

**An Examination of Security Algorithm Flaws in Wireless Networks**

Erica Simcoe, Hirsh Goldberg, and Mehmet Ucal  
Advisor: Dr. Sennur Ulukus

## Table of Contents

1. Wireless Background.....	2
2. Introduction to IEEE 802.11.....	2
3. Wired Equivalent Privacy (WEP).....	4
3.1 Introduction to WEP.....	4
3.2 RC4 Algorithm.....	4
3.2.1 Key Scheduling Algorithm.....	4
3.2.2 Pseudo-Random Generation Algorithm.....	5
3.2.3 RC4 Example.....	5
3.3 WEP Implementation.....	6
4. Wireless Attacks.....	8
5. WEP Implementation Problems.....	8
5.1 Attack Based on IV Reuse.....	9
5.2 Attack Based on Weak IVs.....	10
6. Simulated Attack.....	12
7. Future of Security in WLANs.....	14
8. Conclusion.....	16
9. Acknowledgements.....	16
10. Works Cited.....	17

## **1. Wireless Background**

In today's highly technology-driven world, people increasingly depend on fast, convenient methods to communicate with each other. This growing desire to share data quickly and reliably influenced significantly the development of present day computer networking technology. Initially networks were realized by a physical connection between each computer within the network. Recently, however, the appeal of achieving network communications without the financial and physical burden of the wires themselves has led to many notable innovations in wireless networking technology. As a result, more and more businesses as well as home users are turning to wireless local area networks (WLANs) because of their simplicity, adaptability, and most importantly, mobility.

The general structure of a wireless network, also called the topology, consists of an access point (AP) and several 'clients' or 'mobiles,' each of which is connected or 'associated' with the network. An access point acts as a bridge between a wireless network and a wired one. Wireless networks operate in one of two modes, ad-hoc mode and infrastructure mode. In infrastructure mode, the more common of the two modes, each mobile sends all the information it wishes to send through a central location, the AP [1]. The AP then processes this information and sends it to the proper location; this can either be a wired network which the AP is connected to, or it can be another mobile within the WLAN. Communication with other networks is done solely through the AP; the AP serves essentially as the leader of the WLAN. This mode has applications mainly in corporate settings where employees need to access office equipment such as printers, fax machines, or file servers. In the ad-hoc mode, each mobile is able to communicate directly with any other mobile as long as a data link is set up between the two [1]. Communication links are usually formed spontaneously and only exist for as long as the two clients wish to communicate with each other. Notice that this mode contains no AP.

## **2. Introduction to IEEE 802.11**

Since its emergence, wireless fidelity (Wi-Fi) has become one of the most popular and convenient choices of data communication in today's world. Consequently many protocols governing wireless communications have been developed, and each is unique in how it defines the regulation of data transmission. The most common wireless protocol is IEEE 802.11 which is certified by the Institute of Electrical and Electronics Engineers (IEEE). This is a subcategory of the IEEE 802 standard, which more generally refers to all local area networks (LANs). Specifically, IEEE 802.11 is a set of rules governing devices which operate within the 2.4 GHz radio spectrum, which is part of the Industrial, Science, and Medical (ISM) frequency band. This frequency range is an unregulated band and thus can be used freely, without licensed approval, by anyone with proper equipment.

The IEEE 802.11 standard is broken down further into a set of sub-standards, whose variations will be discussed below. Currently the main sub-standards in the protocol are IEEE 802.11a, 802.11b, and 802.11g. The first one operates at 5 GHz and is the fastest at 54 Mbps. The second one, a revision of the original protocol and by far the most popular sub-standard, operates in the 2.4 GHz band at a maximum speed of 11 Mbps. Finally, the last one provides 20+ Mbps in the 2.4 GHz band. Several research groups are currently attempting to create improved sub-standards of the IEEE 802.11 protocol that address various concerns raised within the current sub-standards. Security issues are presently the biggest of these worries.

In order to understand underlying security matters the Open System Interconnection (OSI) model, which consists of seven different layers, must first be discussed. An illustration of this model is shown below in Figure 1 [2]. When transmission is commenced, data starts at the top of the OSI model at the application layer and is handed off to each successive level until finally it reaches the physical level. Once the data is received, it begins traveling back up the layer structure until it ultimately reaches the application layer on the receiving end. Naturally, the physical layer in the model represents the actual wires that are used to connect devices in a conventional network. In order to capture information from this network, a hacker would have to tap into the physical confines of the wire and then work his way up to any level of the chart he wants in order to achieve data interception.

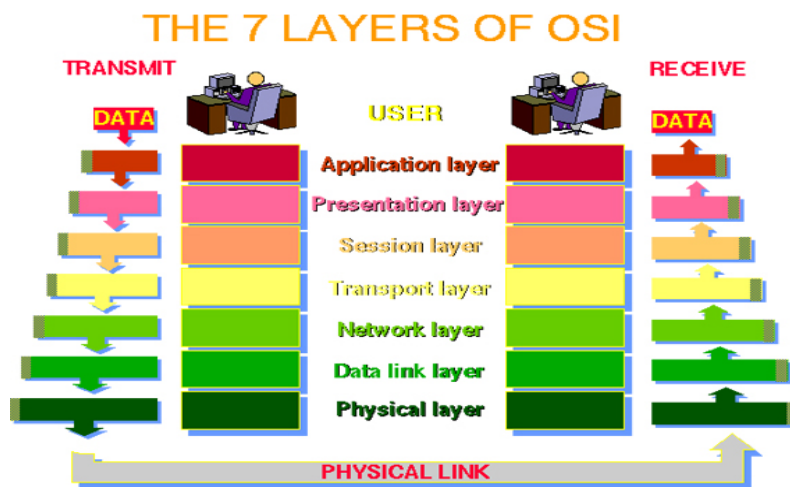


Figure 1: OSI Network Layers

The physical layer is much more vulnerable in a wireless network because the physical medium used to transmit data is simply the air which is open for anyone to access. It is for this reason that wireless networks are significantly harder to secure. Since the physical layer is so much more susceptible to an attack, it is very important that link layer protection is sound. The link layer is responsible for controlling user authentication and data encryption. It is subdivided into two tiers – Logical Link Control (LLC) and Medium Access Control (MAC). With weak link layer protection, it becomes much easier for a hacker to break into the network and capture data. The IEEE 802.11 standard provides guidelines regarding specifically how this link layer protection is to be implemented.

