NTNU

Norwegian University of
Science and Technology

Cryptanalysis of IEEE 802.11i TKIP.

Ammar Lodhi

Master of Telematics - Communication Networks and Networked Services (2 year)

Submission date: July 2010

Supervisor: Stig Frode Mjølsnes, ITEM

Co-supervisor: Martin Eian, ITEM

Norwegian University of Science and Technology
Department of Telematics

Problem Description

A recently found cryptanalytic attack on the 802.11i Temporal Key Integrity Protocol
(TKIP) exploits the fact That TKIP adds the Message Integrity Code
(Michael) primitive while maintaining the WEP checksum mechanism (ICV).
This backward compatibility property enable the Modified
" chopchop " attack two SUCCEED for frames with Mostly known plain text, Such as ARP and DHCP messages .
The task is two under stand this new vulnerability in TKIP , simply reproduce and
experimentally validate the extended attack .
If Possible, try two extendable comfortable or find new applications for the attack .

Assignment given: 15. February 2010
Supervisor: Stig Frode Mjølsnes, ITEM

Problem Description

A recently found cryptanalytic attack on the 802.11i Temporal Key Integrity Protocol (TKIP) exploits the fact that TKIP adds the Message Integrity Code (Michael) primitive while maintaining the WEP checksum mechanism (ICV). This backward compatibility property enables the modified "chopchop" attack to succeed for frames with mostly known plaintext, such as ARP and DHCP messages. The task is to understand this new vulnerability in TKIP, then reproduce and experimentally validate the extended attack. If possible, try to extend or find new applications for the attack.

Abstract

This thesis is based upon the work of Beck and Tews [24]. It presents and experimentally validates the Beck and Tews attack on a network with QoS client associated with a Non-QoS AP.This is done by slightly extending the source code provided by Beck and Tews. A detailed study of the wireless security protocols has also been done followed by description of how original Beck and Tews attack works.

Martin Beck defines a new approach of obtaining keystreams [1] which has been thoroughly analyzed. A description of how the different packets are used in obtaining more usable keystreams has been given. The experimental validation of how extra keystream bytes are obtained through the new approach [1] has been done. This was done using one of the Network security tools.

Preface

This report is the result of the Master's Thesis in information security written in the final semester of Master's programme in Telematics at the Norwegian University of Science and Technology, NTNU. The assignment was given by Stig F. Mjolnes and Martin Eian at the Department of Telematics.

It has been a great and challenging experience working with this area of wireless security. It really demands a lot. Nevertheless this assignment gave me an insight into the clever techniques used by experts in this field to compromise a network. It has been indeed a valuable learning experience.

I would like to thank my supervisor Martin Eian for his regular feedback throughout the thesis work. He has been very supportive. I would also like to thank professor Stif F. Mjolnes and the Department of Telematics for giving me the opportunity to work in this area of information security.

Ammar Lodhi

Trondheim, July 19th, 2010.

Abbreviations and Acronyms

AA	Authenticator address
ACK	acknowledgment
ANonce	Authenticator nonce
AP	access point
ARP	Address Resolution Protocol
BSS	basic service set
BSSID	basic service set identification
CRC	cyclic redundancy
DA	destination address
DS	distribution system
DSCP	differentiated services code point
EAPOL	Extensible Authentication Protocol over LANs
ESA	extended service area
ESS	extended service set
GMK	group master key
GNonce	group nonce
GTK	group temporal key
IBSS	independent basic service set
ICMP	Internet Control Message Protocol
ICV	integrity check value
IV	initialization vector
KCK	EAPOL-Key confirmation key
KEK	EAPOL-Key encryption key
LAN	local area network
LLC	logical link control
MAC	medium access control
MIC	message integrity code

MPDU	MAC protocol data unit
MSB	most significant bit
PDU	protocol data unit
PMK	pairwise master key
PRNG	pseudo-random number generator
PSK	preshared key
PTK	pairwise transient key
QoS	quality of service
SA	source address
SME	station management entity
SNonce	Supplicant nonce
SPA	Supplicant address
SSID	service set identifier
STA	station
TID	traffic identifier
TKIP	Temporal Key Integrity Protocol
TSC	TKIP sequence counter
TTAK	TKIP-mixed transmit address and key
WLAN	wireless local area network
WEP	wired equivalent privacy

Contents

List of Figures

List of Tables

Chapter 1

1 Introduction

The 802.11 networks are growing at a very fast pace because of its easy and fast deployment. Despite the convenience it offers to the end users the security issues related to it are getting much of a concern. So in order to counter for the security issues Wired Equivalent Privacy (WEP) protocol was designed to provide security to users implementing IEEE 802.11 wireless networks. But WEP was broken because it was not subjected to review after its design [23].Several vendors came up with their own solutions but this could not be the ultimate solution as it led to poor interoperatibilty. Consequently IEEE together with the Wi-Fi alliance developed a new standard the Wi-Fi protected access (WPA).It was based on the IEEE 802.11i standard and was designed to work with legacy hardware to ensure backward compatibility. The Temporal Key Integrity Protocol which is part of WPA was designed to solve the security issues related with WEP.

1.1 Motivation

TKIP was thought of being a secure alternative to WEP but Martin Beck and Erik Tews for the first time were able to break it [24]. This led to more successful attacks using the same basic concept provided by Martin Beck and Erik Tews [16].

My real motivation for this thesis has been to dive deep to know about the different vulnerabilities in the IEEE 802.11 wireless networks. It is because knowing the vulnerabilities is the first step in developing a secure solution or atleast taking the necessary countermeasures to it. The real driving force for the thesis has been to discover more weaknesses in the TKIP protocol. I also had a great motivation to learn the different skill sets related to fully understanding a protocol such as Linux, Number Theory, C programming and the command line tools and other tools like hping3.

1.2 Related Work

There was not much work done until Martin Beck and Erik Tews came up with their attack on TKIP. Their attack was not a key recovery attack [24] but was intended to obtain a keystream for communication between an AP to a station. This work laid a very strong basis for further attacks

on TKIP.Finn Michael and Olav Haugen very effectively and successfully used the proof of concept of Martin Beck and Erik Tews to launch the ARP poisoning attack and cryptographic DoS attack. They also developed an improved attack on TKIP that enabled an attacker to decrypt a much larger DHCP ACK message [16].

In recent times Martin Beck came up with more attacks on TKIP in his paper "Enhanced TKIP Michael attacks", the manuscript of which is not yet published.[1]. In this paper Martin Beck used different packets like TCP and ICMP to obtain more keystreams. These attacks are an enhanced version of the Basic Beck and Tews attack. The attacker has to first execute the basic attack to get the IP address of the client and the Access point. It is important to mention here the another work on TKIP done by Martin Eian in his paper "A Practical Cryptographic Denial of Service Attack Against 802.11i TKIP and CCMP" which describes an denial of service attack against IEEE 802.11 networks using TKIP and CCMP protocols. The manuscript of this paper not yet published at the time of writing this thesis.

1.3 Limitations

Because of the time limitations, this thesis would not focus on the following,

1. Development of proof of concept of the attacks proposed by Martin Beck in his paper "Enhanced TKIP Michael Attacks"
2. Experimentation using different kinds of hardware and firmware.
3. Verification of the extended attack [16] due to the new methodology defined by Martin Beck [1].

1.4 Document Structure:

The document is structured in the following manner:

Chapter 2: This chapter gives an insight into the IEEE 802.11 standard. It gives details about how the integrity of a packet in wireless networks in ensured and the flaws associated with it. It also gives the detail of how the wireless security protocols like WEP and TKIP work. Lastly I have described the packet structure with protocol overview of some of the packets like ARP, TCP and ICMP. The whole content of this chapter is relevant to the work presented later.

Chapter 3: This chapter gives detail about the Basic Beck and Tews attack. In this chapter is also described the usage of the Beck and Tews attack in a wireless network with a QoS client associated with a Non-QoS AP.

Chapter 4: This chapter firstly gives an overview of the extended attack [16] and then gives describes in detail the latest proposed attacks by Martin Beck [1].

Chapter 5: This chapter defines the Laboratory environment used for experimentation.

Chapter 6: The practical experimentation method to verify the attacks is described in this chapter.

Chapter 7: This chapter shows the findings from the experiments.

Chapter 8: In this chapter is discussed the methodology and the problems encountered during the experiments. It also gives an overview of some of the future work which could be done in this regard.

Chapter 9: This chapter presents a summary of the whole document and the findings.

APPENDIX

CHAPTER 2

2 BACKGROUND

This chapter gives an insight into the IEEE 802.11 standard. First I have described the components of the 802.11 architecture. Afterwards the general principles involved in information security have been outlined with more emphasis on integrity of the packet. The CRC-32 algorithm has been described in detail in this context with respect to the mathematical background. This has been done because attack on TKIP could not be understood completely without it as one has to know about polynomial arithmetic and the CRC-32 algorithm uses the polynomial arithmetic. It is followed by how the old WEP protocol works. Understanding it is important because the TKIP protocol which has the vulnerability exploited first by Beck and Tews is based upon the WEP protocol. Then the korek chop chop attack which was an attack on WEP has been described along with its mathematical background as it is the basis of the attack by Beck and Tews on TKIP. Following this is the working of TKIP with respect to how it encapsulates and decapsulates a packet and finally the attack on TKIP with the context of how much plaintext bytes does the ARP, ICMP and TCP packets reveal to the attacker. The ARP packets have been used in the basic Beck and Tews attack while the TCP and ICMP packets are used in the Enhanced TKIP Michael attacks by Martin Beck [1] which is an enhancement to the Beck and Tews attack.

2.1 Components of the IEEE 802.11 Architecture

The components of the IEEE 802.11 Architecture have been described below [2]:

2.1.1 Basic Service Set

The basic building block of an 802.11 network is the basic service set (BSS) which is simply a group of stations that communicate with each other. Each Basic service set (BSS) has stations as its members which could communicate only if they are within a specific area. If a station moves out of its area, it can no longer communicate with other stations present in that area. This area is called as the Basic service area (BSA) .There is two network types of Basic service sets.

Independent Networks

Stations in an Independent Basic Service Set (IBSS) communicate with each other directly. The minimum IEEE 802.11 LAN consists of an IBSS with two stations. IBSS are temporary and are set up for a short period of time. Therefore IBSS are referred to as adhoc BSSs or adhoc networks.

Infrastructure Networks

The infrastructure network is the most common structure of wireless networks. In an infrastructure network the communication between stations takes place through an access point. An access point (AP) is an entity that has station functionality and communicates with the Distribution system (DS). The architectural component used to interconnect the BSS is the Distribution system. The distribution system is responsible for finding where a mobile station is physically located and then delivering frames to the access point serving that mobile station. Data moves between a BSS and the DS via an AP.

Hence the Basic service set in an infrastructure network consists of the AP and the surrounding stations associated with the access point. As all communication takes place through the access point therefore the basic service area in infrastructure BSS is defined by those points in which the transmission from the access point could be received.

2.1.2 Extended Service Sets (ESS)

A basic service set covers a small area and could not provide coverage to large areas .Therefore an extended set is created by linking the Basic service sets (BSS) together. Hence wireless networks of large sizes are formed in 802.11.

2.1.3 Address Fields in IEEE 802.11

A frame in IEEE 802.11 contains four address fields and the addresses are 48 bit long. The four address fields are the following.

Destination Address

The destination address is the MAC address of the final station which sends the frame to the higher protocols for processing.

Source Address

This is 48 bit MAC address of the station which acts as the source of the frame.

Receiver Address
The receiver address is the MAC address of the station that has to process the frame. If the station is a wireless station than the receiver address would be the MAC address of this station and which would be equal to the destination address. On the other hand if the frame is destined for a station on a wired network than the receiver address would be the MAC address of the wireless interface of the access point.

Transmitter Address
The transmitter address is a 48 bit MAC address of the wireless interface of the station which transmitted the frame on the wireless medium [1].

Basic Service Set ID (BSSID)
In an infrastructure mode, the basic service set ID (BSSID) is the MAC address of the wireless interface in an access point. It has the same form as the 48 bit MAC address in wired networks. In addition to the BSSID, there is a field called service set identifier (SSID) in wireless networks. "A Service Set Identifier (SSID) is a sequence of characters unique to a specific network or network segment that's used by the network and all attached devices to identify themselves. It allows devices to connect to the correct network when more than one independent networks are operating in nearby areas" [3].

2.2 General Principles
The general security principles encompassing security in wireless networks revolves around confidentiality which is achieved through encryption of packets, integrity of the packet which is ensures by employing different error detecting techniques through the use of algorithms. Availability of resources on a network is another important aspect of security which needs to be ensured. I will emphasize on how the integrity of packet is ensured by the usage of CRC-32 algorithm which is part of the IEEE 802.11b WEP protocol as well as IEEE 802.11i WPA TKIP.

2.2.1 Integrity
The integrity of packets is ensured through error detection techniques. This enables the receiver of the message to find out if the message has been corrupted. The message is corrupted because of the noise on the wireless channel. To detect an error a checksum is computed by the transmitter of the message and is than appended to the message before sending on the wireless channel. The checksum is the function of the message. On the receiver side the appended

checksum is compared to the checksum calculated by the receiver itself which is done by the same function as used at the transmitter. What if both the checksum and the message itself are corrupted in a way that both are complement each other. Here the design of algorithm comes into play.

Design of Algorithm

The algorithm which is used to calculate the checksum ensures that incase both the checksum and message are corrupted, the corrupted checksum is not consistent with the corrupted message. This is achieved by increasing the amount of information in the checksum. Some detection techniques involve complex transformations on the message to inject it with redundant information. The error detecting algorithms are made complex so that the probability of failure of error detection is low. The reason of having a complex algorithm for computing checksum could be illustrated by the help of the following example. Here we assume that the checksum is computed by simply summing the contents of the message modulo 256, which are just the numbers in decimal system.

An Example

Message: 15 10 5
Message with checksum: 15 10 5 30
Message after transmission: 15 12 5 30

In the example above which employs a very simple error deduction algorithm i.e. sum of the message modulo 256, the transmitted checksum is 30. It could be seen that the message after transmission has its second byte corrupted and changes to 12. The receiver detects it by simply computing a checksum on the received message which turns out to be 32 and then comparing it with the transmitted checksum which is 30.The packet is then discarded but what if multiple bytes are corrupted in the transmitted message as shown below.

Message after transmission: 15 8 7 30

As seen the second byte and third byte in the message have been changed but when the checksum is computed it is the same as the checksum of the transmitted message. Thus the message is accepted. This problem is solved by increasing the width of the register i.e. taking a

16 bit register for checksum instead of an 8 bit register. It means that the checksum is computed by summing up the message bytes modulo 65536.This would reduce the probability of unsuccessful detection of error from 1/256 to 1/65536.So increasing the width of the register decreases the probability of failure. There is another problem i.e. the algorithm is not random because any single byte in the message affects only one byte in the checksum register. To overcome this problem we would use a more complex algorithm that causes each byte to have an effect on the entire checksum register. This is called as increasing the chaos in the checksum algorithm.

Consequently two things are important to increase the strength of the checksum function i.e. Width and Chaos. Now when we know that what kind of properties should an error detecting algorithm should have we go into the background of the CRC algorithm [4].

2.2.2 Background of CRC algorithm

The idea behind CRC algorithm is to take a message and then to change it to a binary number. Afterwards dividing this binary number (dividend) by another fixed binary number and in the end taking the remainder as the checksum. At the receiver side the same division is performed and the resulting remainder (checksum) is compared with the transmitted remainder (checksum).

Cyclic redundancy codes (CRC) are used for detecting if a message is corrupted during its transmission, processing or storage. What is important to know that CRC algorithms treat each message as a bit stream represented as a binary polynomial B(x). It then calculates the remainder R(x) by dividing B(x) with a standard ‘generator’ polynomial G(x). The remainder R(x) is transmitted together with the bit stream associated with B(x).When the message is received at the other end; CRC algorithms verify that R(x) is the correct remainder. The remainder is computed by long division using modulo-2 arithmetic. It is to be noted that the additions and subtractions in module-2 arithmetic are ‘carry-less’ as illustrated in Table 1.

Table 1.1: modulo-2 arithmetic

0+0 = 0-0 = 0
0+1 = 0-1 = 1
1+0 = 1-0 = 1
1+1 = 1-1 = 0

In short the additions and subtractions are equal to the exclusive OR (XOR) logical operation. The CRC algorithm is based on the modulo-2 arithmetic, the only difference being that the message represented as a bit stream is appended with 'w' bit zeroes where 'w' is the width of the polynomial Width of the polynomial is the degree of the highest bit For instance the width of 10111 is 4 because the degree of the highest bit 1 at position 5 is 4.Now that we have described the CRC-algorithm I will show how a number could be shown in polynomial representation. Let the number be 24.

Binary representation of 24 = 10111

Polynomial representation $= 1 \times X^4 + 0 \times X^3 + 1 \times X^2 + 1 \times X^1 + 1 \times X^0$

$= X^4 + X^2 + X^1$

In summary the CRC algorithm works like this:

1. Take the message; find its width 'w' (the actual bit position of the highest 1 bit).
2. Take a polynomial 'G(x)' of width 'w'.
3. Append 'w' zeroes to the end of the message 'B(x)'.
4. Divide 'B(x)' by 'G(x)' using CRC arithmetic. The remainder R(x) is the checksum.[4]

The CRC algorithm is used in IEEE 802.11 to protect data during transmission. Protocols like wired equivalent privacy (WEP) and Temporal Key Integrity Protocol (TKIP) use the CRC-32 function to calculate the Integrity check value (ICV) which finally ensures packet integrity. It is to be noted that CRC-32 refers to the 32 bit cyclic redundancy check value. But the way in which the ICV is computed has a flaw. This is because CRC-32 is linear. This flaw is exploited in the Beck and Tews attack on TKIP.

CRC-32 and Linearity

It is not hard to modify the data of a packet with ICV unable to detect the modification. This is because the CRC-32 function which computes the ICV is linear, the mathematical proof of which is given below,

Proof:

If we represent a plaintext of n bits as a n degree polynomial,

$P = P_n X^n + P_{n-1} X^{n-1} + \ldots\ldots\ldots + P_0 X^0$ Where each $P_i = 0$ or 1.

Thus the plaintext with ICV could be represented as,

$PX^{32} + ICV(P) = P_n X^{n+32} + P_{n-1} X^{n-31} + \ldots\ldots\ldots + P_0 X^{32} + ICV(P)$

The above expression represents the whole plaintext message i.e. Plaintext with n bits and appended zeroes which are equal to the width of the divisor which in this case is the width of the CRC-32 polynomial. If the key stream used to encrypt the above plaintext is n+32 bit long and represented by b, than the encrypted packet could be represented in the following manner,

$$PX^{32} + ICV(P) + b$$

But as mentioned that the ICV is linear which means

$$ICV(P + Q) = ICV(P) + ICV(Q)$$

If we modify the packet by an arbitrary data Q, than the modified packet would be represented as,

$$(P + Q)X^{32} + ICV(P + Q) + b$$

$$= PX^{32} + QX^{32} + ICV(P) + ICV(Q) + b$$

$$= ((PX^{32} + ICV(P)) + ((QX^{32} + ICV\ (Q)) + b$$

From the above expression it can be seen that any packet could be altered in ways that the ICV could not detect that change therefore CRC-32 function which computed ICV does not ensures complete packet integrity[5].

2.3 Wired Equivalent Privacy

Furthermore in order to fully understand the flaw in the Integrity check value which is part of the message, we have to first understand how WEP encrypts a packet.

In the Figure 2-1below is illustrated, how WEP encrypts a packet. When a message is to be transmitted, that message is passed through the CRC-32 function. This results in the four byte or 32 bit Integrity check value which is then appended at the end of the message.

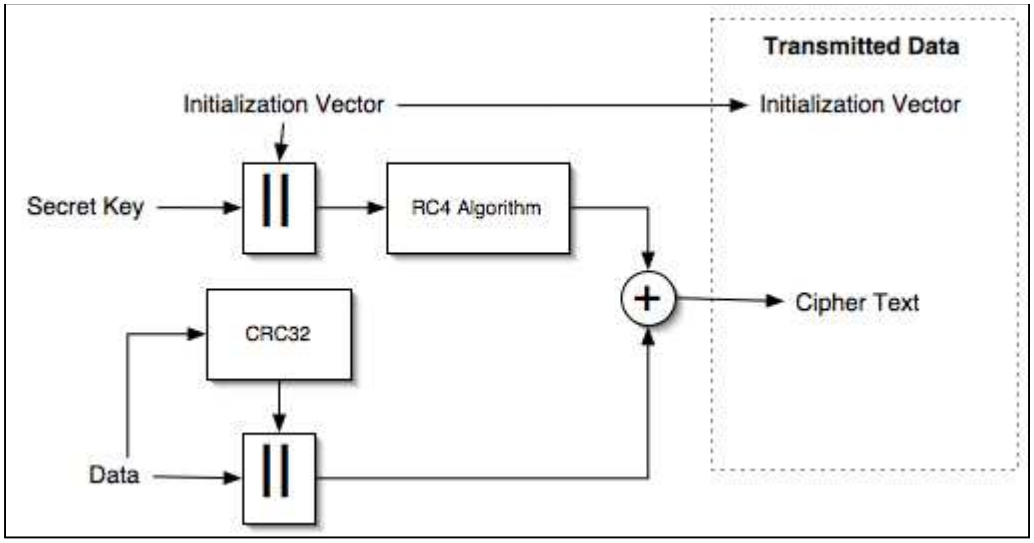

Figure 2-1 WEP Encryption Diagram

The data concatenated with the ICV is then XORed with the pseudorandom string generated by RC-4 algorithm and thus the encrypted data is sent. The inputs to the RC4 algorithm are a 20 bit initialization vector and a secret key. The initialization vector is generated by the transmitting station's wireless card for each message.

