16 hexadecimal numbers,

simulating an actual input on the encryption algorithm. The encryption key is also a list composed of 16 bytes or 16 hexadecimal numbers used to process the plaintext and obtain the ciphertext. Both of these elements are randomly selected as testing input.

Plaintext = 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f

Key = ab 11 ba 22 bc 33 cb 44 cd 55 dc 66 de 77 ed 88

The plaintext is encrypted using the predefined key mentioned before. Figure 3.1 shows the plaintext, on the left, in the matrix form -which can be read top-to-bottom and left-to-right. On the right, we can see the ciphertext obtained from the process of encrypting the plaintext with the predefined encryption key. Here we can see the randomness of the output when comparing the ciphertext to the input. This figure shows the first test run out of ten different test runs made to show the randomness of the output obtained from using the AES encryption algorithm to encrypt the plaintext. All test cases are encrypted using the same key presented above.

00	04	08	0c
01	05	09	0d
02	06	0a	0e
03	07	0b	0f
Plaintext			

ad	80	7d	e3
33	aa	ab	d8
8e	1a	d6	7e
43	05	ce	bd
Ciphertext			

Figure 3.1 Ciphertext obtained from AES encryption (case #1)

Figure 3.2 presents case #2, in which we had the same 16-byte input from case #1, but we shifted all the numbers one position backwards. Even though the inputs are very similar, the ciphertexts created are very different.

01	05	09	0d
02	06	0a	0e
03	07	0b	0f
04	08	0c	00
Plaintext			

0e	70	ea	84
15	4e	16	f1
7c	93	d8	33
17	e7	70	b1
Ciphertext			

Figure 3.2 Ciphertext obtained from AES encryption (case #2)

Figure 3.3 shows a different input, in which the first of the 16 bytes is “11” and all the others are obtained by adding “11” to the previous byte. Notice these are hexadecimal numbers. In this example, we can see that even though each byte of the input possesses the same character, the output obtained has no obvious relation or lacks similarity between the elements.

11	55	99	dd
22	66	aa	ee
33	77	bb	ff
44	88	cc	00
Plaintext			

7e	6b	10	17
af	df	b3	b8
69	38	13	27
e3	09	cf	6b
Ciphertext			

Figure 3.3 Ciphertext obtained from AES encryption (case #3)

In Figure 3.4, the input resembles the one in the previous case, but the input was shifted one position forward for each byte. In this example, we can see that the plaintext and the ciphertext are not alike in any way. At the same time, the ciphertext from this case and the previous case cannot be associated or related in any way.

00	44	88	cc
11	55	99	dd
22	66	aa	ee
33	77	bb	ff
Plaintext			

73	ee	6d	d2
79	b1	6c	3b
77	5a	f7	92
37	7e	97	da
Ciphertext			

Figure 3.4 Ciphertext obtained from AES encryption (case #4)

For Figure 3.5 we selected an input in which each byte had one character from the previous byte. In this case, we can see that the ciphertext obtained from this encryption is not related or similar to the original plaintext or input processed.

01	45	89	cd
12	56	9a	de
23	67	ab	ef
34	78	bc	f0
Plaintext			

f4	4d	61	bc
1a	8f	97	13
29	4d	7d	f3
1b	23	62	3d
Ciphertext			

Figure 3.5 Ciphertext obtained from AES encryption (case #5)

For case #6, we selected all the 16 hex numbers in order. Figure 3.6 shows the hexadecimal numbers ranging from “10” to “1f”. Although this numbers have a one-bit difference from one to the next, the ciphertext output shows a large shift or variety of ranges and values.

