An Overview of Network Security in Wireless Local Area Networks

Jiejun Kong, Mario Gerla
Computer Science Department
University of California, Los Angeles
{jkong,gerla}@cs.ucla.edu

B. S. Prabhu, Rajit Gadh
WINMEC Consortium
University of California, Los Angeles
{bsp,gadh}@wireless.ucla.edu

1 Introduction

Among a variety of wireless data networks that co-exist today, 802.11 Wi-Fi (Wireless Fidelity) is a suite of specifications for wireless local area networks. 802.11 is also referred to as wireless Ethernet because at the link layer it uses frame definitions and random medium access control protocols (CSMA or CSMA/CA) similar to the ones used in 802.3 Ethernet (CSMA/CD). Wireless Ethernet can serve as a wireless extension of an internal Ethernet in an institution which may require strong user authentication to prevent unauthorized users, as well as strong privacy protection to keep information confidential. Unfortunately, the first set of 802.11’s security design is proven to be insecure. Several research groups have independently discovered effective attacks exploring security flaws in the approved 802.11 standards. Open source implementations of these attacks are now widely available [4][28]. This chapter serves as a survey of security problems in 802.11 wireless Ethernets. In this survey we study the state-of-the-art attacks against 802.11 security protocol suite. We present an overview of common attacks, proposed countermeasures, and some known vulnerabilities of these proposals. Compared to many other articles discussing 802.11 security, we will present more technical details to show how the attacks work as well as pros and cons of corresponding countermeasures.

The rest of the chapter is organized as follows. In Section 2, we first describe 802.11 WEP and
related cryptological concepts. We then present an overview of well-known attacks against WEP in Section 3. In Section 4, we further describe a series of proposals that may provide adequate protection to wireless LAN. Finally we summarize the overview in Section 5.

2 802.11 WEP and related cryptology

2.1 Basic cryptology

In a cryptosystem, the protected message is called a plaintext. The processed message that can be seen by all parties is called a ciphertext. A cryptosystem is usually a collection of algorithms that translate plaintext into ciphertext by using secret keys, and vice versa. In modern cryptosystem design, a principle known as “Kerckhoff Desiderata” states that security should not be achieved via obscurity. Instead, all algorithms and protocol design should be made public by default, or the system’s scalability and deployment could be impeded. To satisfy this requirement, the best option probably would be to protect the system using just the secrecy of cryptographic keys. As a result, in a cryptosystem based just on the secrecy of cryptographic keys, if the key gets compromised all services protected by the key will be compromised.

The term cryptology refers to the study of both cryptography and cryptanalysis. Cryptography is the study of creating and using cryptosystems to protect messages so that only authorized parties can use their cryptographic keys or authorization tokens to access critical information. Cryptanalysis is the study of breaking such cryptosystems so that unauthorized cryptanalysts can pay reasonable costs to gain illegal access of critical information.

Depending on the resources available to the potential attackers, the following classes of the cryptanalytic attacks are related to 802.11 security design discussed in this chapter and are ordered by their increased potential to compromise a cryptosystem:

Ciphertext-only attack: The attacker only has the encrypted ciphertext from which to determine the secret plaintext, with no knowledge whatsoever of the latter.
**Known-plaintext attack:** The attacker has certain amount of plaintext and corresponding ciphertext, but out of his choice. These data are said to be “compromised”.

**Chosen-plaintext attack:** The attacker can obtain the ciphertext corresponding to an arbitrary plaintext data of his choice. If the attacker can determine the ciphertext of chosen plaintexts in an interactive or iterative process based on previous results, the attack is called *adaptive chosen-plaintext attack*.

The feasibility of mounting these attacks in practice is very high in wireless networks. In wired networks, adversary has to pass several lines of defense, such as firewalls and security gateways, to gain access to the transmitted data. But in wireless networks there is no clear line of defense inside the wireless part of the network. Adversary can easily intercept needed ciphertexts. Besides, standard network protocols typically use fixed packet formats, hence a wireless adversary has lots of opportunities to know the original plaintext of a specific ciphertext, in particular those data fields in packet headers. Finally, chosen-plaintext attacks are feasible if adversary can trigger encryption upon plaintext it has chosen. In 802.11 environment, this can be achieved by two collaborating adversaries: one communicates with a victim station with chosen plaintexts, and the other intercepts corresponding ciphertexts.