Given a plaintext message, WEP adds the ICV to this plaintext message and then XORs this message with the RC4 pseudorandom string or key stream. The encrypted message is represented as shown below,

$$M = \lfloor P + ICV(P) \rfloor \text{ XOR RC4.}$$

If an attacker wants to modify the message he has to change both the message as well as the ICV so that the packet is valid despite modification. Now an attacker intercepts this message 'M' and modifies the message by flipping some bits of the plaintext P. This can be done by taking the bitmask of the modification 'mod' i.e. to take a pattern of bits such that a XOR operation of the bit pattern with the plaintext 'P_{org}' results in a modified message P_{new}as desired by the attacker as shown below,

$$P_{new} = P_{org}\, XOR\, mod.$$

Once this done the ICV of the original message could not be used anymore with the modified message. So the ICV of the modified message is computed by XOR the original ICV with the ICV of the 'mod' i.e. ICV_{mod}as shown below,

$$ICV_{new} = ICV_{org} XOR\ ICV_{mod}$$

Let the whole new modified plaintext message M_{new}. It could then be written in the following way,

$$M_{new} = P_{new} + ICV_{new}$$

$$P_{org} XOR\ mod + \ ICV_{org} XOR\ ICV_{mod}$$

But as the message in a wireless network is encrypted so therefore we have to take the XOR operation of the RC4 key stream with P_{new} together with ICV_{new}. As the packet captured by the attacker is encrypted therefore encrypting it again will cancel the first operation and M_{new}could be written as below.

$$P_{org} XOR\ mod + \ ICV_{org} XOR\ ICV_{mod}$$

$$P_{org} + \ ICV_{org} XOR\ (\ mod + ICV_{mod})$$

$$M_{new} = M_{org} XOR (mod + ICV_{mod})$$

This above mathematical expression shows that an attacker can create a valid packet by modifying an original encrypted packet by XOR operation with a modified data string of the same length as the original packet. This vulnerability of the ICV was exploited by Korek and the attack was called as the Korek Chop Chop attack which is explained in detail below [6].

2.3.1 Korek Chop Chop Attack

A hacker named as Korek described an attack called Chop Chop attack that exploits ICV vulnerability. It starts with an attacker truncating the final byte of an encrypted packet i.e. final byte of the message in the encrypted packet) the previous computed ICV associated with the packet becomes invalid. Thus we have created an invalid message from a valid message. According to Korek there exists a 'value' depending only on the truncated byte which when

XORed with the invalid message allows an attacker to obtain a valid message i.e. message with a correct CRC.

Firstly the chop chop attack sets the truncated value to 0, XOR's the invalid message with a 'value' that makes it valid and sends it to the AP. There are two situations that could occur. The packet could be accepted by the access point and broadcasted. If that happens then an attacker has obtained the plaintext (the last truncated byte of the packet) and the encrypted byte sent by the access point. So the attacker can obtain the key stream by just the XOR operation of the plaintext byte and the ciphertext byte. The process is repeated to get the other bytes of the message fully deciphering it. On the contrary it could be that the packet could be discarded in which case the attack goes on for maximum 255 times until the packet is accepted. Hence if the maximum size of the frame is 1500 bytes than an attacker could send a maximum number of 384000 messages to decipher the whole frame.

The process is repeated to get the other bytes of the message, fully deciphering it. Please note that, in the case of the Korek chop chop attack, we obtain the plaintext without knowing the WEP key. Let us now go into the mathematical background of the Korek chop chop attack.

Mathematical Background

As we already discussed that in CRC computation a sequence of bytes is represented as a polynomial with the binary elements as the coefficients. Moreover the CRC algorithm computation for WEP uses the 32 bit binary number as its polynomial as defined by IEEE standard and represented by,

$$\boldsymbol{G(X) = X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^{8} + X^{7} + X^{5} + X^{4} + X^{2} + X + 1}$$

A message with a CRC is represented by a polynomial in the following equation:

$$P = P_{one} mod\ R_{crc}$$

Where P = message.

P_{one} = polynomial with all its coefficients from X0 to X31 equal to one.

R_{crc} = CRC-32 checksum.

If P is a message with a CRC checksum than P has the correct checksum if and only if

$$P \; mod \; R_{crc} = P_{one}.$$

As in a Korek Chop Chop attack, we truncate a packet by one byte. We can represent the shortened version of the packet with the truncated byte in the following way:

$$P = Q \times X^8 + P_7.$$

Where P is the packet with truncated byte and P_7 is all those components of packet P with exponents smaller than 8 i.e. the last truncated byte. Now the packet with the truncated byte has not a valid checksum. The packet would have to be altered to make it valid. Now $Q \times X^8$ has the correct checksum if and only if:

$$Q \times X^8 = P_{one} + P_7 \; mod \; R_{crc}$$

Since the binary bit pattern is represented as coefficients of a polynomial with elements 0 and 1 and CRC-32 is simply a sequence of bits.Therefore it is represented as coefficients of a polynomial. Moreover the elements zero and one can undergo the two binary operations of addition and subtraction and have the properties of closure, associativity, commutativity, distributivity in addition to the additive and multiplicative inverses. All these properties are possessed by the finite field. Hence CRC-32 is a finite field.

Hence

$X^8 \; x \; X^{8\,-1} = 1$

$$(X^8)^{-1} = X^{31} + X^{29} + X^{27} + X^{24} + X^{23} + X^{22} + X^{20} + X^{17} +$$
$$X^{31} + X^{29} + X^{27} + X^{24} + X^{23} + X^{22} + X^{20} + X^{17} + \quad X^{15}$$
$$+ X^{14} + X^{13} + X^{10} + X^9 + X^7 + X^5 + X^2 + X$$
$$= \quad R_{INV}$$

Where R_{inv} is the multiplicative inverse of the polynomial representation of the truncated byte. The multiplicative inverse in finite fields is found by the extended Euclidean algorithm but we will not go into the mathematics of it here.

Therefore

$$Q = R_{INV}(P_{one} + P_7) mod \; R_{crc} \quad \_\_\_\_\_(a)$$

But For Q to have a correct checksum it should have the following form,

$$Q = P_{one} \; mod \; R_{crc}$$

By adding a value $P_{cor} = P_{one} + P_{INV}\ (P_{one} + P_7)$ to Q in (a), we will get a new message with a corrected CRC – 32 checksum.

This concludes the Korek Korek chop chop attack which was an attack against WEP. After the weaknesses in WEP were found out the IEEE 802.11i standard came which defines the TKIP protocol the details of which are given below [7].

2.4 Temporal Key Integrity Protocol (TKIP)

The TKIP was designed to cover up the weaknesses of WEP. One of the weaknesses which WEP had was the inability to detect a packet captured by an attacker and then replayed. TKIP was able to cover this by the use of a counter called the TKIP sequence counter, the other being the flaw in the CRC-32 computed checksum, ICV.

TKIP Sequence Counter

When WEP was designed, the successful replay of captured packets by attacker was never considered in IEEE 802.11 standards. TKIP was successful in preventing replay attack. It does it by the use of TKIP sequence counter (TSC) and maintains the most recent sequence counter value received from each station. When a frame is received the TSC is checked against the most recently received TSC. If it is larger than the previous value it is accepted otherwise dropped.

TKIP improves the packet integrity by calculating a keyed cryptographic message integrity code the MIC. In WEP ICV there is a weakness which is exploited by an attacker in a way that even with modification of the packet the ICV still remains valid as already discussed.

MIC

After the design of Mickey and Michelle, Neil Ferguson came up with the MIC (Message Integrity Code) which ensures packet integrity and also detects any kind of Forgery attempts against the packet. MIC runs on legacy hardware without imposing performance degradation of resources but it provides a security level of only 20 bits. Hence MIC is a tradeoff between resource utilization and security.

Working of MIC

MIC algorithm makes use of a secret key which is a 64 bit key and is represented as a sequence of 8 bytes i.e. K0……K7.This sequence of 8 bytes is then converted to two 32 bit words i.e. K0 and K1.These words are little endian which means that the least significant bit is stored first in

the computer memory as opposed to big endian in which most significant bit is stored first. It is noteworthy that all the conversions in the MIC algorithm between the bytes and 32 bit words is little endian because MIC would be running on little endian CPU'S i.e. most of the access points manufactured nowadays use Intel chips in which data is stored in little endian.

MIC is computed on the source address, destination address and priority fields of the wireless frame as well as the data field. In addition to this, a single stop byte (value 0X5a) is added to the frame. But MIC can only be computed when the frame is a multiple of four bytes. Therefore padding of 4 to 7 zero bytes is done at the end of the frame. After the padding the frame is converted into a sequence of 32 bit words.

The input to the Michael algorithm is then the two 32 bit words computed from the secret key of 64 bits represented initially as an 8 byte sequence and the padded frame consisting of the source address, destination address, priority field, data byte, single byte value (0X5a) and padding of zero bytes. The algorithm runs a total of N times (i includes 0 to N-1) where N is the number of 32 bit words making up the padded frame. Consequently two words denoted as V0 and V1 are computed which are converted into a sequence of eight little-endian octets, the MIC value.

Input: Key (K_0 and K_1) and padded frame of 32 bit words.

Output: MIC value (V_0 and V_1).

MIC <= ((k_0, k_1), (M_0, M_{n-1})

(l, r) <= (K_0 , K_1))

For i = 0 to N-1 do

l <= l ^= M_i

(l, r) <= b (l, r)

Return (l, r)

In the above algorithm we could see a block function b which runs over the 32 bit words. The block function 'b' used by the Michael algorithm is Fiestel algorithm and the operations taken by 'b' are below,

Input: (l, r)

Output: (l, r)

r <= r ^= (l <<< 17)

l <= (l + r) mod 2^{32}

r <= r ^ = XSWAP (1)

l <= (l + r) mod 2^{32}

r <= r ^ = (l <<< 3)

l <= (l +r) mod 2^{32}

r <= r ^= (l >>>2)

l <= (l + r) mod 2^{32}

Return (l, r)

As seen above the block function uses alternate additions and XORing. <<< denotes left rotation and >>> right rotation of 32 bit words. The block function b also uses the function XSWAP that exchanges the position of the two least significant bytes with the position of the two most significant bytes in a word.

The Michael algorithm is not a very strong algorithm owing to the fact that an attacker has one chance in a million of modifying a packet with the MIC still being valid. In order to prevent an attacker from modifying a packet the old WEP ICV (CRC -32) is used with the MIC. Therefore these attackers are difficult to success. A very strong countermeasure against packet forgery attacks is prevented by the MIC countermeasures.

Now that we have described in detail the working of MIC we would go into how TKIP has been exploited by attackers starting from the Brute force attacks on TKIP and then the ARP, ICMP and TCP packet structures and protocol overview which reveal the plaintext bytes to the attacker and are the basis of successful attacks on TKIP.

2.4.1 TKIP Data Processing and Operation:

The frame processing by the TKIP provides support for data encryption as well as ensures packet integrity. But how all this is done is shown in the Figure 2-2 below,

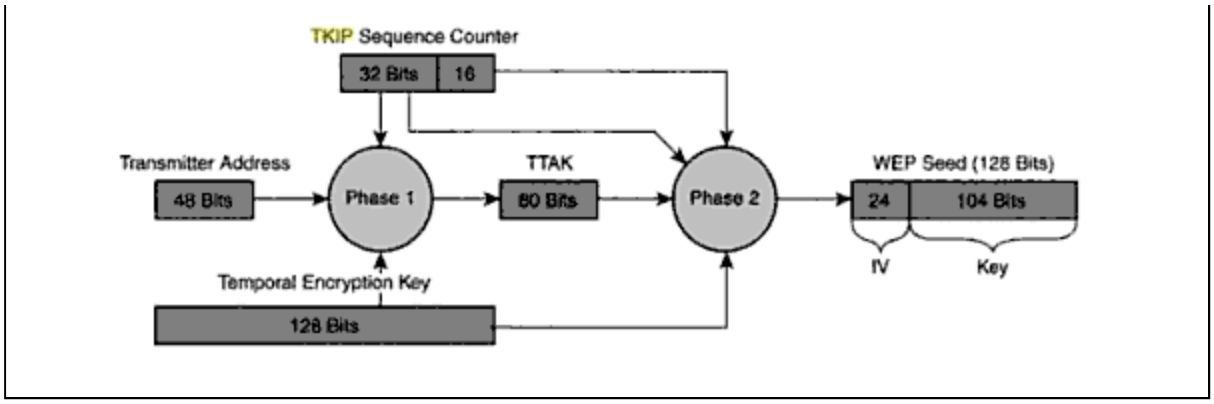

Figure 2-2 Key Mixing TKIP[8]

Phase I key Mixing:

Phase I in data processing by TKIP computes TKIP-mixed transmit address and key (TTAK) of 80 bits. The key is derived from the address of the transmitter of the frame, high-order 32 bits of TKIP sequence counter (TSC) and the 128 bit temporal key.

Purpose: The purpose of Phase I Key mixing is to ensure that the key for one frame is different from the other. For instance when two stations are using the same TSC values they would have different RC4 keys because of Key mixing.

Note: It is noteworthy that whenever the TSC space is exhausted, the temporal key is changed otherwise the traffic sent could be compromised [9].

In brief the inputs to the Phase I key mixing function are the temporal key, TSC and transmitter address and the output is the TTAK and this is done so that the key used from one frame to the other varies significantly. It is to be noted that the only input which changes in Phase I between packets is the TSC value but as it only uses the high-order 32 bits therefore it changes only after 64K or 2^{16}packets.

Phase II Key Mixing:

The introduction of the Phase II key mixing to calculate the per packet key is done in order to account for the low processing power of the 802.11 controllers. Moreover this makes easy to pre-compute the per packet key which must be done because the initialization vector initialization vector increases monotonically. The inputs to the Phase II key mixing function is the 80 bit TTAK, the 128 bit temporal key and the lower 16 bits of the TSC. The output of this phase is the 124 bit encryption key which will be used as an RC4 key just like in traditional WEP.

TKIP Encapsulation

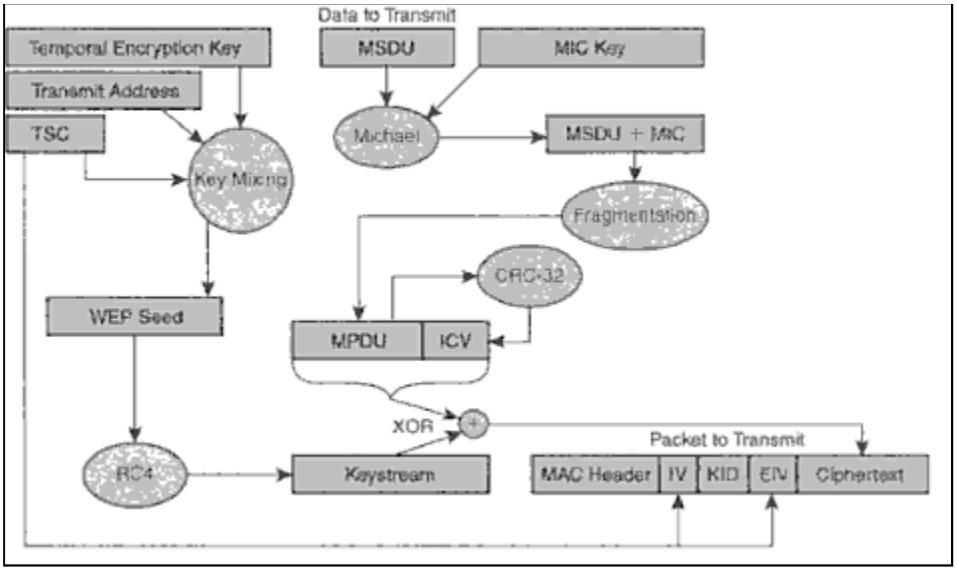

Figure 2-3 TKIP Encapsulation Block Diagram [10]

The Figure 2-3 illustrates the whole process of TKIP encapsulation. TKIP encapsulation is the process of first computing the 8 octet MIC on the MSDU by using the Michael algorithm and the MIC key. If the MSDU together with the MIC is large enough then the MSDU is fragmented into MPDU's and the ICV is calculated by the use of the CRC-32 function. Now the 128 bit per packet key which comprises of IV and the RC4 key is used to encrypt the MPDU.RC4 algorithm is used to calculate the pseudorandom key stream. Finally the MPDU is transmitted as the ciphertext. The 802.11 only encrypts the payload and the MAC header is sent in clear. The TKIP encrypts all the MPDUs generated from one MSDU under the same temporal key.

It is to be noted here that in addition to the ciphertext and the MAC header the MPDU is extended by four octets to accommodate an extension to the WEP IV which is denoted by EIV field in the figure. TSC2 to TSC5 which are the high order 32 bits of the TKIP sequence counter value make up the extended 4 field. In addition to it there is ExtIV bit in the KID field (1 octet). The ExtIV bit is set to one which indicates the presence of the EIV field. Hence it is set to zero for WEP encrypted packets. 5 bits of the KID field are reserved and the remaining two bits represent the Key ID which shall be set to the key index for the key used in cryptographic encapsulation of the frame. Moreover there is the original IV (3 octets) as in WEP. In the IV

field octet one and octet three are the TSC1 and TSC0 respectively while in position 2 is the special octet (TSC1|0x20) & 0x7f [9].

After the packet is encapsulated by TKIP it is sent to the receiver where it is decapsulated.

TKIP Decapsulation

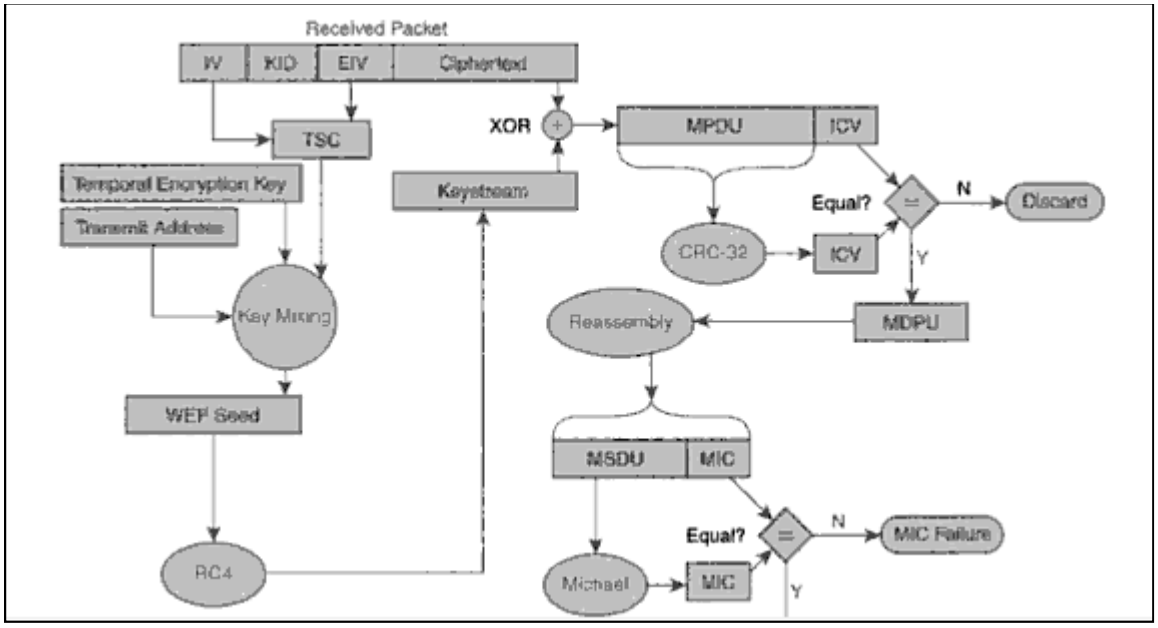

Figure 2-4 TKIP Decapsulation Block Diagram [10]

TKIP decapsulation is the process of recovering the plaintext from the ciphertext MPDU and in the process ensuring the integrity of the packet. When the frame is received by the wireless interface it passes the frame check sequence (4 octets) to ensure that it has not been corrupted in transmission. Afterwards it starts with the extraction of the TKIP sequence counter value from the Extended IV (EIV) and the WEP IV (IV) which is then used to recover the WEP seed used for encryption at the transmitter. When the WEP seed is recovered it is used to get the key stream from the RC4 algorithm. The key stream is XORed with the ciphertext MPDU to get the plaintext MPDU. After this is done the WEP ICV is checked to ensure integrity. If the ICV is incorrect the packet is discarded otherwise accepted. If fragmentation has been done at the transmitter then decapsulation is not done until all the fragments have arrived so that the complete payload with MIC is reassembled. Once the frame is reassembled MIC is calculated over the entire payload. If the computed MIC matches the MIC on the packet, the frame is

passed to higher layers and the sequence counter value is set to the value of the received frame. If calculated MIC is not equal to the MIC on the packet MIC countermeasures are triggered.