While IEEE 802.11 has no physical layer in the sense of wires and cables, it does still transmit and receive data frames via the ether. It accomplishes this using different spectrum modulation techniques. Spread spectrum is a data transfer technique that attempts to reduce interference by using a much larger bandwidth than is really needed. The three different types used primarily in IEEE 802.11 are Frequency Hopping Spread Spectrum (FHSS), Direct Sequence Spread Spectrum (DSSS), and Infrared (IR). Each one utilizes different data transfer speeds. Doing this reduces the peak power while keeping the overall power the same. The IEEE 802.11 standard also specifies rules for the Medium Access Control layer. The most notable rule is Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA). This service is “a mechanism for sensing whether the medium is in use before transmitting” [3]. If the medium is

already transmitting data, the station wishing to send data will wait for a random amount of time and then attempt to re-transmit.

### **3. Wired Equivalent Privacy (WEP)**

#### **3.1 Introduction to WEP**

Currently most WLANs using the IEEE 802.11 standard defend their network using the Wired Equivalent Privacy (WEP) protocol. The primary purpose and goal of employing WEP is to ensure that confidential data is protected from potential attackers, in a similar manner that wired LANs are protected. This is done by guaranteeing data confidentiality and data integrity.

WEP utilizes the RC4 stream cipher algorithm, which is a type of pseudo-random number generator (PRNG), to encrypt and decrypt the data passing through the network. Because it is at the heart of WEP's encryption and decryption process, the RC4 algorithm will be explained in more depth in the next section. This will be followed by a detailed description of WEP's implementation.

WEP is dependent upon a shared key, which is manually entered into every node of the WLAN. This shared key can be 40-bits long or 104-bits long [4]. Because it must be physically entered into each piece of equipment, it is seldom changed. This shared key must remain secret so that the message cannot be compromised by attackers. WEP also relies on a 24-bit initialization vector (IV) for part of the encryption process, which is generated via the wireless PCMCIA card. The IV prepended to the shared key is known as the secret key. The purpose of the IV is to allow the use of the same shared key repeatedly without running the risk of producing the same pseudorandom stream twice. The way WEP manages IV use is one of its biggest flaws and is discussed in depth in Section 5. It should also be observed that WEP includes an integrity check vector (ICV) to ensure that the data in each packet of information is not changed between the transmission and reception path [4]. The ICV is obtained by sending the message through a 32-bit cyclic redundancy check (CRC-32) algorithm.

#### **3.2 RC4 Algorithm**

As mentioned above, the RC4 algorithm is utilized for encryption purposes by the WEP security protocol. RC4, also known as "Ron's Code #4" or "Rivest", was developed at RSA Laboratories in 1987 by Ron Rivest. Originally, the code was kept as a trade secret. However, in 1994, the source code was mysteriously leaked to an online mailing list, and was not a secret anymore [5].

RC4 is a stream cipher, a cipher which encrypts each and every byte of data being sent one byte at a time. In addition, it is a symmetric key algorithm, an algorithm that uses the same key to both encrypt and decrypt the information. In general, RC4 generates a random string (usually 8 bits) which is then XORed with the plaintext byte being sent to produce a ciphertext character. The RC4 encryption algorithm is made up of two major sub-algorithms, the Key Scheduling Algorithm (KSA) and the Pseudo Random Generation Algorithm (PRGA). A more detailed explanation follows.

##### **3.2.1 Key Scheduling Algorithm**

The KSA uses a secret key, which in the case of WEP is the IV concatenated with the shared key, along with an initial state array S to generate a randomly scrambled array. This

randomly scrambled array is simply a permutation of the initial state array S. WEP uses 8-bit RC4, therefore, the KSA consists of an array S that holds  $2^8$  or 256 values [5]. The following algorithm is adapted from [6].

*Initialization-* In this portion of the KSA the initial state array S[i] is generated. S[i] has the form [0,1,2,...253, 254, 255].

$$\begin{aligned} i &= 0 \text{ to } 255 \\ j &= 0 \\ S[i] &= i \end{aligned}$$

*Scrambling-* In this portion of the KSA, the initial state array S[i] is permuted based on the keystream. Below, k[i] represents the keystream and L is the length of the keystream.

$$\begin{aligned} i &= 0 \text{ to } 255 \\ j &= (j + S[i] + k[i \bmod L]) \pmod{256} \\ &\text{swap}(S[i], S[j]) \end{aligned}$$

### 3.2.2 Pseudo-Random Generation Algorithm

The PRGA outputs a streaming key based on the random state array S that was generated by the KSA. It is able to do this by continually shuffling the random permutation stored in S, choosing a different value from the S permutation as the output each time. The following algorithm is adapted from [6].

*Initialization-* This portion of the PRGA simply sets the loop counters to zero.

$$\begin{aligned} i &= 0 \\ j &= 0 \end{aligned}$$

*Generation Loop-* After one generation loop, an 8 bit (1 byte) keystream S[t] is generated. The message  $m_i$  is XORed with this keystream to produce the ciphertext  $C_i$ .

$$\begin{aligned} i &= (i + 1) \pmod{256} \\ j &= (j + S[i]) \pmod{256} \\ &\text{swap}(S[i], S[j]) \\ t &= (S[i] + S[j]) \pmod{256} \\ C_i &= m_i \oplus S[t] \end{aligned}$$

### 3.2.3 RC4 Example

Below is a simple 4 byte example from [7] that illustrates how the RC4 Algorithm works.