### 2.2 802.11 WEP

In the 802.11 protocol suite, Wired Equivalent Privacy (WEP) defines the cryptographic algorithm that is designed to provide wireless LAN with a level of security and privacy that is equivalent to that of the wired LAN and prevent illegal access and protect authorized users from eavesdropping. WEP assumes that a secret key $k$ has already been securely delivered to all communicating parties. Each sender follows the following procedures to protect every payload message $M$: 
Figure 1: Encrypting and decrypting 802.11 packets with WEP

- **Adding Integrity Check Value (ICV):** The 802.11 unit computes a CRC-32 checksum $c(M)$ on the payload $M$. The encryption plaintext is the concatenation\(^1\) $P = (M \parallel c(M))$.

- **Selecting Initialization Vector (IV):** The 802.11 unit randomly selects an IV $v$. The encryption key is $K = (v \parallel k)$.

- **Encryption:** RC4 device accepts the encryption key $K$ and outputs a *keystream*—i.e., a long sequence of pseudorandom bits—denoted by $RC4(K)$. The ciphertext $E$ is obtained by Vernam cipher $E = P \oplus RC4(K)$.

- **Transmission:** The unit prepends original 802.11 packet header and IV $v$ to the ciphertext, then transmits the packet over the radio link.

After a receiver recovers plaintext $P$, it verifies whether $c(M)$ is the CRC-32 checksum of $M$, then forwards the packet to upper layers if the checksum is valid. Figure 1 shows how WEP encrypts and decrypts 802.11 packets.

### 3 A brief overview of common attacks

Since the beginning of the millennium, attacks on WEP have attracted critical attention in both academia and industry. Several network security research groups have published cryptanalysis against

\(^1\)In the chapter “$\parallel$” denotes concatenation, and “$\oplus$” denotes bitwise XOR.
WEP [9][24][25]. They have successfully explored the broadcasting nature of wireless channels to reveal protocol design flaws. Given the capability of overhearing large amount of ciphertexts transmitted in wireless channels, Borisov et al. [9] revealed several insecurities in WEP design including short IV and linear message authentication code. Adversaries can easily launch data confidentiality, message modification, and message injection attacks against the wireless access point and gain unauthorized access to the network. In another research group, the Fluhrer-Mantin-Shamir attack [13] realized by Stubblefield et al. [24][25] can successfully reveal a 128-bit WEP secret key by eavesdropping about 5,000,000 packets, or less than 2,000,000 packets after the attack is improved. These two sets of attacks represent the most significant vulnerabilities discovered so far. Related details are described below. Interested parties may refer to other references for discussions on other vulnerabilities [26, 5].

3.1 Basic insecurities and vulnerabilities of WEP

At the link layer, an adversary is capable of launching ciphertext-only, or known-plaintext, or even chosen-plaintext attacks against security protocols. The design of WEP does not address these threats properly [5][9].

- **IV reuse**: When two WEP packets reuse the same IV \( v \), adversaries can launch attacks against wireless application’s data privacy and data integrity. Suppose the plaintexts in the two WEP packets are \( P_X = X \| c(X) \) and \( P_Y = Y \| c(Y) \), and the corresponding ciphertexts are \( E_X \) and \( E_Y \) respectively, then

  - When IV \( v \) is reused, RC4 device will produce exactly the same pseudorandom keystream \( RC4(K) \) up to the bit length\(^2\) of \( X \| c(X) \) or \( Y \| c(Y) \). If an adversary is capable of launching known-plaintext attack and knows \( P_X \), the adversary can reveal the keystream

    \[
    RC4(K) = P_X \oplus E_X \quad \text{and immediately reveal} \quad P_Y = RC4(K) \oplus E_Y = P_X \oplus E_X \oplus E_Y.
    \]

\(^2\)In CRC-32 checksum system, the shorter one of \( X \) and \( Y \) is identical to an elongated version with zero-bit padded. Thus \( X \) and \( Y \) can be regarded as of same length.
Even when neither $P_X$ nor $P_Y$ is known, the adversary can do further analysis by knowing the bitwise XOR of the plaintexts from the ciphertexts $P_X \oplus P_Y = E_X \oplus E_Y$.

- WEP does not use keyed message authentication code (MAC) [16] to generate cryptographic checksums. In contrast, its integrity checksums are produced by linear CRC-32 function $c$. For two messages $X$ and $Y$,

$$c(X) \oplus c(Y) = c(X \oplus Y).$$

Thus an adversary can generate a new “valid” packet from two existing packets. By WEP’s design the receiver cannot detect the compromise of data integrity.

$$(X \| c(X)) \oplus (Y \| c(Y)) = (X \oplus Y) \| (c(X) \oplus c(Y)) = (X \oplus Y) \| (c(X \oplus Y)).$$

- **Short IV field**: The chance of IV reuse is very large because the IV field used by WEP is only 24 bits wide. In addition, since the size of IV space is only $2^{24}$, an adversary can build a decryption dictionary for all keystreams. As the upper bound of 802.11 packet size is about 2000 bytes, the dictionary size is less than $2^{24} \times 2000$ bytes $= 32$ gigabytes, a storage space available to most modern PCs.

