

Electronic Theses and Dissertations, 2004-2019

2004

Design And Hardware Implementation Of A Novel Scrambling Security Algorithm For Robust Wireless Local Area Networks

Mohit Jagetia
University of Central Florida

 Part of the [Electrical and Computer Engineering Commons](#)
Find similar works at: <https://stars.library.ucf.edu/etd>
University of Central Florida Libraries <http://library.ucf.edu>

This Masters Thesis (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations, 2004-2019 by an authorized administrator of STARS. For more information, please contact STARS@ucf.edu.

STARS Citation

Jagetia, Mohit, "Design And Hardware Implementation Of A Novel Scrambling Security Algorithm For Robust Wireless Local Area Networks" (2004). *Electronic Theses and Dissertations, 2004-2019*. 98.
<https://stars.library.ucf.edu/etd/98>

DESIGN AND HARDWARE IMPLEMENTATION OF A NOVEL SCRAMBLING
SECURITY ALGORITHM FOR ROBUST WIRELESS LOCAL AREA NETWORKS

by

MOHIT JAGETIA
B.E. P.V.P.I.T., India, 2000

A thesis submitted in partial fulfillment of the requirements
for the degree of Master of Science
in the Department of Electrical and Computer Engineering
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2004

Major Advisor: Dr. Taskin Kocak

ABSTRACT

The IEEE802.11 standard for wireless networks includes a Wired Equivalent Privacy (WEP) protocol, which is a popular wireless secure communication stream cipher protocol approach to network security used to protect link-layer communications from eavesdropping and other attacks. It allows user to communicate with the user; sharing the public key over a network. It provides authentication and encrypted communications over unsecured channels. However, WEP protocol has an inherent security flaw. It is vulnerable to the various attacks, various experiments has proved that WEP fails to achieve its security goals. This thesis entails designing, evaluating and prototyping a wireless security infrastructure that can be used with the WEP protocol optionally, thus reducing the security vulnerabilities. We have studied the flaws of WEP and the reasons for their occurrence, and we provide the design and implementation of a novel scheme in Matlab and VHDL to improve the security of WEP in all aspects by a degree of 1000.

The architecture was designed with a consideration for least increment in hardware, thus achieving power and cost efficiency. It also provides flexibility for optional implementation with the available technology by being able to be bypassed by the technology, which allows for non-replacement of existing hardware, common on both, the WEP and the proposed protocols, on the fly.

ACKNOWLEDGMENTS

I am honored to have had the opportunity to work with Dr. Taskin Kocak as my thesis advisor. Without his technical and personal advice, this research work would not have been successful. I am very grateful to my committee members, Dr. Mainak Chatterjee and Dr. Mark Heinrich for their support and time in serving as committee members. I am very grateful to all UCF faculty and staff for their time. I would also like to thank Mr. Faysal Bascci for helping me while developing various attacks to WEP. Finally, I am grateful to Akarsh Ravi for his help towards preparing the documentation for my thesis.

TABLE OF CONTENTS

LIST OF FIGURES	vi
LIST OF TABLES	viii
LIST OF ABBREVIATIONS	ix
INTRODUCTION	1
Wireless Communications	3
Security Issue	4
Open System	5
Digital Signature	5
Challenge-response test	6
Shared Key Authentication	6
Cryptography	7
General Cryptography techniques	9
Symmetric and Asymmetric key cryptography	10
Motivation and Problem Statement	11
Related Work	12
WIRED EQUIVALENT PRIVACY (WEP)	14
Shared key identification system	15
Block Diagram	16
Features of WEP Algorithm	18
Design Specification	18
CRC32	19
Algorithm	19
RC4	20
Algorithm	21
The WEP PRNG	22
Cipher Engine	23
Control Logic and Data Path	23
Implementation	23
CRC32	23
Concatenater	26
Parallel to Serial Converter	27
Random Generator	28

Attacks	29
NOVEL SCRAMBLING ALGORITHM.....	31
Algorithm 1: IV and ICV randomizations / extraction	32
Algorithm.....	32
Algorithm 2: Algorithm for cipher randomization / extraction	32
Algorithm.....	33
Scrambling	34
De-scrambling.....	35
Implementation	35
Simulation Results	38
IV Randomization	38
Cipher Randomization	42
CRC Randomization	45
Analysis.....	48
Algorithm 1	48
Algorithm 2	48
Smart attack	53
Algorithm 1	53
Algorithm 2	55
CONCLUSIONS AND FUTURE WORK.....	57
LIST OF REFERENCES.....	58

LIST OF FIGURES

Figure 1: Encryption Decryption	9
Figure 2: IEEE 802.11 Station Authentication	16
Figure 3: WEP Block Diagram	17
Figure 4: generation of ICV using CRC32 algorithm.....	25
Figure 5: Concatenater.....	26
Figure 6: Parallel to Serial Converter	27
Figure 7: Random Generator	29
Figure 8: Architecture of the implemented patching algorithm.....	37
Figure 9: IV Randomizing	39
Figure 10: IV Randomizing showing insertion of random octet	40
Figure 11: a) IV a) before b) after application of Algorithm 1: IV and ICV randomizations / extraction.....	41
Figure 12: Cipher Randomization showing different states	42
Figure 13: Cipher-text randomization showing random insertion positions	43
Figure 14: Cipher-text randomization showing random insertion.....	43
Figure 15: Cipher Randomizing (idle, initialize, insert, read states)	44
Figure 16: Cipher Randomizing a) before b) after application of Algorithm 2: Algorithm for cipher randomization / extraction	45
Figure 17: CRC Randomizing	46
Figure 18: Zoomed view showing the IDLE, READ_PORT, SEARCH state transitions	47
Figure 19: CRC Randomizing	47

Figure 20: Packet formats for the WEP and modified scrambled WEP by the scrambling algorithm.....	48
Figure 21: Density of random octets.....	50
Figure 22: The number of octets in the modified packet Vs the number of incoming octets.....	51
Figure 23: PSR versus n.....	52
Figure 24: Computational Complexity and Chunk position.....	53

LIST OF TABLES

Table 1: Chunk specification and randomization of binary values in chunk.....	36
---	----

LIST OF ABBREVIATIONS

n: number of bytes (octets) received as WEP cipher-text

N: number of bytes (octets) result of application of SA

CS: chunk size

CP: chunk position

RO: number of random octets

dRO: density of random octets

INTRODUCTION

Internet enabled wireless devices continue to proliferate and are expected to surpass traditional Internet clients in the near future. Wireless technology has become an integral part of today's life. The use of wireless networking is rapidly rising with an ever-increasing need for businesses to cut costs and to provide mobility to workers. Wireless technology has spread to devices from small-embedded systems to large general purpose PCs. This is due to cheaper prices, faster speeds and also due to the need for greater mobility. However, data security and privacy remain major concerns in the current generation of "wireless Web" offerings; it is desirable to have as much data privacy as possible. If this data is threatened by phenomenon such as hacking, being accessed by unauthorized individuals, stolen or attacked by viruses, disaster is bound to occur. Thus data security is a major issue in communications. Hence in today's networked world security is at a premium. All such offerings today use a security architecture that lacks security.

Wireless network security is very essential, as it is not bound to any region. Any unauthorized person can read, modify or use the private data being transmitted over a network. As wireless platforms mature, grow in popularity, and store valuable information, hackers are stepping up their attacks on such wireless targets. This is a problem to be considered because wireless devices, including smart cellular phones and personal digital assistants (PDAs) with Internet access, were not originally designed with security as a top priority. Now, however, wireless security is becoming an important area of product research and development. As in the wired world, wireless security boils down to protecting information and preventing unauthorized system access. However, it is challenging to implement security in small-footprint devices with

low processing power and small memory capacities and these use unreliable, low bandwidth wireless networks.

IEEE 802.11 Wireless Fidelity (Wi-Fi) standard [1, 48, 52, 53] refers to a family of specifications developed by the IEEE for wireless LAN technology. It specifies an over-the-air interface between a wireless client and a base station or between two wireless clients. The need for the 802.11 standard came due to the emergence of various proprietary wireless systems, which were incapable of interoperation. The standard becomes popular due to high data rates, fast and easy encryption techniques. It modeled on the ISO's OSI Model [5, 6, 7]; but the standard is only concerned with the physical layer and the lower part of the Data Link layer (the MAC (Medium Access Control [8, 9, 10, 11, 12] sub-layer). This architecture uses fixed network access points with which mobile nodes can communicate. These network access points are sometime connected to landlines to widen the LAN's (Local Area Network's) capability by bridging wireless nodes to other wired nodes. It is designed to provide a wireless local area network (WLAN) with a level of security and privacy comparable to what is usually expected of a wired LAN, using Wired Equivalent Privacy (WEP) [1, 47, 48, 52, 53] security protocol. WEP is a popular wireless secure communication stream cipher protocol approaches to network security used to protect link-layer communications from eavesdropping and other attacks. It allows users to communicate with other users sharing the public key over a network. It provides authentication and encrypted communications over unsecured channels. WEP seeks to establish similar protection to that offered by the wired network's physical security measures by encrypting data transmitted over the WLAN. Data encryption protects the vulnerable wireless link between clients and access points; once this measure has been taken, other typical LAN security mechanisms such as password protection, end-to-end encryption, virtual private

networks (VPNs) [13, 14, 15], and authentication can be put in place to ensure privacy. WEP is used to protect link-layer communications from eavesdropping and other attacks. However, it has an inherent security flaw. It is vulnerable to various attacks; experiments have proved that WEP fails to achieve its security goals. These serious security flaws have been discovered in the protocol, stemming from misapplication of cryptographic primitives. The flaws lead to a number of practical attacks that demonstrate that WEP fails to achieve its security goals. For example, a WEP network can be made vulnerable by simply, passively gathering and analyzing the packets as they are transmitted through the air. This type of analysis can be performed by a standard personal computer with off the shelf hardware and freely available software. It is by no means completely secure. Thus WLANs using the WEP protocol are vulnerable to attacks. These so called wireless equivalent privacy attacks appear in the form of intercepting and modifying the transmissions, and gaining access to restricted networks. In this thesis, we propose an algorithm to patch the WEP protocol against these attacks.

This thesis develops a minimal required solution for the various attacks in the WEP protocol. It offers guidelines to develop a practical and a viable infrastructure for robust WEP implementation with least increment in hardware thus providing power and cost efficiency.

Wireless Communications

Wireless communications uses radio waves to transmit data between devices. Even with advantages such as mobility and portability, there are some limitations that are to be considered.

A few advantages of wireless technology are:

- No cables: No physical connection required from one device to another.

- Mobility: Freedom of movement and flexibility in placing the devices.
- Portability: One can easily carry a wireless device.
- Connecting ports: Devices can have dynamic number of connecting ports.

The limitations of wireless networks include issues such as security, speed, and not being able to change the number of channels without having to change technologies, unlike in wired networks where the number of channels (wires) can be changed for higher data rates.

Popular wireless technologies and standards are

- WAP, GPRS, 3G
- Blue tooth
- IEEE 802.11Std

Security Issue

The information security is the protection of information against unauthorized access to or modification of information against the denial of service to authorized users or the provision of service to unauthorized users. As the wireless network has no physical boundary, any unauthorized user can access in the wireless range and thus access the information. Security not only requires for user authentication but also requires for data to be protected from eavesdropping. There are various kinds of authentication services.

- Open system (Null authentication system)
- Digital signature
- Challenge response text
- Shared key authentication

Open System

Open System authentication [19] is a null authentication algorithm. It involves a two-step authentication transaction sequence. The first step in the sequence is the identity assertion and request for authentication. The second step in the sequence is the authentication result. If the result is “successful,” the STAs (Stations) shall be mutually authenticated.

Digital Signature

Digital signature [19, 20] is a method of authenticating digital information, in the same sense that an individual signing a paper document (or applying the seal of an organization) authenticates it.

It is itself simply a sequence of bits conforming to one of a number of standards in the area. The whole system depends on the fact that anyone can transform a message using a public key, but the private key is needed to reverse that transformation. Most digital signatures rely on public key cryptography to work.

In Digital signature method, the receiver generates a key pair consisting of two related "keys": public key and private key. Then it encrypts the data with its private key. After which, the receiver publishes their private key, which is used for encrypting the data and then transmitting the encrypted data which can only be decrypted with the private key; the key pairs are generated in such a way that its impractical to obtain private key from the public key.

Challenge-response test

Challenge-response authentication relies on the possession of a secret of some sort to perform authentication. A very simple example is asking for a password, where the challenge is asking for the password, and the adequate response is the correct password. A sophisticated algorithm is RSA [21, 22, 23, 24, 25, 26, 27] method to avoid communicating password or any other private information.

Shared Key Authentication

Shared Key Authentication [27, 28, 29, 30, 31] also known as Public-key cryptography or Asymmetric-key cryptography, is a form of cryptography in which two digital "keys" are generated, one private and one public. These keys are used for encrypting or signing messages; one key is used to encrypt a message and another is used to decrypt it, or one key is used to sign a message and another is used to verify the signature. The public key can encrypt or sign messages that can only be verified using the private key, and vice-versa, so it is critical that the private key be kept secret. The two keys are related mathematically - a message encrypted by the algorithm using one key can be decrypted by the same algorithm using the other key. A few examples of asymmetric algorithm are mentioned below:

- Diffie-Hellman [32, 33]
- RSA encryption algorithm
- ElGamal [34, 35, 36]
- Elliptic curve cryptography [36, 37]

Protocols using asymmetric key algorithms include:

- DSS (Digital Signature Standard) [38, 39, 40] which incorporates the Digital Signature Algorithm
- PGP (Pretty Good Privacy) / GPG (GNU Privacy Guard) [41]

SSH (Secure SHell) is both a program and a network protocol for logging into and executing commands on a remote computer. It is intended to replace rlogin, telnet and RSH (Remote SHell), A program to provide a means of executing commands on a remote host without the need to (r)login), and provides secure encrypted communications between two non-trusted hosts over an insecure network

- Secure Socket Layer now implemented as an IETF (Internet Engineering Task Force) standard (TLS (Transport Layer Security)).