2.4.2 Key Hierarchy:

The complete working of TKIP could not be understood without knowing the key hierarchy. Key hierarchy generates the keys which are used for data encryption as well as data origin authentication during the four way handshake. Key management is an important part of secure wireless communication. In IEEE 802.11i standard, 4-way handshake protocol is designed to exchange key materials and generate a fresh pairwise key for subsequent data transmissions between the mobile supplicant and the authenticator.

Pairwise Key Hierarchy

The pairwise key Hierarchy with respect to the TKIP refers to the derivation of the 512 bits Pairwise Transient Key (PTK) from the 256 bit Pairwise Master Key (PMK) as shown in the figure 2.5. The 256 bit PMK is expanded into the 256 bit PTK by using a PRNG called the pseudo random function (PRF-512) which takes other inputs in addition to the PMK i.e. the smallest numerical value and the largest one amongst the Authenticator address and the Supplicant address i.e. Min(AA and SPA) and Max (AA, SPA) as well as the smallest of the authenticator Nonce and supplicant Nonce i.e. Min (ANonce, SNonce) and Max (ANonce, SNonce).The Min and Max operations for IEEE 802.11 addresses are the conversion of these addresses into positive integers and then to output the smallest or the largest value. From the PTK which is 512 bits long are derived three keys.

EAPOL Key confirmation Key (KCK) which is used to provide data origin authentication in the four way handshake and the Group Key Handshake messages. Thus this key is used to prove the possession of the Master Key.

EAPOL-Key Encryption Key (KEK) is used to provide data confidentiality in the 4 way Handshake and Group Key Handshake messages by encrypting the EAPOL messages.

Lastly there is 256 bit Temporal Key which is computed as bits 256-383 (CCMP) or bits 256-511 (for TKIP) of the PTK which could be represented in the following manner as shown in the figure[9].

TK←L (PTK, 256,128)

TK←L (PTK, 256, 256)

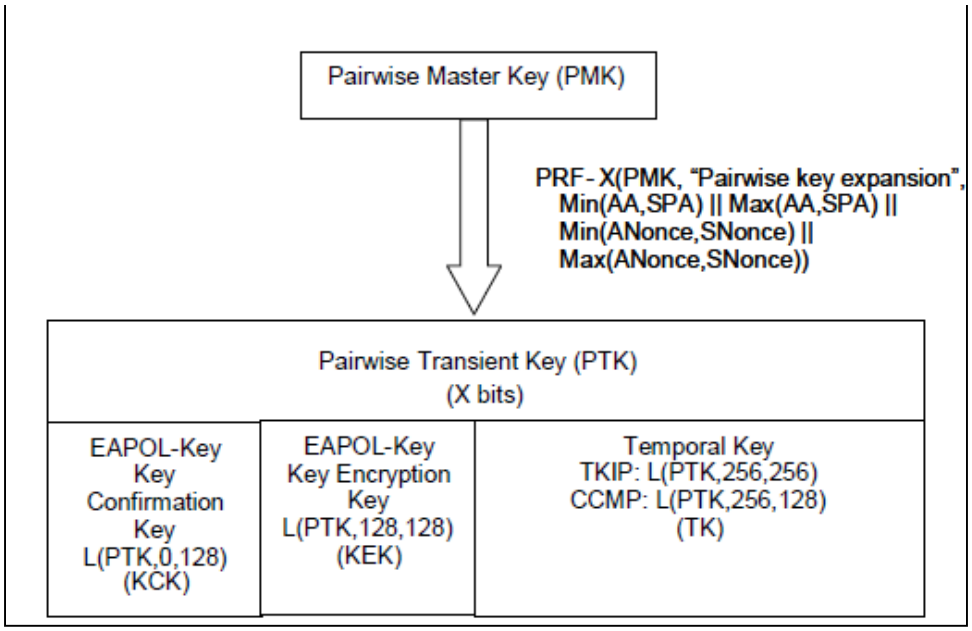

Figure 2-5 Pairwise Key Hierarchy Block Diagram [9]

Group Key Hierarchy

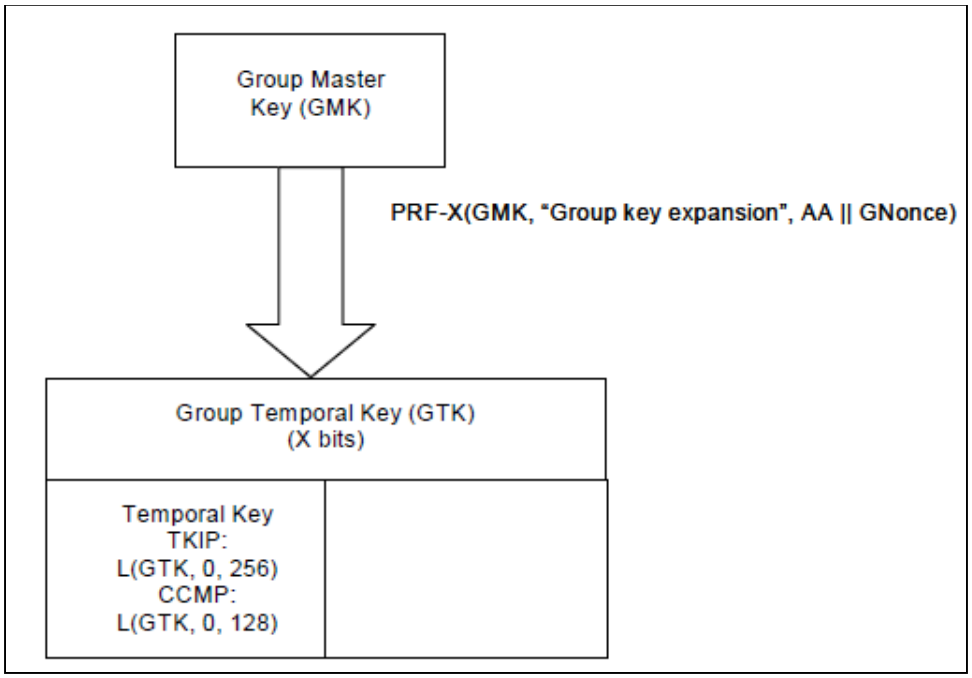

Figure 2-6 Group Key Hierarchy Block Diagram [9]

The Group Key Hierarchy is used to derive the keys which protect the multicast and broadcast transmissions. Here the authenticator maintains a group master key (GMK) which is 128 bits from which the Group Temporal Key (256 bits) is derived. The whole process is represented as following:

GTK<- PRF-256(GMK," Group Key Expansion"||AA||GNonce)

Where X= 256 for TKIP in the Fig and X=128 for CCMP.

From the Group Temporal Key (GTK) is derived the temporal key (TK) which is bit 0-255 of the GTK and represented as following.

TK<-L (GTK, 0,256).

After going into the details of the Key Hierarchy I would now give an insight into what the MIC failure events for an Authenticator and a supplicant.

2.4.3 TKIP Countermeasures:

When a frame is received at the receiver, it will check the FCS, ICV and TSC for all related MPDU's. If an MPDU has an invalid FCS or an incorrect ICV or TSC i.e. a TSC value that is less than or equal to the TSC value of the most recent MPDU, the MPDU's are discarded. Afterwards the MPDU's are assembled into a MSDU and the MIC is checked. If a MIC failure is detected in the received MSDU, it indicates an active attack. When this happens TKIP takes some countermeasures. All the stations and the STAs and APs when detect MIC failure would do the following:

1. A MIC failure event is noted and logged.
2. If the APs and stations detect two MIC failures within 60 s all TKIP communication is disabled for 60 seconds. This slowdown makes it difficult for an attacker to launch an attack quickly.
3. The AP and the stations delete their copies of the master keys. Here it is to be noted that the Pairwise Transient Key (PTK) is changed in case of a supplicant and the Group Transient Key (GTK) in case of an authenticator. The Authenticator and supplicant are usually the STAs and the APs. Furthermore the pairwise keys are used for the protection

of traffic between a station and the AP while Group keys are used to protect broadcast/multicast traffic from the AP to the STAs which are associated with it.

TKIP Countermeasures for an Authenticator:

The MIC failure event in case of an authenticator means the detection of a MIC failure on a received unicast frame from a supplicant i.e. AP or it could also mean the reception of the MIC failure report frame. If the authenticator receives a unicast frame with a MIC error it does the following.

1. It discards the frame.
2. A timer or MIC failure counter is used to log the MIC failure events and incremented each time when a MIC failure event occurs.

In case the authenticator receives the MIC failure report frame from the supplicant it does the following and which is illustrated in Figure 2.7 below

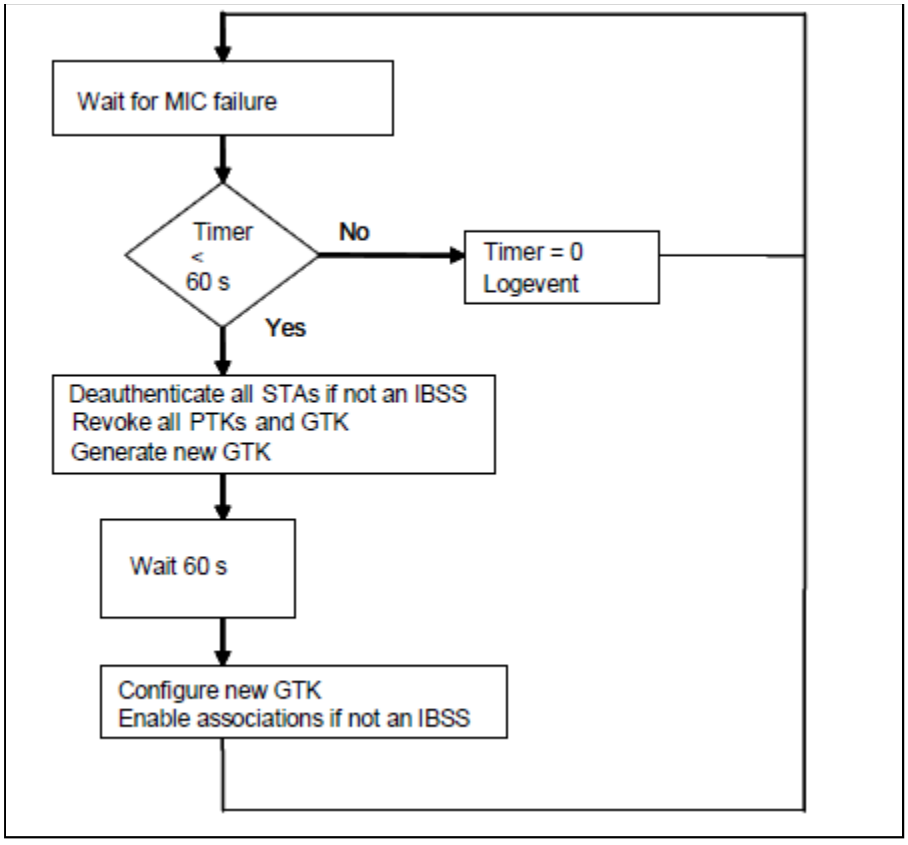

Figure 2-7 TKIP Countermeasures Authenticator [9]

As shown in the figure If the MIC is the first failure within the last 60 seconds the countermeasure timer is initialized but if more than one MIC failure event takes place within the last 60 seconds then the authenticator will deauthenticate all STAs associated with it and will delete all the PTKs and GTK. At this moment a new GTK is generated but it is not used for 60 seconds. After one minute the MIC failure counter is reset and the new PTK configured [8].

TKIP Countermeasures for a Supplicant

In case of a supplicant station which receives a frame with a MIC error it discards the frame and increments the MIC failure counter. It also generates the MLME-MICHAELMICFAILURE.indication primitive .The supplicant on receiving a MLME-MICHAELMICFAILURE.indication primitive from the station, sends a Michael MIC failure report to the AP. If in less than 60 seconds more than one MIC failure occurs the supplicant deletes the PTK and the GTK and deauthenticates the station from the authenticator .After 60 seconds it reassociates with the same authenticator [9].

2.5 Attacks on TKIP

Before the Beck and Tews attack published in November 2008 the only successful attacks against the TKIP were the Brute Force attacks. Robert Moskowitz in 2003 was able to successful do this kind of attack against TKIP and was known as the cracking of WPA-PSK.

The PSK or preshared Key is a 256 bit number or a passphrase which is 8 to 63 bytes long.The PSK is converted to the 256 bit PMK that is used to derive the PTK used in the 4 way handshake. When the PSK is a passphrase then the PMK is derived from the PMK in the following manner.

PMK = PBKDF2 (passphrase, SSID, SSIDlength, 4096, 256)

Where PBKDF2 is a method from PKCS#5 v 2.0. and PKCS is a password based cryptography standard. The method is applied on the concatenated string of passphrase, SSID and SSIDlength and then hashed 4096 times to generate the 256 bits PMK.

The PTK is derived from the PMK and the two MAC addresses and 2 nonces used in the four way handshake so an attacker in possession of the PSK and the 4 way EAPOL handshake is able to know all about the key Hierarchy. The capturing of 4 way handshake is done by simply

deauthenticating the station and then sniffing the wireless traffic. When the four way handshake is captured then the remaining attack is done offline. The capture file is taken as input to the office dictionary attack which is done using aircrack-ng.When the attacker guesses the password by use of a dictionary a PMK is calculated from the password and then tested with the four way handshake to see if the guess was correct.

The 802.11i standard points out that:
"A passphrase typically has about 2.5 bits of security per character, so the passphrase of n bytes equates to a key with about 2.5n + 12 bits of security. Hence, it provides a relatively low level of security, with keys generated from short passwords subject to dictionary attack. A key generated from a passphrase of less than about 20 characters is unlikely to deter attacks" [11].

Consequently a passphrase of shorter length is very easily subjected to dictionary attacks. Brute force attacks were the first of its kind on TKIP but it was not until the IEEE 802.11e amendment which later laid the basis for some of the other attacks on TKIP. The amendment which was done is described below.

2.5.1 IEEE 802.11e-QOS/WMM Amendment

"This amendment defines the medium access control (MAC) procedures to support Local Area Network (LAN) applications with quality of service (QoS) requirements. The procedures include the transport of voice, audio, and video over IEEE 802.11 wireless LANs (WLANs)" [11].

Before the IEEE 802.11e amendment the MSDU was transported on best effort basis. However in IEEE 802.11e standard the QOS facility was defined which uses a traffic identifier (TID) to specify differentiated services on a per MSDU basis.

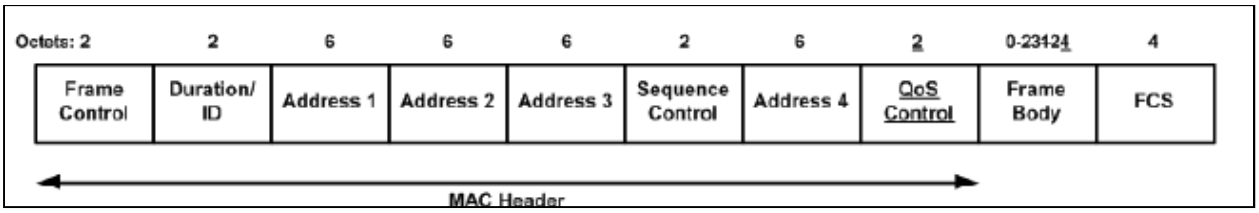

Figure 2-8 IEEE 802.11 MAC Frame Format [12]

In the above figure 2.8, a general MAC format is shown. In the IEEE 802.11e amendment, the QOS control field of two octets was inserted and all QOS data frames contain the QOS control information field. The first two octets in the above frame belong to the frame control field, the first two bits of which are the protocol version field. The next two bit is the Type field (Data, Management and Control) and the next 4 bits is the Subtype field. If the bit b7 of the subtype field is set to 1 then that frame is a QOS data frame when the bit 'b3b2' is set to 10 i.e. data frame in the Type field. The subtype field was added in IEEE 802.11e amendment. In the above frame format the QOS control field which is part of every QOS data frame and indicates the traffic class to which the frame belongs. It also shows other QOS information that varies with each frame type and subtype. It has five sub fields but we will only discuss that subfield that is relevant to the attack on TKIP i.e. Traffic Identifier or TID field which is four bits and therefore has sixteen possible values but these 16 values of TID are not yet supported in implementations. The attack on TKIP makes use of this field such that TID 0 through TID 7 correspond to eight different channels having different QOS specifications and hence used to prioritize traffic. TID 0 channel is having the lowest priority while the TID 7 has the highest priority.

Wifi Multimedia or WMM is the subset and implementation of the IEEE 802.11e standard. In Wifi networks having WMM functionality is meant that both the Access point (AP) and clients can run applications that require QOS. For Multimedia applications AP and clients should have WMM enabled in it. WMM defines four access categories or channels which have different QOS requirements and hence support different type of traffic. The four channels are Voice having the highest priority with TID 6 and 7 followed by video with TID 4 and 5, Best effort (devices that don't require QOS support) with TID 0 and 3 and WMM background i.e. low priority traffic (file download and print jobs) having TID 1 and 2 [12].

I would explain how Beck and Tews used the IEEE 802.11e –QOS/WMM to their advantage in order to mount an attack on the TKIP using ARP in the next chapter but to understand it completely the ARP protocol overview is necessary.

2.6 Address Resolution Protocol (ARP)

An understanding of the ARP Protocol is important because the Beck and Tews attack on TKIP uses the ARP packets in its attack.

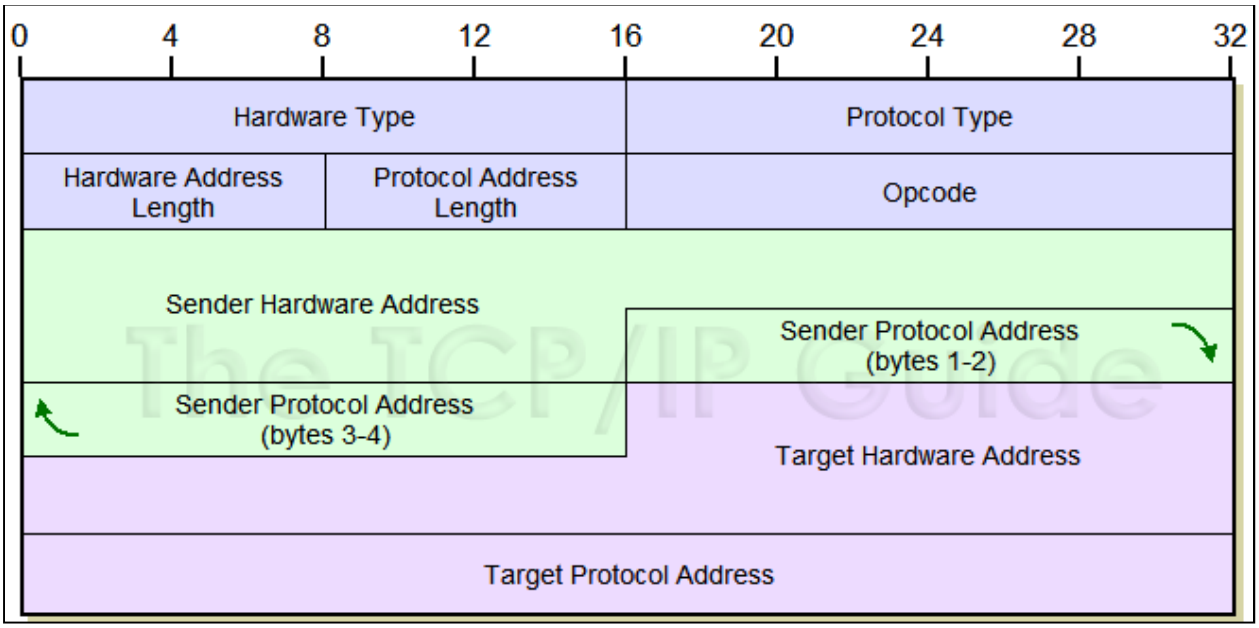

Figure 2-9 ARP Packet Structure [13]

The ARP packet structure is shown in the Fig 2.9. It is a small packet which is 28 bytes long. It accommodates the layer 2 and layer 3 addresses i.e. MAC and IP addresses. The “Hardware type” field specifies the type of hardware used for the network which is transmitting the ARP packet. For instance for an Ethernet network its value is 1 while for IEEE 802 Networks it is 6. The “protocol type” field which is the next field specifies the type of layer 3 addresses used in the message for instance for IPv4 addresses, the value is 2048 (0800 hex).The next field is the “Hardware Address Length” which specifies the length of the hardware addresses. The value of this field is 6 for both Ethernet and IEEE 802 MAC addresses. The “Protocol Address Length” is the field which shows how long the layer three addresses are. For IPv4 addresses this value is 4.Following is the “opcode” field which specifies the type of the ARP message. Value 1 is used for ARP request and value 2 shows that the message is the ARP reply.” Sender Hardware Address” and “Sender Protocol Address” is the hardware address and IP address of the sender.

The “Target Hardware Address” is the hardware address of the machine to which the ARP message is being sent and Target Protocol address is the IP address of the target machine.

In case of an IEEE 802.11 network the same procedure will be adopted except that now the hardware type has the value 6 and ARP request is sent to the access point which broadcasts the ARP packet to all the stations on the local wireless network. If we see the packet in the wireshark capture it is like this, who has IPA (B)? Tell IPA (A) where IPA (B) is the address of the machine for which the MAC address is required and IPA (A) is the address of the sender of the ARP request.