- **Revealed keystream**: An adversary can use a revealed keystream $RC4(K)$ to inject random messages and launch other attacks. For example, since WEP’s access control is based on a symmetric key challenge-response scheme, an adversary can easily use the revealed keystream to produce a valid response, then gain access to the network.

### 3.2 Fluhrer-Mantin-Shamir attack

RC4 was arguably the most widely used stream cipher when WEP was designed. Unfortunately, Fluhrer et al. [13] discovered that the initial output in the RC4 keystream is disproportionately
affected by a small number of key bits\(^3\). Moreover, the secrecy of RC4 key is vulnerable to related key cryptanalysis [7]. It is based on the observation that adversaries may know the difference between two cipher keys though both keys are not completely revealed yet. In WEP, the cipher key used by RC4 device is the concatenation \(v\|k\) where \(v\) is the exposed IV and \(k\) is the shared WEP secret key. As the result, related key cryptanalysis against RC4 is enabled by the design. Fluhrer-Mantin-Shamir attack is practical for any key size and for any modifier size, including the 24 bit recommended in the original WEP and the 128 bit recommended in the revised version WEP2. After numerous different exposed IVs are used, an attacker can derive the WEP secret key by analyzing the initial word of the keystreams.

As depicted below, RC4 algorithm is comprised of two modules: Key Scheduling Algorithm (KSA) and Pseudorandom Generation Algorithm (PRGA).

<table>
<thead>
<tr>
<th><strong>KSA(K):</strong></th>
<th><strong>PRGA(S):</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initialization:</strong></td>
<td><strong>Initialization:</strong></td>
</tr>
<tr>
<td>For (i = 0..255)</td>
<td>(i = 0)</td>
</tr>
<tr>
<td>(S[i] = i)</td>
<td>(j = 0)</td>
</tr>
<tr>
<td>(j = 0)</td>
<td></td>
</tr>
<tr>
<td><strong>Scrambling:</strong></td>
<td><strong>Generation Loop:</strong></td>
</tr>
<tr>
<td>For (i = 0..255)</td>
<td>(i = (i + 1) \mod 256)</td>
</tr>
<tr>
<td>(j = (j + S[i] + K[i \mod l]) \mod 256)</td>
<td>(j = (j + S[i]) \mod 256)</td>
</tr>
<tr>
<td>Swap((S[i], S[j]))</td>
<td>Swap((S[i], S[j]))</td>
</tr>
<tr>
<td>Output (z = S[(S[i] + S[j]) \mod 256])</td>
<td>Output (z = S[(S[i] + S[j]) \mod 256])</td>
</tr>
</tbody>
</table>

Figure 2: RC4’s Key Scheduling Algorithm and Pseudorandom Generation Algorithm

KSA turns an \(l\)-byte random key (whose typical size is 40-256 bits) into a scrambled initial

\(^3\)Earlier credits go to Ian Farquhar who posted a warning to sci.crypt in 1996 that there was an approximately 35% probability that the first byte of the keystream generated by RC4 would be the same as the first byte of the key.
permutation $S$ of $\{0, \ldots, N-1\}$, then PRGA uses this permutation to generate a pseudo-random output sequence that can be used in stream cipher.

In this section we briefly describe the known IV attack (as named in [13]), that is, how related key cryptanalysis can reveal the secret part (i.e., the WEP secret key) upon knowing exposed IV prefix. Once the WEP secret key is revealed, cryptographic protection provided by the key is no longer effective.

\[
\begin{array}{cccc}
S & 1 & X & X+Y \\
& X & Y & Z \\
& & & \ldots.
\end{array}
\]

**Figure 3: RC4's Resolved Condition (after KSA)**

The first PRGA keystream output word depends only on three specific permutation elements. If the state of the permutation immediately after KSA is as shown in Figure 3, then the value labeled $Z$ will be output as the first word. In addition, let $S_i$ denote the snapshot permutation of $S$ at round $i$. If the key setup reaches a stage where $i$ is greater than or equal to 1, $X = S_i[1]$ and $Y = S_i[X]$, then (if we model the remaining swaps in the key setup as in a truly random process) with probability greater than $e^{-3} \approx 5\%$, none of the elements referenced by these three values will participate in any further random swaps. This is because

\[
\left(1 - \frac{3}{N}\right)^N \geq e^{-3}
\]

for $N$ rounds. The probability that none of the three positions is chosen for swapping at each round is $(1 - \frac{3}{N})$.