10	14	18	1c
11	15	19	1d
12	16	1a	1e
13	17	1b	1f
Plaintext			

46	ca	92	4a
c2	61	88	2b
2c	92	f3	6a
d7	e8	16	82
Ciphertext			

Figure 3.6 Ciphertext obtained from AES encryption (case #6)

Figure 3.7 shows the case #7. This case resembles the previous one in the range of the plaintext input. For this example, we used the range from “a0” to “af”. If we examine the input processed and the output obtained, we find no relation between the two. This is one of the proficient characteristics of the AES encryption algorithm

a0	a4	a8	ac
a1	a5	a9	ad
a2	a6	aa	ae
a3	a7	ab	af
Plaintext			

4e	9c	df	64
05	1b	66	53
47	13	bd	fd
0b	4e	cc	3f
Ciphertext			

Figure 3.7 Ciphertext obtained from AES encryption (case #7)

Figure 3.8 has the plaintext and ciphertext of the test case #8. For this test case, we selected an input that would not be numerically associated and would not be in a specific range. The ciphertext obtained in this case is also very different from the input and at the same time, there is no one-to-one relation between the plaintext and the ciphertext.

40	84	c8	0c
51	95	d9	1d
62	a6	ea	2e
73	b7	fb	3f
Plaintext			

93	65	03	5a
20	e2	00	60
b3	07	68	ff
7f	83	09	dc
Ciphertext			

Figure 3.8 Ciphertext obtained from AES encryption (case #8)

For case #9, we selected the input obtained from the FIPS-197 [23] AES standard publication. From this example, we can be assured that the encryption process is being done correctly, and the ciphertext obtained from this plaintext can be verified from the example presented in the original document. The ciphertext obtained from this plaintext shows no obvious correspondence from the plaintext to the ciphertext.

32	88	31	e0
43	5a	31	37
f6	30	98	07
a8	8d	a2	34
Plaintext			

e6	68	de	a2
dd	6a	53	4a
18	a2	84	3f
ec	75	43	e1
Ciphertext			

Figure 3.9 Ciphertext obtained from AES encryption (case #9)

For our last example, test case #10, we used a different approach. We used the same hexadecimal number or byte for each of the bytes in the plaintext matrix. Figure 3.10 has the plaintext matrix composed of the same 16 hexadecimal numbers. In this example, like in the other test cases, the ciphertext created has no order association to the original plaintext. Even though all the characters in the input are the same, the characters in the output are not closely related to each other.

11	11	11	11
11	11	11	11
11	11	11	11
11	11	11	11
Plaintext			

2f	09	f6	68
fd	10	92	1e
fd	a2	e5	65
76	a5	4a	da
Ciphertext			

Figure 3.10 Ciphertext obtained from AES encryption (case #10)

The next part of the experiment simulates the avalanche effect. As mentioned earlier, the avalanche effect is a property that encryption algorithms aim to have. This property can be seen when changing one bit in the plaintext and then watching the change in the outcome of at least half of the bits in the ciphertext. One purpose for the avalanche effect property is that if by changing only one bit there is a large change then it is harder to perform an analysis of the ciphertext, when trying to come up with an attack.

The first step is to change the plaintext's first bit to "*01*" instead of "*00*", and recording the output of the ciphertext matrix. Just from changing one bit, we can see that the ciphertext's results are very different from the original ones. Here we can see the consequence a simple change can cause, in the result when performing the AES encryption.

Figure 3.11 shows the original plaintext selected to be encrypted with the AES algorithm. As seen before, we can say that the ciphertext created is random, given that the input cannot be directly related to the output obtained.

00	04	08	0c
01	05	09	0d
02	06	0a	0e
03	07	0b	0f
Original Plaintext			

ad	80	7d	e3
33	aa	ab	d8
8e	1a	d6	7e
43	05	ce	bd
Ciphertext			

Figure 3.11 Ciphertext obtained from AES encryption (case #11)

In Figure 3.12, we perform the first "avalanche effect" test scenario. By changing the first element of the plaintext by one bit, the ciphertext obtained has no obvious correspondence. We should try to find similarities between the ciphertexts obtained from case #11 and case #11a. However, no similarities are visible at a glance.

01	04	08	0c
01	05	09	0d
02	06	0a	0e
03	07	0b	0f
Modified Plaintext			

ea	30	c4	3b
46	66	c2	7c
c1	6f	92	38
cb	29	c4	b8
Ciphertext			

Figure 3.12 Avalanche effect seen after AES encryption (case #11a)

Figure 3.13 shows the same encryption performed before, with all of the 16 bytes of the plaintext being of the same value and the ciphertext output being of no obvious correspondence. No obvious relation can be made from the plaintext to the ciphertext obtained. We use this test case as a guide to compare with the two subsequent test cases. In test case #12, we can see that the ciphertext obtained has no relation to the plaintext or even to the encryption key, showed at the beginning of the section.