The basic Beck and Tews attack makes use of the ARP packets because they are easily recognizable because of their characteristic length. Moreover the unknown fields in the ARP packet are the source protocol address i.e. sender IP address and the Target Protocol address i.e. destination IP address. The sender hardware address and the destination hardware address in the ARP packet are always sent in clear as described in TKIP encapsulation and are therefore not needed to be determined by the attacker.

After the successful attacks on TKIP by Beck and Tews, Martin Beck came up with “Enhanced TKIP attacks” based on the ICMP and TCP packets the details of which are given in the coming topics.

2.7 ICMP protocol

Martin Beck uses the ICMP packets in his attack called Michael Reset attack which would be described in chapter 4. Therefore it important to know how much byte is known in the ICMP packet. RFC 792 specifies the ICMP protocol. The ICMP packets are sent using the basic IP header. The purpose of the ICMP messages is to get a feedback about the communication network. For instance it is used when a datagram could not reach its destination. As the ICMP messages are sent using the basic IP header therefore the protocol field of the IP header has the value of 1 indicating that the IP header is followed by ICMP message.

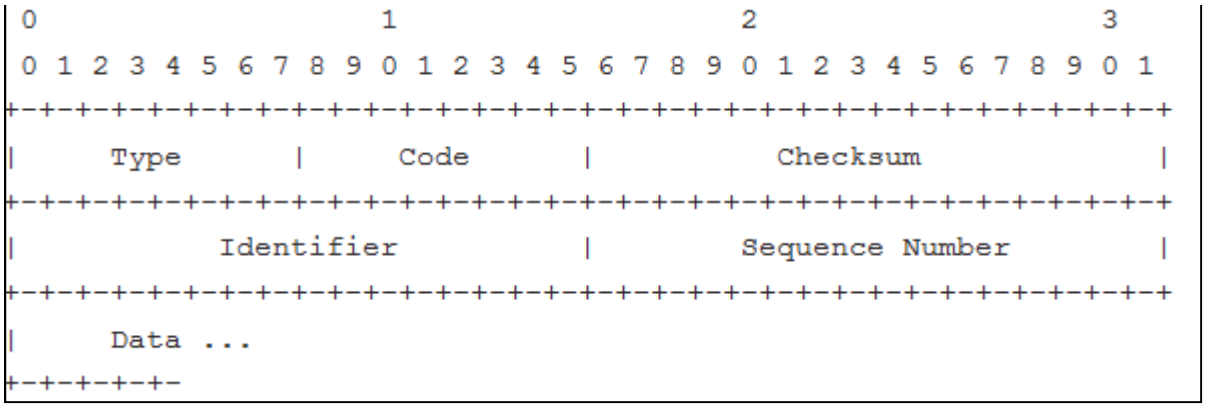

Figure 2-10 : ICMP Header structure [14]

In the figure 2.10 is shown the ICMP Header which follows the IP header.ICMP ping echo or 'ping' and echo reply or 'reply' are the two well known ICMP packets used in network troubleshooting to find out if there are any problems with the routes the packets are taking to get to the other side.

As shown in the figure the first octet is the "Type" field which has the value showing the kind of the ICMP message.For ping request its value is 8 and for ping reply its value is 0.The next field is the "code" field which is always equal to 0 for both requests and replies.The next two bytes field is checksum field. "The checksum is the 16-bit ones's complement of the one's complement sum of the ICMP message starting with the ICMP Type" Ref: RFC….Following it is the two octet identifier field which is set in the echo requests and echoes back in the reply to be able to keep different requests and replies together.Next is the 2 byte sequence number field which is usually starts with 1 and incremented for each packet sent.The sequence number and the code field are used together in matching the echo request with replies.The last field is the data field whose length varies depending upon the value of the type and code field in the ICMP header. It is used as a data padding and varies with the frame used on the LAN.

To conclude the known fields are type field, code field,identifier and the sequence field followed by the variable data field which could be padded and inserted by the attacker itself by using packet crafting tools i.e. hping3.Finally all the fields being known, the attacker could compute the checksum.

IP Header and Known Plaintext Bytes

As the IP header encapsulates the ICMP header therefore we would be interested in how much plaintext bytes does the IP header reveals to the attacker.The version field which is four bits long is the first field in the IP header and its value is 4.Internet header Length is the following four bit field indicates the length of the internet header in 32 bit words.The value of "IHL" field is 4 when the IP header does not contain the option field.If in case an option field is used then the value of the IHL field is 6. Therefore the version field and the Internet Header Length field could be guessed by the attacker and could have the value of 0x45 or 0x46 ,the latter being the case when the option field is used in the IP header.

The Differentiated services field is the next field in the IP header which has supreseded the IPV4 Type of service 'TOS' field which was defined in RFC 791. The differentiated service field is one octet long.Six bits of the field are used as a codepoint (DSCP) to select the PHB a packet experiences at each node and the remaining bits are the unused bits.PHB ensures what priority and policy would be applied to a packet at a particular hop.The primary purpose of the differentiated service field is to allow different levels of service to be provided to the traffic streams on the network. Moreover the packets could be assigned the default per hop behaviour in which case there would not be any specific treatment of a packet at a specific hop and the packets would be forwarded on a best effort basis.The value of the DSCP would than be '000000'. The differentiated service field when DSCP is set to default per hop behaviour could also be guessed by the attacker as it has the value '000000'. The following field which is the total length field showing the IP header length and data can also be calculated,given the total length of the transmitted packet. Moreover the source IP address and the destination IP address in the IP header are chosen by the attacker himself. The destination IP address would be the address of the client machine found by using the Basic beck and Tews attack on TKIP.So in total in an IP header twelve bytes are known.

Briefly an ICMP packet is used because it consists of the IP header and ICMP header in which 17 bytes of plaintext data is known in total .In ICMP header the type and code field of one byte each as well as identifier field and sequence field of two bytes each and in IP header the version

and IHL field (one byte) , differentiated services field(one byte) and protocol field (one byte), source and destination IP addresses (8 bytes) are known.

2.8 TCP packet structure and Protocol overview:

The Enhanced TKIP attack by Martin Beck makes use of the TCP-SYN packets therefore an overview of the TCP packet stucture and the protocol is of utmost importance here in this section.

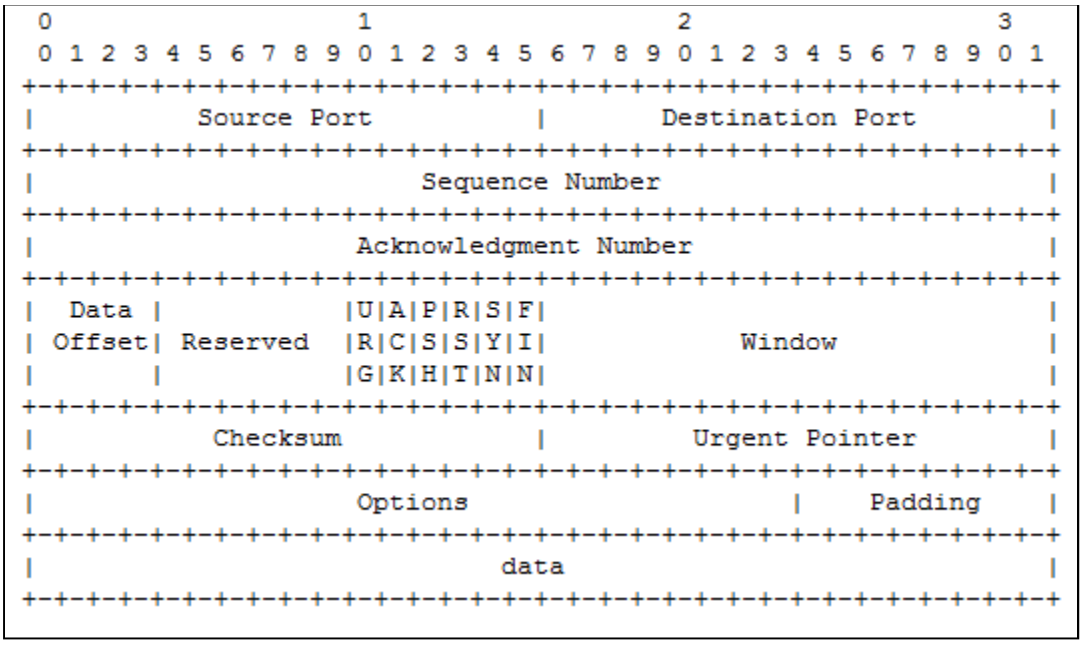

Figure 2-11 TCP Header structure [15]

RFC 793 gives the TCP protocol overview and the TCP packet structure [15]. It is to be noted that we are interested about the TCP protocol because TCP SYN packet reveals a lot of plaintext bytes to the attacker.

As shown in the figure 2.11 the first and second fields are the two octets source port and the destination port.RFC 1700 gives the details of the well known TCP port numbers.For the purpose of providing services the contact ports have been defined in RFC 1700 and these contact ports are called the “well known ports”.

Then is the sequence number and Acknowledgement number fields which are both 32 bits fields.The sequence number field indicates which packets are received. If the SYN flag is set to 1, the Sequence Number field provides the initial sequence number (ISN and the sequence number of the first data octet is one greater than the number that appears in the sequence number field i.e. (ISN+1).The acknowledgment number acknowledges a received segment. This field is the next sequence number the receiving computer is expecting to receive or in short, the sequence number of the last byte received +1.

Data offset which is 4 bits field specifies the length of the TCP header in 32 bit words The next 6 bits are reserved for future use while the following field of 6 bits contains six flag bits for instance RST,URG and ACK. The urgent pointer flag is used to identify that an incoming data is urgent. The packets having urgent pointer flag set do not have to wait until the previous segments are consumed by the receiving end but are sent directly and processed immediately. The window field follows next and is a 16 bit field and by which the sender of the segment indicates to the receiver about the number of data octets which it could receive.The 16 bit checksum field is the next field which ensures that the packet is not corrupted. When the Urgent Pointer flag is set to '1', then the Urgent Pointer field specifies the position in the segment where urgent data ends. The option field which could be a maximum of 40 bytes contain optional information. Therefore a TCP header could have a minimum data offset value of 5 indicating a header length of 20 bytes and it could have a maximum data offset value of 15 indicating a header length of 60 bytes.The padding field is the field which makes the whole packet a multiple of 32 bit words and the last field is the data field[15].

2.8.1 Three Way Handshake

The process of establishing a connection between two hosts A and B involve the exchange of three messages: the initiating host for instance A sends a SYN message, the receiving host B sends a message which contains the acknowldegement of the SYN message as well as its own SYN i.e. SYN/ACK and finally the host A sends an Ack for B SYN message.To illustrate this is shown the following figure.

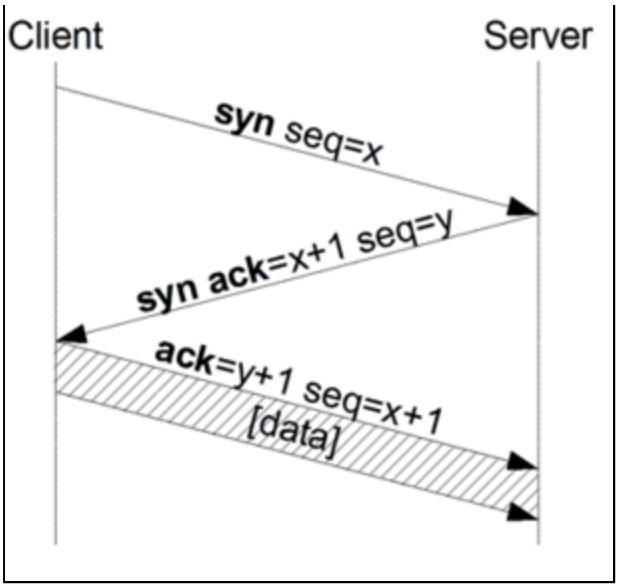

Figure 2-12 TCP Handshake

In figure 2.12 as shown first the client sends the SYN packet which contains its sequence number e.g. 'X' to which the server responds with the SYN/ACK packet.The SYN/ACK packet contains the acknowledgement number 'X+1' indicating the next sequence number which the server expects to receive from the client.In the SYN/ACK packet it also sends its own sequence number i.e. Y.Finally in the last message the client responds with the Ack packet with acknowledgement number Y+1 and sequence number X+1.The sequence number value is the intial sequence number incremented by one i.e. 'X+1'.After the establishing of connection the data transfer occurs as shown in the figure.

TCP-RST Packet

The attacker could use the TCP SYN packet but we are interested to know how much plaintext bytes could this packet reveal. As Martin Beck points out in his paper "Enhanced TKIP Michael Attacks" an attacker could use a TCP SYN packet with the source IP address being spoofed to the IP address of an access point.The forged SYN packet is then send to a wireless client which then sends the SYN/ACK packet to the access point . But as there is no information about this connection with the access point therefore it responds with a TCP-RST packet.As the attacker wants to obtain the keystream therefore he is interested to know how much plaintext bytes are known in the encrypted TCP-RST packets. In the TCP-RST packet the sequence number field

will carry the value of initially sent TCP-SYN packet incremented by one. The acknowledgement field will carry the value of zero because the sender of the TCP-RST packet will not indicate to the receiver about the value of the next sequence number it wants to receive which is quite understandable.The header fields and the flags field in theTCP header have the value of 0x50 and 0x04 indicating the 20 byte TCP header and the setting up of the Reset flag.Moreover the window field will be having a zero value as the connection has not been established and indicating the size of the window to the client is unnecessary.Moreover the urgent pointer field will be zero as the there is no data transfer between the access point and the client.Thus the whole TCP header of the TCP-RST packet is known to the attacker as the checksum could be computed [1].

Chapter 3

3 BECK AND TEWS ATTACK ON TKIP

Martin Beck and Erik Tews for the first time attacked TKIP and the protocol which was considered to be secure alternative to WEP was no more secure. They successfully developed a proof of concept of the attack in the form of a tool called tkiptun-ng which is now part of the Aircrack-ng suite. The attack was a modified form of the Korek chop chop attack on WEP and could be called as the TKIP ICV chop chop attack. This chapter gives details of this attack, application areas and the countermeasures which could be taken against this attack.

3.1 Requirements

In order for the Beck and Tews attack to work there are some requirements which need to be fulfilled. The enabling of QOS on AP is one of the requirements while the other is the setting up of the key renewal interval to duration which is more than the completion time of the attack. It is to be noted that Key renewal interval is usually set to 3600 seconds and as such the attack works. The requirement for the key renewal interval to be more than the completion time is because when the key renewal occurs during the attack a new PTK is generated. This means that if the attacker is successful to obtain a keystream through the Beck and Tews attack corresponding to a PTK, that keystream would only be valid for that specific key renewal interval. Thus an attacker could not inject packets into the network using the obtained keystream as the PTK has changed. Why the attack works on QOS enabled networks is elaborated later when the TKIP ICV chop chop attack is explained.

3.2 Beck and Tews Attack

The beck and Tews attack is not a key recovery attack but an attack in which the AP to STA keystream is obtained after successfully capturing an ARP packet and decrypting it though TKIP ICV chop chop attack which is a modified form of the Korek chop chop attack. After the obtaining of the keystream, an attacker reverses the Michael algorithm and obtains the MIC key. Finally the attacker can inject custom packets into the network.

The attack consists of the following stages which are elaborated in detail. These stages are also illustrated by means of a flowchart as shown below [17],

- Client Deauthentication.
- TKIP ICV chop chop attack.
- Guessing the remaining bytes.
- Reversing the Michael algorithm.

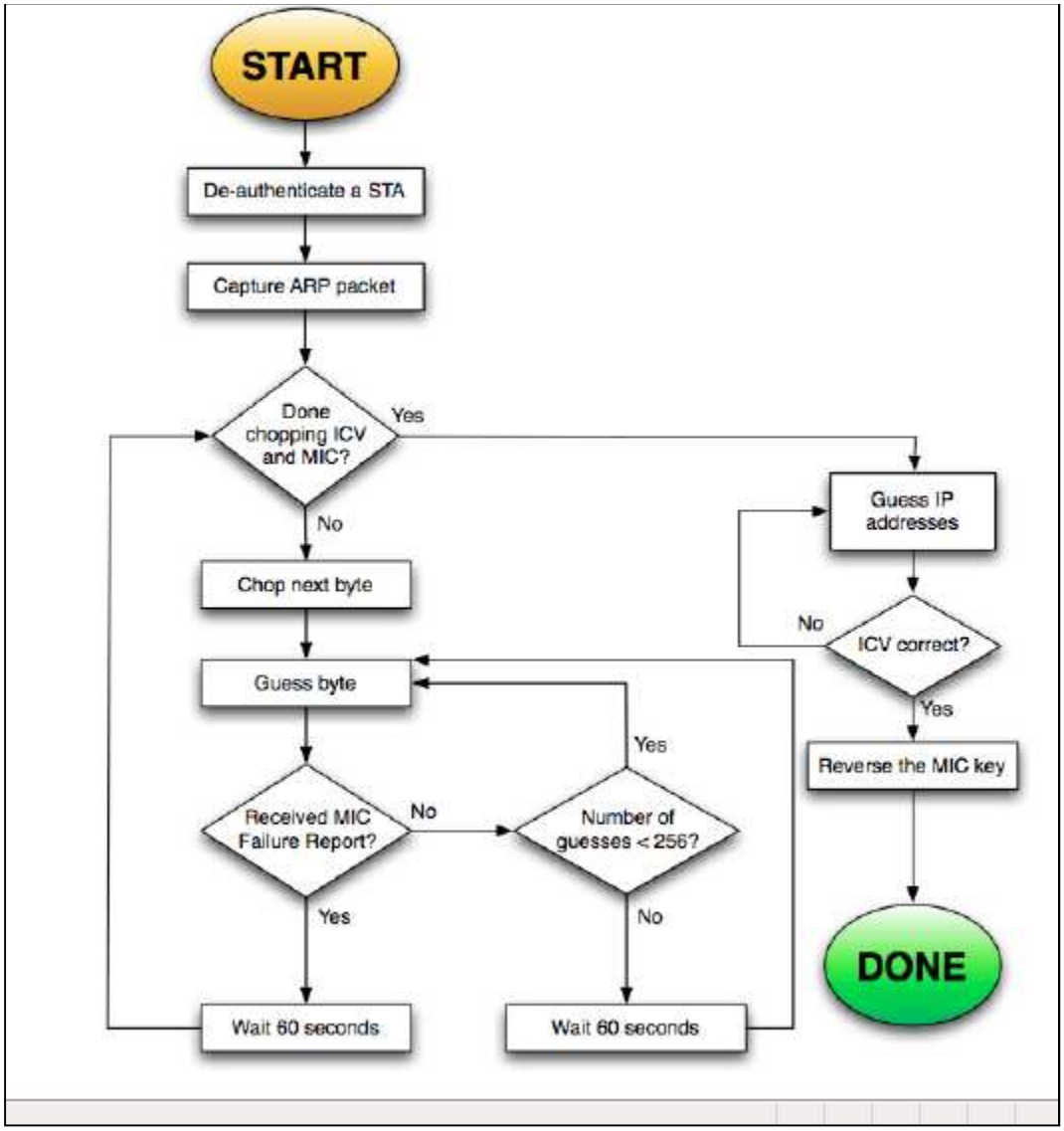

Figure 3-1 Flowchart of Beck and Tews Attack [17].

Client De-Authentication

Before the TKIP ICV chop chop attack starts a STA associated to an AP is de-authenticated. The purpose of de-authenticating a STA is that by doing so an EAPOL handshake takes place between the AP and the STA. In an EAPOL handshake, the PTK and GTK are generated which are used to encrypt the unicast traffic and the broadcast or multicast traffic respectively. Finally after performing an EAPOL handshake, control packets i.e. ARP packets are exchanged between the STA and AP in order to know the MAC addresses of each other. This ARP packet is important as it is the packet which the attacker is interested about because most of the ARP packet is known plaintext. Moreover an ARP packet is easily recognizable because of its characteristic length. But it is important to know that the ARP packet from the AP to the STA is the one which the attacker would finally capture , the reason of which would be known in the next stage i.e. TKIP ICV chop chop attack.