If after KSA $S$ is in a resolved condition, then the value $S[S[1] + S[S[1]]]$ will be output as the first word by PRGA. With probability less than $1 - e^{-3} \approx 95\%$, at least one of the three values will participate in a swap, and be set to an effectively random value, which will make the output value effectively random. However, the chance of 5% is non-negligible.

Now we analyze the weakness of KSA and see how a resolved condition can be set up. Consider
the WEP scenario where the IV is prepended to the secret key. In this circumstance, assuming we have an $I$-byte exposed IV and an $(l - I)$-byte secret key ($K[I + 0], K[I + 1], \ldots, K[l - 1]$). These values can be mapped into real values used in 802.11. In WEP, $I = 3$ and $l = 8$. In WEP2, $I = 16$ and $l = 32$. Our goal is to derive information on a particular byte $B$ of the secret key (i.e., to find out $K[I + B], 0 \leq B \leq l - I - 1$).

Let’s see a simple example. In WEP, $I = 3$ and $l = 8$. Suppose the cryptanalyst has inductively revealed the sequence of secret key bytes ($K[3], \ldots, K[A + 2]$), now he wants to reveal $K[A + 3]$. The entire $(l - I)$-byte secret key ($K[I + 0], K[I + 1], \ldots, K[l - 1]$) can be inductively revealed in this way as the process starts from the scratch line where $A = 0$ and $K[3]$ is the target. The cryptanalyst examines a series of IVs of the form $(A + 3, N - 1, X)$. At the first KSA round, $j$ is advanced by $A + 3$, and the $S[i]$ and $S[j]$ are swapped, resulting in the key setup state shown in Figure 4.

$$
\begin{array}{c|c|c|c|c|c|c}
\text{exposed IV} & 0 & 1 & 2 & 3 & \text{exposed secret so far} & \text{secret} \\
S \text{ (swapped)} & i_0 & 1 & 2 & & & 0 \\
\end{array}
$$

Figure 4: KSA round 1

Then, on the next round, $i$ is advanced, and then the advance on $j$ is computed, which happens to be $(N - 1) + 1 \mod N = 0$. Then $S[i]$ and $S[j]$ are swapped, resulting in the state shown in Figure 5.

Then, on the next round, $j$ is advanced by $X + 2$, which implies that each distinct IV assigns a different value to $j$, and thus beyond this point, each IV acts differently. Unfortunately, since the cryptanalyst knows the value of $X$ and ($K[3], \ldots, K[A + 2]$), he can compute the exact behavior of the key setup until he reaches round $A + 3$. At this point, he knows the value of $j_{A+2}$ and the exact values of the permutation $S_{A+2}$. If $S_{A+2}[0]$ or $S_{A+2}[1]$ has been disturbed (i.e., $S_{A+2}[0] \neq A + 3$
Figure 5: **KSA round 2**

or $S_{A+2}[1] \neq 0$, the cryptanalyst rests the case and the $K[A + 3]$ is not revealed. Otherwise, $j$ is advanced by $S_{A+2}[i = A + 3] + K[A + 3]$, and then the swap is done, resulting in the state shown in Figure 6.

Figure 6: **KSA round (A + 3)**

We are looking for a resolved condition where $(S_{A+3}[1] + S_{A+3}[S_{A+3}[1]]) = 0 + A + 3 = A + 3$.

Then with more than 5% probability $S_{A+3}[A + 3]$ will be output as the first byte in RC4’s keystream. As described previously, in particular in [9], there are many ways to recover this key stream and to know $S_{A+3}[A + 3]$.