To avoid the eavesdropping security issue, data is usually encrypted using cryptographic techniques and transmitted over the network, and the authorized personnel having the proper access rights and keys can access the data.

IEEE 802.11 provides standard for designing wireless LAN services with two different levels of security systems.

- Open System
- Shared Key authentication service.

Cryptography

Cryptography [45, 46] is a process by which information is converted into an encrypted version that is difficult (ideally, impossible) for any unauthorized person to convert back to the original information, while still allowing the intended reader to do so. In traditional

cryptography, the sender and receiver of a message know and use the same secret key; the sender uses the secret key to encrypt the message, and the receiver uses the same secret key to decrypt the message. This method is known as secret key or symmetric cryptography. The main challenge is getting the sender and receiver to agree on the secret key without anyone else finding out. If they are in separate physical locations, they must trust a courier, a phone system, or some other transmission medium to prevent the disclosure of the secret key. Anyone who overhears or intercepts the key in transit can later read, modify, and forge all messages encrypted or authenticated using that key. The generation, transmission and storage of keys is called key management. All cryptosystems must deal with key management issues. Because all keys in a secret-key cryptosystem must remain a secret, secret-key cryptography often has difficulty providing secure key management, especially in open systems with a large number of users.

The other popular cryptography technique is public key cryptography, which uses asymmetric keys, one being private and the other being public. The private key is kept a secret; the public key can be known by anyone.

Sets of public and private keys match from a cryptographic standpoint. For example, the sending station (e.g., NIC (Network Interface Card) or access point) can encrypt data using the public key, and the receiver uses the private key for decryption. The opposite is also true. The sending station can encrypt data using the private key, and the receiving station decrypts the data using the public key.

General Cryptography techniques

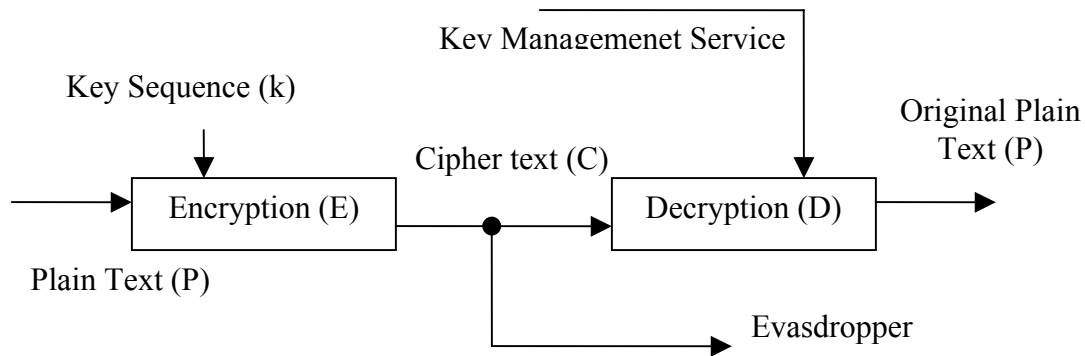


Figure 1: Encryption Decryption

The process of disguising data (binary) in order to hide its information content is called Encryption (denoted by E) (see Figure 1). Data that is not enciphered is called Plaintext (denoted by P) and data that is enciphered is called Cipher-text (denoted by C). The process of turning cipher-text back into plaintext is called Decryption (denoted by D). A cryptographic algorithm, or cipher, is a mathematical function used for enciphering or deciphering data. Modern cryptographic algorithms use a key sequence (denoted by k) to modify their output. The encryption function E operates on P to produce C: At the sending end the encryption function operates on the P to produce E:

$$E_k(P) = C \quad (1)$$

At the receiving end the decryption function operates on C to produce P:

$$D_k(C) = P \quad (2)$$

The WEP algorithm is symmetric in nature. Hence we can use the same key for enciphering as well as deciphering:

$$D_k(E_k(P)) = P \quad (3)$$

Symmetric and Asymmetric key cryptography

The primary advantage of public-key cryptography is increased security and convenience: private keys never need to be transmitted or revealed to anyone. In a secret-key system, by contrast, the secret keys must be transmitted (either manually or through a communication channel) since the same key is used for encryption and decryption. A serious concern is that there may be a chance that an enemy can discover the secret key during transmission.

Another major advantage of public-key systems is that they can provide digital signatures that cannot be repudiated. Authentication via secret-key systems requires the sharing of some secret and sometimes requires trust of a third party as well. As a result, a sender can repudiate a previously authenticated message by claiming the shared secret was somehow compromised by one of the parties sharing the secret

A disadvantage of using public-key cryptography for encryption is speed. There are many secret-key encryption methods that are significantly faster than any currently available public-key encryption method. Nevertheless, public-key cryptography can be used with secret-key cryptography to get the best of both worlds. For encryption, the best solution is to combine

public- and secret-key systems in order to get both the security advantages of public-key systems and the speed advantages of secret-key systems. Such a protocol is called a digital envelope.

Public-key cryptography may be vulnerable to impersonation, even if users' private keys are not available. A successful attack on a certification authority will allow an adversary to impersonate whomever he or she chooses by using a public-key certificate from the compromised authority to bind a key of the adversary's choice to the name of another user.

Motivation and Problem Statement

WEP used in Wi-Fi standard uses RC4 [4, 24, 43] encryption algorithm, which operates by expanding a short key into an infinite pseudo-random key stream. If an attacker flips a bit in the cipher text, then upon decryption, the corresponding bit in the plaintext will be flipped. If an eavesdropper intercepts two cipher texts encrypted with the same key stream, it is possible to obtain the XOR of the two plaintexts. Knowledge of this XOR can enable statistical attacks to recover the plaintexts. The statistical attacks become increasingly practical as more cipher text that uses the same key stream become known. Once one of the plaintexts becomes known, it is trivial to recover all of the others. To ensure that a packet has not been modified, WEP uses an Integrity Check Value (ICV) field in the packet. To avoid encrypting two cipher text with the same key stream, an initialization vector (IV) is used to augment the shared key and produce a different RC4 key for each packet. The major attacks [16, 17, 18, 47, 52, 53] to WEP are given as follows:

1. Active attack: Modification of the packet by modifying the ICV.
2. Passive attacks:

- a. Integrity violation by analyzing the IV
- b. Table based attack for decrypting every packet that is sent over the wireless link.

In order to avoid these attacks, a novel-scrambling algorithm is proposed in this work. The algorithms randomize the data and prevent access from unauthorized users by adding some standard randomness to it. This random characteristic is a function of the private attribute shared between transmitter and receiver only. In this approach the randomness is achieved by RC4 algorithm, and the distribution of the randomness is provided with different algorithms to increase the complexity of rectifying the encrypted data and optimize utilization of the randomness.

Related Work

Various groups are working on different approaches in order to make WEP more robust and impractical to break. IEEE 802.11i group is working on integrating TKIP [54, 55, 56] (Temporal Key Integrity Protocol) with it which is attempting to harden IV based attacks and authentication issues by changing the SK (Secret Key) over the certain amounts of packet transfer from particular user.

Another group at Wi-Fiplanet has proposed to use AES (Advanced Encryption Standard) encryption method with it to secure it from all other known encryption attacks to WEP.

WPA [57, 58, 59] (Wi-Fi Protected Access) is another approach to attain more security. It incorporates both the TKIP and the AES and achieves both authentication and encryption security. WPA includes both the Temporal Key Integrity Protocol (TKIP) and 802.1x mechanisms, which together provide dynamic key encryption and mutual authentication for

mobile clients. WPA thwarts hackers by periodically generating a unique encryption key for each client.

TKIP introduces new algorithms to WEP, which includes extended 48-bit initialization vectors and associated sequencing rules, per-packet key construction, key derivation and distribution function, and a message integrity code (referred to as "Michael"). WPA can interface with an authentication server, such as Remote Authentication Dial-In User Service, using 802.1x with EAP. The authentication server is a storehouse for user credentials. This function enables effective authentication control and integration into existing information systems.

Another alternative is to use IPsec [60] for wireless security adding VPN [61] functionality to the client.

The TESLA [62] Broadcast Authentication Protocol despite using purely symmetric cryptographic functions (MAC functions), TESLA achieves asymmetric properties. We discuss a PKI application based purely on TESLA, assuming that all network nodes are loosely time synchronized.

All the above approaches are leading to give partial or full security to the wireless communication comparable to the wired network. The approach which gives full security requires replacing the existing hardware completely and not downward compatible with existing technology. In our approach we have achieved very high security in all aspects with downward compatibility, non replacement of current hardware and technology with minimal requirement of hardware and computational power.

WIRED EQUIVALENT PRIVACY (WEP)

Wired Equivalent Privacy (WEP) is a part of the IEEE 802.11 Wireless Fidelity (Wi-Fi) standard. It is a system to secure Wi-Fi networks. It is a data encryption method designed to protect the transmission between 802.11 wireless clients and Access Points (APs) and to provide a Wireless Local Area Network (WLAN) with the same level of security as can be expected from the security provided to a wired network. The WEP algorithm is a form of electronic codebook in which a block of plaintext is bitwise XORed with a pseudorandom key sequence of equal length. The WEP algorithm generates the key sequence based on shared key authentication, which uses RC4 algorithm to encrypt the message and CRC32 to keep the integrity. Wired Equivalent Privacy is defined as protecting authorized users of a wireless LAN (Local Area Network) from casual eavesdropping. This service is intended to provide functionality for the wireless LAN equivalent to that provided by the physical security attributes inherent to a wired medium. It provides an adequate level of security for most home networks. It is a popular wireless secure communication stream cipher protocol approaches to network security used to protect link-layer communications from eavesdropping and other attacks. It allows users to communicate with other users sharing the public key over a network. It provides authentication and encrypted communications over unsecured channels. WEP seeks to establish similar protection to that offered by the wired network's physical security measures by encrypting data transmitted over the WLAN. Data encryption protects the vulnerable wireless link between clients and access points; once this measure has been taken, other typical LAN security mechanisms such as password protection, end-to-end encryption, virtual private networks

(VPNs), and authentication can be put in place to ensure privacy. WEP is used to protect link-layer communications from eavesdropping and other attacks.

Shared key identification system

Shared key authentication supports authentication of STA (station). It accomplishes this without the need to transmit the secret key in the open; however, it does require the use of the WEP privacy mechanism. Therefore, this authentication scheme is only available if the WEP option is implemented. The required secret, shared key is presumed to have been delivered to participating STAs via a secure channel that is independent of IEEE 802.11. This shared key is contained in a write-only Management Information Base (MIB) attribute via the MAC management path; so that the key value remains internal to the MAC. During the Shared Key authentication exchange, both the challenge and the encrypted challenge are transmitted. This facilitates unauthorized discovery of the pseudorandom number (PRN) sequence for the key/IV pair used for the exchange. Implementations should therefore avoid using the same key/IV pair for subsequent frames.

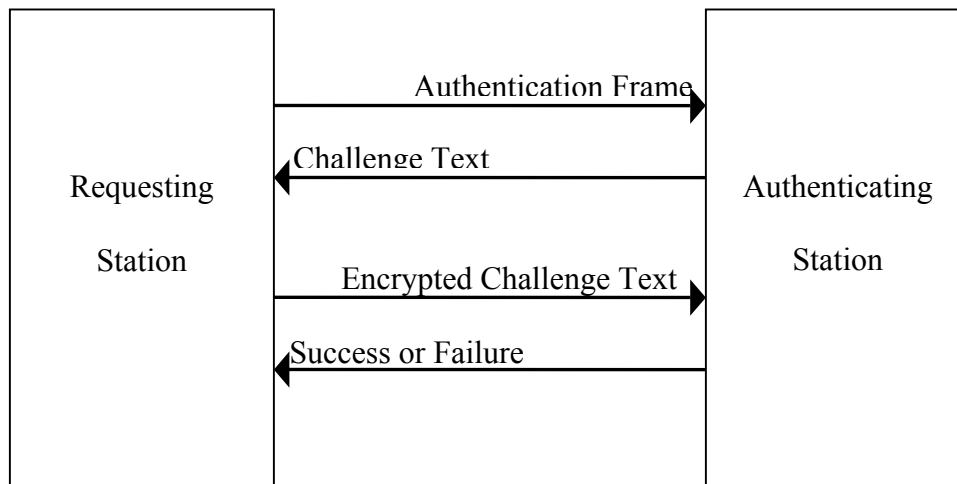


Figure 2: IEEE 802.11 Station Authentication

Block Diagram

WEP is a symmetric algorithm in which the same key is used for encipherment and decipherment. The secret key is concatenated with an initialization vector (IV) and the resulting seed is input to a PRNG (Pseudo Random Number Generator). The PRNG outputs a key sequence k of pseudorandom octets equal in length to the number of data octets that are to be transmitted in the data plus 4 (since the key sequence is used to protect the integrity check value (ICV) as well as the data). Two processes are applied to the plaintext. To protect against unauthorized data modification, an integrity algorithm operates on P to produce an ICV.