TKIP ICV Chop Chop attack:

After capturing an ARP packet going from the AP to STA, the attacker performs the TKIP ICV chop chop attack which is illustrated in the following figure,

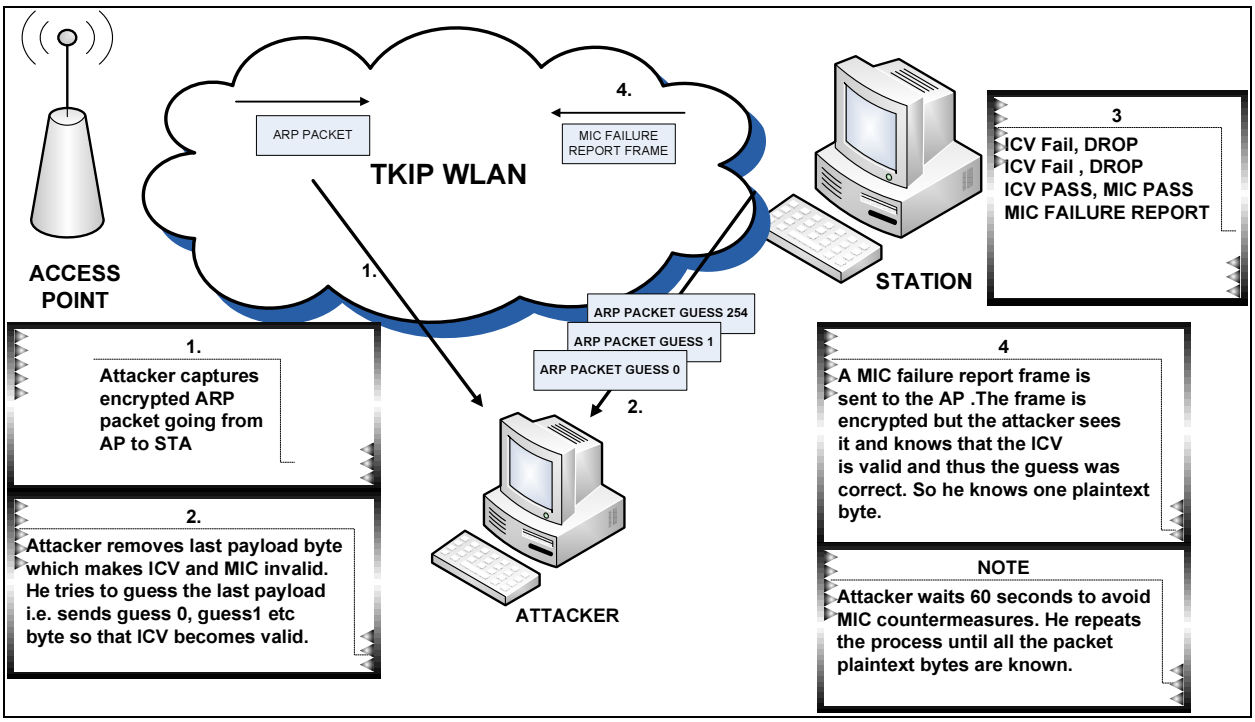

Figure 3-2 TKIP ICV Chop Chop Attack

As illustrated in the Figure 3-2 the TKIP ICV chop chop attack starts by capturing a packet going from the AP to the STA. This attack is a modification of the korek chop chop attack as the ARP packets after truncation of its last byte is sent to the STA and not towards the AP as in case of the korek chop chop attack described in Sec 2.3.1. Moreover the Korek chop chop attack described relies on the validation of the truncated packet by the access point if the guess was correct and finally the sending of the validated packet to the wireless network. Thus the attacker knows gets to know about the plaintext bytes. In contrast to it, the TKIP ICV chop chop attack relies on the TKIP countermeasures taken by the STA, when the guess was correct and finally the sending of the MIC failure report frame by the STA towards the AP.

As shown in Figure 3-2, when the attacker removes the last payload byte the ICV and MIC of the packet becomes invalid. Afterwards the attacker tries to make the ICV valid. This is done by guessing a ‘value’ and using this ‘value’ to obtain a modified data stream which when XORED with the truncated packet makes it valid as already discussed in Sec 2.3.1. This ‘value’ in this context is called “guess” as shown in fig 3.2.If the value or guess is correct, the packet becomes valid in which case the ICV is validated but the MIC still would still be invalid which is because an active attack is undergoing. When this occurs the STA sends a MIC failure report frame towards the AP as shown in 3.2. The attacker sees that and thus knows that the ICV test has passed which indicates to the attacker that the guess was correct. Finally the attacker now knows the last plaintext byte value.

It is important to note that when the attacker detects the MIC failure report frame, he waits for 60 seconds before chopping the next byte. It is because if the AP detects two MIC failures within a time span of 60 seconds TKIP countermeasures are activated which means that the authenticator (AP) will de-authenticate all STAs associated with it and will delete all the PTKs and GTK. At this moment a new GTK is generated but it is not used for 60 seconds and after one minute a new PTK is also configured [9]. This concludes that the last plaintext byte value now known to the attacker and corresponding to a specific PTK is no more valid for the new configured PTK and thus the attack could not proceed further. After waiting for 60 seconds to avoid MIC countermeasures the attacker goes on with the remaining plaintext bytes until all the packet is known to him.

Another important point which was also mentioned in the Sec 3.1 of this chapter was the enabling of the QOS networks for this attack to work. Why this condition of QOS enabled networks should be fulfilled for this attack to work could be explained by means of the following figure.

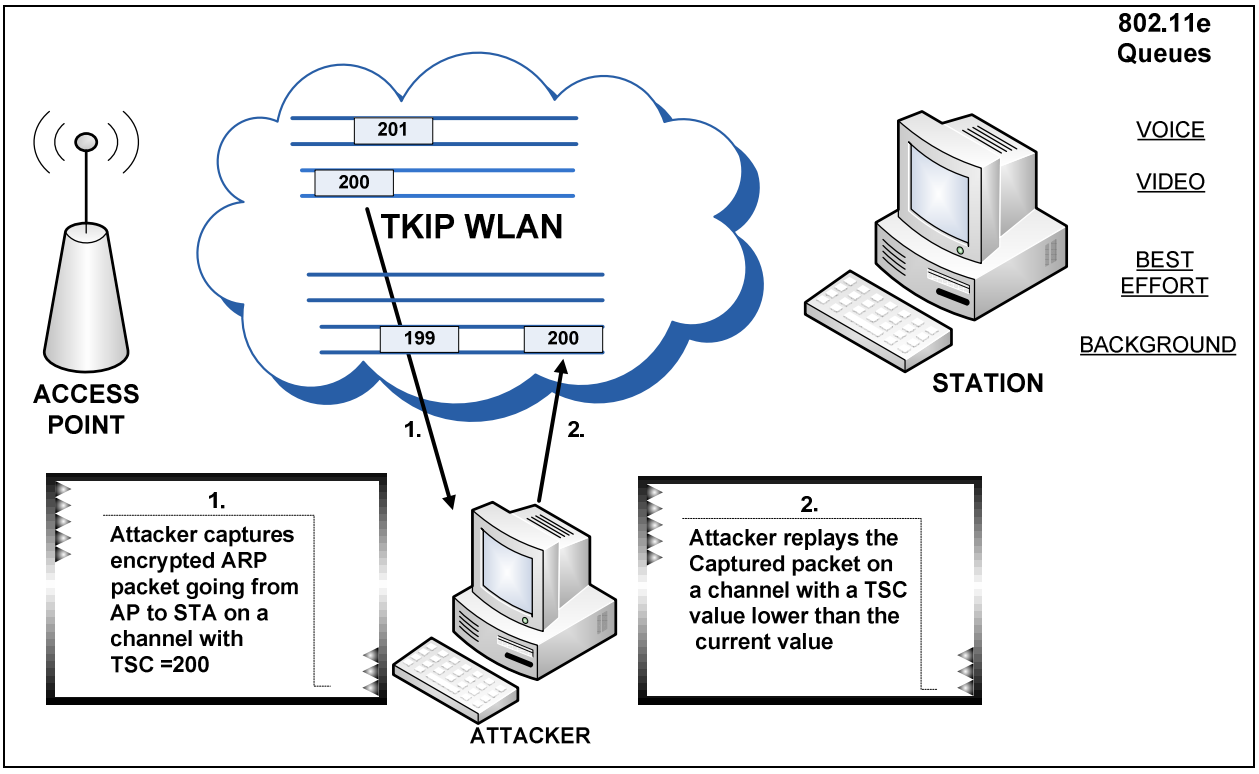

Figure 3-3 802.11e Queues

The enabling of the QOS on the network is important for this attack to work because than the traffic would be prioritized and all the traffic is not sent on the same channel. In the Wireless Multimedia (WMM) which is a subset of the IEEE 802.11e draft, four channels are specified for the traffic to be sent, Voice has the top most priority followed by Video, best effort traffic and finally Background. As shown in Figure 3-3 there is a weakness in IEEE 802.11e which is exploited by the attacker. The weakness is that the sequence is enforced across multiple queues and this enforcement is not independent of each other. To put it simpler it means that a packet if captured on a channel meant for video streams and having TSC value 200 could be sent on a channel meant for Background traffic where the current TSC value is 199 and therefore less than the value of the captured packet. This weakness could have been avoided if the sequence

counters were enforced independently on each channel. Consequently the TKIP ICV chop chop attack is conducted on a channel which has a TSC value lower than the one on which the ARP packet was captured as shown in the figure 3.3.

Another important worth mentioning here is that the TKIP ICV chop chop attack could be conducted persistently on a channel with a lower TSC value than the channel on which the ARP packet was captured. This is because the TSC value is only updated if both the ICV and MIC are correct. In this case the attack relies on correct ICV and incorrect MIC so TSC value will not be updated on the channel on which the attack is run.

Guessing the remaining Bytes:

As is already been discussed in chapter 2 in the section dealing with the ARP packet structure Sec 2.6 which says that an ARP packet has known plaintext bytes except the source and destination IP addresses (8 bytes) and incase of encrypted packet the 8 byte MIC and 4 byte ICV are also unknown. As we also know from the previous discussion that an attacker waits one minute between every byte chopped in order to avoid MIC countermeasures therefore one plaintext byte is known in one minute and 20 bytes would talk about 20 minutes. To speed up the attack an attacker could guess the source and destination IP addresses.

As mentioned in RFC 1928 that the internet Assigned Numbers Authority (IANA) has reserved the address blocks 10.0.0.0-10.255.255.255, 172.16.0.0-172.31.255.255 and 192.168.0.0-192.168.255.255 for private internets in order to prevent the exhaustion of IPv4 address space therefore an educated guess could be made about the source and destination IP addresses. Once the guess about source and destination IP addresses is done, it is verified by computing the ICV of the whole packet with the guessed source and destination IP addresses and then comparing it with the already decrypted ICV.

Reversing the Michael Algorithm:

Once the whole ARP packet which was sent from AP to the STA is known to the attacker, the attacker has obtained the AP to STA keystream. Now if the attacker is in possession of the MIC key, he can successfully inject custom packets into the network. But how can an attacker know a MIC key.

As Michael was not designed to be invertible which means that it is not a one way function the MIC key could be found it .As already discussed in Sec 2.4 related to MIC, that the input to the

Michael algorithm is the MIC key and the plaintext MSDU with SA, DA priority fields and the output is the plaintext MSDU with MIC. So if MIC is invertible then an attacker who already knows the plaintext MSDU and the MIC could compute the MIC key by reversing the Michael algorithm. When the MIC key is known, an attacker can inject custom packets into the network but there is a limitation as to how many packets could be injected.

3.3 Limitations

The Beck and Tews attack obtains the AP to STA keystream after which it computes the MIC key and then injects the packets in the network. But we already discussed and illustrated in Figure 3-3 802.11e Queues, that in products with WMM support, four queues are used to prioritize traffic in which the best effort or regular traffic is sent on channel 0. Therefore 3 custom packets could be sent on the remaining three channels on which the TSC value is lower than the channel on which regular traffic is sent. Moreover in some implementations there are 8 channels i.e. TID 0 to TID 7 of which TID 0 is used for regular traffic therefore the custom packets could be sent on the 7 remaining channels. Consequently 3-7 packets could be sent to the network. It was also discussed that Sec 2.5 that there is TID subfield of 4 bits which is a part of QOS control field and thus has 16 possible values. This means there is a potential of sending 16 packets which has not been tested as yet.

3.4 Practical TKIP Attack Example

A practical TKIP attack is illustrated in the following figure,

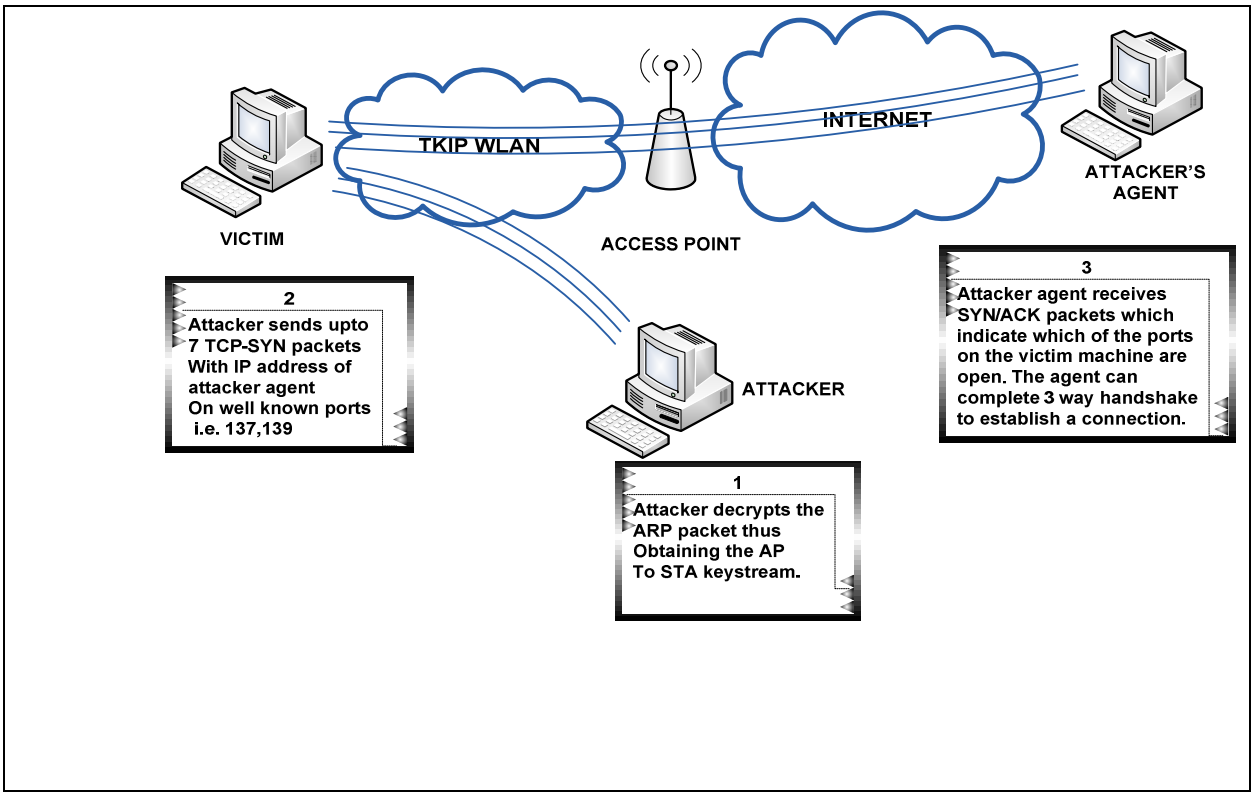

Figure 3-4 A Practical TKIP attack

As illustrated in Figure 3-4 A Practical TKIP attack an attacker after successfully doing the Beck and Tews attack and finding the AP-STA keystream by decrypting the ARP packet and knowing the MIC key sends upto 7 TCP-SYN packets by spoofing the IP address of the attacker agent which is a STA on the WAN. The TCP-SYN packets are sent to the well-known ports of the victim machine. Thus if some of the specified ports are open on the victim it would respond with the SYN-ACK packets towards the attacker agent machine. After receiving the SYN-SCK packets the attacker agent would know which ports on the victim are open. The attacker agent machine could then complete the TCP three way handshake and afterwards could do a malware attack on the victim machine.

3.5 Countermeasures

The best way to mitigate against the Beck and Tews attack is to use the AES-CCMP. The downside of using it is that AES-CCMP could not be implemented on legacy hardware. Another countermeasure is that the frequency of key rotation could be increased. As already discussed that key renewal interval is by default set to 3600 seconds which is long enough for the attack to

be successful. Therefore the key renewal interval should be small and as pointed out by Beck and Tews it should be brought down to 2 minutes. The downside of it is that AP would be burdened. The key renewal interval is the time duration during which the PTK remains valid. If the key renewal occurs during the attack, the keystream already obtained by the attacker and corresponding to a specific PTK is no more valid as the PTK has changed.

It is important to mention that the attack could be executed against networks where QOS is not enabled on the AP. This means disabling QOS on an AP could not be a countermeasure anymore. Even if it was a countermeasure against the Basic Beck and Tews attack as mentioned by Olav and Finn than it could not be used as a countermeasure in 802.11n High Throughput networks where QOS requirements are stringent.

3.6 Beck and Tews Attack against TKIP with QOS Disabled in AP

Beck and Tews pointed out that the attack could also work for a non QOS networks. The idea was worked upon and it was found out that the attack could be successfully run with a QOS client associated with a non-QOS Access point. This idea was first taken from the paper “A Practical Cryptographic Denial of Service Attack Against 802.11i TKIP and CCMP” [17].

As the QOS data frames contain the 2 byte QOS control information field in the MAC header as discussed in Sec 2.5.1. Therefore for this attack to work the encrypted ARP response from the AP is taken by the attacker and the two byte QOS control field is inserted into the regular data frame to convert it into a QOS frame. Moreover bit b7 should have the value 1 in the QOS data frame so this bit is flipped in the data frame to make it a QOS data frame.

Chapter 4

4 ENHANCED TKIP MICHAEL ATTACKS

This chapter deals with some of the related work which has been done to extend the Beck and Tews attack on TKIP. Furthermore the Enhanced TKIP Michael attacks by Martin Beck [1] will be explained in detail, the theory of which will be validated in chapter 6.

4.1 Extended Attack on DHCP ACK packets

Halvorsen and Haugen extended the Beck and Tews attack into a modified chop chop attack on DHCP ACK packets [16]. DHCP is a common mechanism to assign an IP address dynamically to a station on a network. The DHCP packets are constructed on UDP and DHCP ACK packet is the final packet in the DHCP message exchange. In the table below is given the four types of messages sent in the exchange [18].

Table 4-1 DHCP Message Exchanges

Message	**Description**
Discover	A client broadcasts Discover message to locate available DHCP servers. (AP)
Offer	DHCP server (AP) responds with the offer message containing the configuration parameters i.e. the IP addresses, the DHCP server has currently.
Request	The client sends a DHCP request in response to the offer from the server. By sending this message the client requests offered parameters from one server and declines others. It also means verifying a previously allocated IP address after system reboot or disconnection or extending the lease time of the IP address.
ACK	Finally in the DHCP ACK message the server assigns the IP address to the client.

As in these experiments the attack works after the client is deauthenticated or disconnected from the WLAN, therefore the client just has to send the two messages i.e. the DHCP request and DHCP ACK. These two messages are sent to acquire the previously assigned IP address [18].

So Halvorsen and Haugen were able to attack the DHCP ACK packets, the purpose of doing it was to obtain a keystream of 596 bytes more than the keystream of 48 bytes obtained through the Beck and Tews attack which was performed on the ARP packets. The Halvorsen and haugen attack on DHCP ACK packets recovers the AP to station keystream just as the Beck and Tews attack on ARP packets.

Details Of the attack

Halvorsen and Haugen did their improved attack on the 330 to 584 byte DHCP ACK packets. The format of these packets is hardware dependent therefore its length varies. Most of the data in the DHCP ACK packets is known plaintext except the two IP addresses i.e. IP address of the client and IP address of the server. In addition to it the 4 byte transaction ID: a random number chosen by the client and used by the client and the server to match the requests with the response is unknown. Finally the 4 byte ICV and the 8 byte MIC of the encrypted DHCP ACK message are unknown. The two IP addresses could be found by the Beck and Tews attack and therefore an attacker is left with 16 unknown bytes.

Since the transaction ID is in the middle of the packet and the data after and before it is almost known therefore an attacker does not proceed with the modified chop chop attack. Rather the known data before and after the transaction ID is inserted through a method called the simulate chop chop and thus the chopped array is updated with the already known data. The transaction ID is than guessed by chopping the bytes and finally the ICV is verified for all the known data and the MIC key is reversed [16].

4.2 Man in the Middle Attack on TKIP

Ohigashi and Morrii applied the Beck and Tews attack to the MITM attack. The attack was called as the practical message falsification attack on WPA. The objective of this attack was to reduce the execution time of the Beck and Tews attack to recover a keystream after the first successful attack to duration of one minute.

As according to Beck and Tews after the first execution of their attack, the recovery of a keystream takes 4-5 minutes. The reason is that an attacker already possesses the MIC key now from the first attack. The only unknown bytes to the attacker are the four byte ICV and the IP addresses of the STA and the AP. The modified chop chop attack will be done to find the four byte ICV and takes 3 minutes. It is because the attacker has to wait for one minute after getting a MIC failure report in order to avoid MIC countermeasures. Finally the IP addresses of the STA and the AP could be guessed. The MIC key is used to compute the MIC and finally the ICV is computed using the CRC-32 checksum. For each guess the MIC and then the ICV is computed and then compared with the decrypted ICV to validate the guess.

But according to Ohigashi and Morrii their attack takes only one minute in order to recover a keystream after the first attack. They used the MITM attack in the WLAN to intercept the communication until the chop chop attack ends. This is done using directional antennas so that the sender of the packets does not detect the attack.

Ohigashi and Morrii's attack only one minute as compared to Beck and Tew's attack duration of three minutes. As according to them after the Beck and Tews attack is run for the first time, an attacker knows the MIC key and the IP address of the access point because it is fixed. The only unknown bytes are the four byte ICV and the last byte of the STA IP address. They proceed further from now hereon by chopping the last one byte of the ICV instead of chopping all the four bytes. Then the attacker makes the 2^8 guesses for the last byte of the STA IP address. For each guess, he computes the MIC and then the ICV and compares the last byte of the ICV with the already decrypted last byte of the ICV. If both of them are identical then an attacker has successfully recovered the keystream [19].