**KSA**  
*Initialization*  
i = 0 to 3  
j = 0  
S[i] = [0,1,2,3]

$$k[i] = [1,7,1,7]$$

*Scrambling*

First Iteration ( $i = 0, j = 0, S = [0, 1, 2, 3]$ ):

$$j = (j + S[i] + k[i]) = (0 + 0 + 1) = 1$$

Swap  $S[0]$  with  $S[1]$ :  $S = [1, 0, 2, 3]$

Second Iteration ( $i = 1, j = 1, S = [1, 0, 2, 3]$ ):

$$j = (j + S[i] + k[i]) \pmod{4} = (1 + 0 + 7) \pmod{4} = 0$$

Swap  $S[1]$  with  $S[0]$ :  $S = [0, 1, 2, 3]$

Third Iteration ( $i = 2, j = 0, S = [0, 1, 2, 3]$ ):

$$j = (j + S[i] + k[i]) = (0 + 2 + 1) = 3$$

Swap  $S[2]$  with  $S[3]$ :  $S = [0, 1, 3, 2]$

Fourth Iteration ( $i = 3, j = 3, S = [0, 1, 3, 2]$ ):

$$j = (j + S[i] + k[i]) \pmod{4} = (3 + 2 + 7) \pmod{4} = 0$$

Swap  $S[3]$  with  $S[0]$ :  $S = [2, 1, 3, 0]$

**PRGA**

*Initialization*

$$i = 0$$

$$j = 0$$

*Generation Loop*

Recall  $S = [2, 1, 3, 0]$

$$i = (i + 1) \pmod{4} = 1$$

$$j = (j + S[i]) \pmod{4} = 0 + 1 = 1$$

Swap  $S[i]$  and  $S[j]$ :  $S = \{2, 1, 3, 0\}$

$$t = [S[i] + S[j]] \pmod{4} = 2$$

$$S[t] = S[2] = 3$$

and so on . . .

### 3.3 WEP Implementation

The first step in the WEP process is the checksum. WEP takes the data and passes it through the CRC-32 algorithm, to produce the ICV. This value is attached to the end of the data. The final plaintext is of the form data+ICV. It is this combination that goes through the encryption and decryption process [8].

The next step WEP completes is encryption. The IV along with the shared key, are used by the RC4 algorithm to generate a keystream. This keystream is simply a long string of pseudorandom bits, which are a function of both the IV and shared key. The keystream and plaintext are XORed together, resulting in what is known as the ciphertext. This ciphertext, along with the unencrypted IV and a bit denoting that the ciphertext is a WEP encrypted packet, is sent over the radio link. The IV must be sent unencrypted because it is used as a part of the decryption process [8]. A flowchart depicting this process is illustrated below in Figure 2.

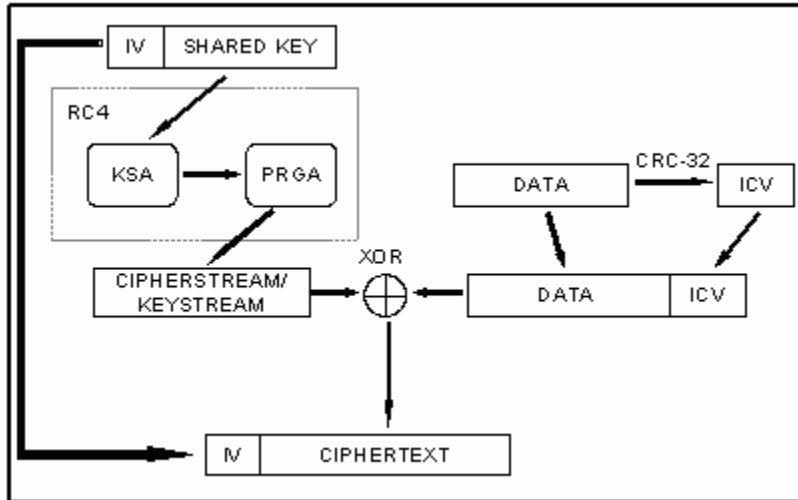


Figure 2 – WEP encryption flow chart

The decryption process is quite simply the reverse of the encryption process. The mobile receiving the encrypted packet splits the information into the ciphertext and IV. The IV is then concatenated with the shared key to form the same secret key that was used to encrypt the data. The secret key is then passed through the RC4 algorithm and the original keystream is regenerated. This keystream is XORed with the ciphertext to produce the data and ICV. Once separated, the data is sent through the CRC-32 algorithm and a new ICV is formed. If the two ICVs match, the packet is accepted [8]. This process is illustrated in another flowchart, Figure 3.

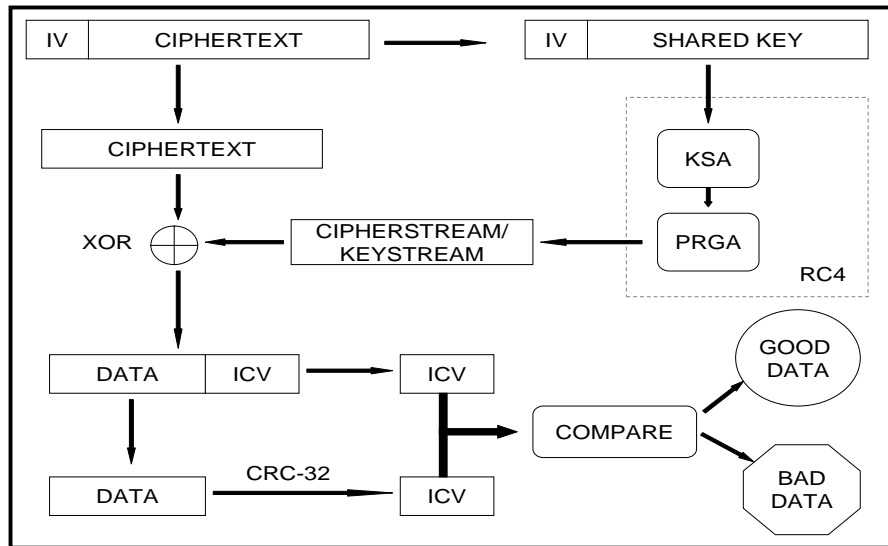


Figure 3 – WEP decryption flow chart

#### 4. Wireless Attacks

Attacks against WLANs can be grouped into two main categories: active and passive. “A passive attack is an attack in which an unauthorized party gains access to an asset but does not modify its content or engage in communication with any node in the network” [9]. Passive attacks involve eavesdropping and traffic analysis. Eavesdropping is when the attacker monitors

packet transmissions for message content. A good example of eavesdropping is a person sniffing packets and essentially “listening” to packet transmission on a WLAN between two devices. This type of attack is extremely easy and commonplace today especially with the advent of a wide variety of free, easily accessible wireless sniffers. In traffic analysis, the attacker gains valuable information by analyzing the traffic flow for patterns of communication. “A considerable amount of information is contained in the flow of messages between communicating devices” [9]. A passive attack is normally undetectable, while an active attack, can usually be detected.