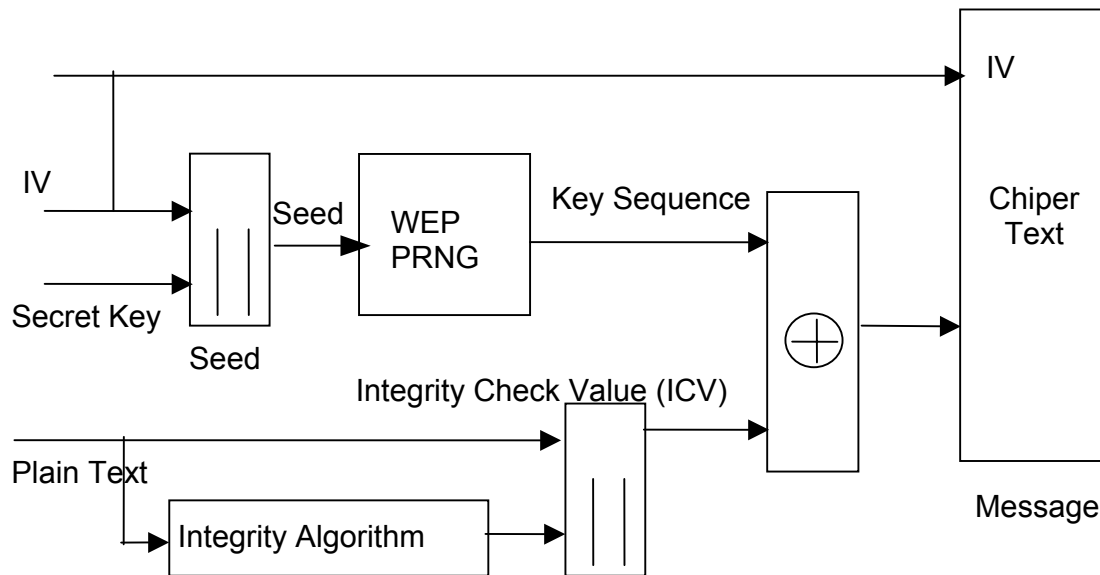


Figure 3: WEP Block Diagram

Encipherment is then accomplished by mathematically combining the key sequence with the plaintext concatenated with the ICV. The output of the process is a message containing the IV and cipher-text. The WEP PRNG is the critical component of this process, since it transforms a relatively short secret key into an arbitrarily long key sequence. This greatly simplifies the task of key distribution, as only the secret key needs to be communicated between STAs. The IV extends the useful lifetime of the secret key and provides the self-synchronous property of the algorithm. The secret key remains constant while the IV changes periodically. Each new IV results in a new seed and key sequence, thus there is a one-to-one correspondence between the IV and k . The IV may be changed as frequently as every MPDU (MAC protocol data unit) and, since it travels with the message, the receiver will always be able to decipher any message. The IV is transmitted in the clear since it does not provide an attacker with any information about the secret key, and since its value must be known by the recipient in order to perform the decryption.

The WEP algorithm is applied to the frame body of an MPDU. The IV, frame body, ICV triplet forms the actual data to be sent in the data frame.

Features of WEP Algorithm

- Reasonably strong: Relies on the difficulty of discovering the secret key through brute force attack
- Self synchronizing: Important when mobile stations go in and out of coverage
- Computationally efficient: Easily implemented on software and hardware
- Exportable: Can be exported outside the US
- Optional: It is an option not required in an 802.11 compliant system

Design Specification

The frame body of the data frame has the following format:

WEP protected frames are having 3 major parts IV, encrypted text, and ICV. The first four octets of the frame body contain the IV field for the MPDU (MAC Protocol Data Unit). The PRNG (Pseudo Random Number Generator) is fed with a 64 bit seed, which is obtained from concatenation of 24 bit IV and 40 bit SK (secret Key). The 24 LSBs (Least Significant Bits) of seed are IV and rest 40 bits are SK. The IV is followed by the MPDU, which is followed by a 32-bit ICV. The WEP ICV is obtained from Integrity Check algorithm that is CRC-32, WEP achieves encryption by combining k with P using bitwise XOR. The WEP mechanism is invisible to the entities outside the MAC data path. WEP design incorporates the following modules

- ICV (CRC32)

- RC4 PRNG
- Cipher Engine
- Control logic and Data Path

CRC32

A cyclic redundancy check (CRC) [49, 50, 51] is the result of a type of calculation made upon data, such as network traffic or computer files, in order to detect errors in transmission or duplication. CRCs are calculated before and after transmission or duplication, and compared to confirm that they are the same.

A cyclic redundancy check (CRC) is the result of a type of calculation made upon data, such as network traffic or computer files, in order to detect errors in transmission or duplication. CRCs are calculated before and after transmission or duplication, and compared to confirm that they are the same. It is used by the WEP to produce the ICV. It performs a mathematical calculation on a block of data and returns a 32-bit number called the checksum that represents the content and organization of that data. The IEEE 802.11 protocol uses the CRC to verify whether the data received at the receiver is the same as the data that was sent; i.e. whether the data has modified after transmission or not.

Algorithm

The essential mathematical operation in the calculation of a CRC is binary division, and the remainder from the division determines the CRC.

shiftregister = initialize shiftregister with 0s

```

while (there are more input bits) {
    if (MSB of shiftregister == 1)
        shiftregister = (shiftregister leftshift 1) xor polynomial
    else:
        shiftregister = shiftregister leftshift 1
        xor next bit from the string into LSB of shiftregister
}

```

CRC32 = shiftregister value

Applications of CRC:

- Verifying transmitted information
- Sending and receiving records
- Modifying files and records
- Verifying emails between 4D databases

WEP incorporates the most commonly used CRC32 polynomial given by

$$X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X^1$$

This polynomial generates a 32-bit fingerprint checksum.

RC4

RC4 is a symmetric key, secret key, stream cipher of RSA Security. It is a pseudo-random number generator initialized from a secret key of up to 256 bytes. The RC4 algorithm it generates a "keystream" which is simply XORed with the plaintext to produce the cipher-text

stream. Decryption is exactly the same as encryption. One reason for the algorithm's popularity is its simplicity.

Algorithm

The RC4 algorithm consists of an initialization stage, which uses the key to initialize the pseudo-random number generator:

```
for i = 0 ... 255
    S[i] = i
for i = 0 ... 255
    j = (j + S[i] + key[i mod key_length]) mod 256
    swap (S[i],S[j])
```

Once the generator has been initialized, both encryption and decryption is performed using values output from the generation stage. The process of encryption and decryption is as follows:

```
i = 0
j = 0
while(Entire message is encrypted/decrypted)
    i = (i + 1) mod 256
    j = (j + S[i]) mod 256
    swap(S[i],S[j])
    k = S[(S[i] + S[j]) mod 256]
    output the k as psedo random byte
end while
```

Output k is then used to XOR with the input RC4

The WEP PRNG

The WEP PRNG is very critical to the process of security as it converts the relatively short secret key into an arbitrarily long key sequence. Hence only the secret key needs to be distributed among the STAs. The PRNG uses the RC-4 algorithm.

RC-4 uses a private and a public key.

e : Public key

d : Private key

n : number used to find e (this is also a public key)

To generate the keys, two large prime numbers p and q are chosen such that:

$$n = p \times q \quad (4)$$

Randomly an encryption key is chosen such that e and $(p-1) \times (q-1)$ are relatively prime.

The decryption key is computed using Euclid's algorithm such that:

$$e \times d = 1 \pmod{(p-1) \times (q-1)} \quad (5)$$

d and n are relatively prime.

In the end, publish e and n , keep d secret and discard p & q .

Dynamically set the initial state to configure any starting random integer and corresponding sequence.

Cipher Engine

WEP Cipher Engine is a simple entity that performs bitwise XOR of the 8-bit plaintext and 8-bit keystream. The resultant will be 8-bit encrypted text. This gives the WEP a simple, fast but very powerful encryption engine as it is merging the properties of plaintext and random keystream together without any increase in the size of the output.

Control Logic and Data Path

As the ICV algorithm is working bit by bit and the rest of the unit works on byte, and the resultant 32 bits are fed 8 bit at a time thus a parallel to serial converter is used at the beginning and end of ICV module.

The controller first initializes and randomize the RC4 PRNG and once the RC4 PRNG is ready to use the data path of keystream and input data is enabled. At the end of the packet input data path is disabled and the ICV serial to parallel cov2 data path is enabled, results in encrypted ICV at the end of the packet.

Implementation

CRC32

On the application of reset or the end of the beginning of the CRC calculation; the 32 bit CRC registers have reset to zeroes and a bit value is given to the module from the parallel to serial converter at every clock cycle. A dataavailable signal is used to indicate that datapath has

valid data for the period. After the `dataavailable` goes low the controller waits for the last bit to propagate to the last register which takes 32 clock cycles. For this time period a `crc_busy` signal goes high, indicating that the ICV module is busy calculating CRC32 value of the data; and at the end of calculation (which is `data_avalble + 32` clock cycles) the signal goes low indicating the current 32-bit output in CRC32 is valid CRC output of last input data packet. The value is held until the next `data_available` or reset signal. On application of Reset clear out all 32 registers with zeroes. A active (high) `data_available` signal with inactive (low) reset pulled up the busy signal notify other modules that CRC calculation has begin and CRC register contents are invalid. An input data bit fed from parallel to serial converter is sampled at the rising edge of the clock and shifted into the CRC shift registers from LSB side. The busy signal remains high for next 32 clock cycle after the `data_available` goes low, which ensure the propagation of the last input bit till the end of the CRC register. The valid or correct CRC result is notified by high to low transition of the busy signal. The value is held in the register until the next `data_available` or reset signal.

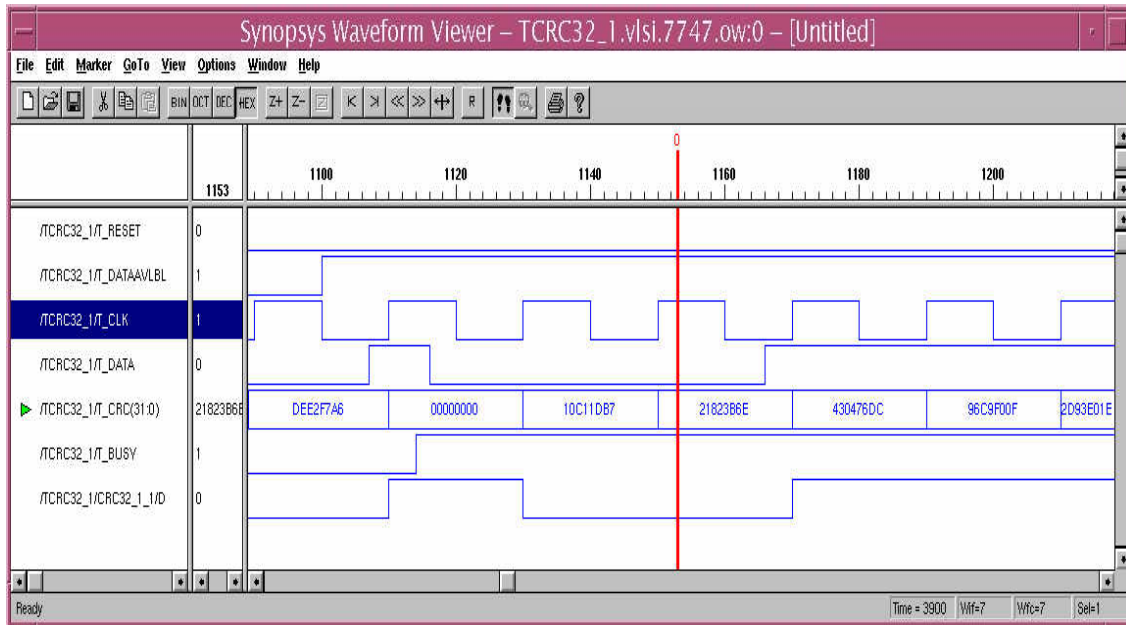


Figure 4: generation of ICV using CRC32 algorithm

The implemented CRC function is realization of well known CRC polynomial

$$X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 X^4 + X^2 + X^1$$

used in various error recovery algorithms and WEP. This implementation is simple shift register and XOR network in loop back.

Concatenater

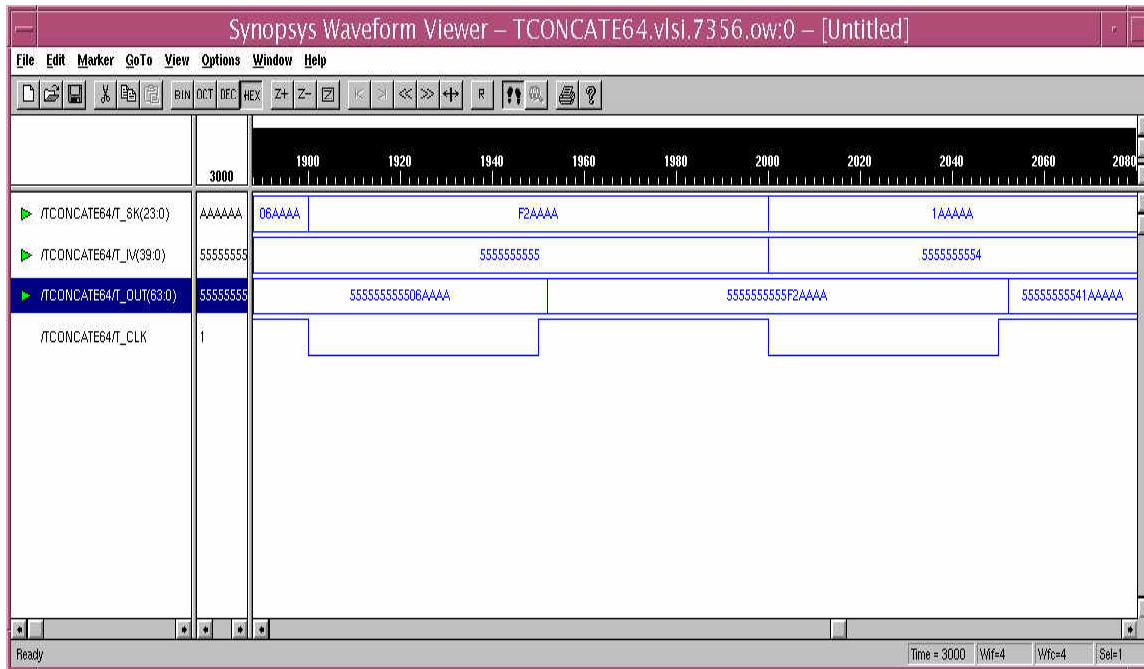


Figure 5: Concatenater

Simulation shows the generation of seed as a result of concatenation of IV and SK. On every rising edge of clock IV and SK are sampled and held in the 64 bit register. The lower 24 bit (0 thru 23) of 64 bit are sampled IV and next 40 bit (24 thru 63) are SK. Simulation shows that change in any one or both at any time will reflect at next rising edge.