4.3 Enhanced TKIP Michael Attacks

According to Martin Beck, by sending a TCP-SYN packet by spoofing it to the IP address of a Linux access point or STA running Linux, an attacker can get 7 new 60 byte keystreams without any fragmentation. This is one of the attacks which is part of the enhanced TKIP Michael attacks and is called the TCP attack [1].

As we know that In an IEEE 802.11 network most of the packets are IPv4 and ARP packets. In an IPv4 packet, 2 bytes are known plaintext which has been mentioned in Sec. 2.7, 4 bits each of

the version field and IHL field together (1 byte) could have the value 0x45 or 0x46 in case there is an IP option field used which is known to the attacker. The Differentiated service field which is one octet has the value 0x00 is the next field known to the attacker in the IPv4 packet. The total length field which is two octets is the next field and is also known. This is because the total length field indicates the total length of the transmitted packet including the internet header length and the data. The data length is selected by the attacker himself so he calculates the value of the total length field as he known the length of the internet header. The IP header follows the LLC/SNAP header (8 byte) in an IEEE 802.11 network and is known. Therefore in an IPv4 packet, a total of 12 keystream bytes are known to the attacker.

As IEEE 802.11 network a longer packet could be sent through fragmentation and the network supports up to 16 fragments [1]. Hence an attacker can fragment an arbitrary packet into 16, encrypt each fragment with the 8 byte keystream out of the 12 bytes the remaining four bytes being used for ICV. This allows us to send a packet with a size of 120 bytes as the last fragment needs to carry the MIC. It is to be noted that the ICV is computed for each MPDU while the MIC is computed for the whole MSDU. An illustration of the fragmentation of a MSDU into MPDU is given below.

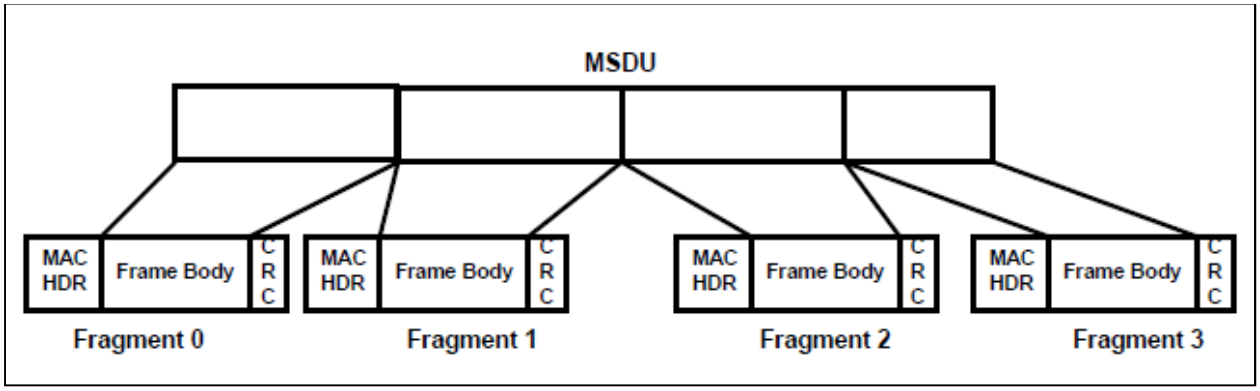

Figure 4-1 A Fragmented MSDU

4.3.1 TCP Attack:

The TCP attack proposed by Martin Beck is illustrated in Figure 4-2,

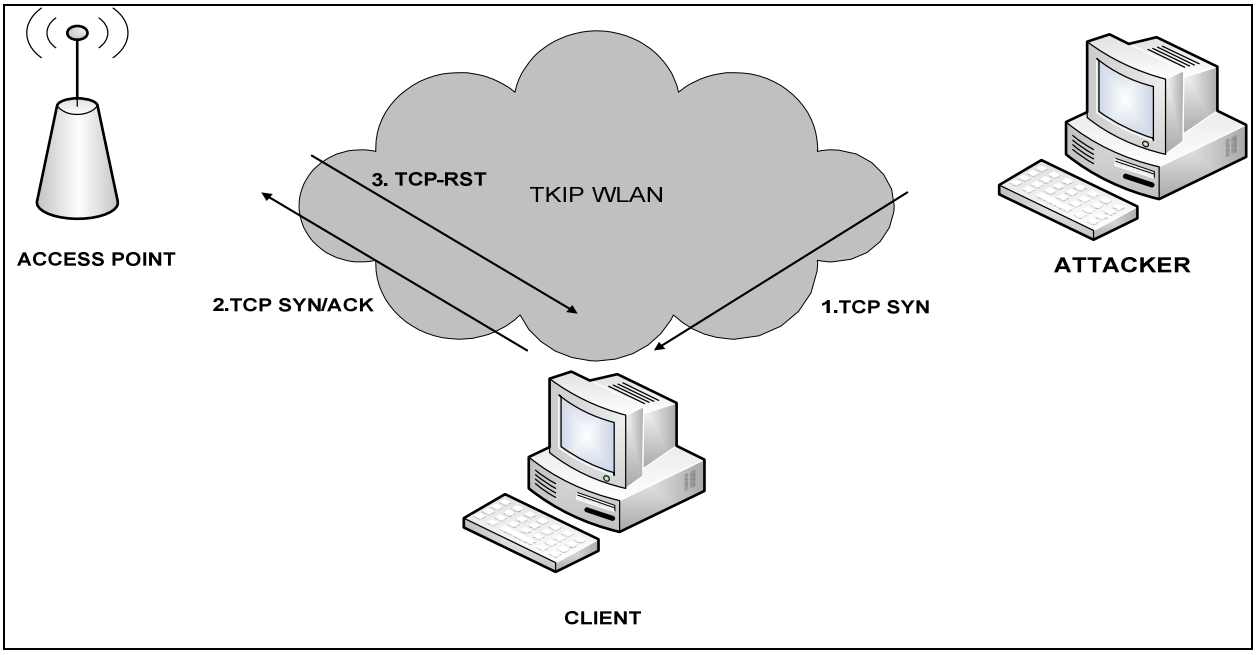

Figure 4-2 TCP Attack

Before the TCP attack, a Basic Beck and Tews attack is conducted on a TKIP WLAN. By doing so an attacker gets the IP address of the Access point and the client. After this the attacker finds which of the ports on the client machine are open so that he can send the TCP-SYN packets towards the open port of the client.

The attacker while sending the TCP-SYN packet spoofs it to the IP address of the AP which he found out by the Beck and Tews attack. The client after receiving the TCP-SYN packet sending the TCP SYN/ACK packet to the access point.

The access point after getting the TCP-SYN/ACK packet responds with the TCP-RST packet indicating that the access point has no previous information about this connection. The TCP SYN packet sent by the attacker in the first instance is 56 bytes with 8 byte LLC header, 20 byte IP header, 20 byte TCP header and 8 byte MIC. In order to send this packet the attacker would have to use the 8 byte of already known keystream from the IP header. Thus 7 fragments are needed to send the whole TCP-SYN packet. As same response would be sent by the access point

on the 7 channels (IEEE 802.11e QOS channels) owing to fragmentation an attacker would never get more usable keystreams.

4.3.2 TCP Attack with Linux Access Point

According to Martin Beck in order to get more usable keystreams without any fragmentation the TCK attack is run on a TKIP WLAN with Linux AP. As already discussed that incase of the IP packets 12 bytes are already known to the attacker.

But with the AP running Linux all the fields of the IP header in the TCP-RST packet could be found. The identification field (2 bytes) in case of a TCP-RST is zero followed by flags and fragment offset field (2 bytes) which carry the value 0x40 indicating that 'DF' flag is set (No fragmentation to occur) and fragment offset field value is zero. As the access point is running Linux so the next field the TTL field is known which is set to 0x40 as a default by Linux systems and as TCP is running over IP so the protocol field has the value 0x06.The source and destination IP addresses are also known to the attacker which are the addresses of the access point (spoofed IP address) and the client respectively. These were known by the Basic Beck and Tews attack. Therefore all the fields are known so the checksum could be computed. So in a TCP-RST packet the whole IP header is known.

In the TCP header of the TCP-RST packet the source and destination ports (4 bytes), the sequence number field, the acknowledgment field (8 bytes), the data offset and flags field (12 bits), the window and the urgent point field (4 bytes) of the TCP header are also known to the attacker as discussed in section 2.8. Finally the checksum could be computed. This concludes that 60 byte keystream constituted by the 8 byte LLC, 20 byte IP, 20 byte TCP, 8 byte MIC and 4 bytes ICV is known to the attacker and is used to encrypt the whole TCP-SYN packet without the need for fragmentation. This generates 7 new 60 byte keystreams on each of the channels of IEEE 802.11e QOS. Consequently this attacks result in more usable keystreams.

4.3.3 Remote TCP Attack

The remote TCP attack which is part of the Enhanced TKIP Michael attacks is illustrated in the figure below. This attack is done when an attacker does not have a Linux system available.

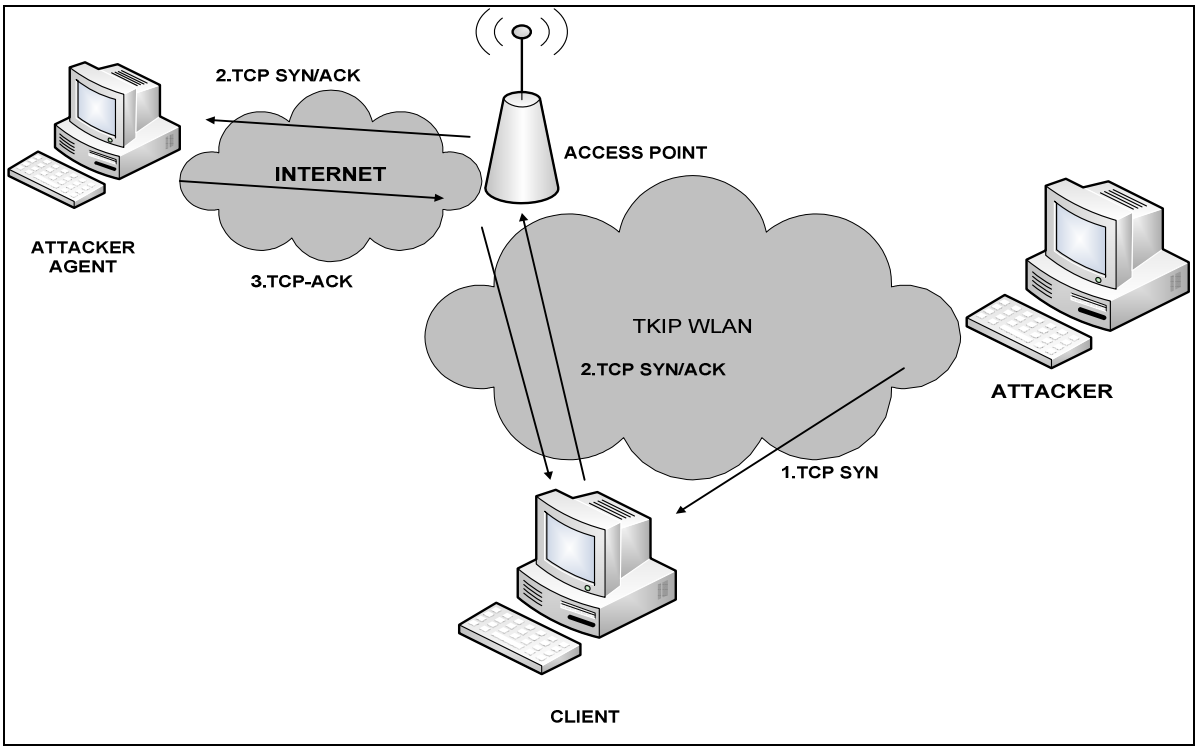

Figure 4-3 Remote TCP Attack

The above attack which is the remote TCP attack is almost the same as the practical TKIP attack example discussed in Chapter 3. Once the IP addresses of the client and the access point is obtained through the Basic Beck and Tews attack, he can inject the TCP-SYN packet by spoofing the IP address of a remote machine on the internet. By doing so, the attacker can specify the exact content and size of the final packet in the TCP handshake i.e. TCP-ACK packet because the attacker agent machine is the control of the attacker. This way an attacker gets keystreams as the plaintext which is the TCP-ACK packet was chosen by the attacker because he is in control of the remote machine and the encrypted TCP_ACK packet is also finally obtained by the attacker. This way an attacker can get arbitrary length keystreams because he can specify extra padding in the TCP-ACK packet. This type of attack requires that the attacked wireless network should have a connection to the internet.

4.3.4 Michael Reset Attack

Martin Beck in his paper titled “Enhanced TKIP Michael Attacks” proposes another clever by which an arbitrary captured packet is decrypted by resetting the internal station of the Michael algorithm.

As already discussed in Section 2.4, that in order to compute the MIC a secret key of 8 bytes which is then converted to two 32 bit words is used. The data which is in the form of 32 bit words is encrypted using this key to compute the MIC. According to Martin Beck the Michael reset attack starts with the calculation of what he calls magic words. These magic words are then inserted between an arbitrary packet captured on the WLAN TKIP and an ICMP echo request. Afterwards the ICMP echo request is sent to a client machine on WLAN with the IP address spoofed to the address of the WAN port of the attacker station. This causes the ICMP response to be sent to the WAN port of the attacker. As the response goes through the internet therefore the arbitrary captured packet gets decrypted as it reaches the WAN port of the attacker station.

The important and very clever technique is the calculation of the magic words to reset the internal state of the Michael algorithm. By resetting the internal state of the algorithm, an attacker causes the MIC of the captured packet to be valid for the whole packet which consists of the ICMP echo; the magic words itself and the captured packet.

The Michael reset attack is illustrated in the following figure,

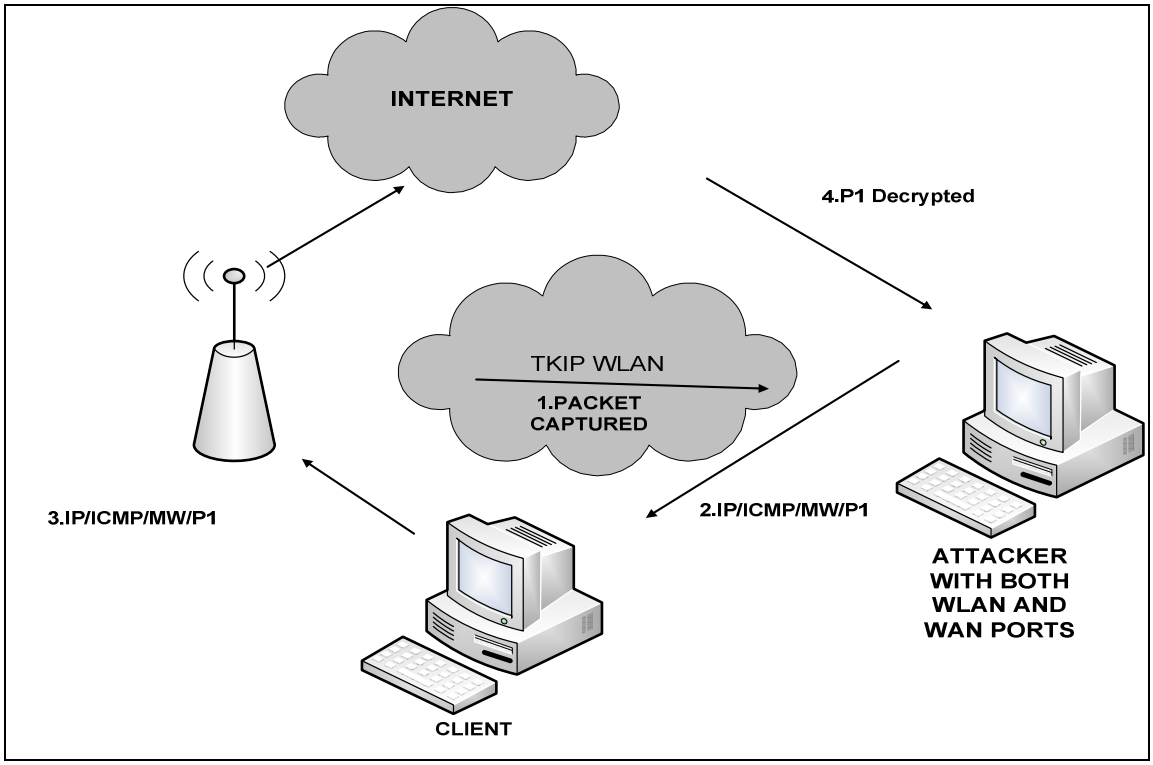

Figure 4-4 Michael Reset Attack

As shown in the Figure 4-4 the attack starts when an attacker captures an arbitrary packet on the WLAN.As the attacker knows the IP address of the client machine through the Basic Beck and Tews attack he sends a ICMP echo request concatenated with the two magic words and the captured packet.

Thirdly the client machine sends the response to the IP address of the WAN port of the attacker because the ICMP echo request was spoofed to that IP. Finally the response is sent through the internet and is therefore decrypted. This way an attacker gets the keystream for the captured packet as he now possesses both the plaintext and the ciphertext. This keystream can be used to send other packets.

Chapter 5

5 LABORTARY ENVIRONMENT

The chapter is gives a description of the laboratory environment i.e. the hardware and software used in the experiments.

5.1 Hardware

The hardware used in the experiments consists of three entities i.e. the victim, the attacker and the access point.

It is very important to select a wireless interface card which is compatible with the software used in the experiments. There are always two manufacturers involved with the wireless cards, one is the brand manufacturer of the wireless card and other is manufacturer of the chipset. Both the chipsets with its driver should be compatible with the software used. In the experiments, I have used the Atheros and Ralink chipsets which supports the software used in the experiments. The brand manufacturers of the cards are Linksys, Dlink and 3com.

5.1.1 Victim

The victim is a machine which has windows 7 installed on it. It was chosen with the intention to make the Beck and Tews attack successful against the windows machine which had not been the case in the previous work done. Besides the attack was also tested on Ubuntu Release 2.6.31-19-generic as the operating system. Wireless interface cards of two types were used on the victim. One was the 3 com Wireless a/b/g, Atheros AR5413 chipset and the other was D-Link AirPlus G DWL-G510 (Ralink, RT 2561/RT61).

5.1.2 Attacker

The attacker is a machine which runs the tkiptun-ng tool which is the proof of concept of the Beck and Tews attack and the operating system running on this machine was Ubuntu Release 2.6.13-19-generic.

5.1.3 Access point

The access point is Linksys WRT54GL v and supports IEEE 802.11e QOS/WMM which is the basic requirement for running the Beck and Tews attack.

A Summary of the hardware and software used in the experiments is given below,

Table 5-1 Attacker

Model – Attacker	Dell Optiplex GX 270.
CPU	Intel Pentium 4, 2.80 GHZ.
Memory	2 GB.
Operating system	Ubuntu Release 2.6.31-19-generic.
Wireless interface	Linksys Wireless-G 2.4 GHZ 802.11g Model No: WMP54G ,Ralink chipset- rt61pci
MAC Address	00:1C:10:6D:60:86

Table 5-2 Victim

Model –Victim	Dell Optiplex GX 270
CPU	Intel Pentium 4, 2.80 GHZ.
Memory	2GB
Operating System	Windows 7 build 7600, Ubuntu Release 2.6.31-19-generic.
Wireless interface	3 com Wireless a/b/g, Atheros AR5413 chipset,ath_pci
MAC Address	00:22:57:EA:2A:39

Table 5-3 Access Point

Model-Access point	Linksys WRT54GL v1.1
Firmware	v4.30.11, Aug. 17, 2007
Standards supported	IEEE 802.3, IEEE 802.3u, IEEE 802.11g, IEEE 802.11b, IEEE 802.11e QOS/WMM.
Wireless security	WEP, WPA/WPA2 Personal, WPA/WPA2 Enterprise, RADIUS.
BSSID	00:1E:E5:64:F1:15
Router MAC Address	00:1E:E5:64:F1:13

5.2 Software

The software used in the experiment was the following,

5.2.1 The Aircrack-ng Suite

The Aircrack-ng suite is an open source security software having version 1.1 released now. It is very important wireless security software because it contains the aircrack-ng which is a tool used for cracking WEP and WPA-PSK networks. Moreover it contains the tkiptun-ng tool which is a proof of concept of the Beck and Tews attack.

The other tools which were part of the suite and were used in my experiment were the airodump-ng tool which was used for packet capturing, airmon-ng which was used to create a virtual interface in monitor mode.

Just recently in aircrack-ng version 1.1, there is an addition of another tool called the airdrop-ng which is a program used for targeted, rule-based deauthentication of users. It can target the clients and deauthenticate them based on MAC address, type of hardware, (by using an OUI lookup, IE, “APPLE” devices) or it can simply deauthenticate all users. By dauthenticating all clients it functions in the same way as the aireplay-ng tool which is also part of the aircrack-ng suite [20].

5.2.2 Wireshark

Wireshark is an open source packet analyzer which is used to capture packets and then give details of the captured packets. The tool captures live packet data on an interface and then gives detailed protocol information about those packets. It was used in the experiments to examine security problems as well as to learn security protocol internals. Wireshark can do packet filtering and therefore makes it easier to find out the desired packets. In Figure 5-1 is given the different filters for different types of frames [21].