Even though it is possible for one to detect an active attack, that does not mean an active attack is preventable. “An active attack occurs when an unauthorized party makes modifications to a message, data stream, or file” [9]. More specifically, some form of communication is set up between an attacker and one or more nodes in the network. Essentially, this attack involves changing data in the packet(s). Active attacks fall into four categories: masquerading, replay, message modification, and denial of service. Masquerading is when “an attacker impersonates an authorized user and thereby gains certain unauthorized privileges” [9]. In a replay attack, the attacker intercepts packets, and then proceeds to retransmit the messages while posed as a legitimate network user [9]. Message modification is when an attacker alters a “legitimate message by deleting, adding to, changing, or reordering the packet data” [9]. In a denial of service attack, the attacker prevents or prohibits the normal use or management of communication facilities.

These attacks help demonstrate that none of the goals of WEP are achieved since confidentiality and integrity can all be broken using one or more of these attack methods. Thus, WEP cannot be completely trusted for security. While active attacks can often be the most dangerous and fatal kind of attacks, passive attacks are usually the most popular since they do not require as much technical knowledge to be carried out and there is less risk of getting caught. The information that can be gathered as a direct result of these passive attacks can then be used to perform many of the active attacks. This project, as well as the rest of this paper, concentrates on a variety of ways a passive attack can be carried out.

## 5. WEP Implementation Problems

Although WEP does offer a reasonable level of security against a hacker with very little experience, the protocol is certainly not immune to attacks from more sophisticated hackers. An attacker with knowledge about the details of the protocol can break the WEP algorithm and obtain access to transmitted data. Unfortunately, WEP has several design flaws that significantly affect the security of the protocol, specifically in its implementation of the RC4 algorithm.

One of the most obvious of these design flaws is the algorithm WEP uses to produce the ciphertext. Reiterating from above, we see that the following equation is used:

$$\text{Ciphertext} = \text{Plaintext} \oplus \text{Keystream}$$

Using properties of the XOR function, we see that this is equivalent to:

$$\text{Keystream} = \text{Ciphertext} \oplus \text{Plaintext}$$

The significance of this is that if a hacker is able to obtain some plaintext before it is encrypted, he is then able to XOR that plaintext with the captured ciphertext to obtain the keystream. This

fact, along with WEP's tendency to reuse keystreams, makes the algorithm significantly more vulnerable to an attack.

Obtaining all of the plaintext is not necessarily an easy task; however, some properties concerning the way packets are sent often make it significantly simpler to determine a portion of the plaintext. When a packet is sent across a wireless network, a specific pre-defined header is attached that contains information which helps the packet travel to the right destination. Because of the way the header protocol is defined, the information is always in the same place. In many cases, the hexadecimal value 0xAA, a part of the SNAP encapsulation header, is the first plaintext byte of the packet and is thus the first byte XORed [10]. Using this information, a hacker can then determine the password.

By exploiting another design flaw involving the implementation of the 32-bit cyclic redundancy check (CRC-32), it is possible for an attacker to flip selected bits of the message, and still have the message pass the ICV test [11]. Thus, the contents of an important message could be compromised with the recipient having no knowledge of the attack. Because the attacker presumably inserts known data, this is another way in which a hacker can determine plaintext before it is encrypted.

The design flaws mentioned above are common; however, the two most important flaws which ultimately lead to WEP's vulnerability are detailed in the following two sections.

### 5.1 Attack Based on IV Reuse

One significant design flaw concerns the length of the initialization vector (IV). The IV is 24 bits long; therefore, there are  $2^{24}$  different IVs. This may seem like a large number, but a simple analysis reveals that even if a different IV is used for each successive packet, the entire IV space will be used up extremely quickly. For example, an access point sending 1500 byte packets at an average of 11Mbps, will run out of the available space in about 5 hours [4]. Even worse, some wireless network cards reset the IV to zero each time they are reinitialized, and increment the IV by one for each packet transmitted [12]. This makes the possibility of collisions among low IVs, even more likely. With enough IV collisions an attacker can collect sufficient information to use statistical analysis or even logical guesses to decipher the shared key. Once the IV space is entirely exhausted, WEP provides no key management algorithm and thus, IVs are almost certain to be reused.

The easiest way to achieve this attack is to intercept wireless traffic, trace the IEEE 802.11 frames and XOR the ciphertexts that are produced with the same IV. For example, let us say we have two plaintexts to be transmitted ( $D_1$  and  $D_2$ ), and we cipher them using RC4, with the same keystream  $K_x$  (same IV):

$$D_1 \oplus K_x = P_1 \quad D_2 \oplus K_x = P_2$$

Assuming the attacker intercepts the packets  $P_1$  and  $P_2$ , and then all one has to do is a simple XORing [10]:

$$P_1 \oplus P_2 = (D_1 \oplus K_x) \oplus (D_2 \oplus K_x) = D_1 \oplus D_2$$

From this,  $D_1$  and  $D_2$  can be recovered by performing statistical analysis. In addition to statistical analysis, WEP's checksum can be used to referee among guesses for the few that cannot be eliminated by probabilities. Also, intercepting a third or a fourth frame under the same IV can

reduce the possibilities for  $D_1$  and  $D_2$  even more. In this way, the attacker can recover the plaintext and the keystream without any knowledge about the plaintext in advance.

After recovering the plaintext of an intercepted message, the attacker, by performing statistical analysis or through other means, also knows the value of the keystream used to encrypt the message. At this point, the attacker can use this keystream to decipher all the packets ciphered with the same IV. Over time, the attacker can build a decryption table of keystreams corresponding to each IV. Since WEP frames are small and because there are only  $2^{24}$  possible IVs, the full table has a relatively small space requirement, roughly 24 GB. An example of a decryption table from [7] is shown below.