Parallel to Serial Converter

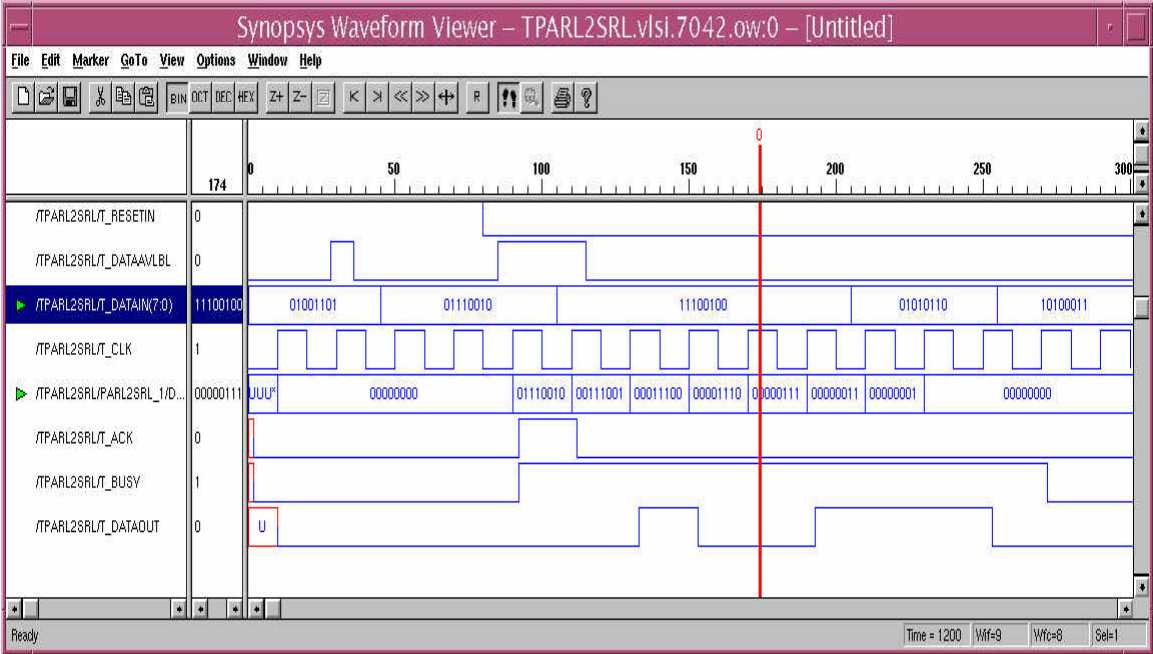


Figure 6: Parallel to Serial Converter

An handshaking signal *ack* is used to let other units know that current input data has sampled and data at input port can be changed without any effect on the current output; This ensure the data has delivered to ICV unit and rest of the time (9 clock cycles of parallel to serial converter) could be utilize for the next data settlement. Busy signal is activated (high) from sampling to the shifting output, which notifies other units that parallel to serial converter is processing data and cannot sample any input data at this time. This in conjunction with ack signal shows the other units that output at output post is a valid serial data output. The output data will be valid when busy signal will be high and ack signal has not received.

The data bits in the shift register are shifted bit by bit at rising edge of clock; a 3-bit counter is used to shift 8 bits. Counter reseted at high ack. Busy signal is dropped low after 7. Output is shifted to right that is from MSB to LSB and output starts with the LSB and ends with MSB. Zeroes are inserted from MSB side, which ends all zeroes at the end in the shift register. The shifting process starts as soon as a data_available signal comes and an ack has responded back.

Random Generator

A handshaking signal 'SET' is used to initialize the random generator every time IV changes, and ACK is outputted to acknowledge that the input SEED has been read and can be changed. Initially, the random generator has the unknown contents. The small initial setup time is required for the random generator, after initialization the generator is capable to generate a random byte every clock cycle.

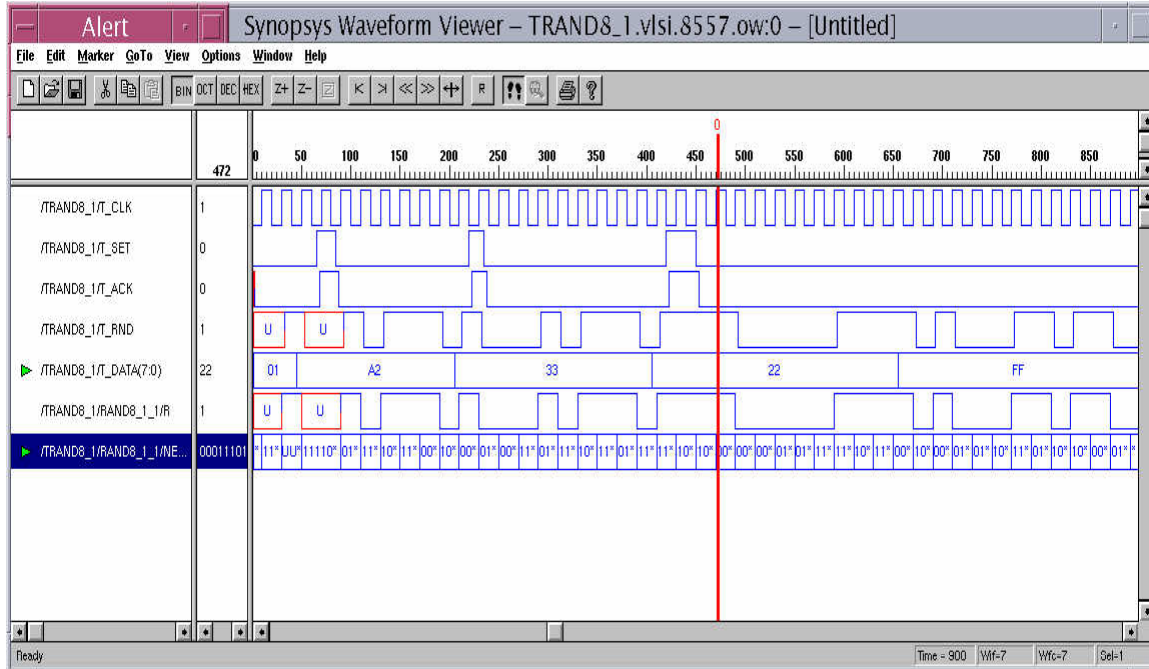


Figure 7: Random Generator

Attacks

The simple passive gathering and analyzing of packets as they are transmitted through the air can compromise a WEP network. This type of analysis can be performed by a standard personal computer with off the shelf hardware and freely available software.

WEP uses RC4 encryption algorithm, which operates by expanding a short key into an infinite pseudo-random key stream. If an attacker flips a bit in the cipher-text, then upon decryption, the corresponding bit in the plaintext will be flipped. If an eavesdropper intercepts two cipher-texts encrypted with the same key stream, it is possible to obtain the XOR of the two plaintexts. Knowledge of this XOR can enable statistical attacks to recover the plaintexts. The statistical attacks become increasingly practical as more cipher-text use the same key streams

that are already known. Once one of the plaintexts becomes known, it is trivial to recover all of the others. To ensure that a packet has not been modified, WEP uses an Integrity Check Value (ICV) field in the packet. To avoid encrypting two cipher-texts with the same key stream, an initialization vector (IV) is used to augment the shared key and produce a different RC4 key for each packet. The major attacks to WEP are given as follows:

- Active attack: Modification of the packet by modifying the ICV.
- Passive attacks:
 - Integrity violation by analyzing the IV
 - Table based attack for decrypting every packet that is sent over the wireless link.

NOVEL SCRAMBLING ALGORITHM

In this chapter we propose two algorithms to patch the WEP protocol against the described attacks in chapter WIRED EQUIVALENT PRIVACY (WEP).

We have attempted a minimal required solution for the various attacks in the WEP protocol. In this chapter we will discuss the about the design, implementation and analysis of the algorithms to develop a practical and a viable infrastructure for robust WEP implementation. We have designed two algorithms and called Scrambling Algorithm (SA) to randomize the contents of WEP.

In the SA, a random octet is inserted at a random position. The random position is obtained by RC4 as a function of the secret key. Currently, the octets contain random information, however, these octets can be utilize for further improvement in security of the packets (e.g., dynamically changing secret keys (TKIP)). Octet insertion is applied to three different fields in the packet format, namely, ICV, IV and cipher-text, to reduce the vulnerability for each of the attacks mentioned above. One octet is inserted for both ICV and IV. However, due to the length of the cipher-text and the need to improve security, more octets are inserted to the cipher-text. In the cipher-text, the SA distributes the random octets at random positions in such a way that density of octets reduces along with the length of the packet, which ensures insignificant increase in the packet size for large packets.

Each field (IV, cipher-text, ICV) uses one RC4 key-stream octet, to find random position for insertion of random octet. Thus every packet requires 3 RC4 octets from the key-stream, thus the key stream is divided into sets of 3 octets, and where one set is used for each WEP frame. We have two major algorithms using these sets for inserting the random octet at a random position.

Algorithm 1: IV and ICV randomizations / extraction

Algorithm 1: IV and ICV randomizations / extraction uses the first and last octet of the set to randomize cipher-text to randomize IV and ICV. It uses 5 lower bits of first octet from a set to randomize IV and 5 lower bits of third octet to randomize ICV. It inserts a random octet in IV and ICV at random position depending upon the value of these 5 bits.

Algorithm

For entire transmission

Fetch packet_number and IV/ICV from 802.11 protocol

Fetch random data content for the octet from memory

Calculate_random_position(RC4(secret_key)

*[i *packet_number],field_length)*

Insert/Extract the octet at the calculated random position

End for

where i is the number of RC4 octets used per packet for randomness.

Algorithm 2: Algorithm for cipher randomization / extraction

Algorithm 2: Algorithm for cipher randomization / extraction uses the second octet of the set to randomize cipher-text. The distributive algorithm used in Algorithm 2: Algorithm for cipher randomization / extraction incorporates the LSBs of octet to find the random point in the chunk. The size of the chunk is increased exponentially to utilize the different patterns of second octet and to create high random density at the starting of the cipher field (explained in analysis).

Distributive algorithm ensures the insertion of the random octet at the random position throughout the chunk, which is always in the range of 0 to current chunk size. A chunk is a portion of input stream whose size is increasing logarithmically and dependent on the chunk number or chunk position. Each chunk is twice the size of its previous chunk and the first chunk is 1 byte wide. This is due to the random pointer being pre-pended with one bit to point the random position, which results in twice as many points as the earlier one, so the size is doubled every time. If pre-pending is not binary but rather octal or hex then chunk sizes will be 8 times or 16 times of its predecessor.

Algorithm

For Entire transmission

Fetch packet_number from 802.11 protocol

Reset Cipher_octet_cntr to 0;

Reset octets_processed to 0;

While not end of the cipher-text

If (random_position == current position) then

*/**

*(random_position = distributiveAlgorithm (No_of_inserted_octets,
packet_number)) == Cipher_octet_cntr*

**/*

*(Fetch random data content for the octet from memory Insert the octet in the current
position) OR (Remove the current octet from stream)*

```

    octets_processed = octets_processed+1;
End if
Insert an octet of cipher-text;
Cipher_octet_cntr = Cipher_octet_cntr +1;
Fetch next cipher octet;
End While
End For

SubProc: Return pos distributiveAlgorithm (No_of_inserted_octets, packet_number)
    pos = RC4(secret_key)[packet_number*i+2][0 to No_of_inserted_octets]
        /* packet_number*i+2 points to the second octet among the current set of RC4 octets used */
End subProcedure;

```

Scrambling

Performing the insert action in the above algorithms results in the Scrambling of the original WEP cipher-text. This action causes the insertion of the random octet at the random position obtained from distributive algorithm. Insertion could be the part of another protocol enhancing security or other robustness, but must not be the function of SK, because combination of Scrambling algorithm and it may cause major leak of Secret key information.

De-scrambling

Algorithms are designed in such a way that same design implementation can be used to achieve both scrambling and descrambling, thus not necessarily requiring different hardware, which further helps in reduction of hardware and thus power, cost and size. Performing the extract action in the above algorithms results in the descrambling of the scrambled WEP cipher-text. The extracted octet is then made available for other processors if used for other enhancements.

Implementation

The cipher-text is randomized in decreasing density of randomness. We are dividing cipher-text into virtual chunks of different sizes and adding random contents at random positions in each chunk. The size of a chunk is determined by the formula: $2^{(\text{chunk number})}$, as shown in Table 1.