11	11	11	11
11	11	11	11
11	11	11	11
11	11	11	11
Original Plaintext			

2f	09	f6	68
fd	10	92	1e
fd	a2	e5	65
76	a5	4a	da
Ciphertext			

Figure 3.13 Ciphertext obtained from AES encryption (case #12)

Figure 3.14 contains test case #12a, which shows a change in the first byte of the plaintext from "11" to "10". By changing only one bit of the plaintext (from "10001" to "10000" binary), the changes in the ciphertext are excessive. The avalanche effect property is very important for encryption algorithms, since a "bad" avalanche effect could provide opportunity for possible attackers to develop cryptanalytic attacks.

10	11	11	11
11	11	11	11
11	11	11	11
11	11	11	11
Modified Plaintext			

22	71	77	7c
00	c8	fe	5d
fc	29	1f	ee
4b	3c	03	1b
Ciphertext			

Figure 3.14 Avalanche effect seen after AES encryption (case #12a)

Another example on the avalanche effect can be seen in the results for test case #12b. Figure 3.15 has a small change from test case #12, where we change the first byte of the plaintext from "11" to "12". By comparing the ciphertexts obtained from both test cases, we can see that none of the bytes is similar to each other.

12	11	11	11
11	11	11	11
11	11	11	11
11	11	11	11
Modifies Plaintext			

ff	93	63	9b
4b	88	d2	58
73	9d	ae	c0
6a	33	ac	51
Ciphertext			

Figure 3.15 Avalanche effect seen after AES encryption (case #12b)

The second part of the avalanche effect experiment is to perform a change in the ciphertext and verify the effect on the plaintext. Therefore, we will work backwards in this step (in comparison to the previous one). For this part, we use the original ciphertext obtained from the encryption of the original plaintext at the beginning of the experiment. Then, we modify the ciphertext to change the first two characters of the matrix to "*ff*". Figure 3.16 has a representation of this change. The modified ciphertext is the same as the original ciphertext except for the first hexadecimal character.

ad	80	7d	e3
33	aa	ab	d8
8e	1a	d6	7e
43	05	ce	bd
Original ciphertext			

ff	80	7d	e3
33	aa	ab	d8
8e	1a	d6	7e
43	05	ce	bd
Modified Ciphertext			

Figure 3.16 Ciphertext Avalanche Effect

The procedure here involves a change in the first character of the ciphertext from "*ad*" to "*ff*". The next step involves running the program, and recording the output for the plaintext matrix. Figure 3.17 has the final values of the plaintext obtained from decrypting the ciphertext with the predefined key.

ff	80	7d	e3
33	aa	ab	d8
8e	1a	d6	7e
43	05	ce	bd
Ciphertext			

3e	28	97	1a
78	d5	a6	42
d2	45	39	e1
4a	78	e2	4f
Plaintext			

Figure 3.17 Plaintext Avalanche Effect AES encryption

In the above experiments, we have examined the randomness of the output as well as the avalanche effect property of the AES encryption cipher. In the next section, we will focus on performing a Triple AES encryption. We will try the encryption with one, two, and three different keys.

3.2 Triple AES Encryption

For this part of the project, we examine a Triple AES encryption by performing encryptions and decryptions with the use of one, two, and three different keys. Like in the previous part of the project, the elements in the plaintext and in the keys are 16 bytes or 16 hexadecimal characters. The values for the plaintext, key #1, key #2, and key #3 are presented next. These values are used for all of the simulations in this section.