$IV_1$	Cipherstream <sub>1</sub> ( $K_1$ )
$IV_2$	Cipherstream <sub>2</sub> ( $K_2$ )
...	...
$IV_x$	Cipherstream <sub>x</sub> ( $K_x$ )
...	...
$IV_{2^{24}}$	Cipherstream <sub><math>2^{24}</math></sub> ( $K_{2^{24}}$ )

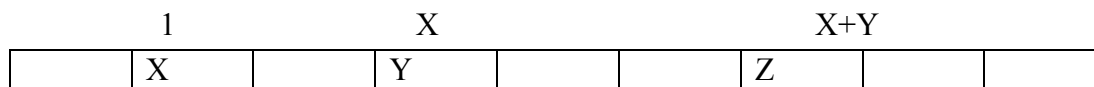
Once such a table is available, the attacker can immediately decrypt each subsequent ciphertext almost in real time with very little work.

In order to avoid this attack, the same keystream, should not be used twice. Notice that the key size is not important in this attack. The IV space has to be longer than 24 bits to avoid IV collisions in a short amount of time. However, as the size of the IV space gets longer, the size of the decryption table will get bigger. Making the IV space larger merely delays the IV collisions; IV reuse is inevitable. The shared key must be changed when all IVs are used but WEP does not provide key management protocol to do this and it is a trouble to change the shared key since it has to be done manually for each host in the network [7].

## 5.2 Attacks Based on Weak IVs

There are several weaknesses discussed in [6]. The one we will focus on is the existence of a large class of weak keys. The pseudo-random stream generated by RC4, known as the keystream, is strongly related to the IV, shared key combination, which we will refer to as the secret key. This correlation is a weakness that ultimately enables hackers to gain access to the shared key. Typically in WEP the key used to initialize the KSA is made up of the shared key, which is manually entered into every computer, and the IV, which is generated via the wireless PCMCIA card. In the case of WEP, the IV precedes the shared key. Also, the IV is sent as plaintext, along with the encrypted data. It can be shown that if the same shared key is used in conjunction with numerous different IVs, and the hacker can gain access to the first word or byte of the ciphertext corresponding with each IV, the hacker can eventually recover the shared key.

Cracking the key depends highly on knowing a portion of the plaintext. In the case of WEP, the first word is almost always the SNAP header, 0xAA, which is an IEEE-defined layer 2 encapsulation header format [8]. With this information, it is easy to determine which packets will be able to leak information about the secret key. When a packet is



resolved,  $i \geq 1$ ,  $X=S_i[1]$ , and  $X+Y= S_i [1]+ S_i [S_i [1]]$  [6]. In this case, there is probability greater than .05 that these values will not be swapped again, and in that situation,  $S[S[1]+S[S[1]]]$  will be the output of the first word [6]. Therefore information regarding the secret key can be obtained. Because there is a 5% chance that the packet is resolved, there is a 95% chance that swapping will occur. However, if enough packets with different weak IVs are obtained (60+), the actual value of  $S[S[1]+S[S[1]]]$  can be detected because it will be repeated [6].

Next will be a discussion regarding how it is possible to attack the RC4 algorithm when you have known IVs. First we make the assumption that the IV is  $I$  words long, and that we have a shared key of length  $l$  ( $K[0], K[1], . . . K[l-1]$  [6]. It is our goal to determine the value of a particular word or byte  $B$  of the shared key  $K[B]$ . This will be done by searching for IV values such that, after the first  $I$  steps of the algorithm,  $S_i[1] < I$  and  $S_i[1] + [ S_i[1]]= I+B$  [6]. After step  $I+B$  of the algorithm, it is highly probable that a resolved condition will be present and the output value will most likely be described by the equations below taken from [6].

$$OUT = S_{I+B-1}[j_{i+B}] = S_{I+B-1}[j_{I+B-1} + K[B] + S_{I+B-1}[I + B]]$$

$$K[B] = S_{I+B-1}^{-1}[out] - j_{I+B-1} - S_{I+B-1}[I + B]$$

Therefore, if we know the first byte or word of output along with the values of  $j_{I+B-1}$  and  $S_{I+B-1}$ , the value of word or byte  $B$  of the shared key can be determined. As mentioned above, this model works only 5% of the time, however collecting and testing enough IVs will inevitably allow one to produce the shared key. In the case of WEP  $I=3$  because the IV is a 3 byte, 24 bit stream, and  $l$  can be anywhere from 1-255. We look for IVs of the form  $[A+3, 255, T]$  for around 60 different values of  $T$ , where  $A$  is the index of the shared key byte currently being guessed and  $T$  can be any number, 0-255 [7]. There are about 9000 known weak IVs that fall into this category, and it takes between 2000 and 3000 in order to fully recover a 104 bit key [7].

The following is an example from [7] that illustrates how the first byte ( $A=0, B=0$ ) of the RC4 keystream is ascertained.  $Y$  is the input of RC4; the first three values form the IV, and  $Y[3]-Y[N-1]$  form the shared key so  $Y[3]$  is the shared key's first byte. A chart of  $Y$  is shown below.

3	255	T	Y[3]	Y[4]	Y[5]	...
---	-----	---	------	------	------	-----

$S$  is the initialized KSA permutation table shown below.

0	1	2	3	4	5	...
---	---	---	---	---	---	-----

$i_0=0$  and  $j_0=S[0] + Y[0]=0+0+3=3$  Therefore we swap the 0<sup>th</sup> and 3<sup>rd</sup> positions in  $S$ .

3	1	2	0	4	5	...
---	---	---	---	---	---	-----

$i_1=1$  and  $j_1 = j_0 +S[1]+Y[1]=3+255+1=259 \text{ mod } 256=3$  Therefore we swap the 1<sup>st</sup> and 3<sup>rd</sup> positions in  $S$

3	0	2	1	4	5	...
---	---	---	---	---	---	-----

$i_2=2$  and  $j_2 = j_1 + S[2] + Y[2] = 3 + 2 + T = 5 + T$  Therefore we swap the 2 and 5 + T positions in S. Remember T is known.

3	0	$S[j_2]$	$S[j_3]$	...	...	...
---	---	----------	----------	-----	-----	-----

$i_3=3$  and  $j_3 = j_1 + S[3] + Y[3] = (5+T) + 1 + Y[3] = 6+T+ Y[3]$  Therefore we swap the 3 and 6+T+ Y[3] positions in S.