At KSA round $(A + 3)$, the cryptanalyst knows the permutation $S_{A+2}$ and the value of $j_{A+2}$. Now he also knows the value of $S_{A+3}[A + 3]$, then he knows its location in $S_{A+2}$, which is the value of $j_{A+3}$. He can use the known information to compute $K[A + 3]$:

$$K[A + 3] = j_{A+3} - j_{A+2} - S_{A+2}[A + 3].$$
The case described above is a special case. In general, by searching for IV values such that, after the first $I$ KSA rounds, $S_I[1] < I$ and $S_I[1] + S_I[S_I[1]] = I + B$. Then, with high likelihood (probability $\approx e^{-\frac{2B}{M}}$ if we model the intermediate swaps as in a truly random process), we will be in a resolved condition after KSA round $(I + B)$, and then the most probable first byte of keystream output will be

$$keystream = S_{I+B-1}[j_{I+B}] = S_{I+B-1}[j_{I+B-1} + K[B] + S_{I+B-1}[I + B]].$$

Or, in other words, if we know the value of $j_{I+B-1}$ and $S_{I+B-1}$, then given the first byte of keystream output, we can predict the value

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

where $S^{-1}[X]$ denotes the location within the permutation $S$ where the value $X$ appears. This prediction is accurate more than 5% of the time, and effectively random less than 95% of the time. By collecting sufficiently many values from different IVs, we can reconstruct $K[I + B]$.

Because the known IV attack only uses the first byte of keystream output from any given secret key and IV, the attack is applicable to secret key and IV of any size. This means the elongated key design adopted by WEP2 is not an effective countermeasure. The substitutes of WEP and WEP2 are being developed to address the new challenges. They will be reviewed in later sections of this chapter.

### 3.3 Other simple attacks

In addition to these serious cryptanalytical attacks, there are other simple attacks utilizing network and system loopholes. For example, most 802.11 (Wi-Fi) access points have a built-in feature called MAC Address Filtering, which is widely used and allows the network administrator to enter a list of MAC (Media Access Control) addresses that are allowed to communicate on the network.
On the other hand, most 802.11 (Wi-Fi) NICs allow you to configure the MAC address of the NIC in software. Therefore, if you can sniff the MAC address of an existing network node, it is possible to join the network using the MAC address of the victim node. This shows anonymity protection is needed in wireless LAN as well.

4 New wireless LAN security protocols

4.1 WPA

Knowing the vulnerabilities of Wired Equivalent Privacy (WEP), the Wi-Fi Alliance, in conjunction with the IEEE, has begun an effort to bring strongly enhanced, interoperable Wi-Fi security to replace WEP. The result of this effort is Wi-Fi Protected Access (WPA), which is a specification of standards-based, interoperable security enhancements that strongly increase the level of data protection and access control for existing and future wireless LAN systems. Designed to run on existing hardware as a software upgrade, Wi-Fi Protected Access is derived from and will be forward-compatible with the upcoming IEEE 802.11i standard.

Wi-Fi Protected Access (WPA) protects wireless LANs using technology like IEEE 802.1x, RADIUS, and TKIP. 802.1x provides authentication services, likely from a RADIUS server. EAP provides a variety of algorithms for authenticating the client with the RADIUS server. TKIP ensures confidentiality and integrity for wireless communications. WPA operates in two modes: PSK (Pre-shared Key) and Enterprise. In the PSK mode, a shared password is used in authentication. It is essentially the same key management scheme assumed in 802.11 WEP, so that authentication server (e.g., RADIUS) is not required. In the Enterprise mode, an authentication server is needed in service provisioning.

802.1x, RADIUS, and TKIP are described in details below.
4.1.1 802.1x

802.1x leverages existing standards such as EAP [8], which stands for “Extensible Authentication Protocol”. EAP lets the network use a variety of algorithms for authenticating the client with an authentication server.

![Diagram of 802.1x network](image)

**Figure 7: Roles and control flows in 802.1x**

802.1x defines three roles: (1) *Supplicant* is a user or client requesting authentication; (2) *Authentication Server* is the server providing authentication service; (3) *Authenticator* is the being-requested device to be accessed by the supplicant. The conversation among the three roles is depicted in Figure 7. In a typical scenario, an authenticator (i.e., the wireless access point) is configured to allow open and shared authentication. The initial client authentication is open because dynamically derived encryption keys are not yet available at this stage. Once the client associates with the authentica-
tor, the depicted 802.1x authentication conversation (between Access blocked and Access allowed) grants access to supplicants with valid credentials. Prior to EAP-Success, the authenticator filters all non-EAPoL traffic from the supplicants. Upon EAP-Success, the authenticator removes the filter.

802.1x can be run on top of both wired LAN (802.3) and wireless LAN 802.(802.11) with minimal changes. It is particularly well suited for wireless LAN applications because it requires very little processing power on the part of the authenticator (access point). In addition, 802.1x also support user-based identification based on Network Access Identifier [1], which enables support for roaming access in public domains [3].