Plaintext= 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f

Key 1= ab 11 ba 22 bc 33 cb 44 cd 55 dc 66 de 77 ed 88

Key 2= 00 ff 11 ee 22 dd 33 cc 44 bb 55 aa 66 99 77 88

Key 3= a0 a1 a2 a3 a4 a5 a6 a7 a8 a9 aa ab ac ad ae af

The notation used in the following steps will be E for encryption, D for decryption, k_1, k_2, and k_3 for the corresponding keys, and p for the plaintext being processed. Figure 3.18 shows the process of AES encryption, decryption, and encryption. For this test case, we only use one key. Therefore, although we do two encryptions, it only uses the same key for the processes. Therefore, the final ciphertext for this part is equivalent to the ciphertext obtained when doing a normal (single) AES encryption.

$$y_2 = E_{k1} (D_{k1} (E_{k1} (p)))$$

00	04	08	0c
01	05	09	0d
02	06	0a	0e
03	07	0b	0f
Plaintext			

ad	80	7d	e3
33	aa	ab	d8
8e	1a	d6	7e
43	05	ce	bd
Ciphertext			

Figure 3.18 Triple AES one key encryption

The next step for this part is the encryption, decryption, encryption, and decryption, with the same key. Figure 3.19 shows the output of this process. As it turns out, the output is the same plaintext that served as input.

$$x_2 = D_{k1} (y_2) = D_{k1} (E_{k1} (D_{k1} (E_{k1} (p))))$$

ad	80	7d	e3
33	aa	ab	d8
8e	1a	d6	7e
43	05	ce	bd
Ciphertext			

00	04	08	0c
01	05	09	0d
02	06	0a	0e
03	07	0b	0f
Plaintext			

Figure 3.19 Triple AES one key decryption

Figure 3.20 shows the ciphertext obtained from the encryption, decryption, and encryption process with two different keys. The purpose for using two keys for encryption is to increase the complexity level of the performed algorithm.

$$y_1 = E_{k1} (D_{k2} (E_{k1} (p)))$$

00	04	08	0c
01	05	09	0d
02	06	0a	0e
03	07	0b	0f
Plaintext			

6e	c3	2f	15
ec	7e	94	8a
01	6b	de	32
90	61	8f	f4
Ciphertext			

Figure 3.20 Triple AES two key encryption

Figure 3.21 shows the encryption, decryption, encryption, and decryption of the plaintext. In this step, we make the final decryption with key 1, because the last encryption made used that same key. The decryption however does not have as a result a plaintext. Instead, it ends up with a decrypted text or ciphertext of the encryption with key 1 and the decryption with key 2.

$$x_1 = D_{k1} (y_1) = D_{k1} (E_{k1} (D_{k2} (E_{k1} (p))))$$

6e	c3	2f	15
ec	7e	94	8a
01	6b	de	32
90	61	8f	f4
Ciphertext			

f4	1c	81	0a
30	15	cc	70
e7	ed	69	88
54	98	1b	d3
Decrypted text			

Figure 3.21 Triple AES two key decryption

For the third part of the Triple AES experiment, we use three different keys to perform encryption. The process in this stage involves performing an encryption with key 1, a second encryption with key 2, and a third encryption with key 3.

$$y_3 = E_{k3} (E_{k2} (E_{k1} (p)))$$

00	04	08	0c
01	05	09	0d
02	06	0a	0e
03	07	0b	0f
Plaintext			

b4	f1	3c	5e
b8	d1	09	04
e2	09	a5	f1
60	7b	11	f2
Ciphertext			

Figure 3.22 Triple AES three key encryption

After performing the Triple AES encryption, the next step is to perform a Triple AES decryption. To correctly perform this, we have to perform decryption with k3 first, then decryption with k2, and finally decryption with k1. Figure 3.23 shows the input ciphertext obtained from the Triple AES encryption performed in the before process, and then the output plaintext obtained from performing the Triple AES decryption. In this case, we can see that the plaintext obtained is the same one that was encrypted in the previous process.

$$x_3 = D_{k1} (D_{k2} (D_{k3} (y_3)))$$

b4	f1	3c	5e
b8	d1	09	04
e2	09	a5	f1
60	7b	11	f2
Ciphertext			

00	04	08	0c
01	05	09	0d
02	06	0a	0e
03	07	0b	0f
Plaintext			

Figure 3.23 Triple AES three key decryption

Triple AES is a proposed encryption solution to a possible future vulnerability the AES standard might have. However, further studies and experiments are needed to develop serious suggestions or conclusions.