At this point we have done all of the necessary KSA. There is a 5% chance that the current first four values of S will be the input of the PRGA. This is what we count on to be able to attack the system. Now we will examine what happens if the first four values are in fact the input to the PRGA.

The initialized PRGA permutation table is the same as the last KSA table.

3	0	$S[j_2]$	$S[j_3]$	...	...	...
---	---	----------	----------	-----	-----	-----

$i_0=1$  and  $j_0 = j_0 + S[1] = 0 + 0 = 0$  Therefore we swap the 1<sup>st</sup> and 0<sup>th</sup> positions in S.

0	3	$S[j_2]$	$S[j_3]$	...	...	...
---	---	----------	----------	-----	-----	-----

Now we can determine the first value of the shared key.  $S[S[i_0] + S[j_0]] = S[S[1] + S[0]] = S[3] = S[j_0] = S[6+T+Y[3]]$ . This value is equal to the keystream's first byte; which we will call KS. We can obtain the keystream's first byte by XORing the cipherstream's first byte with 0xAA, the first plaintext byte and SNAP header. The formula then becomes,  $KS = S[6+T+Y[3]]$ . Rearranging terms,  $Y[3] = S^{-1}[KS[0]] - 6 - T$ . In this case  $S^{-1}[p]$  is the index of S where p occurs and T is the third byte of the IV [7].

## 6. Simulated Attack

There are currently several research groups on the web that have used [6] to write various codes attacking WEP's weak IVs. The first known publicly available program that implemented this attack was WEPCrack. It has been documented that this program successfully cracked WEP and ascertained the shared key of a vulnerable wireless network. AirSnort was released shortly after WEPCrack, and it has become more widely known and used. AirSnort grabs and analyzes data moving across wireless networks. It operates by passively monitoring transmissions, and then computing the shared key when enough packets have been gathered. WEPCrack and AirSnort, both Linux based programs, demonstrate that wireless networks using WEP for security purposes are not secure at all; in fact, they are very vulnerable to unauthorized use of, and access to, their internal infrastructure.

It was our goal as a part of our research to implement the weak IV attack using one of the programs available for free online. We chose AirSnort for two main reasons. First, due to its popularity, it was better documented on the web. Second, there was a beta Windows version available online. After installing AirSnort on our Windows laptop we were able to get the

program to run, and capture packets, however it crashed the system after about 2 hours. This happened several times in a row, so we were unable to crack WEP on a Windows machine. As a result we were forced to obtain a Linux box and install AirSnort on it. We were able to successfully accomplish this and crack WEP.

An explanation and downloadable file of AirSnort can be found at [13]. Installing AirSnort is not an easy process, once the tar file is extracted, many more steps must be taken before the program will function. The most important part of installing AirSnort on a Linux machine is to ensure that the wireless driver is working properly. It is documented that the best PCMCIA cards to use with this program are PrismII and Orinoco, but others may work as well. Regardless of what brand the card is, it must be able to be placed in monitor mode so that it can passively sniff packets. For some cards this can be accomplished by downloading a patch from the AirSnort website, otherwise the driver code itself must be modified. Also, the most recent version of the library known as Libpcap must be installed on the Linux box. This library enables packets to be captured and analyzed by the AirSnort program. Once these processes are completed AirSnort should execute.

For our simulation we configured a vulnerable access point, with a Service Set Identifier (SSID) of mysterynet, to communicate with a laptop using 40-bit WEP encryption. We chose a shared key with a hexadecimal value of AA:BB:CC:DD:EE. From the laptop we ping flooded the access point to elevate the number of packets transmitted and received. Then we ran AirSnort on another independent laptop and watched as packets were collected from mysterynet. Two hours later, after about two million packets and 800 weak IVs the WEP key was cracked. A screen shot of the end result is shown below in Figure 4. This hack occurred so quickly because we were only simulating attacking a wireless network by ping flooding the AP. Cracking an actual wireless network would take longer depending upon the amount of traffic present. The end result of our simulation was as we expected; WEP is susceptible to weak IV attacks.

The screenshot shows the AirSnort application interface. At the top, there are controls for scanning (radio button) or channel selection (dropdown menu set to 6). The network device is set to wlan0 and the driver type is Host AP/Orinoco. The 40-bit crack breadth is set to 3 and the 128-bit crack breadth is set to 2. Below these controls is a table with the following columns: C, BSSID, Name, WEP, Last Seen, Last IV, Chan, Packets, Encrypted, Interesting, PW: Hex, and PW: ASCII. The first row, marked with an 'X', shows a successful crack for the 'mysterynet' SSID with a WEP key of AA:BB:CC:DD:EE. Other rows show various other SSIDs like 'umd', 'interauth-3g', 'nist', and 'interauth-wlan' with their respective statistics.

C	BSSID	Name	WEP	Last Seen	Last IV	Chan	Packets	Encrypted	Interesting	PW: Hex	PW: ASCII
X	00:06:25:50:52:4A	mysterynet	Y	Thu Jul 29 11:13:38 2004	7A:B3:2D	6	2932051	2901122	817	AA:BB:CC:DD:EE	.....
	00:40:96:47:92:3D	umd	Y	Thu Jul 29 11:13:38 2004	42:42:03	6	57642	27286	0		
	00:40:96:47:86:F9	umd	Y	Thu Jul 29 11:13:38 2004	AA:AA:03	6	68352	37375	0		
	00:40:96:47:86:F2	umd	Y	Thu Jul 29 11:13:38 2004	42:42:03	6	49279	16638	0		
	00:0F:90:14:F3:30	umd		Thu Jul 29 11:13:38 2004	00:00:00	6	18224	0	0		
	00:00:24:C0:3D:C4	interauth-3g		Thu Jul 29 11:13:38 2004	00:00:00	3	22497	0	0		
	00:60:1D:21:5E:E5		Y	Thu Jul 29 11:13:38 2004	54:4E:57	8	4523	444	0		
	00:00:24:C0:3A:F8	nist		Thu Jul 29 11:13:38 2004	00:00:00	7	11312	0	0		
	00:40:96:46:88:52	umd		Thu Jul 29 11:13:38 2004	00:00:00	6	25163	0	0		
	00:00:24:C0:3D:C0	interauth-wlan		Thu Jul 29 11:00:28 2004	00:00:00	11	8	0	0		
	FF:FF:FF:FF:FF:FF			Thu Jul 29 11:13:36 2004	00:00:00		2254	0	0		