Table 1: Chunk specification and randomization of binary values in chunk

Chunk no C_p	Max chunk size $C_s = 2^{C_p}$	Chunk size C_s	Chunk size after randomized C_{s+1}	Min-max of valid random positions 0- C_s
0	20	1	1 – 2	0 – 1
1	21	1 – 2	1 – 3	0 – 2
2	22	1 – 4	1 – 5	0 – 4
3	23	1 – 8	1 – 9	0 – 8
4	24	1 – 16	1 – 17	0 – 16
5	25	1 – 32	1 – 33	0 – 32
6	26	1 – 64	1 – 65	0 – 64
7	27	1 – 128	1 – 129	0 – 128
8	28	1 – 256	1 – 257	0 – 256

We have implemented the proposed algorithm in MATLAB to verify the functionality at the system level. Furthermore, the algorithm is behaviorally modeled in VHDL to obtain hardware simulation and verification. The architecture of the implemented patching algorithm is given in Figure 8. The algorithm works as a post-processor to the WEP protocol. It takes the WEP input and applies randomness to it as specified in section 2. In the architecture, BAB (Bit Addressable Memory Bank) is having 2 banks of bit addressable memory. Bank 1 is, of size 32 bits, used to hold input WEP content either IV or ICV. Bank 2 is, of size 48 bits, used to hold the resultant randomized output of IV or ICV. Cntr 1 is used to point bits in Bank 1 for the IV starting of the packet and decreases across the length of the packet.

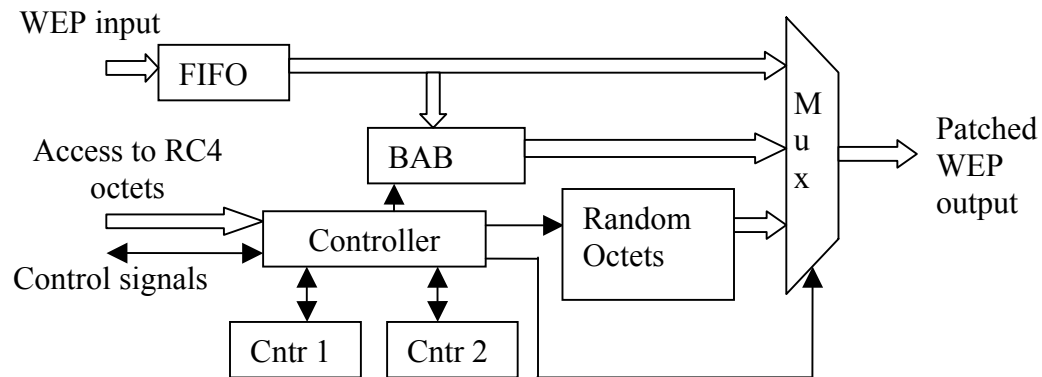


Figure 8: Architecture of the implemented patching algorithm

Random insertion positions in above test simulation are calculated by the controller (shown in architecture), which has access to RC4 octets from WEP implementation. In the above test, the RC4 octet has a binary value of "00011100". The last bits of the obtained RC4 octet are used to give different positions in the m th chunk between the range of 0 and the maximum chunk size.

The dashed region in Fig. 3 is zoomed in Fig. 4 to illustrate the insertion of random octet in details. In every clock cycle, input is read and forwarded to output if the current state is *READ_INP_PORT* state. When the current state is *INSERT* state, a control signal is sent to FIFO to wait for a clock cycle; in which the system inserts the random content.

Simulation Results

IV Randomization

Above simulation result demonstrate the implementation of Algorithm 1: IV and ICV randomizations / extraction.

Signal *clock* is a global clock input signal to the system generated by the clocking circuitry of the system. All the events are synchronized at the clock transition. Signal *reset* is another global reset input signal to the system, which resets the system to the initial condition where all required registers are initialized and the system is brought up into IDLE state.

in read state in 3 consecutive clock cycles and read in *READ* and *FILL* states. *New_data_holder* is a 32 bit, bit addressable register that is used to store new generated IVs (actual IV + 8 bit unrelated data). On every clock transition, reset input is sampled and on high, reset system goes into *IDLE* state. Current state is *IDLE* (see Figure 10) due to high Reset input. In the *RESET* state all the registers and counters are reset to zero(s). At low Reset, input system waits for *start_end_signal*, which is high on start and end of the packet.

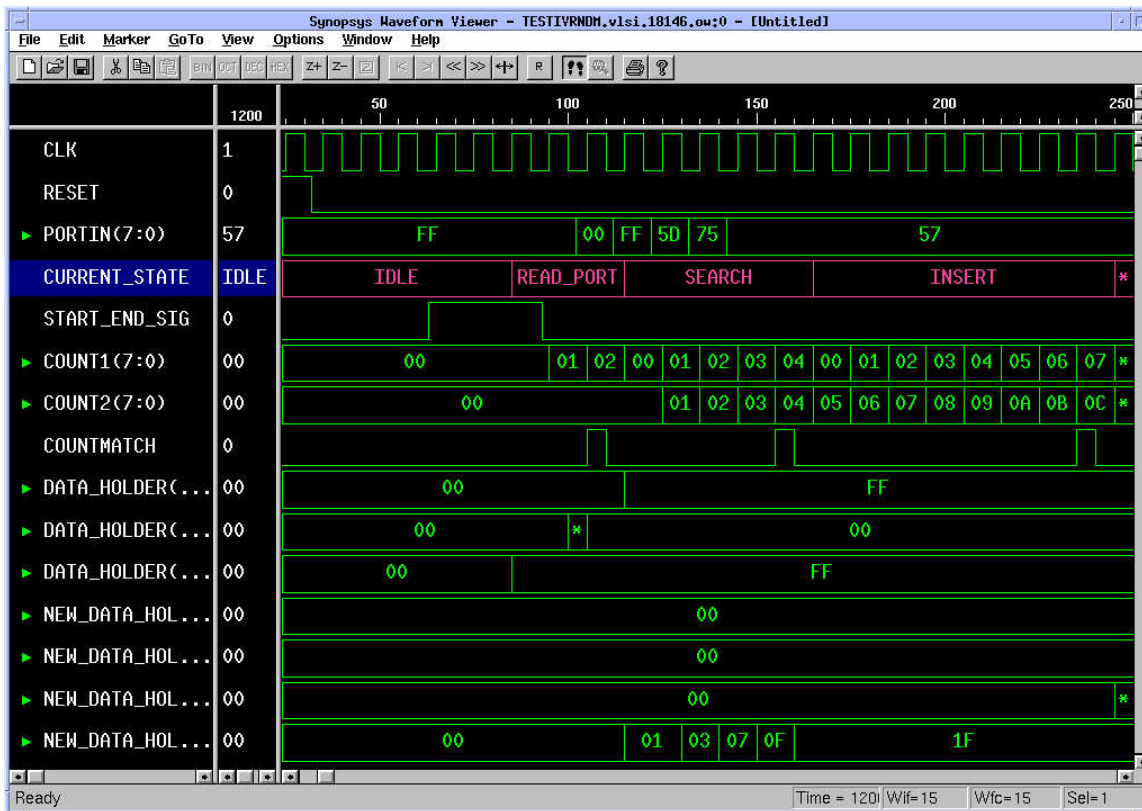


Figure 10: IV Randomizing showing insertion of random octet

(If state is *IDLE*; a high on it states packet start else packet end). On high *start_end_signal*, system goes into *READ* state and reads 3 bytes from port. System enters into *SEARCH* state after

read state and counter 1 and counter 2 increments every time; in the design counter1 is used to point the bit position of *data_holder* register holding input-24 bits of IV and counter2 is used to point bit position of *new_data_holder* register which will be the resultant modified 32 bit IV. One bit is copied from *data_holder* to *new_data_holder* register, upon the match of first four bit of RC4(1) and counter1 system goes into *INSERT* mode, during which counter 2, which is already initialized with 0, keeps incrementing causing insertion of 8 random bits

After inserting a random byte, system enters into *FILL* state where the remaining bits are filled as they are. After filling the bits, system enters into *IVdone* mode and signals to next system for further processing and finally returns to the *IDLE* state and waits for the next input packet.

The result of this application is the insertion of an 8 bit random octet into IV at random positions obtained from RC4 keystream. Figure 11 demonstrates the result.

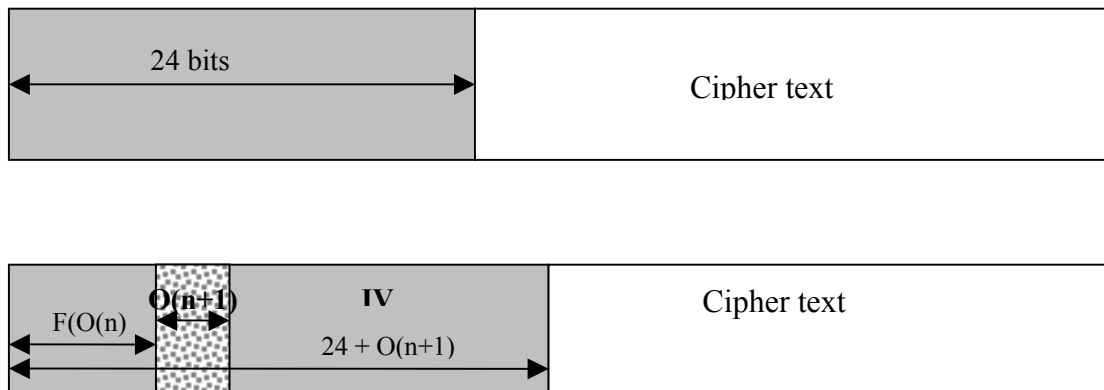


Figure 11: a) IV a) before b) after application of Algorithm 1: IV and ICV randomizations / extraction

Cipher Randomization

The simulation result shown below demonstrates the implementation of Algorithm 2:

Algorithm for cipher randomization / extraction.