3.3 Brute-force Attack on AES

Finally, this is the last part of the experiment. In this part, we want to simulate a brute-force attack to AES with full knowledge of the plaintext and ciphertext, and partial knowledge of the key, except for the final element. The plaintext used in this part of the brute-force attack is presented next.

Plaintext = 00 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e 0f

In Figure 3.24, we can see the plaintext and the corresponding ciphertext. After having these values, we proceed to try to find the last character in the encryption key. As in the previous sections, the values from the plaintext and ciphertext matrixes can be read in a top-to-bottom, right-to-left sequence.

00	04	08	0c
01	05	09	0d
02	06	0a	0e
03	07	0b	0f
Plaintext			

30	4d	6c	e0
43	0a	cb	44
91	ef	6d	cf
b3	eb	d5	31
Ciphertext			

Figure 3.24 Brute Force Attack Ciphertext (case #1)

At this point, we have 15 out of 16 characters that make up the encryption key. Therefore, we proceed to perform a brute-force (or an exhaustive checking) attack to find the missing character. The partial key with the missing last character is shown next.

Partial Key = 12 23 34 45 56 67 78 89 9a ab bc cd de ef f0 ***?***

The result from this execution of the attack is the missing part of the key. In this case, the missing character is the hexadecimal number “24”. Therefore, for this example, the exhaustive checking attack had to run and check for the corresponding keys from value “00” to “24” (hexadecimal) or from 0 to 36 (decimal) times to obtain the correct key. The time to run the Matlab program to break the last key is approximately 6 minutes. The complete key after performing the brute-force attack is shown next.

Complete key= 12 23 34 45 56 67 78 89 9a ab bc cd de ef f0 ***24***

Case #2 of a brute-force attack can be seen in the next figure. The plaintext used for case #2 is shown next. In this case, since all of the values of the plaintext are the same, it is more difficult to know in which position each of the values is placed.

Plaintext = 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11

Figure 3.25 has the original plaintext and the ciphertext obtained after performing an AES encryption on the plaintext.

11	11	11	11
11	11	11	11
11	11	11	11
11	11	11	11
Plaintext			

2f	09	f6	68
fd	10	92	1e
fd	a2	e5	65
76	a5	4a	da
Ciphertext			

Figure 3.25 Brute Force Attack Ciphertext (case #2)

The partial key for this particular case is the one written below. For this case, like in the previous, we have 15 out of the 16 components of the encryption key. We will perform a brute-force attack to obtain the missing character.

Partial Key = ab 11 ba 22 bc 33 cb 44 cd 55 dc 66 de 77 ed ***?***

The result for this execution was that the attack provided the missing part of the key, which was “88”. Therefore, for this example, the exhaustive checking attack ran approximately 136 times to get this result.

Complete key= ab 11 ba 22 bc 33 cb 44 cd 55 dc 66 de 77 ed ***88***

The time to run the Matlab program to break this last key is approximately 22 minutes.

Although a brute-force attack takes a long time to execute, it is one of the most assertive ways to find the key. The downside of a brute-force attack is associated to the amount of characters that is has to find and the length of the key, for instance in both cases seen before, the key length is 128 bits or 16 bytes. This is for the 128-bit AES

standard. However, for the longer length AES standards, the brute-force attacks are more complicated, and take longer time to break the key.

3.4 Result Discussions

The results obtained from the simulation study are discussed in this section. The AES is a strong encryption algorithm that in the future will become widely used. For the purpose of this project, different analyses were performed to describe the process behind AES. In the results obtained from the first part, the ciphertext obtained from one plaintext showed a difference in most or all of the bytes when compared to the next ciphertext obtained from other plaintext. This could be seen when doing the encryption from the initial plaintext to the final ciphertext. When we performed the randomized output simulation, we tried to find if some obvious correspondence could be seen when going from the encryption of different plaintexts to the ciphertext. However, this was not possible since the output observed no relation or similarity, indicating, to some extent, the efficiency of the encryption algorithm. The avalanche effect property displayed by the different examples of AES encryption on different plaintext demonstrated that AES encryption is an appropriate measure for securing data integrity.