4.1.2 RADIUS authentication

RADIUS [21][20][31] is not part of WPA or 802.11i standard, but it is the de facto back-end authentication protocol. A RADIUS server customized for 802.11 wireless LAN is designed on the same core architecture as a wired RADIUS server. The related service includes user-specific check, session time-out, idle time-out, and framed IP address, etc. However, protocols like SLIP, PPTP and L2TP are particularly designed for wired dial-ups, they are not suitable in a wireless environment. Therefore, RADIUS servers customized for wireless LAN need not implement these features.

For backward compatibility, RADIUS servers customized for wireless LAN may implement proxy supports, which let the network maintain accounting information and user data in an existing database that holds information pertaining to conventional RADIUS usage, such as dial-up, VPN and firewall-access accounts.

A RADIUS server is essentially a centralized database storing authentic IDs and associated attributes. It can perform a look-up authentication or a log-in authentication. In the look-up mode, the server authenticates the user against the ID store and returns user-specific attributes from the ID store with the access-accept packet. In the log-in mode, the user's credentials are used to log in to the authentication system. If the credentials are validated, the user receives an access-accept packet.
4.1.3 TKIP

TKIP (Temporal Key Integrity Protocol) is an interim standard allowing for backward compatibility. It will be replaced by standards with better security supports, such as CCMP. However, the backward compatibility with TKIP is necessary because most legacy Wi-Fi (802.11) hardware does not support the AES (Advanced Encryption Standard [19]) algorithm used by CCMP.

![TKIP's key mixing](image)

Figure 8: TKIP's key mixing

TKIP enforces a policy that the RC4 cipher key must be refreshed every 10,000 packets. This prevents a Fluhrer-Mantin-Shamir attacker to accumulate enough information to reveal a constant cipher key. In addition, the RC4 cipher key is not a constant. As depicted in Figure 8, TKIP has a key-mixing function to generate the RC4 cipher key per packet. Finally, TKIP uses a 48-bit IV (versus the 24-bit IV in WEP). At least 44-bit of the IV field is used as a counter to avoid IV reuse and replay attacks. Any packet with out-of-sequence IV counter will be discarded. Now two IVs collide after about $2^{44}$ frame transmissions, which would typically take thousands of years.

To ensure message integrity, a keyed hashing for message authentication (HMAC) algorithm named Michael [11] is used in each direction of wireless transmission, where a 64-bit key is used in Michael to generate a 64-bit cryptographic checksum (also known as message integrity code-MIC). In WEP, CRC-32 is linear and is only applied on payload. In TKIP, Michael is not only cryptographically strong, but also applied on both payload and header. This further decreases the chance of checksum collision and enlarges the covered range of integrity protection.
4.2 802.11i

WPA is viewed by many merely as an interim measure. IEEE 802.11i is the draft security standard for 802.11 wireless network protection. Nevertheless, 802.11i relies heavily on many existing WPA components.

802.11i introduced the RSN (Robust Secure Network) protocol for establishing secure communications. Like WPA, RSN uses 802.1x to authenticate wireless devices to the network. Two new protocols based on Advanced Encryption Standard (AES), namely CCMP and WRAP, are introduced to protect data privacy and integrity.

4.2.1 802.11i key management

Key management is a critical service in any security protocol suite using cryptography to protect the system. The key management module of 802.11i is far more sophisticated than the flawed 802.11 WEP and 802.1x. There are two classes of keys: pairwise keys and group keys. In pairwise key management, a hierarchy of three types of keys protects the network:

1. Master Key (MK) which is established upon appropriate authentication;

2. Pairwise Master Key (PMK) which is derived from MK per connection/session; The difference between MK and PMK is due to security policy. Since authentication server is the appropriate place to verify a wireless station’s identity and grant associated privileges, MK is the fresh token representing the relation between the authentication server and an authenticated wireless station. On the other hand, PMK is the fresh token representing the relation between an access point and the authenticated station. In terms of PMK, the authentication server can be regarded as a trusted third party.

3. Pairwise Transient Key (PTK) is the collection of operational keys including Key Confirmation Key (KCK), Key Encryption Key (KEK), and Temporal Key (TK). They are derived from PMK. The entire derivation process uses random nonces selected by both the access point and
the wireless station, thus man-in-the-middle attack is not feasible. KCK is used to prove the possession of PMK and to bind the wireless station to the access point via the PMK. KEK is used to distribute Group Transient Key (GTK) which belongs to group key management described below. TK is used as cipher key to secure data traffic.

In group key management, the access point selects a random Group Transient Key (GTK) and encrypts it with KEK. The wireless station uses KEK, which is derived from the shared PMK, to decrypt and know the GTK. Messages of this GTK key exchange handshake are authenticated by KCK, which is also derived from the shared PMK. GTK can be shared among a number of wireless stations and the access point, thus can be used to protect broadcast and multicast traffic.