Figure 4 – Screenshot of Successful WEP Secret Key Crack (Taken from AirSnort)

There are several measures that could be taken to avoid this attack. First, all weak keys could be filtered out, and discarded before they are used to encrypt data [7]. Second, the IV and shared key could be hashed, before passing them through the RC4 algorithm [7]. A hashing function can be defined as “a (mathematical) function which maps values from a large (possibly

very large) domain into a smaller range. The function satisfies the following properties: 1. it is computationally infeasible to find any input which maps to any pre-specified output; 2. it is computationally infeasible to find any two distinct inputs which map to the same output” [14]. Using a hashing function makes the IV significantly more secure. Another possible solution would be to discard the first several outputs of RC4 [7]. This would eliminate the correlation there currently is between the RC4’s input and the output’s first few bytes. The above solutions are just some simple actions that could be taken to make the IEEE 802.11b standard more secure than it currently is.

## **7. Future of Security in WLANs**

Although the solutions suggested in the previous section provide a higher level of security, WEP is still an easy target for serious attackers even after performing these extra safety measures. Thus, it can be concluded that WEP is easy to crack with the right tools and enough patience. The proposed precautions are only intended to alleviate the associated problems with WEP until an improved and much more sophisticated security algorithm is established.

For home and small office environments, WEP remains useful for deflecting eavesdroppers. Larger companies and users wishing to transmit highly classified information might wish to strengthen WEP by deploying it with other third-party security solutions. There are two important upgrades developed for WEP: Temporal Key Integrity Protocol (TKIP) and 802.1X. TKIP is an upgrade for the encryption standard while 802.1X is an upgrade for the authentication standard.

TKIP attempts to fix the well known problems of short encryption keys and small initialization vectors. This protocol, which is downward compatible with IEEE 802.11a, b, and g, still uses the RC4 algorithm so one can upgrade the existing hardware to support the standard. TKIP uses a 48-bit IV, whereas WEP uses 24-bit vectors. The longer IV space reduces the repetition of IV collision, meaning that it limits the cryptographic attacks. In this protocol, the IV is now encrypted instead of being sent in plaintext as it is in WEP. TKIP also utilizes a longer encryption key (128 bits) than WEP in order to address the short key problem.

Another important aspect of TKIP is that it uses per-packet keying. In this protocol keys are dynamically generated and distributed by the authentication server. A client’s MAC address, a shared base key, and a packet’s sequence number create a unique key for each packet [15]. Also, TKIP rotates the broadcast key periodically. This characteristic, combined with the per-packet keying, eliminates the predictability which the attackers depend upon to crack the WEP key. As a final point, TKIP uses a Message Integrity Check (MIC) to prevent problems with undetected WEP modification attacks allowed by the CRC-32 algorithm. The MIC algorithm is much stronger and more secure than the CRC-32 algorithm used by WEP. MIC ensures packet-tampering detection immediately upon encryption by using a cryptographically protected one way hash in the data. Thus, MIC prevents the attacker from capturing, changing, and resending the data packets.

IEEE 802.1X, which was originally designed for Ethernet networks, has useful applications in wireless networks. The important characteristic of 802.1X is that there is support for mutual authentication between the client and the network. In 802.1X, the client sends the user’s credentials to the authentication server via the AP when a user requests access to the network. If the server accepts these credentials, the master TKIP key is sent to both the client and to the AP. “After this four-way handshake, in which the client and AP acknowledge one another and install the keys, the process is completed” [16].

All of the authentication requests in 802.1X are handled by the Extensible Authentication Protocol (EAP). EAP provides a very flexible platform for vendors to implement their own authentication mechanisms. For example, EAP can handle the presentation of all the user credentials in the form of digital certificates, secure IDs, smart cards, and unique user names and passwords. Several common EAP methods in use today are: EAP-Transport Layer Security (EAP-TLS), EAP-MD5, EAP-Tunneled Transport Layer Security (EAP-TTLS), and Protected Extensible Authentication Protocol (PEAP).

Advanced Encryption Standard (AES) is one possible alternative to WEP encryption. This standard was adopted as an official government standard by the U.S. Department of Commerce [17]. Instead of the RC4 algorithm, it uses another mathematical algorithm called Rijndael. This is a symmetric encryption algorithm that has a variable key length and block length (in AES, the block length is restricted to 128 bits). The user can choose from various key sizes such as 128-, 192-, or 256-bits and this makes it much more difficult to decipher the key than WEP. However, there are two downsides for AES. The first is that AES takes a longer processing time than other standards. Also, it requires a new chipset, meaning that it is not downward compatible with today's WLAN devices using the IEEE 802.11a, b, and g standards.

There are two main protocols that can be permanent replacements for WEP as the IEEE 802.11 wireless standard: Wi-Fi Protected Access (WPA) and IEEE 802.11i (also known as WPA2). WPA, which was introduced in 2002 by the Wi-Fi Alliance, is designed to secure all versions of the IEEE 802.11 devices. It is a software-upgradeable security solution which means it is compatible with WEP enabled systems. WPA combines TKIP encryption scheme with 802.1x/EAP authentication to greatly enhance security. MIC is also added to protect against packet forgery. If WPA is enabled, enterprises can work securely over the wireless network without any add-ons to the network. The IEEE 802.11i standard, created by a committee known as Task Group i (TGi), provides even more enhanced authentication, authorization, and encryption capabilities. This latter protocol was ratified on June 24, 2004.

The newly ratified IEEE 802.11i provides much more security for wireless networks. This standard requires AES for encryption purposes. It does, however, support WPA and is backwards compatible with most legacy WEP equipment. WPA and IEEE 802.11i are compatible with each other assuming AES encryption is available within the network structure. IEEE 802.11i is very similar to WPA in that it uses 802.1X/EAP authentication to ensure mutual authentication and dynamic key management. Counter mode with CBC-MAC Protocol (CCMP) is a required component for protection in IEEE 802.11i standard. It is the equivalent of TKIP in WPA. "CCMP computes a Message Integrity Check (MIC) using the well known, and proven, Cipher Block Chaining Message Authentication Code (CBC-MAC) method." [18]. Messages are encrypted in 128-bit chunks using CBC mode. This is a much more complicated process and cracking CCMP encryption is significantly harder than cracking RC4 encryption.