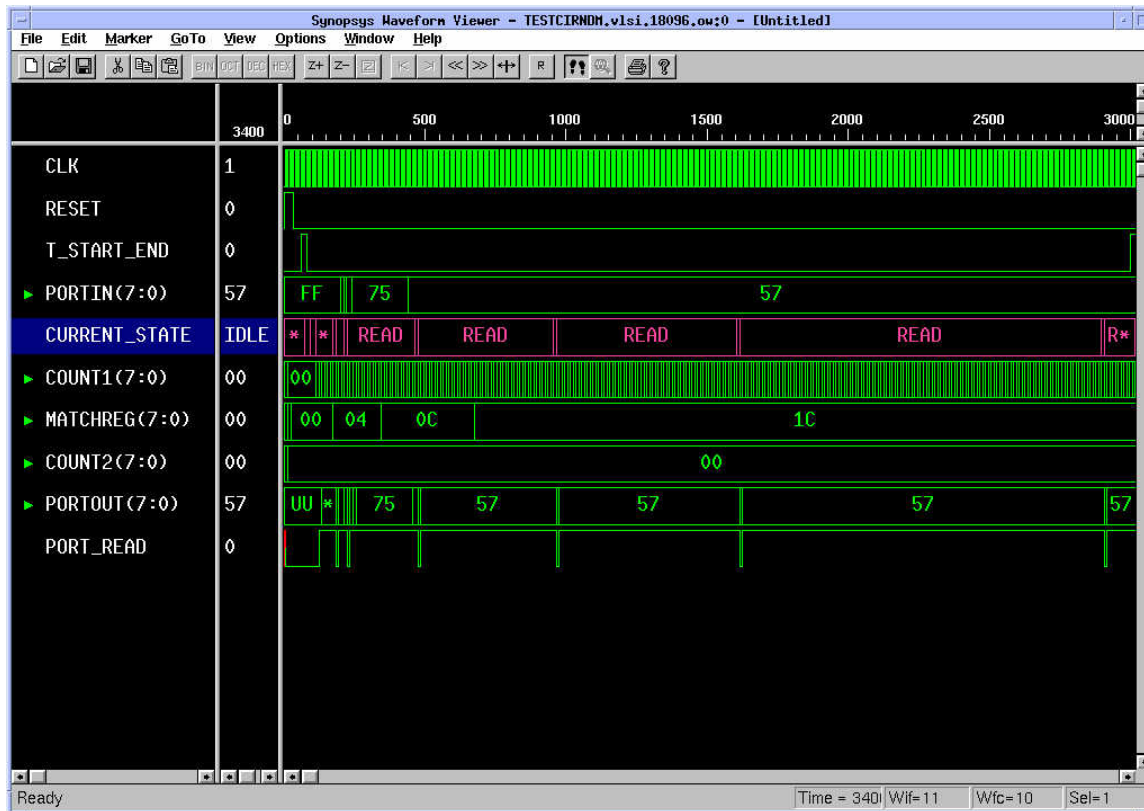


Figure 12: Cipher Randomization showing different states

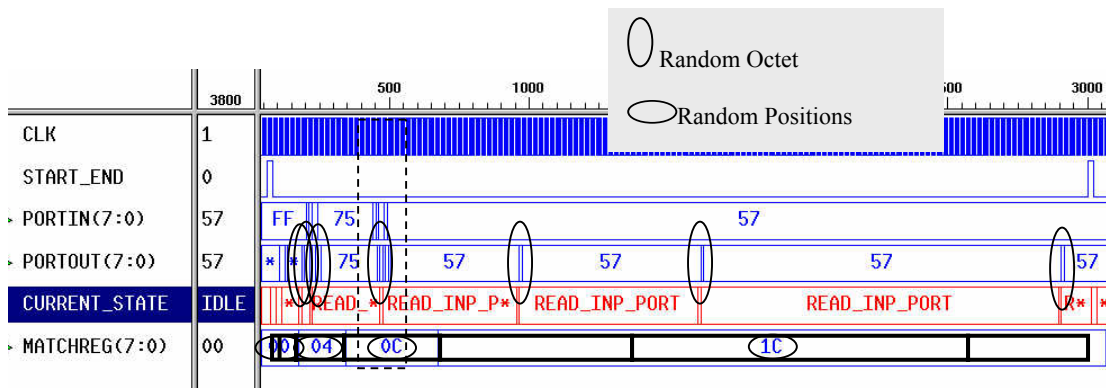


Figure 13: Cipher-text randomization showing random insertion positions

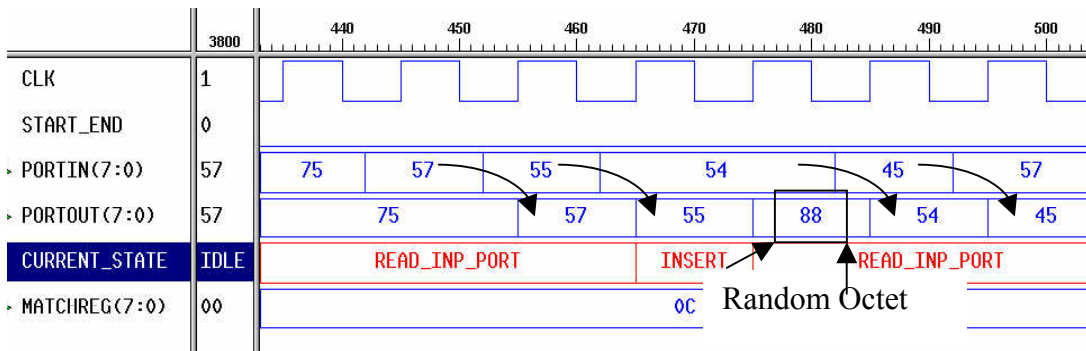


Figure 14: Cipher-text randomization showing random insertion

Signal clock and Reset, *PortIn*, *CURRENT_STATE*, *Count1* and *Count2* function in a similar manner as described in the previous section. Signal *T_start_end* is input to the system, which shows that previous system has completed the task. Signal *Port_read* is output signal that indicates to the previous unit that data has been read (if it is high) else data is required to be held at the input port for next clock. On every clock transition, reset input is sampled and on high reset system goes into *IDLE* state. A high on *T_start_end* denotes the completion of the previous stage and the start of the Cipher Randomizing module. Current state is *IDLE* due to high Reset

input (see Figure 12). In the *IDLE* state all the registers and counters are reset. At low Reset, input system waits for *Tstart_end*, which is high on completion of the previous stage. On high *Tstart_end*, the system goes into *INITIAL* state where it finds if the insertion point is at the very beginning or not by reading and comparing second octet of current set of RC4 with the counter1 value. If the first position is the insertion point then the system directly goes into *INSERT* state, else it goes into *READ_PORT* state where it reads a byte from the port and makes available the randomized output to the next unit.

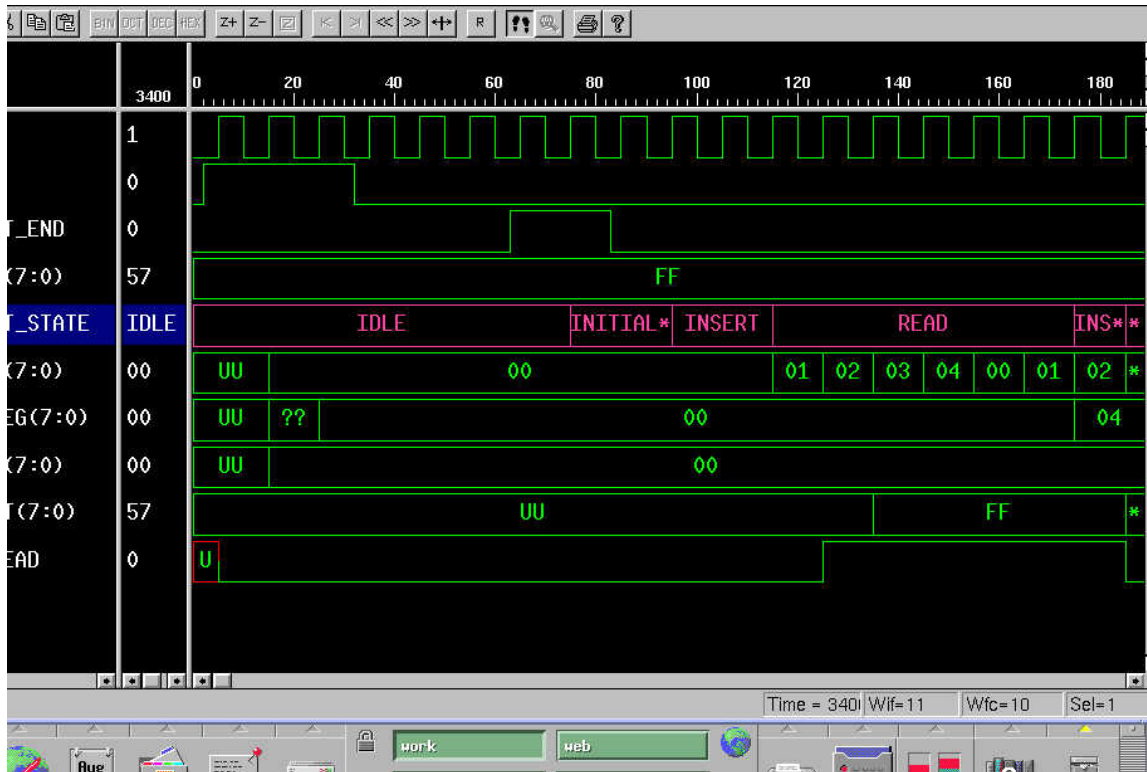


Figure 15: Cipher Randomizing (idle, initialize, insert, read states)

The system then remains in the *READ_PORT* state and continues transmitting the input port data as long as it does not encounter the next insertion point; and then again the system goes into the

INSERT state if it is not the end of the packet. In the *INSERTING* state, the system transmits the random byte rather than the input port byte and forces the read_port signal to 0 thereby acknowledging the source of the data that the byte has not yet been read and thus needs to be available during the next clock cycle. At the end of the packet, the system comes into *CIDONE* state, which requests the next unit to begin functioning. We use a four byte FIFO at the input port from which data is read, so that ‘end of the packet’ can be detected 4 bytes earlier, and thus the last 4 bytes can then be diverted to CRCrandomizer.

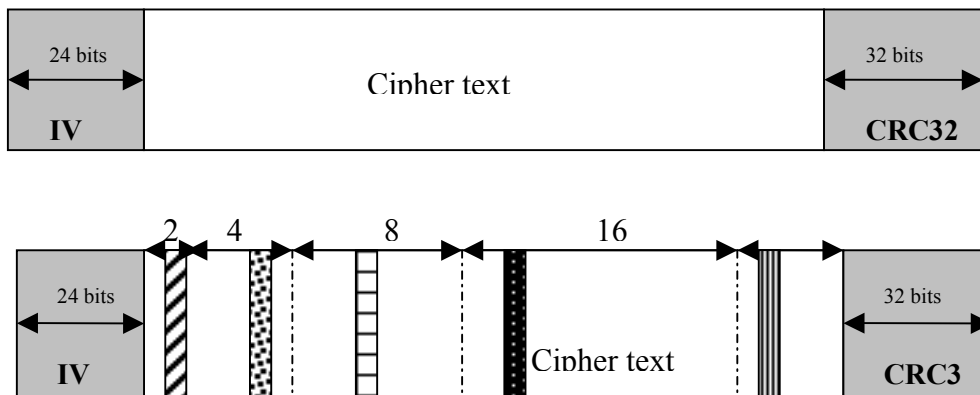


Figure 16: Cipher Randomizing a) before b) after application of Algorithm 2: Algorithm for cipher randomization / extraction

Figure 16 demonstrates the change in the cipher text due to the application of the Algorithm 2: Algorithm for cipher randomization / extraction.

CRC Randomization

The simulation results shown below demonstrate the CRC randomization using Algorithm 1: IV and ICV randomizations / extraction. This module works very similar to the IV

randomizer module. There are a few differences, them being: the Data at the input port is sampled for 4 consecutive clock cycles; unlike in the IV randomizer where it is 3 cycles during the *READ* state. *Data_holder* is a 32-bit, bit addressable register used to store a 4 byte (32 bits) CRC. *New_data_holder* is a 40 bit, bit addressable register used to store new, modified CRC (actual CRC + 8 bit unrelated data).