4.2.2 CCMP

CCMP (Counter Mode with Cipher Block Chaining Message Authentication Code Protocol) is the mandatory encryption protocol in the 802.11i standard. It uses CCM mode [29] and 128-bit keys, with a 48-bit initialization vector (IV) for replay detection.

In CCMP, each sender follows the following procedure to protect every payload message $M$:

Unprotected:

```
+-----------------+---+---+
|     Header     |  H |  M |
+-----------------+---+---+
```

Encapsulate $\rightarrow$ Decapsulate

Cryptographic processing:

```
+-----------------+---+---+---+
|     Header     |  H |  P |  M |
+-----------------+---+---+---+
```

Encrypted with AES in CTR mode

```
+-----------------+---+---+---+
|     Header     |  H |  P |  M |
+-----------------+---+---+---+
```

Authenticated with CBC–MAC mode

```
+-----------------+---+---+---+---+
|     Header     |  H |  V |  I |  P |  M |
+-----------------+---+---+---+---+
```

Transmission:

Figure 9: Encrypting and decrypting 802.11 packets with CCMP
• **Adding Message Integrity Code (MIC):** The 802.11i unit computes a cryptographic checksum $MIC(H||M)$ on the header $H$ and payload $M$. In CCMP, the Cipher Block Chaining Message Authentication Code (CBC-MAC) mode is used to produce the $MIC$.

• **Encryption:** The unit then encrypts payload $M$ and $MIC$ separately. Payload $M$ is encrypted in CTR mode with counter values counting from 1. $MIC$ is encrypted in CTR mode with counter value 0.

• **Transmission:** The unit prepends IV before the payload $M$ and transmits the packet over the radio link.

After a receiver recovers $M$ and $MIC$, it verifies whether $MIC$ is the CBC-MAC checksum of $M$, then forwards the packet to upper layers if the checksum is valid. Figure 1 shows how CCMP encrypts and decrypts 802.11 packets.

### 4.2.3 WRAP

WRAP (Wireless Robust Authenticated Protocol) is an encryption protocol in the 802.11i standard. WRAP is based upon the Offset Codebook (OCB) mode proposed by Rogaway et.al [22].

OCB is a composite mode ensuring data privacy and integrity at the same time. OCB is efficient. The overhead of OCB’s composite operations is about the same workload of encryption alone—given a message of $n$ 128-bit blocks, encryption alone needs $n$ AES cipher calls, and the OCB mode only needs $n + 2$ AES cipher calls (versus $2n + 2$ calls in CCM mode).

The major drawback of WRAP is not a technical one. OCB mode is proven to be provably secure by its designers, and has so far survived cryptanalysis. Unfortunately, three different parties have filed for patents on WRAP. These intellectual property issues caused the IEEE to adopt CCMP as the standard. The WRAP protocol becomes an optional component as some vendors have already implemented WRAP-conforming hardware.
4.3 Known vulnerabilities

4.3.1 802.1x design trade-offs and vulnerabilities

802.1x recommends an implementation to use RADIUS [21][20]. But this is not an imperative requirement. An implementation can employ a variety of authentication methods, such as PKI certification based on X.509 [15], Kerberos [23], etc. This design choice gives wireless LAN administrators certain flexibility in their management. However, there are trade-offs. This design is suitable for a closed wireless network, but not open wireless networks for mobile users. For example, even if a mobile supplicant has installed and enabled its 802.1x support in its home wireless LAN, it cannot connect to a wireless LAN with another type of 802.1x support.

Mishra and Arbaugh [18] reported two potential vulnerabilities of 802.1x: (1) 802.1x should mandate mutual identity authentication (or class 2 authentication named in WTLS [27]) in all scenarios, so that an adversary cannot launch man-in-the-middle attack and insert itself between a supplicant and an authenticator. Fortunately, EAP-TLS [2] does provide mutual identity authentication. Also this man-in-the-middle attack is detectable. In other words, the supplicant and the authenticator can detect anomalous communications, such as duplicated identities, if they are within one-hop radio transmission range. (2) Not all 802.11 frames and EAP frames are appropriately authenticated. This contributes not only to the feasibility of man-in-the-middle attack, but also to session hijacking, where an adversary—the man in the middle—de-associates the supplicant and acts as the supplicant itself. The key to address these vulnerabilities is to implement appropriate per-connection identity authentication and per-frame message authentication. As described previously, the design of 802.11i key management scheme can be considered as a related effort along this track.