In the second part of the experiment, we introduced the use of Triple AES. We supplied this as a possible solution when performing encryption. Instead of performing the encryption with only one key, a secondary key and/or a third key are used to make the algorithm somewhat more complicated. Although the AES standard has different available key sizes of 128, 192, and 256, using Triple AES as a security measure can be

another security solution. Since Triple AES has a final key length of 128*3=384-bit key, it could work as a better security solution that 256-bit key AES since the Triple AES key would be larger. As seen before, one of the possible problems of Triple AES could be that the execution time takes longer for 3DES than from DES, meaning that there could be a possibility than Triple AES will execute slower than AES. However, Triple AES has not been extensively studied, since it has not been recommended or suggested in the past.

Finally, in the third part we demonstrated that a brute-force attack could take place, given a partial knowledge of the key. However, the knowledge has to be much, to be able to find the missing character in a suitable amount of time. Given the complexity of the AES encryption algorithm by the large key sizes, it is very difficult to perform a brute-force attack towards it. The AES algorithm's default block size is 128 bits, if or when this size comes to be too small, in which case it can become vulnerable to attacks, the larges block sizes of 192 and 256 can become the defaults or the ones used at that moment.

The results obtained from this project can be subdivided into the importance of an encryption tool such as the AES standard, and at the same time, the importance of having good encryption while transferring data through the wireless mediums. If the wireless data being transferred is vulnerable to attacks, it is important that the encryption algorithm used to ensure data integrity be sufficiently secure to withstand serious attacks. In the next section, we present some suggestion on using AES for wireless LANs security.

CHAPTER 4

SUGGESTIONS ON USING AES FOR WIRELESS NETWORK SECURITY

The purpose for this project is to have better understanding of the DES, 3DES, and AES encryption standards and at the same time, develop possible applications for it in the wireless networking environment. Based on the literature the AES encryption algorithm seems like a powerful encryption standard, however further studies and simulations are needed to be able to determine that AES encryption is one of the most secure encryption available at this time. AES is a new standard that can be used for wireless networks. Although it is not the default security protocol used at this time, we suggest its use since it is the selected encryption standard that will become the default standard in the future. It is being added to the 802.11i standard though AES-CCMP protocol. At the same time, it is being regarded as the "mandatory" implementation mode for 802.11i.

AES has been the focus for many studies since it was made standard in 2001. The encryption algorithm has been added to different security solutions. However, it is still regarded as something new. It has been selected as the default encryption algorithm for the future wireless networks as well as the one for wired networks. The 128-bit AES has been suggested for the security of unclassified data. At the same time, the larger key sizes

like 192-bit and 256-bit keys are being taken into consideration for the protection of classified and sensitive data.

The Triple AES encryption is a variation of the AES encryption scheme where AES is applied to the data three times before it is transmitted through the wireless network. Since AES with a single key may not be sufficiently secure for brute-force attacks, it can be made more secure by encrypting multiple times with different keys simultaneously. In Triple AES, the keyed AES function is iterated three times (encryption, decryption, encryption) and generates the ciphertext for the block. Each iteration uses an independent key: k_1, k_2, and k_3. Triple AES usually uses two different keys. First, the data is encrypted with the first key, then decrypted with the second key, and then encrypted with the first key again. To decrypt, the order of the functions is reversed: decrypt with k_3, encrypt with k_2, and decrypt with k_1. When all three keys k_1, k_2, and k_3 are the same, Triple AES is equivalent to AES.

From the study, we know that Triple AES encryption can be taken into account in the future, if attacks on AES encryption are developed. Even though AES has different longer key modes, Triple AES could be used. The fact that 3DES was seen as a possibility and a security solution when attacks on DES started to be successful, leaves us with a possibility that Triple AES could serve as well for the future vulnerabilities the AES standard might encounter.

Wireless Security will continue to evolve, as well as the different attacks performed, however, for the upcoming time AES will be a good enough encryption algorithm that will be able to protect the transfer of data through a wireless medium. With

the 802.11i standard appearing, AES will continue to gain popularity. It will be widely used as the default encryption standard, and although it will require hardware upgrade, it will be widely supported.