Wireshark 802.11 Display Filter Field Reference

Frame Type/Subtype	Filter
Management frames	wlan.fc.type eq 0
Control frames	wlan.fc.type eq 1
Data frames	wlan.fc.type eq 2
Association request	wlan.fc.type_subtype eq 0
Association response	wlan.fc.type_subtype eq 1
Reassociation request	wlan.fc.type_subtype eq 2
Reassociation response	wlan.fc.type_subtype eq 3
Probe request	wlan.fc.type_subtype eq 4
Probe response	wlan.fc.type_subtype eq 5
Beacon	wlan.fc.type_subtype eq 8
Announcement traffic indication map (ATIM)	wlan.fc.type_subtype eq 9
Disassociate	wlan.fc.type_subtype eq 10
Authentication	wlan.fc.type_subtype eq 11
Deauthentication	wlan.fc.type_subtype eq 12
Action frames	wlan.fc.type_subtype eq 13
Block ACK Request	wlan.fc.type_subtype eq 24
Block ACK	wlan.fc.type_subtype eq 25
Power-Save Poll	wlan.fc.type_subtype eq 26
Request to Send	wlan.fc.type_subtype eq 27
Clear to Send	wlan.fc.type_subtype eq 28
ACK	wlan.fc.type_subtype eq 29
Contention Free Period End	wlan.fc.type_subtype eq 30
Contention Free Period End ACK	wlan.fc.type_subtype eq 31
Data + Contention Free ACK	wlan.fc.type_subtype eq 33
Data + Contention Free Poll	wlan.fc.type_subtype eq 34
Data + Contention Free ACK + Contention Free Poll	wlan.fc.type_subtype eq 35
NULL Data	wlan.fc.type_subtype eq 36
NULL Data + Contention Free ACK	wlan.fc.type_subtype eq 37
NULL Data + Contention Free Poll	wlan.fc.type_subtype eq 38
NULL Data + Contention Free ACK + Contention Free Poll	wlan.fc.type_subtype eq 39
QoS Data	wlan.fc.type_subtype eq 40
QoS Data + Contention Free ACK	wlan.fc.type_subtype eq 41
QoS Data + Contention Free Poll	wlan.fc.type_subtype eq 42
QoS Data + Contention Free ACK + Contention Free Poll	wlan.fc.type_subtype eq 43
NULL QoS Data	wlan.fc.type_subtype eq 44
NULL QoS Data + Contention Free Poll	wlan.fc.type_subtype eq 46
NULL QoS Data + Contention Free ACK + Contention Free Poll	wlan.fc.type_subtype eq 47

Figure 5-1 Filters For different frames in Wireshark[22].

5.2.3 DD-WRT

DD-WRT is a linux based open source firmware for WLAN routers. This firmware supports all current WLAN standards (802.11a/b/g/n) and has a lot of other characteristics including bandwidth management. If we compare the DD-WRT to the software preinstalled on many of the WLAN routers DD-WRT is more reliable and stable [26].

5.2.4 Command Line Tools

Some of the command line tools which were used in the experiments are described with respect to why and how they were used.

Ifconfig

Ifconfig is used for network interface configuration therefore ifconfig stands for interface configurator. It can be used to activate or deactivate the computer's NIC or for changing of

machine's IP address, netmask or broadcast address. It also allows the user to view information about the configured network interfaces.

The ifconfig can be used to set a new MAC address on an interface by first setting the interface down and then setting the new MAC address and then making the interface up again.

When an interface is marked down the system will not transmit messages through that interface and making an interface up enables the interface again.

- ***ifconfig wlan0***
  This command is used to view the network setting on the wireless adapter in the machine.

- ***ifconfig wlan0 down***
  This command causes the interface wlan0 not to transmit or receive any information.

- ***ifconfig wlan0 up***
  This command causes the wlan0 interface to become up again so as to transmit and receive information.

- ***ifconfig eth0 hw ether* 00:66:88:44:23:89**
  The above command is used to set the hardware address of the interface, if the device driver supports this operation. The keyword "hw" is followed by the name of the hardware class "ether" and the printable ASCII equivalent of the hardware address i.e. 00:66:88:44:23:89. Hardware classes currently supported include ether (Ethernet), ax25 (AMPR AX.25), ARCnet and netrom (AMPR NET/ROM).

Macchanger

A GNU/Linux utility for viewing/manipulating the MAC address of the network interfaces by using the following command. This can be done by the ifconfig command but it takes only one command through macchanger to set the mac address.

- **# macchanger --mac=01:23:45:67:89 wlan0**

Iwconfig

Iwconfig is a command line tools which is used to configure wireless interfaces. Iwconfig is similar to ifconfig but it is dedicated to the wireless interfaces. It is used to set the parameters of the network interface that are specific to the wireless operation i.e. frequency, channel etc.

Mode

Using the iwconfig we can set the operating mode of the device. The operating mode could be Adhoc (No access point), Managed (More than one Access point with roaming), Master (Synchronization master or an access point), Repeater (Node forwards packets between other wireless nodes), Secondary (Node acts as a backup master or repeater), Monitor (The node acts as a passive monitor and only receives packets).

In the attack on TKIP, the mode of the wireless interface was set to monitor because in this mode an interface is able to inject and monitor packets without being associated with the access point. This could be done using the following command.

- ***iwconfig wlan0 mode monitor channel* 11**

Where wlan0 is the interface and 11 is the desired wireless channel.

Hping3

Hping3 is command line TCP/IP packet crafter. It can be used to create IP packets with ICMP, TCP and UDP payloads. Hping works on unix-like systems i.e. Linux, FreeBSD, NetBSD, OpenBSD, Solaris, MacOS X and windows. How the packets like TCP, ICMP and UDP are sent is explained in the following text [25].

TCP SYN Scan

The TCP SYN Scan is used to craft TCP packets and it is the default behavior of hping3. The attacker simply has to specify the TCP flags, the target IP address and the destination port in order to craft a TCP packet and consequently the TCP SYN scan is done. This is done by the usage of the following command.

- ***hping3 -S* 192.168.0.1 *-p 80 -c 1***

Where –S indicates that the setting up of the SYN flag, 192.168.0.1 is the IP address of the target machine –p 80 indicates that the TCP port 80 of the target machine is being scanned and –c1 indicates that only one TCP SYN packet is being sent otherwise hping3 will continue sending probes. Thus the above command will scan port 80 on the target machine 192.168.0.1 and the target machine will respond with SYN/ACK packet, which is indicated by the SA return packet. The SA return packet also indicates that the port 80 on the target machine is open otherwise an RA packet is returned.

But if an attacker wants to send TCP-SYN packets on multiple ports to find out which ports are open on the target machine, it could use the following command.

- ***hping3 -S* 192.168.0.1 *-p ++80***

This command will send TCP-SYN packets on multiple ports starting from port 80 and incrementing for each subsequent TCP-SYN packet. In order for an attacker to go through all the responses to see which of the ports are open is difficult so in order to overcome this, an attacker could use "grep" so that interesting responses could be displayed only. The usage of grep is shown in the following command.

- ***hping3 –S* 192.168.0.1 *-p ++50 | grep SA***

grep SA means that the attacker is interested in only knowing which of the ports are open.
We have defined how hping3 can be used to send TCP packets and to see which of the ports on a target machine are open. This utility will be used in my experiments to send TCP packets on a wireless network because in this way an attacker can recover a keystream.

ICMP packets

Hping3 can be used to send the ICMP packets. ICMP protocol is used to send the ping request to see if a host is alive or not. This could be done by using the following command.

- ***hping3 -1* 192.168.0.1**

Where -1, numerical value shows that the packet is a ICMP echo request.

Hping3 can be used to craft packets in such a way that an attacker can also specify the payload bytes in the packet. This could be done by the following command.

- ***hping3 -S* 192.168.0.1 *-d 80 -p 80 -E /home/ammar***

Where –d 80 specifies the length of the payload which is 80 bytes and –E /home/ammar specifies that the payload bytes are taken from the file "ammar".

Chapter 6

6 EXPERIMENTS

6.1 Methodology

To start with, I had very little idea how this attack worked and which practical issues could be issued while carrying out the experiments. So while carrying out the experiments I got more and more knowledge about how to encounter various issues in order to accomplish the objectives of the experiments. Therefore an iterative methodology was employed during the experiments. The previous carried out work done was on a QoS enabled network but I tested the attack on a QOS enabled client associated with a non-QoS AP. The idea was taken from [17] and by the time of writing this thesis the manuscript is yet to be published.

6.2 Preparation

In order to conduct the Beck and Tews attack with a non QOS AP associated with a QOS client the following procedure was adopted.

Step 1

First a virtual interface was created using airmon-ng. This interface was created for packet sniffing and injection during the attack. The interface was created using the following command.

- ***airmon-ng start wlan0* 11**

Where 11 is the channel which is set on the Access point and wlan0 is the physical interface.As a result of this command a virtual interface mon0 is created in the monitor mode. Monitor mode is a special mode used to sniff all the packets in the wireless network. It is also used for packet injection.

Step 2

Secondly the already created virtual interface MAC address was set to the MAC address of the client being attacked. This was achieved by using the following commands.

- ***ifconfig mon0 down.***
- ***Macchanger –m* 00:22:EA:2A:39 *mon0***
- ***Ifconfig mon0 up.***

Where 00:22: EA: 2A:39 is the MAC address of the client station being attacked.

6.3 Beck and Tews Attack with QoS Client Associated with Non_QoS AP

The original Beck and Tews attack was added to the Aircrack-ng suite as a proof of concept in the November 2008.The proof of concept is named the tkiptun-ng. The experiment was conducted with the tool which is still in the development phase. The tool was in revision SVN 1673 when the experiments were conducted. A patch was applied to the original source code and the attack with QOS client associated with the NON-QoS AP was successfully conducted.

The attack was executed with the following command,

- ***tkiptun-ng –x 50 –h* 00:22:57:EA:2A:39 *–a* 00:1E:E5:64:F1:15 *mon0***

Where –x specifies the injection rate of packet which is 50 packets per second, –h specifies the STA MAC address and –a the BSSID of the network

6.4 TCP Attack with Linux Access Point

As point out by Martin Beck in his latest unpublished paper “Enhanced TKIP Michael attacks” TCP attacks is a very clever way of getting more usable keystreams when the access point is running Linux. The attack has already been discussed in Sec. 4.3.2.It is to be noted that the attack was run just to validate the theory behind the attack. The keystream was never obtained.

6.4.1 Building a Linux Access Point

In order to validate the theory of the TCP attack which is part of the Enhanced TKIP Michael Attack, I build a Linux access point. Initially the idea was to build an access point on of the Linux STA in the Lab using hostapd. But after doing all the necessary work of taking the wireless card in the Master mode and making the necessary changes in the hostapd.conf and setting up a Bridge it was found that the AP was unstable. The problem was that on the client machine the BSSID of the built access point could be seen for a moment and then disappeared.

After doing some research it was found out that the wireless card with Atheros chipset and madwifi driver used in the experiment was unstable to be used with hostapd. The wireless card in my case was AR5413.

Therefore the DD_WRT which is a linux based firmware was installed on the Linksys WRT 54GL wireless router. The firmware was successfully flashed on the Linksys router with the necessary precautions for instance hard resetting before and after the firmware installation taken. The file version installed was mini_generic.bin file. It is to be noted that DD_WRT firmware installation is very simple but incorrect installation without taking the necessary precautions such as hard reset and incorrect file installation could hard brick the router.

6.4.2 Preparation for the Attack

It is important to mention that the purpose of the experiment was to validate the theoretical background of the TCP attack by Martin Beck and the keystream was never obtained through this experiment.

Step 1

In order to run the TCP attack, it is to be found out what which of the ports are open on the client machine. In order to find this out a TCP Scan was done by the following command.

- ***hping3 -S* 192.168.1.123 *-p ++1 | grep SA***

The port numbers 135,139 and 445 were the ports which were open. In the above command grep SA indicates that only those replies with flag SA set are showed. It is to be noted that the firewall on the client machine was disabled and no applications or services were run on the client machine which is not a real world scenario. As a matter of fact in real life there are many services running on a machine such as application and gaming, common servers like telnet and web so an attacker could easily find one of these ports open to launch a TCP attack.

Step 2

The TCK attack was run using the packet crafting tool Hping3. The following command was used.

hping3 -a* 192.168.1.1 *–S* 192.168.1.123 *–p 135

Where –a is the spoofed IP address of the Linux based AP and –S in the TCP SYN packet sent to the target machine with IP address 192.168.1.123 and –p 135 indicates that the packet was sent to the port 135 of the target machine.

6.5 Michael Reset Attack

The Michael Reset attacks described in theory by Martin Beck uses the ICMP echo messages to decrypt a captured packet as already described in chapter 4. This methodology by Martin Beck

uses the wireless LAN also having a connection to the WAN port of the attacker. Moreover it also requires the calculation of the magic words to reset the internal state of the Michael algorithm [1]. There is a need to be setting up of a bridge on the attacker machine because he has to get the decrypted captured packet on the WAN port and not the WLAN port.

Being said all that the setting up of bridge is simple but the calculation of the magic words to reset the internal state of the Michael algorithm is a hard task. Therefore the following approach in the form of ICMP NAT Transversal attack was adopted.

6.5.1 ICMP NAT Transversal Attack

I think that it is easier to get a keystream through a modified form of the Michael Reset attack in which one does not have to concatenate the captured packet with the ICMP echo request. In this method an ICMP echo is sent from a machine on the internet and the encrypted response is obtained on the TKIP wlan and then finally obtaining the keystream.The following command was used to send an ICMP echo request and port forwarding was enabled to let through the packet.

- ***hping3* 192.168.0.1 *-d 1460 -p 80 -E /home/ammar***

where 192.168.0.1 is the IP address of the Linksys router, 1460 is length of the payload of the packet and it was selected to set the maximum size i.e. MTU which is 1500 bytes and ammar is the filename with a written payload.

Chapter 7

7 RESULTS

This chapter deals with the results obtained from the experiments.

7.1 Verification of Beck and Tews Attack with QoS Client Associated with Non_QoS AP

The Beck and Tews attack with a client associated with NON_QOS AP was verified .For this attack to work one has to understand the theory of the Basic Beck and Tews attack as well as how the QOS data frames are constructed. The theory is explained in detail in the Background chapter. The tool is in still in development but it worked and obtained a keystream and MIC and then injected an encrypted ARP packet to the client machine. A successful completion of the attack is shown in the following figure.

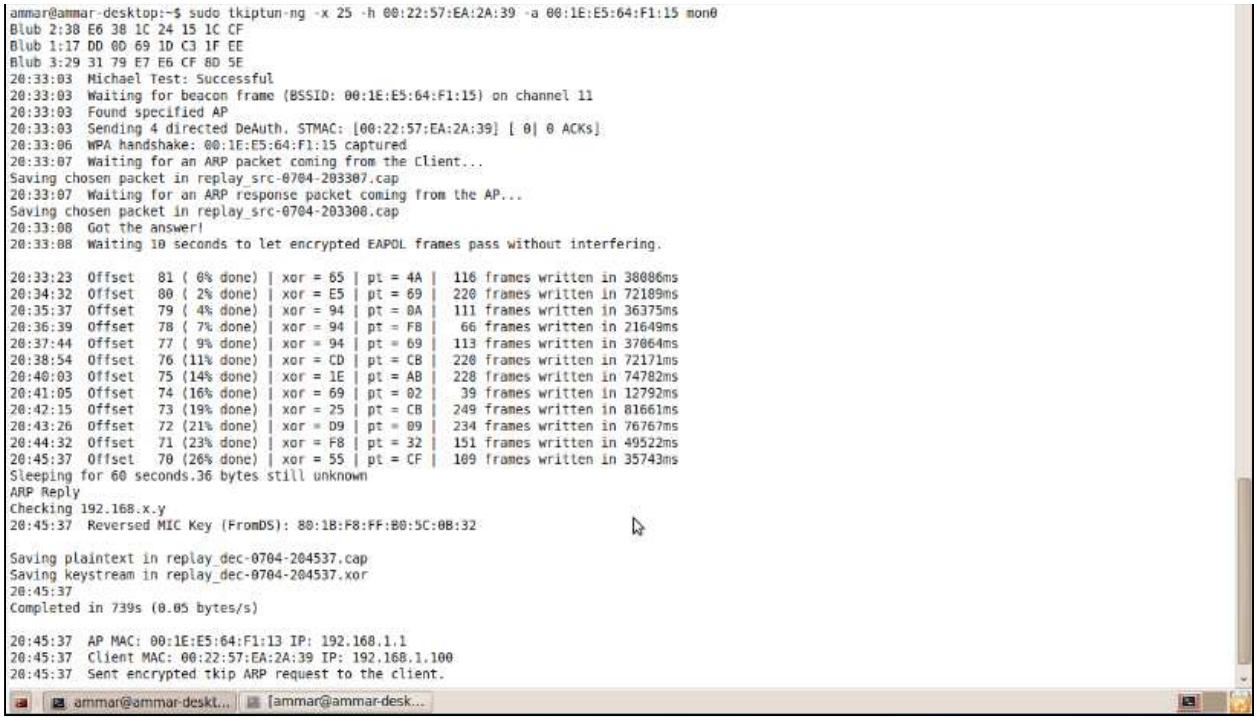

Figure 7-1 Beck and Tews Attack on a Non-QOS AP

The attack was executed with an injection rate of 25 packets per second while the implementation's default injection rate is 10 packets per second. Thus with the default injection

rate, the whole attack would take 13 minutes and 34 seconds. This is because the number of bytes to guess by chop chop attack is 12 bytes. Averages of 128 guesses per byte are done in the attack which means that the total numbers of packets sent are 1536 packets. Thus 153.6 seconds are taken to send 1536 packets. There is an interval of 60 seconds after each correct guess to avoid countermeasures and there are 11 such intervals. Therefore the total time of the attack is 13 minutes and 34 second without including the time for initialization.

In addition to it there is a time in which the tool initializes with the some of the MIC failures which are not detected. As a result of these MIC failures the tool waits for an additional minute for each MIC failure. Consequently the total time for the attack is 15 to 20 minutes. With an injection rate of 25 packets per second the attack would take 12 minutes and 2 seconds without including the initialization time.

The attack was also conducted with an injection rate of 100 for which the time taken by the attack to complete was 11 minutes and 16 seconds as shown in the following figure.

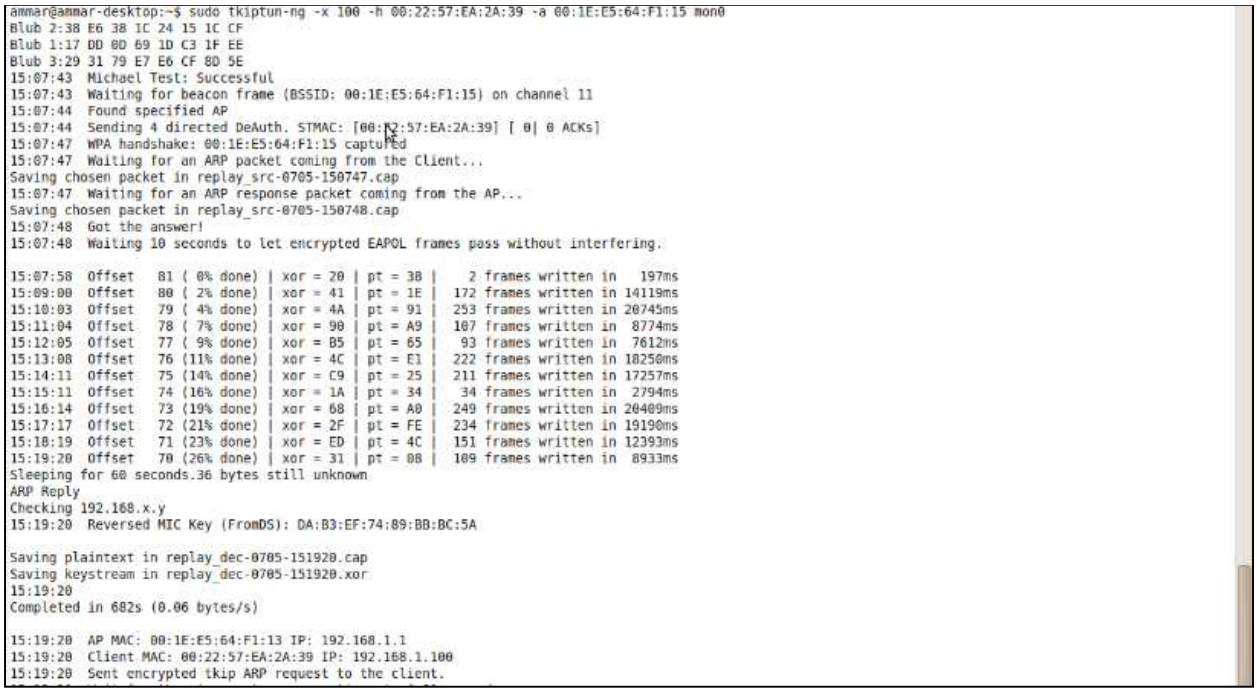

Figure 7-2 Attack with Injection Rate of 100 packets/second

Finally when the attack was conducted with an injection rate of 150 packets per minute the attack could not be completed successfully as shown in Figure 7-3.