4.3.2 Michael vulnerabilities

Michael is an efficient message authentication algorithm, but it is much weaker than other counterparts like MD5 and SHA1. Due to “birthday paradox”, it is expected that one MIC collision
will happen for every $2^{32}$ packets (versus $2^{64}$ in MD5 and $2^{80}$ in SHA1). It is shown in the original document [11] that this number can be decreased to $2^{29}$. Nevertheless, this value is considered to be big enough and will unlikely affect a wireless session.

Recently there is an unpublished technical report [30] showing that Michael can be easily inverted by a known-plaintext attacker given a single known plaintext message and its known MIC. The designer of Michael seemed to be aware of this vulnerability since the need of keeping the MIC values secret is stated several times in the original document [11]. The technical report further shows a related message attack that utilizes the invert function as a subroutine to reveal Michael’s secret key with relaxed conditions.

### 4.3.3 AES vulnerabilities

No successful cryptanalysis against AES/Rijndael is publicly known at this time. But there are some interesting results unveiled. Ferguson et al. [12] pointed out certain interesting algebraic properties of AES/Rijndael which implies that the security of AES/Rijndael relies on a new computational hardness assumption. But it is unknown whether effective attacks against the hardness assumption are feasible in the future. Fuller and Millan [14] also discovered an interesting mathematical property of AES/Rijndael. Again it is not even an attack.

A more recent result published by Courtois and Pieprzyk [10] illustrated an attack against AES/Rijndael and Serpent. The authors described XSL attack which is a better-than-brute-force attack against Serpent, and possibly one against Rijndael as well⁴. AES is a newly proposed standard. Studies on related key cryptanalysis and other cryptanalytic attacks against AES are inconclusive and may potentially reveal more vulnerability and even insecurity.

⁴This is a controversial issue. Interested parties may refer to [http://www.cryptosystem.net/aes/](http://www.cryptosystem.net/aes/) for more details.
4.3.4 Composition of authentication and encryption

Security protocols/applications, such as SSL/TLS/WTLS, IPsec, and SSH, apply both symmetric authentication (HMAC) and encryption to the transmitted data. However, these popular protocols have chosen a different method to combine authentication and encryption. The three methods used by SSL, IPsec and SSH are referred to as authenticate-then-encrypt (abbreviated AtE), encrypt-then-authenticate (EtA), and encrypt-and-authenticate (E&AE), respectively:

- **SSL:** \( a = Auth(x), C = Enc(x||a) \), transmit \( C \)
- **IPsec:** \( C = Enc(x), a = Auth(C) \), transmit \( (C||a) \)
- **SSH:** \( C = Enc(x), a = Auth(x) \), transmit \( (C||a) \)

where \( x \) is a message; \( Enc(\cdot) \) is a symmetric encryption function; \( Auth(\cdot) \) is a message authentication code; and ‘||’ denotes concatenation—in this notation the secret keys to the algorithms are implicit.

Under the generic assumptions that HMAC functions are secure against chosen-message attacks and symmetric encryption functions are secure against chosen-plaintext attacks. Bellare & Namprempre [6], Krawczyk [17] proved that IPsec’s EtA method is a generically secure method for implementing secure channels.

In contrast, the E&A method used in SSH is vulnerable to privacy attacks even if we use a secure HMAC function. The AtE method used in SSL has the same problem unless specific encryption modes are used, for example, two very common forms of encryption: CBC mode and stream ciphers.

802.11i CCMP protocol adopts the AtE approach. It is considered secure if CTR mode effectively realizes a strong pseudorandom generator as CBC mode does. 802.11i WRAP protocol utilizes OCB mode, which achieves message privacy and message authentication simultaneously. Thus composition vulnerability is not applicable to WRAP.
5 Summary

Compared to deploying an 802.11 wireless LAN, protecting the deployed wireless LAN is a relatively more difficult job. In this survey we limit our scope to 802.11 security attacks and proposed countermeasures. We have illustrated design flaws of original 802.11 WEP standard in details. We also show that countermeasures are feasible. Currently WPA/TKIP is considered as an inexpensive and effective means to address wireless cryptographic attacks. In the near future proposed solutions like CCMP and WRAP will provide better protection to wireless users. We will continuously watch the ongoing game of cryptanalysts versus cryptographers (or attackers versus counter-attackers). Though we believe that a secure wireless LAN environment will be realized in the foreseeable future, the final conclusion should not be made today because more investment both in terms of resource and time are needed, and there will be more security surprises waiting for us ahead.

References