CHAPTER 5

CONCLUSIONS

Wireless networks have different uses and security concerns. For these and other reasons, many different available solutions have been proposed and employed. AES is one possible solution to the security threats. Wireless transmissions should use AES or another encryption algorithm as the default algorithm for transmission. One of the biggest security threats is the signal interception from unauthorized users. If there is no known way to avoid signal interception, then a good encryption algorithm is needed to provide security of the transmitted data.

Wireless network security attacks are concerns for people with available wireless connections. New security solution have been implemented and proposed. AES encryption is new, since it was made a standard in 2001. However, it has already been implemented in many ways, and seen as the possible solution to security problems.

AES forms part of the new 802.11i standard, and it was selected as the main encryption algorithm for the future. As part of the 802.11i standard, two main encryptions will be used. These are TKIP, for backwards compatibility with WEP and AES-CCMP, which is based on the AES encryption. One possible problem for the use of AES as the default security algorithm is the need for hardware update that goes with it. Nonetheless,

AES is the most secure of both algorithms. It is recommended that AES be used as the default encryption for WLANs.

Through this project, we provide a survey on the current available encryption algorithms including DES, 3DES, and AES for wireless LANs. We study the detailed performance of the AES algorithm by simulations, including the randomness of output and avalanche effect, and the possibility of brute-force attack on AES. We then present the Triple AES encryption concept, and explore its performance and operations. Triple AES is a promising encryption algorithm but further future scientific studies are needed to be able to provide concrete conclusions. We show examples of the security of the AES standard and at the same time, we provide a starting point for future work. Finding a way to utilize AES in the wireless networks and wireless security is an effective way to provide security and integrity to the wireless data. We also suggest that Triple AES encryption would be a good solution, in case other encryption standards become vulnerable to attacks.

Future work related to this topic could focus on additional developments of the Triple AES for WLANs as well as some new studies and surveys on the security vulnerabilities of the 128-bit AES encryption. Although 128-bit AES encryption has been proven to withstand many known attacks, it has been said this key length is vulnerable to some cryptanalytic attack. From previous literature and rules of thumb, the AES encryption algorithm has good avalanche effect property and at the same time, it can withstand brute-force attacks with partial knowledge of the key. Although a normal brute-force attack takes much longer and takes into account not having knowledge of the

previous key, the results obtained from this project are very interesting for the development of other projects.

BIBLIOGRAPHY

[1] Altunbasak, H. Owen, H. *Alternative Pair-wise Key Exchange Protocols for Robust Security Networks (IEEE 802.11i) in Wireless LANs*. SoutheastCon, 2004. Proceedings. IEEE , 26-29 Mar 2004, 77 – 83.

[2] Arbaugh, W. A. *An Inductive Chosen Plaintext Attack Against WEP and WEP2.* IEEE 802.11 Working Group, Task Group I (Security), 2002.

[3] Arbaugh, W.A., Shankar, N., Wan, Y.C.J. and Zhang, K. *Your 802.11 Wireless Network Has No Clothes.* IEEE Wireless Communications, December 2002, 44-51.

[4] Arbaugh, W.A. *Wireless security is different*. Computer, Volume: 36, Issue: 8, Aug. 2003, 99 – 101.

[5] Borisov, N., Goldberg, I. and Wagner, D. *Intercepting Mobile Communications: The Insecurity of 802.11.* 7th Annual International Conference on Mobile Computing and Networks, Rome, Italy, 2001.

[6] Carli, M., Rosetti, A., and Neri, A. *Integrated security architecture for WLAN*.Telecommunications, ICT, Volume: 2 , 23 Feb.-1 March 2003, 943 – 947.

[7] Candolin, C., and Kari, H.H. *A security architecture for wireless ad hoc networks.* MILCOM 2002. Proceedings , Volume: 2 , 7-10 Oct. 2002,1095 – 1100.

[8] Chlamtac, I., Conti, M., and Liu, J. *Mobile Ah hoc networking: imperatives and challenges.* Ad Hoc Networks I, Elsevier BV, 2003, 13-64.