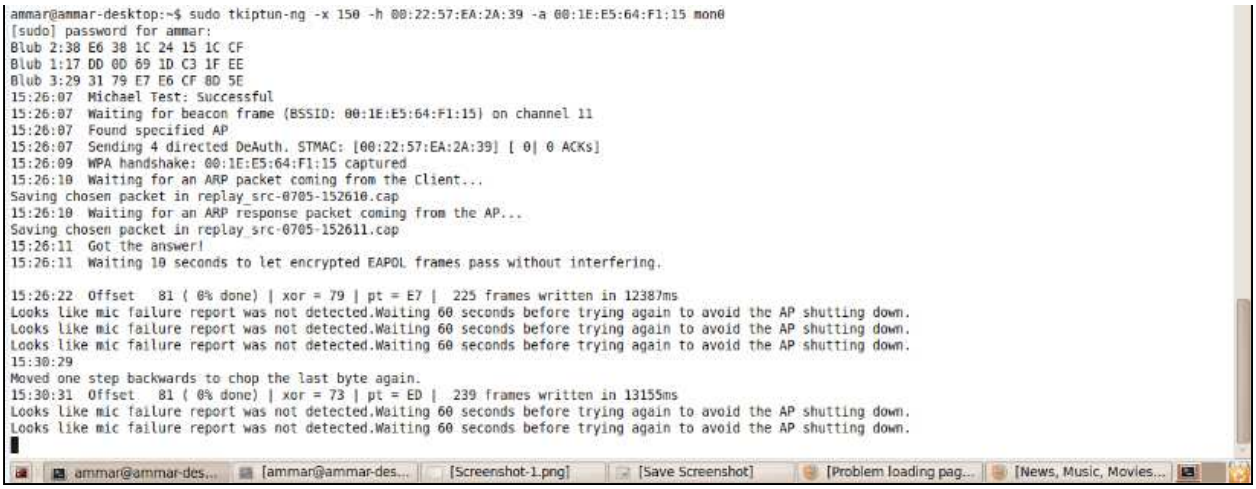

Figure 7-3 Attack with injection rate of 150 packets/second.

It is to be noted that the MIC failure report is sent in an EAPOL-key frame. The EAPOL key frames are used to exchange cryptographic keys .This is done through the Group Key handshake and the4 way handshake. The IEEE standard mentions that a Group key handshake to update the Group Temporal key occurs when an event occurs in the STA's station management entity (SME) [9]. Here in this case a MIC failure occurs because the guess is correct.

As shown in figure 7.4, the EAPOL-Key frame consists of a field called key information. This field contains Key MIC bit which is set as shown in the figure indicating that this EAPOL key frame contains the MIC. It also contains the error bit which is also set as shown in Figure 7-4 and indicates that a MIC failure occurred in the TKIP MSDU.

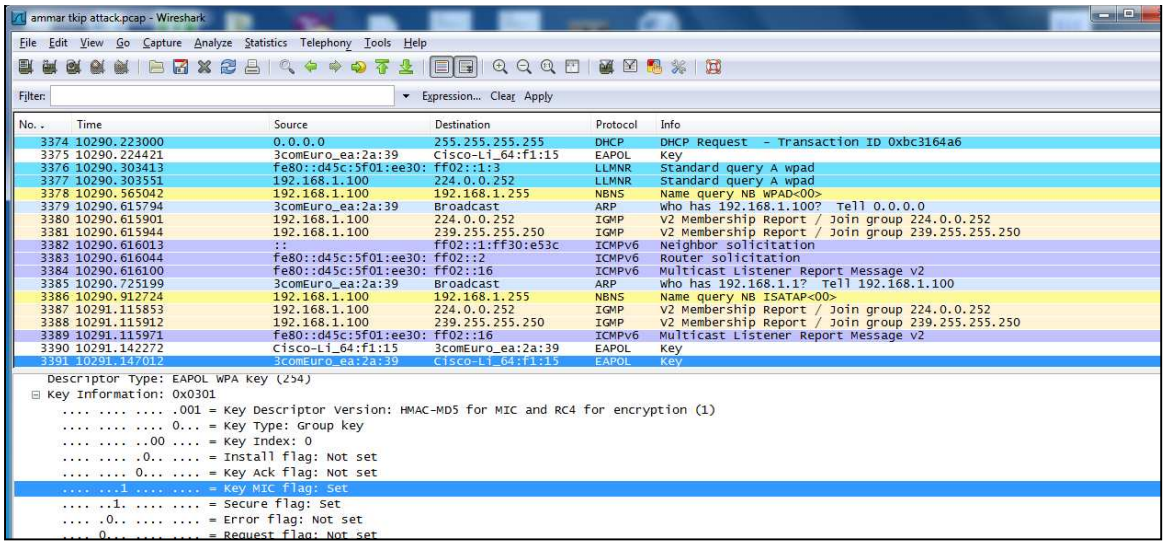

Figure 7-4 Wireshark capture showing MIC failure.

7.2 TCP Attack with Linux Access Point

The theoretical background related to the TCP Attack with Linux Access point was validated. As according to Martin Beck, when a TCP-SYN packet is sent to the client machine's open port with the spoofed IP address of the Linux access point, the Access point sends the TCP-RST packet.

All the values of the fields in the 20 byte TCP and 20 byte IP header in the TCP-RST were the same as pointed out by Martin beck in his paper "Enhanced TKIP Attacks [1]. Furthermore the experiment to validate it was also conducted with a Netgear Wireless router MR814v2 with the factory set firmware and it was found out that the IPID field in this case was not zero and was different for every segment. Hence it is obvious that usage of the Linux access point enables an attacker to know the two extra keystream bytes in the form of identification field in the IP header. The wireshark capture of the attack is shown below.

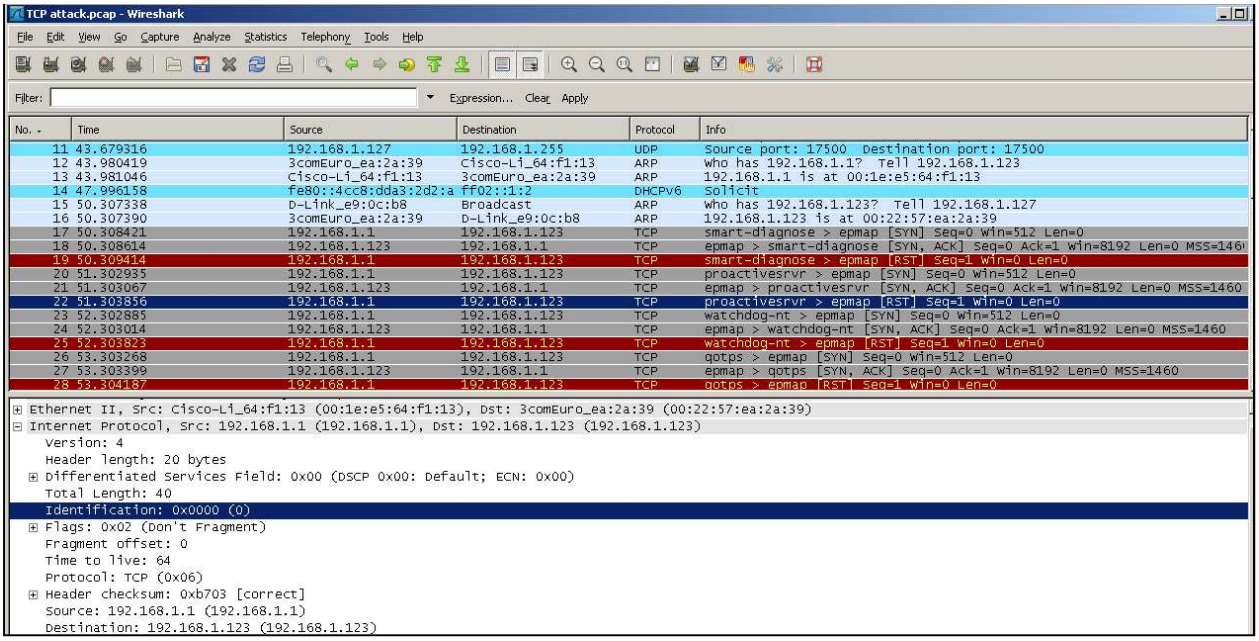

Figure 7-5 TCP Attack showing IPID field zero.

7.3 ICMP NAT Transversal Attack

The ICMP ping request could not transverse the router although the NAT was disabled. The reasons behind not achieving the desired results are explained in the Chapter 8.

Chapter 8

8 DISCUSSION

This chapter deals with the discussion about the problems which arose during the working of the thesis and how they were dealt with. It also gives an overview of the approach which could be adopted to overcome some obstacles which could not be removed. Finally it gives a direction which could be followed in future regarding working on TKIP.

The problem description comprised of understanding the vulnerability in TKIP and experimentally validating the extended attack [16]. Meanwhile the attack of obtaining keystreams in a new way using TCP SYN packets, proposed by Martin Beck [1] prompted me to explore this new idea.

8.1 Attack against Windows 7 Machine

The experimentation in the thesis started with a windows 7 machine configured as a client on the *TKIP wlan* and the intention was to launch the Basic Beck and Tews attack against this machine with the most recent SVN development at that time. But the attack could not work and it was noted that after the client is deaunthenticated, the attacker machine looks for an ARP packet and reads packets coming from the client indefinitely and hence the chopping part of the attack could not start. This Basic attack though worked against a Linux machine. To successfully run this attack against a windows machine is an area which could be looked upon in future.

8.2 Network Manager

During the experiments an iterative approach was adopted. I took up this work as a challenging task with very little knowledge of working on Linux machine and C programming. Furthermore I had some knowledge of wireless security but it was inadequate. So I started and had to face some problems during the work. During the experimentation it was observed that Basic Beck and Tews attack did not completed successfully. Apparently everything was configured properly but it was not until the Network Manager was disabled only then the attacks success rate increased.

8.3 High Injection Rate

The attack when the network is configured with a QoS client associated with a QoS AP was tried with different injection rates just to test how much minimum duration the attack could take. It was found that the minimum duration the attack took was 11 minutes, 16 seconds with an injection rate of 100 packets per second. This was also observed that when the injection was too high i.e. 150 packets per second the attack could not complete. This is because with an injection rate too high the MIC failures from the client station are detected at a later time and for a byte guessed incorrectly so the attack did not proceed further.

8.4 TCP Attack with Linux Access Point and Problems with AR5413

The Enhanced TKIP Michael attacks [1] were explained in chapter 4. They were analyzed and some of the theory behind the attack was verified through experimentation. Martin Beck says that it is possible to guess the whole TCP-RST packet in response to a TCP-SYN packet sent to a client's open TCP port with source IP address spoofed to the IP of a Linux access point by the attacker. During the experiments a Linux access point was configured with dd-wrt and it was found that one is able to get the two extra keystream bytes in the form of IPID field the TCP-RST packet as claimed by Martin Beck. Initially I wanted to configure a Linux station as an access point using hostapd but after all the necessary configuration steps I observed that the ESSID on the supplicant appeared for a moment only. It appears that the machine running hostapd get stuck beacons although the startup log of hostapd is not reporting errors. As I tried to reconfigure the systems again the problem persisted. It looks as if the AR5413 Atheros chipset is broken [27]. Finally I installed dd-wrt which is Linux based firmware on the flushing the old firmware on the Linksys-WRT54GL router[26] and I was able to verify the IPID field of the TCP-RST packet was zero.

8.5 Port Forwarding and ICMP Echo Request

The Michael reset attacks [1] is a difficult attack to implement as firstly it requires the capturing of a packet then its concatenation with ICMP echo request together with the two magic words [1] to reset the Michael algorithm so that the MIC for the captured packet is valid for the whole new packet. This is very clever technique to decrypt any captured packet but the downside of this its implementation. I think and as also pointed out in the last two chapters a modified form of this

attack and which is easy is the sending of just the ICMP echo request to a TKIP wlan and then capture the encrypted ICMP echo request packet sent by the AP so as to obtain the keystream. As during the experiments I found out that ICMP echo could not be sent from outside to a local wlan. So I configured the access point so that it is exposed to all the internet traffic from outside by turning off NAT or enabling DMZ but I still was not able to achieve the results. Further research into the issue that ICMP packets cannot pass through NAT [29].The reason is that ICMP is buried deep into the stack and to forward an ICMP packet one has to recompile the kernel [29].It is to be noted that turning off NAT enables TCP and UDP traffic to pass through which was also observed during the experiments. This could be another way of obtaining a keystream by sending TCP SYN packets to an internal machine on TKIP wlan and then getting an encrypted TCP-ACK packet on a wlan port of the attacker machine.

During the experiments the windows 7 firewall was disabled in order to send a TCP-SYN packet to an open port incase of a TCP attack. This is not a real world scenario. Infact nowadays because of many users interested in multiplayer gaming and file transfers from a remote machine; it is quite possible that one can find some of the ports open.

Finally to maintain such a Lab environment with different operating systems and firmware on AP, it has been quite challenging and at times exhausting. But it has been indeed a great learning experience

8.6 Further Work

8.6.1 Metasploit Framework

As explained in Sec 3.4 with the Practical TKIP attack example, after successfully obtaining a AP-STA keystream from the Beck and Tews attack, an attacker sends the TCP-SYN packets targeting the well known ports on a target machine on each of the 7 channels.

The machine responds with the TCP-ACK packet to a machine on the internet because the IP address in the TCP-SYN packets is spoofed. From there onwards the attacker agent machine present on the internet completes the three way handshake.

The metasploit framework could be used to send exploits to the target machine. This framework contains chunks of code written in Ruby language which could be reused as exploits. I think it’s

a very fast and quite interesting to use this framework to find vulnerabities in an operating system of a target machine. [30].

8.6.2 Attack on Networks supporting TKIP and CCMP

The extended attack [16] used the DHCP ACK packet to obtain a keystream. According to [9] in networks using CCMP as a protocol to encrypt unicast traffic, TKIP is used to encrypt broadcast or multicast traffic when both TKIP and CCMP are supported in such networks. We also know that DHCP offer message is one of the broadcast messages in DHCP message exchange therefore the extended attack [16] can be used against the DHCP offer message in such networks where TKIP and CCMP are both supported and CCMP is used to encrypt unicast traffic. This could be one of areas which could be explored and experimentally validated.

8.6.3 Enhanced Michael TKIP attacks

Due to time limitations an analysis of the enhanced TKIP Michael attacks was done. A proof of concept has not yet been developed. This could be implemented as a generic code and the TKIP attacks would be extended in practice.

Chapter 9

9 CONCLUSION

Despite the security concerns regarding wireless technology it is growing in development and utilization. To have a knowledge of the threats which the wireless networks are facing nowadays one needs to have a comprehensive knowledge of the IEEE 802.11 standards.

The thesis started with an insight into the vulnerability with TKIP. In this regard the WEP failure to provide the adequate security was discussed. The CRC-32 algorithm which is an algorithm for computing ICV was also discussed and the flaw in it which makes an invalid message into a valid one was also looked upon.

The original beck and Tews attack together with Beck and Tews attack on a network with QoS client associated with a Non-QoS AP was analyzed and experimentally validated. The most recent attack methodology [1] which uses TCP-SYN and ICMP packets to obtain keystreams were thoroughly analyzed. The verification of the claims made by Martin Beck in case of a TCP attack were validated.

There were some problems which were encountered during the thesis i.e. ICMP echo requests not transversing through the NAT and the issues with the AR5413 while running hostapd. Finally a future course of action regarding attacks on TKIP has been outlined.

TKIP was developed as a secure alternative to WEP but it is no more secure now. The vulnerability found in TKIP by Martin Beck and Erik Tews has already paved the way to more extended attacks on TKIP.It seems that it will not be too long for the experts to find flaws in the most secure wireless security protocol CCMP.

APPENDIX

The patch to make the Basic Beck and Tews attack work on a network with QoS client associated with a Non-QoS AP is shown below.

The command used to patch the tkiptun-ng.c file is given below,

- patch -p0 < tkiptun-ng.c.patch

```
--- tkiptun-ng.c.orig2010-03-24 19:59:21.316729507 +0100
+++ tkiptun-ng.c     2010-03-24 21:07:03.538475994 +0100
@@ -1172,6 +1172,23 @@

        nb_pkt_read++;

+       // z = MAC header length
+       z = ( ( h80211[1] & 3 ) != 3 ) ? 24 : 30;
+
+       if( ( h80211[0] & 0x80 ) != 0x80 )
+       {
+         // Create a QoS frame from a non-QoS frame
+         // QoS data
+         h80211[0] |= 0x80;
+         // Move frame body 2 bytes to the right to make room for QoS control field
+         for(i=*caplen+1; i>z+1; i--)
+         h80211[i] = h80211[i-2];
+         // Add 2 bytes QoS control field
+         *caplen += 2;
+         h80211[24] = 0x00;
+         h80211[25] = 0x00;
+       }
+
        if( filter_packet( h80211, *caplen ) != 0 )
            continue;
```

REFERENCES

[1] B. Martin (2010), Enhanced TKIP Michael Attacks http://download.aircrack-ng.org/wiki-files/doc/enhanced_tkip_michael.pdf Last accessed July 1st, 2010.

[2]G. Matthew, (2002).The 802.11 MAC In Wireless Networks: The Definitive Guide. O'Reilly.

[3]Javvin Technologies, SSID: Service set Identifier. www.networkdictionary.com/wireless/s.php Last accessed May 1st, 2010.

[4]W.N.Ross, (1993), A Painless Guide to CRC Error Detection Algorithms. http://www.ross.net/crc/download/crc_v3.txt Last accessed May 2nd, 2010.

[5]W.Jesse, (2001), Overview of 802.11 Security. http://www.ieee802.org/dated Last accessed May 10th, 2010.

[6]F.Seth (2006), Byte Sized Decryption of WEP with Chop Chop http://www.informit.com/guides/content.aspx?g=security&seqNum=196 Last accessed May 10th, 2010.

[7]Korek (2004), Chop Chop (Experimental WEP attacks) http://www.netstumbler.org/f50/chopchop-experimental-wep-attacks-12489/ Last accessed May 15th, 2010.

[8]B.Andrew, S.Krishna (2005), Key Mixing Algorithm In Cisco Wireless LAN Security.Cisco Press.

[9]IEEE Std 802.11-2007 Information Technology-telecommunications and Information Technology and Information exchange between Systems and Local and Metropolitan Area Networks-specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.

[10]B.Andrew, S.Krishna (2005), TKIP Encapsulation & TKIP Decapsulation In Cisco Wireless LAN Security.Cisco Press.Cisco Press.

[11] IEEE Std 802.11i-2004 Information Technology-telecommunications and Information Technology and Information exchange between Systems and Local and Metropolitan Area Networks-specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Amendment 6: Media Access Control (MAC) and Security Enhancements

[12] IEEE Std 802.11e-2005 Information Technology-telecommunications and Information Technology and Information exchange between Systems and Local and Metropolitan Area Networks-specific Requirements-Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Amendment 6: Media Access Control (MAC) and Quality of Service Enhancements.

[13]. ARP Message Format . http://www.tcpipguide.com/free/t_ARPMessageFormat.htm Last accessed May 29, 2010.

[14]J.Postel, RFC 792-Internet Control Message Protocol. http://www.faqs.org/rfcs/rfc792.html.1982.

[15] RFC 793-Transmission Control Protocol. http://www.faqs.org/rfcs/rfc793.html, 1981.

[16] H.M Finn, H.Olav (2009), Thesis on Cryptanalysis of IEEE 802.11i TKIP.NTNU, Trondheim, Norway. http://dl.aircrack-ng.org/breakingwepandwpa.pdf

[17] E.Martin (2010), .A Practical Cryptographic Denial of Service Attack Against 802.11i TKIP and CCMP, Manuscript not yet published.

[18]R.Droms (1997), RFC2131-Dynamic Host Configuration Protocol. http://www.faqs.org/rfcs/rfc2131.html

[19] Morrii & Ohigashi (2009), A Practical Message Falsification Attack on TKIP http://jwis2009.nsysu.edu.tw/location/paper/A%20Practical%20Message%20Falsification%20Attack%20on%20WPA.pdf

[20]Aircrack-ng Project. Latest SVN Development Sources. http://www.aircrack-ng.org/doku.php?id=install_aircrack#installing_aircrack-ng_from_source Last accessed June 10th 2010.

[21] Wireshark Project. http://www.wireshark.org/about.html. Last accessed June 10th 2010.

[22]SANS Institute, Wireshark 802.11 Display Filter Field Reference http://www.willhackforsushi.com/papers/80211_Pocket_Reference_Guide.pdf

[23]Barnes, Douglas (2002). Network America: Wireless security? Read it and WEP. http://www.vnunet.com/Features/1133066 Last accessed June 27th, 2010.

[24] B.Martin, T.Erik (2008). Practical attacks against WEP and WPA. http://dl.aircrack-ng.org/breakingwepandwpa.pdf Last accessed March 10th, 2010.

[25]Hping team, Active Network Security tool. http://www.hping.org/ Last accessed May 10th 2010.

[26]. dd-wrt community. Dd-wrt installation .http://www.dd-wrt.com/site/index.Last accessed July 1st, 2010.

[27] Madwifi project. Ticket No. 2253 (new defect). http://madwifi-project.org/ticket/2253

[28]Mitchell .B, DMZ – demilitarized Zone.http://compnetworking.about.com/cs/networksecurity/g/bldef_dmz.htm

[29]Linux forums. NAT does not forward ICMP.http://www.linuxquestions.org/questions/linux-newbie-8/how-to-configure-linksys-wrt54gl-router-to-forward-icmp-echo-requests-663415/

[30]Metasploit Team, Metasploit 3.4 Developer's Guide Chapter 01: Introduction,http://www.metasploit.com/redmine/projects/framework/wiki/DeveloperGuide_01