It is not necessary for small businesses or home/office users to switch to IEEE 802.11i if they already have WPA technology installed since this switch means new investments in wireless devices. Also, WPA provides the necessary security requirements for these users and it is compatible with IEEE 802.11i. Many businesses looking for new WLANs will find IEEE 802.11i very attractive and they certainly should consider new investments in hardware to improve their security. Enterprises with WPA, however, have to justify whether AES security is worth the cost of replacing the equipment.

## **8. Conclusion**

In the past decade, wireless networks have exploded into both corporate and home settings, mainly because people enjoy the ease and convenience WLANs afford them. However, as with all new and unrefined technologies, unexpected problems have surfaced. The most outstanding problem with current WLANs is the security vulnerabilities that exist. The most popular standard in present day WLANs is the IEEE 802.11b standard, which implements the WEP algorithm in order to encrypt data. Unfortunately, because of the faulty way in which the RC4 algorithm is implemented within WEP, the IEEE 802.11b standard is compromised. It is unsafe to transmit confidential data via WLANs that use the IEEE 802.11b standard.

As a result of WEP's security being jeopardized, many new ideas and fixes, intended to make WEP either more secure or to replace it entirely, have been researched by different organizations. Several of the new algorithms have been tested and seem to correct – or at least mitigate – the typical problems associated with the current WEP implementations. As the wireless industry continues to grow, the hope is that wireless networks will eventually be as secure as their modern day wired counterparts.

## **9. Acknowledgements**

The authors of this paper would like to thank the following people for their help and support during the course of this summer research project: Dr. Sennur Ulukus, Shabnam Shafiee, and Onur Kaya. Our sincere thanks go to Nick Petroni, Jr. for his tireless and extremely helpful work.

## 10. Works Cited

- [1] W. A. Arbaugh, N. Shankar, and Y.C. Justin Wan., "Your 80211 wireless network has no clothes," *IEEE Wireless Communications*, vol. 9, Dec. 2002, pp 44 – 51.
- [2] "The Seven Layers of the OSI Model" n.d. [Online] *Webopedia Reference*, Available: [http://webopedia.internet.com/quick\\_ref/OSI\\_Layers.asp](http://webopedia.internet.com/quick_ref/OSI_Layers.asp).
- [3] J. LaRocca and R. LaRocca, *802.11: Demystified*. New York: McGraw-Hill, 2002.
- [4] P.C. Mehta, "Wired Equivalent Privacy Vulnerability", LevelOne Security Essentials Track, April 2001
- [5] S. Fluhrer, I. Mantin, and A. Shamir, "Attacks on RC4 and WEP," *RSA Laboratories, Cryptobytes*, vol. 5, no. 2, Summer/Fall 2002
- [6] S. Fluhrer, I. Mantin, and A. Shamir. Weaknesses in the key scheduling algorithm of RC4. In Proc. 8<sup>th</sup> Workshop on Selected Areas in Cryptography, LNCS 2259. Springer-Verlag, 2001.
- [7] R. Giller and A. Bulliard, "Security Protocols and Applications 2004: Wired Equivalent Privacy", Swiss Institute of Technology, Lausanne, Mar. 3, 2004
- [8] C. Peikari and S. Forgie, "Cracking WEP" (2003) [online], Available <http://www.airscanner.com/publications.html#articles>, June 20, 2004
- [9] "Overview of Wireless Security Threats and Risks," n.d. [Online] Available: [www.acns.fsu.edu/network/pdf/Overview%20of%20Wireless%20Security%20Threats%20and%20Risks.pdf](http://www.acns.fsu.edu/network/pdf/Overview%20of%20Wireless%20Security%20Threats%20and%20Risks.pdf).
- [10] C. Peikari and S. Forgie., *Wireless: Maximum Security*. Indiana: Sams, 2002.
- [11] D.L. Pepyne, "SPRiNG: Synchronized Random Number for Wireless Security," *Wireless Communications and Networking*, vol. 3, pp. 2027-2032, Mar 16-20, 2003.
- [12] N. Borisov, I. Goldberg, and D. Wagner, "Intercepting mobile communications: The insecurity of 802.11," In Proc. 7<sup>th</sup> ACM Conference on Mobile Computing and Networking (MOBICOM'01), Rome, Italy, 2001.
- [13] AirSnort Homepage. n.d. [Online] Available: <http://airsnort.shmoo.com>.
- [14] L. Wheeler, "X9F Taxonomy and Glossary" n.d. [Online] Available: <http://www.garlic.com/~lynn/x9fgloss.htm>
- [15] B. Potter, "Wireless Security's Future," *IEEE Security & Privacy*. pp. 68-72 July/Aug 2003.

- [16] “Wi-Fi Protected Access: Strong, standards-based, interoperable security for today’s Wi-Fi networks.” Wi-Fi Alliance. [Online] Available: [www.wi-fi.org/OpenSection/pdf/Whitepaper\\_Wi-Fi\\_Security4-29-03.pdf](http://www.wi-fi.org/OpenSection/pdf/Whitepaper_Wi-Fi_Security4-29-03.pdf). Apr. 29, 2003.
- [17] “Securing Wi-Fi Wireless Networks with Today’s Technologies.” Feb. 6, 2003. Wi-Fi Alliance. [Online] Available: [www.wi-fi.org/OpenSection/pdf/Whitepaper\\_Wi-Fi\\_Networks2-6-03.pdf](http://www.wi-fi.org/OpenSection/pdf/Whitepaper_Wi-Fi_Networks2-6-03.pdf)
- [18] D. Lavery, “WPA versus 802.11i (WPA2).” Openxtra Network Management. <http://www.openxtra.com/articles/wpa-vs-80211i.htm>.
- [19] E. Griffith, “802.11i Security Specification Finalized,” n.d. [Online] Available: <http://www.wi-fiplanet.com/news/article.php/3373441> Jun 25, 2004.