[9] Erten, Y.M. *A layered security architecture for corporate 802.11 wireless networks*. Wireless Telecommunications Symposium, 2004, 14-15 May 2004,123-128.

[10] Feil, H. *802.11 Wireless Network Policy Recommendation For Usage Within Unclassified Government Networks*. The Aerospace Corporation, 2003, 832-838.

[11] Fluhrer, S., Shamir, A. and Mantin, I. *Weaknesses in the Key Scheduling Algorithm of RC4.* Selected Areas of Cryptography, Toronto, Canada, 2001.

[12] Gupta, V. and Gupta, S. *Securing the wireless internet.* Communications Magazine, IEEE , Volume: 39 , Issue: 12 , Dec. 2001, 68 – 74.

[13] Hao Yang, Haiyun Luo, Fan Ye, Songwu Lu, Lixia Zhang. *Security in mobile ad hoc networks: challenges and solution.*, Wireless Communications, IEEE, Volume: 11 , Issue: 1 , Feb 2004, 38 – 47.

[14] Lee, C.K.L., Xiao-Hu, L., and Yu-Kwong, K.. *A multipath ad hoc routing approach to combat wireless link insecurity.* Communications, 2003. ICC '03. IEEE International Conference, Volume: 1 , 11-15 May 2003, 448 – 452.

[15] Lidong Zhou, Haas, Z.J. *Securing ad hoc networks.* Network, IEEE, Volume: 13, Issue: 6, Nov.-Dec. 1999, 24 – 30.

[16] Potter, B. *Wireless security's future*. Security & Privacy Magazine, IEEE, Volume: 1, Issue: 4, July-Aug. 2003, 68-72.

[17] Walker, J. *Unsafe at Any Key Size: An Analysis of the WEP Encapsulation.* IEEE 802.11 Task Group E, 2000.

[18] Wang Shunman, Tao Ran, Wang Yue, and Zhang Ji. *WLAN and its security problems.* Parallel and Distributed Computing, Applications and Technologies, 2003. PDCAT'2003. Proceedings of the Fourth International Conference on , 27-29 Aug. 2003. 241 – 244.

[19] *802.11 Security Checkpoint*. Light Reading Wireless Oracle, Vol.2, No.3, March 2003.

[20] Burr, W.E. *Selecting the Advanced Encryption Standard.* Security & Privacy Magazine, IEEE. Volume 1, Issue 2, Mar-Apr 2003. 43 – 52.

[21] Buchholz, J. *Matlab Implementation of the Advanced Encryption Standard.* http://buchholz.hs-bremen.de/aes/aes.htm

[22] Jamil, T. *The Rijndael algorithm.* Potentials, IEEE Volume 23, Issue 2, April-May 2004. 36 – 38.

[23] FIPS-197. *Advanced Encryption Standard (AES)*. 2001. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

[24] IEEE P802.11 Task Group I. *Status of Project IEEE 802.11i*. 2004. http://grouper.ieee.org/groups/802/11/Reports/tgi_update.htm

[25] FIPS 46-3. *Data Encryption Standard (DES). 1999.* http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

[26] National Institute of Standards and Technology. *Cryptographic Toolkit Encryption*. 2004. http://csrc.nist.gov/CryptoToolkit/tkencryption.html

[27] Sun Microsystems, Inc. *Crypto-Politics: Decoding the New Encryption Standard*. 2005. http://research.sun.com/features/encryption/

[28] Sanchez-Avila, C., and Sanchez-Reillol, R. *The Rijndael block cipher (AES proposal): a comparison with DES*. Security Technology, 2001 IEEE 35th International Carnahan Conference on 16-19 Oct. 2001. 229 – 234.

[29] Stallings, W. *Cryptography and Network Security: Principles and Practices*. Pearson Education, Inc., NJ, Third Edition, 2003.

[30] The IEEE Computer Society. *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications.* IEEE Std 802.11i. 2004.

[31] Tanenbaum, A.S. *Computer Networks.* Prentice Hall, NJ, Fourth Edition, 2003. 738-741.

[32] Shimonski, R. *Security+ Study Guide and DVD Training System.* Syngress Publishing, MA, First Edition, 2002. 159-219.