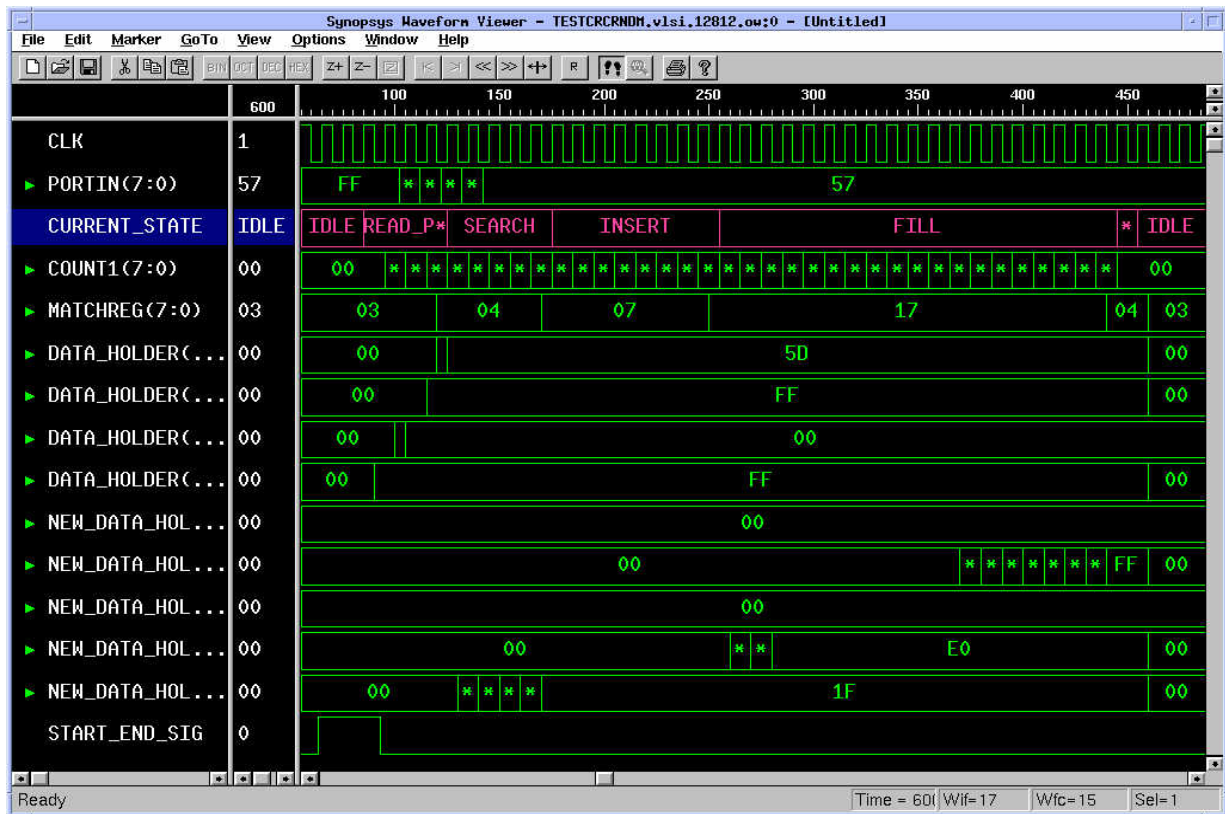


Figure 17: CRC Randomizing

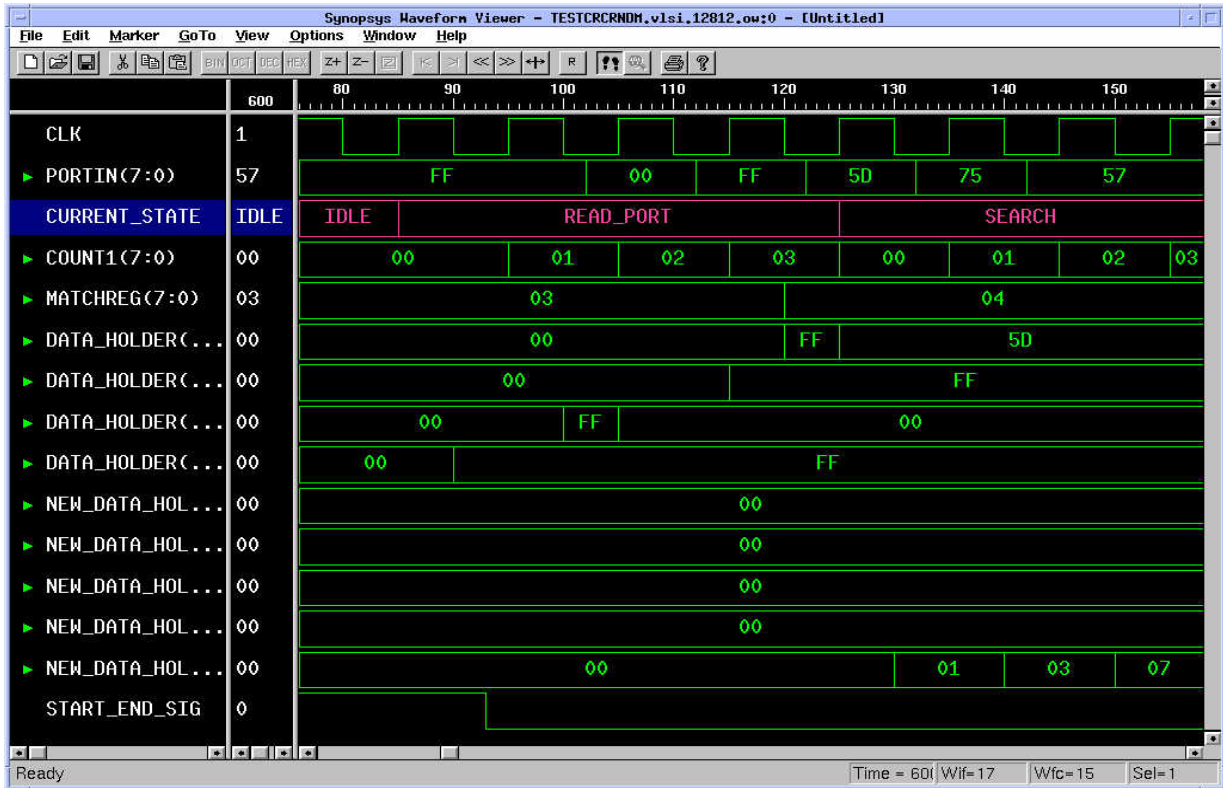


Figure 18: Zoomed view showing the IDLE, READ_PORT, SEARCH state transitions

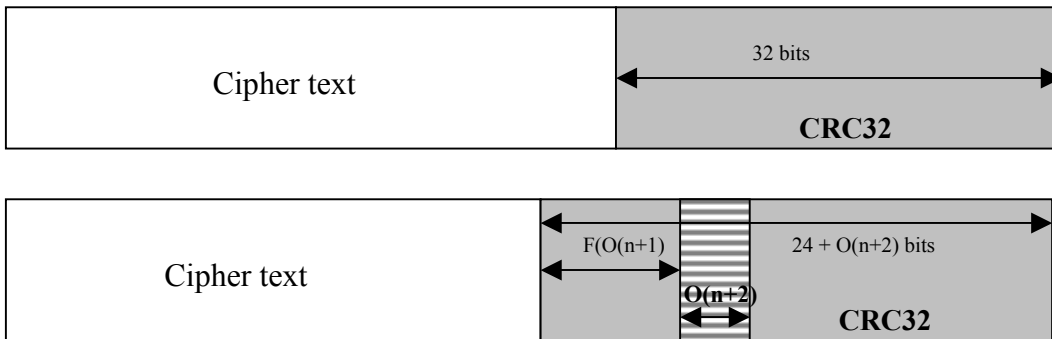


Figure 19: CRC Randomizing

The module functions similar to the IV randomizer. The application results in the change in the CRC as shown in the Figure 19.

The affect of the application due to these algorithms to IV, Cipher, CRC fields will result in the randomization of the whole MPDU as shown Figure 20.

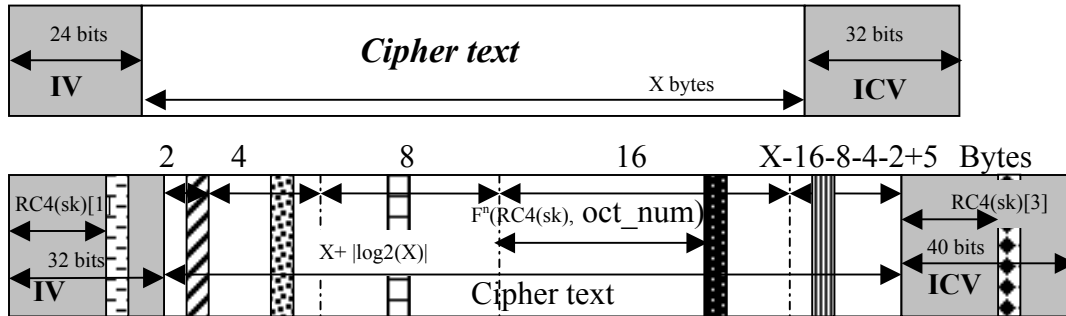


Figure 20: Packet formats for the WEP and modified scrambled WEP by the scrambling algorithm

Analysis

Algorithm 1

Insertion of an 8-bit random octet in 24-bit IV at any random position, obtained from Calculate_random_position function, results in 6144 (24×2^8) different patterns of the same IV. This means an attacker needs to analyze 6,144 more patterns to decrypt the message in case of an IV collision. Thus, the improvement in security is 6144 times for IV based attacks. The same improvement in the ICV based attacks is 8192 (32×2^8) times as ICV is 32 bits long.

Algorithm 2

Calculation of achieved randomization:

We insert 1 octet per chunk thus

No of octets inserted = No of chunks processed.

No of chunks processed = \log_2 (number of cipher-text octets processed)

C_P as a function of incoming octets can be written as

$$C_P(n) = \log_2(n) \quad (6)$$

Each insertion has s positions among which one has to be selected randomly where s = chunk size.

Before applying the algorithm, the size of a chunk C_S at position C_P can be calculated as

$$C_S(C_P) = 2^{C_P} \quad (7)$$

Since every chunk has one random octet in it, the total number of random octets inserted can be given as

$$R_o(n) = 1 + C_P(n) = 1 + \log(n) \quad (8)$$

For each chunk obtained randomization is:

$$Randomization(C_S) = C_S^{28}$$

For m chunks the total randomization will be

$$2^m \times 2^{(m-1)} \times \dots \times 2^1 = 2^{\left(\frac{m \times (m-1)}{2} + 8\right)}$$

Therefore the total improvement in security for cipher table based attacks $2^{\left(\frac{m \times (m-1)}{2} + 8\right)}$

The density of inserted random octets in the cipher-text can be given as

$$d_{Ro}(n) = \frac{Ro(n)}{n} \quad (9)$$

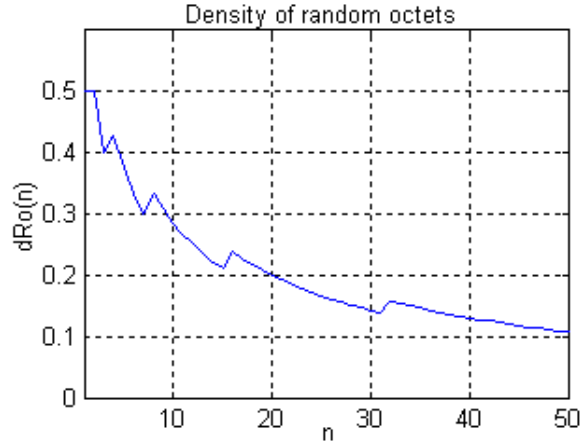


Figure 21: Density of random octets

As depicted in Figure 21, the density makes a peak at the first 2 bytes of the insertion, and then it reduces logarithmically. As it requires lesser computational power to retrieve original WEP cipher-text from small number of input octets, we use distributive algorithm to maintain high density of randomness at the beginning of the frame and reduce logarithmically over the length of the frame, using distributive algorithm.

The number of octets in the modified packet will be given by

$$N(n) = n + Ro(n) \quad (10)$$

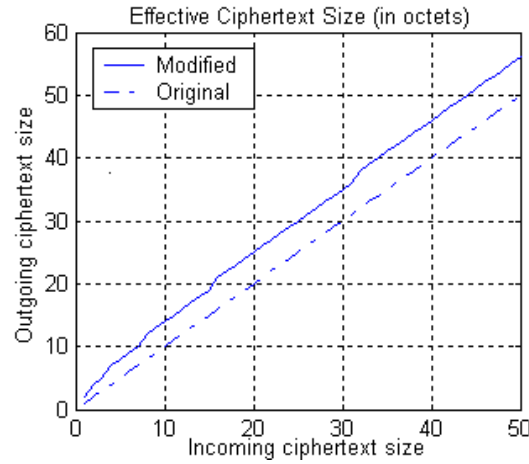


Figure 22: The number of octets in the modified packet Vs the number of incoming octets.

Figure 22 illustrates the change in the ciphered section of WEP packet due to the Algorithm 2: Algorithm for cipher randomization / extraction, which shows that for 50 input octets, the modified cipher will be 56 octets, and for 1024 octets it will be 1035 octets.

Now, let us calculate the probability of an intruder successfully retrieving the cipher-text stream from the scrambled WEP output.

First, the probability of finding a random octet in a chunk can be given as

$$P_{FR}(n) = \frac{1}{C_S(n)+1} \quad (11)$$

Then, the probability of successfully retrieving the whole cipher-text (i.e., breaking the scrambling algorithm for the cipher-text randomization - The probability of finding all random octets) will be

$$P_{SR}(n) = \prod_{i=1}^n P_{FR}(i) \quad (12)$$

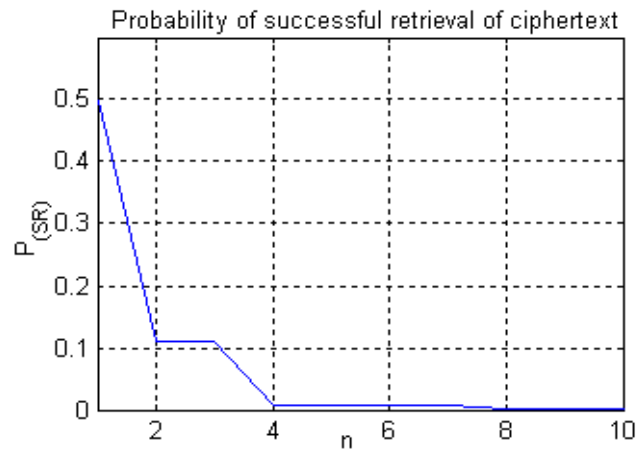


Figure 23: PSR versus n

As the number of cipher-text octets increases linearly, the probability of breaking the scrambling algorithm for the cipher-text randomization decreases exponentially. For the packets with a cipher-text of 5 octets or more, the probability becomes 0.00097 or less.

Computational Complexity: calculated in the terms of the number of different patterns generated for same data pattern.

Different patterns for the same 24-bit IV can be given as:

$$C_{x1}(n) := 2^8 \cdot 24 \quad (13)$$

Different pattern for the same n data bits can be given as:

$$C_{x2}(n) := (2^8 \cdot \log(n, 2)) \cdot C_p(n) \quad (14)$$

Different pattern for the same CRC can be given as:

$$C_{x3}(n) := 2^{8 \cdot 32} \quad (15)$$

Increased total Complexity difficulty can be given as:

$$C_x(n) := C_{x1}(n) + C_{x2}(n) + C_{x3}(n) \quad (16)$$

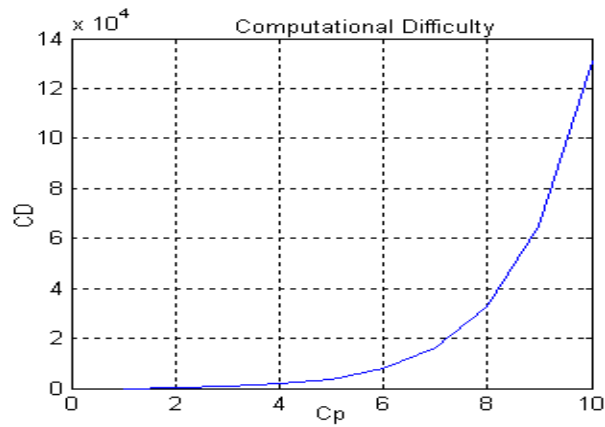


Figure 24: Computational Complexity and Chunk position

Curve shows the increase in the requirement of computational power to recover the WEP cipher-text from the scrambled cipher text.

Smart attack

Algorithm 1

In Algorithm 1 attacker will try to match the bits of IV with the already created IV bank. On a mismatch attacker will skip the next 8 bits and will try to match the remaining to the IV.

This attack will require at most 24 matching for IV and 32 for ICV for every match in the bank. But not only 255 different IVs can have same scrambled IV pattern but also same IVs can have 24 same scrambled patterns.

For example, if an attacker got IV_x, IV_y, IV_z as follows:

IV_x = 1101 0011 0101 1011 0010 0010 0101 1100

IV_y = 1101 0011 0101 1011 0010 0010 0101 1100

IV_z = 1101 0011 0101 1011 0010 0010 0101 1100

By bit comparison attacker can guess IV_x, IV_y, and IV_z are same

But actual contents at transmitter are

IV_x = 1101 1011 0010 0010 0101 1100

With random byte (0100 1101) inserting at 2nd position

IV_y = 1101 0011 0010 0010 0101 1100

With random byte (1101 0110) inserting at 6th position

IV_z = 1101 1011 0010 0010 0101 1100

With random byte (0011 0101) inserting at 4th position

Here IV_x and IV_z are generated from same IV and IV_y has generated from different IV, but attacker cannot figure it out and has to analyse for all the cases

IV_x = 1101 0011 0101 1011 0010 0010 0101 1100

IV_y = 1101 0011 0101 1011 0010 0010 0101 1100

IV_z = 1101 0011 0101 1011 0010 0010 0101 1100

This demonstrates that not only 24 cases are needed to analyze as that will match the IV_x to IV_y. Attacker has to match 24 cases of every match in the bank (255) that will be 24*255 cases. Also we have assumed that IV bank has created already, but lot more problems are

involved in building the bank, as to find out the received IV1 and IV2 are different or same. As both results in same scrambled IV that again creates problem if we have IV collision as IV1 and IV3, which attacker does not know and has to make different row in the bank for every one of these input IV or ICV.

Algorithm 2

In Algorithm 2 attacker will try to reveal the random bits used in the previous chunk. As for the new chunk all the bits used to point random positioning in the previous chunk are used and one bit is appended to the random chunk pointer, an attacker can only look for the new bit in the chunk and by analyzing the packet chunk by chunk attacker may have less complexity. If attacker successfully guesses the random byte in all the chunks, its computational difficulty will be 2^{Cp} . And if cannot guess the random byte correctly in any chunk than it will be $(2^{(Cp*(Cp-1)/2})$

But in real case an attacker has to analyze all the cases (max computational difficulty) as he does not know before hand that the result of guess is correct, even if he has guessed correctly.

For example, if the attacker assumes that the first byte of chunk 0 is random (when bit 0 of current used set of RC4 key-stream (RC4(SK)[set][2][0]) is 0) and analyzes the next chunk accordingly, the next chunk says either 0th or 2nd position could be random insertion.

Thus, it gives

2 insertion positions will be for the first chunk

2 insertion positions will be for the next chunk if 1st guess is correct

2 insertion positions will be for the next chunk if 2nd guess is correct

2 insertion positions will be for the next chunk if 3rd guess is correct

2 insertion positions will be for the next chunk if 4th guess is correct

And so on...

That will result in 2^{C_p} conditions if all the guesses are correct

If the 2nd guess is incorrect than

2 insertion positions will be for the next chunk first chunk

2 insertion positions will be for the next chunk 1st guess is correct

4 insertion positions will be for the next chunk if 2nd guess is incorrect

8 insertion positions will be for the next chunk as 3rd guess is uncertain

16 insertion positions will be for the next chunk as 4th guess is uncertain

And so on, that gives $2 * 2 * 2^{(4-2)} * 2^{(4-3)}$ patterns to match

Thus, if the zth guess is incorrect than total computational difficulty will be

$$2^z * (2^{(C_p-Z)} \dots 2^1) \quad (17)$$

But as attacker does not know before hand that his guess is correct so he has to analysis all the cases obtained from eq 17.

CONCLUSIONS AND FUTURE WORK

A novel, scrambling algorithm is proposed to patch WEP protocol. The algorithm is implemented in MATLAB and VHDL and is verified through simulations. Mathematically, it achieves an aggregate of $6144 + 8192 + 2^{((m \times (m-1)/2) + 8)}$ times improvement in WEP security for table based attacks, where m is the number of chunks in cipher-text. The hardware implementation of the algorithm requires only adding two counters, few registers and a simple controller. Thus the algorithm provides a robust WEP security system without substantially increasing the overall implementation cost.

This implementation increases the data security in various aspects (integrity, confidentiality) by a degree of thousand, with very little change in hardware and the packet size.

This implementation is compatible with WEP. It is integrated into the WEP protocol, so that it could either replace the WEP processor or work as a pre and post processor of WEP. It can also easily incorporate other protocols to change the Secret Key dynamically after a certain number of packet transmissions.

Issues of concern that can be further researched on are:

- Optimization of IV/ CRC32 algorithms for insertion of random octets at random points.
- Utilization of inserted random octets in error recovery or data compression.
- To incorporate TKIP into the algorithm.
- To use byte-wise operation on IV and CRC, which currently uses bit-wise.
- Other insertion approaches such as octet shuffling (trade off between size and efficiency) rather than the interpolation process as discussed in this thesis.

LIST OF REFERENCES

- [1] Std 802.11b, *IEEE*, 1999.
- [2] Borisov, N. Goldberg, I. and Wagner D., “Intercepting Mobile Communications: The Insecurity of 802.11”, *Proc. of the 7th Annual International Conference on Mobile Computing and Networking, July 2001*. <http://www.isaac.cs.berkeley.edu/isaac/wep-draft.pdf>
- [3] Walker, J. “Unsafe at any key size; an analysis of the WEP encapsulation”, *IEEE Document 802.11-00/362, Oct. 2000*.
- [4] Eason, G. Noble, B. Sneddon, I. “The RC4 Encryption Algorithm”, *RSA Data Security, Inc., 1992*.
- [5] [http://www.filibeto.org/sun/lib/networking/internetworking_technology_overview/Open%20Systems_Interconnection_\(OSI\)_Routing_Protocol.pdf](http://www.filibeto.org/sun/lib/networking/internetworking_technology_overview/Open%20Systems_Interconnection_(OSI)_Routing_Protocol.pdf)
- [6] <http://www2.rad.com/networks/1994/osi/layers.htm>
- [7] <http://www.pcsupportadvisor.com/nasample/t04124.pdf>
- [8] The editors of IEEE 802.11, “Wireless LAN medium access control (MAC) and physical layer (PHY) specifications,” *Draft Standard IEEE 802.11, P802.11/D1: 1997*.
- [9] Chen, K. “Medium access control of wireless LANs for mobile computing,” *IEEE iVetwork, vol. 8, no. 5, pp. 50–63, Sep. 1994*.
- [10] Chhaya, H. Gupta, S. “Throughput and fairness properties of asynchronous data transfer methods in the IEEE 802.11 MAC protocol” *Personal, Indoor, Mobile, and Radio Communication Conference, 1995*.
- [11] Diepstraten, W. “A wireless MAC protocol comparison” *IEEE P802.11-92/51*.
- [12] Weinmiller, J. Woesner, H. Wolisz, A. “Analyzing and improving the IEEE 802. 11-MAC protocol for wireless LANs,” *MASCOTS '96, San Jose, USA, Feb. 1996*.
- [13] <http://www.clark.net/timw/vpn/Tech/vpn.pdf>
- [14] Venkateswaran, R. “Virtual private networks” *IEEE, March 2001*.
- [15] Brady, J. “Virtual private networking-the flexible approach” *IEE Colloquium on IEEE, 25 Jun. 1997*.
- [16] Borisov, N. Goldberg, I. Wanger, D. “Security of the WEP Algorithm”, *University of California at Berkeley; Feb. 2001*. <http://www.isaac.cs.berkeley.edu/isaac/wep-faq.html>
- [17] Borisov, N. Goldberg, I. Wanger, D. “Intercepting Mobile Communications: The insecurity of 802.11”, *University of California at Berkeley; Sep. 2001*, <http://www.isaac.cs.berkeley.edu/isaac/mobicom.pdf>
- [18] Shunman, W. Ran, T. Yue, W. Ji, Z. "WLAN and it's security problems", *PDCAT'2003, 29 Aug. 2003*.

- [19] Ishibashi, H. Yamai, N. Abe, K. Ohnishi, K. Matsuura, T. "A protection method against unauthorized access and address spoofing for open network access systems", *PACRIM. 2001, IEEE Pacific Rim Conference*, 28 Aug. 2001.
- [20] <http://sconce.ics.uci.edu/seminar/slides/signature.pdf>
- [21] Koc, C. "High-Speed RSA Implementation", *RSA Security*, Nov. 1994, <ftp://ftp.rsasecurity.com/pub/pdfs/tr201.pdf>
- [22] Robshaw, M. "Stream Ciphers", version 2.0, *RSA Security*, Jul. 1995. <ftp://ftp.rsasecurity.com/pub/pdfs/tr701.pdf>
- [23] Koc, C. "RSA Hardware Implementation", *RSA Security*, Apr. 1996. <ftp://ftp.rsasecurity.com/pub/pdfs/tr801.pdf>
- [24] Rivest, R. "RSA Security Response to Weaknesses in Key Scheduling Algorithm of RC4", *RSA Security*, Sep. 1, 2001. <http://www.rsasecurity.com/rsalabs/technotes/wep.html>
- [25] Carlson, J. "Cryptography", Nov. 17, 2001. <http://www.math.utah.edu/~carlson/scienceday/Cryptography.pdf>
- [26] Messerschmitt, D. "RSA Asymmetric Encryption", *University of California*, 1999.
- [27] Kim, Y. "Key Establishment Protocols", Sep. 4, 2001. <http://sconce.ics.uci.edu/seminar/slides/chap12.pdf>
- [28] Diffie W. "The First ten years of public key cryptography" in *SIMMONS, G.J. (Ed) 'Contemporary cryptology', IEEE press 1992.*
- [29] Rivest, R. Shamir, A. Adleman, L. "A method of obtaining digital signatures and public key cryptosystems", *Commun, ACM* 1978.
- [30] Dolev, D. Yao, A. "On the security of public key protocols". *IEEE Transactions on Information Theory*, IT, 1983.
- [31] Grecas, C. Maniatis, S. Venieris, I. "Towards the introduction of the asymmetric cryptography in GSM, GPRS, and UMTS networks", *Computers and Communications, 2001. Proceedings. Sixth IEEE Symposium*, 5 July 2001.
- [32] Okamoto, T. Uchiyama, S. Fujisaki, E. "Efficient probabilistic public-key cryptosystem based on the Diffie-Hellman problem", *Electronics Letters* pp. 326 - 327.
- [33] Abdalla, M. Bellare, M. Rogaway, P. "DHIES: An encryption scheme based on the Diffie-Hellman Problem" Sep. 18, 2001.
- [34] Yuan, H. "Investigation of the efficiency of the elliptic curve cryptosystem for multi-applications smart card" *KES '98*, 23 Apr. 1998.
- [35] Mohammed, Emarah, El-Shennawy, "A blind signature scheme based on ElGamal signature", *EUROCOMM 2000. Information Systems for Enhanced Public Safety and Security. IEEE/AFCEA*, 17 May 2000
- [36] Sutikno, S. Surya, A. Effendi, R. "An implementation of ElGamal elliptic curves cryptosystems", *IEEE APCCAS 1998*. 27 Nov. 1998.

- [37] Robshaw, M. Yin, Y. "Elliptic Curve Cryptosystems". *RSA Laboratories Technical Note June 27, 1997* http://www.rsasecurity.com/rsalabs/technotes/elliptic_curve.html
- [38] Nel, Kuhn, "Generation of keys for use with the digital signature standard (DSS)", 1993 *IEEE South African Symposium, 6 Aug. 1993*
- [39] Harn, L. "Modified key agreement protocol based on the digital signature standard", *Electronics Letters pp. 448 - 449, 16 Mar. 1995.*
- [40] Adam, J. "Data security-cryptography = privacy?" *Spectrum, IEEE, Aug. 1992.*
- [41] Bidder, A. Weiler, N. "Key Exchange (KX) - a next generation protocol to synchronise PGP Keyservers", *WET ICE 200., Twelfth IEEE International Workshops, 11 Jun. 2003.*
- [42] Gan, L. Simmons, S. Tavares, S. "A new family of stream ciphers based on cascaded small s-boxes", Electrical and Computer Engineering, 2001. *Canadian Conference, 16 May 2001.*
- [43] Tsoi, K. Lee, K. Leong, P. "A massively parallel RC4 key search engine", *Field-Programmable Custom Computing Machines, 2002., 10th Annual IEEE Symposium, 24 Apr. 2002.*
- [44] Rivest, R. "RSA Security Response to Weaknesses in Key Scheduling Algorithm of RC4", <http://security.ece.orst.edu/koc/ece575/rsalabs/rc4.pdf>
- [45] Meyer, C. "Cryptography-a state of the art review", *CompEuro '89., VLSI and Computer Peripherals. 12 May 1989.*
- [46] Piper, F. "Encryption", *ECOS 97, pp. 28-30 Apr. 1997.*
- [47] Zouridaki, C. "Security in Wireless Local Area Networks, Security in IEEE 802.11", *Technical Note, http://ece.gmu.edu/crypto/zouridaki/presentations/IEEE_802.pdf*
- [48] Dubrawsky, I. "Wireless (In)Security", *Technical Note, May 2002.* <http://www.unixreview.com/documents/s=2427/uni1020280760693/>
- [49] Black, R. "Fast CRC32 in Software", *Technical Note* <http://www.cl.cam.ac.uk/Research/SRG/bluebook/21/crc/crc.html>
- [50] Scott, R. "Understanding Cyclic Redundancy Check", *ACI Technical Support Technical Note* <http://www.4d.com/docs/CMU/CMU79909.HTM>
- [51] "Cyclic Redundancy Check (CRC)", http://www2.rad.com/networks/1994/err_con/crc.htm
- [52] Burke, K. "Wireless Network Security, 802.11/802.1X", *Technical Note* <http://www.cs.fsu.edu/~yasinsac/wns02/19b.pdf>
- [53] Wagner, D. "Wireless Security", *Technical Note, University of California, Berkeley,* <http://www.cs.berkeley.edu/~daw/talks/FCC02.ppt>
- [54] Letanche, O. "Proposed TKIP 48bit IV Frame Format", *Wireless LANs, IEEE P802.11Conf., 11 June, 2002.* <http://grouper.ieee.org/groups/802/11/Documents/DocumentHolder/2-281.zip>

- [55] Harkins, D. Lefkowitz, M. "TKIP Mixing Suggested Changes", *Wireless LANs,, IEEE P802.11Conf.*, 12 Nov., 2002. <http://www.ieee802.org/11/Documents/DocumentHolder/2-733.zip>
- [56] Yu, J. "From LAN (802.3) to Wireless LAN (802.11)", *IEEE Communications Society*, 12 Jan 2002. <http://www.ewh.ieee.org/r4/chicago/WLAN-Challenge.pdf>
- [57] Bailey, D. "Monterey 802.11i/802.15 Liaison Report", *Technical Report*, 13 Sep., 2002.
- [58] Moreton, M. "WPA Coordination Changes", *Wireless LANs,, IEEE P802.11Conf.*, 8 May. 2003.
- [59] Geier, J. "How WPA works", *Network World*, 31 Mar., 2003.
<http://www.nwfusion.com/research/2003/0331wpa.html>
- [60] IETF's IPsec draft specs <http://www.ietf.org/ids.by.wg/ipsec.html>
- [61] Singer, M. "Certicom Serves Up Wireless VPN Using 802.11b", *Silicon Valley Internet*, 19 Oct., 2001. http://siliconvalley.internet.com/news/article/0,2198,3531_907461,00.html
- [62] Perrig, A. Canetti, R. Tygar, J. Song, D. "The TESLA Broadcast Authentication Protocol", *RSA Laboratories, Volume 5, No. 2, Summer / Fall 2002*.
http://www.rsasecurity.com/rsalabs/cryptobytes/cryptobytes_v5n2.pdf