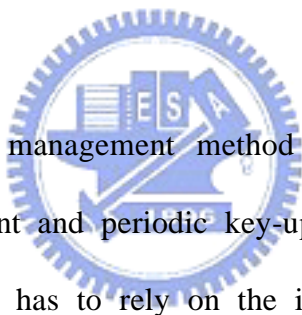


Chapter 3

Proposed Key Management Method



The proposed key management method can be divided into two parts, authentication management and periodic key-updated management. The periodic key-updated management has to rely on the information that exchanges in the authentication management. This dependent design is different from the original IEEE 802.11 authentication, which is only designed for station's access permission from joining the wireless network. In this chapter, we present several designed issues, which is involved in our method. Then, we present the proposed of authentication and periodic key-updated management method. Finally, a real case is given to show how the periodic key-updated management proceeds.

3.1 Designed Issues

The objective of our proposed key management is to strengthen the security of wireless LANs for IEEE 802.11 standard. First, we state the design goals. These goals can be considered as the contributions of the method. Next, we describe the features of the main components for the method. These components include key-mapping table, clock synchronization, hash function, and the challenge text of authentication.

3.1.1 Design Goals



The design goals are given as follows.

1. IEEE 802.11 compliance: The IEEE 802.11 is successful in commercial applications. There have been millions of devices used around the world. Therefore, the proposed method can be compliant with vast existing IEEE 802.11 wireless equipments.
2. Stations revocation: In the IEEE 802.11 standard, all stations communicate with the AP using the same share WEP key. But it will be difficult to revoke a

station from access, because you have to change the shared key manually and announce the changed to those legitimate stations. In our proposed method, we can revoke any station in any time automatically and smoothly.

3. Short periodically and seamlessly key updated: This is one of the most important issues in key management. The WEP key has to be updated before an enough broken amount of packets. In theoretically, it only requires few million packets to recover a 128 bits WEP key. Suppose we are using IEEE 802.11g, 54MHz, and each packet size is 500 bytes, the key is possible to be broken around 1 minute. In our proposed method, the key can be updated in a short period, and the updated should be seamless to avoid the throughput slowed down.

4. Minimize the long-term key used: Any key could be broken, if an attacker collects enough samples and patterns. A very simple way to avoid the risk is to reduce the secret key used times. In our proposed method, whenever we finish the authentication, we use temporary group keys in following communication.

3.1.2 Key-Mapping Table

The proposed method uses the individual-key for authentication. Thus, we need to maintain a table that binds the individual-key for each station, in which we use the MAC address as the station's identity. Fortunately, the IEEE 802.11 defines an optional key-mapping table already. We can simply use the existing key-mapping table in AP to establish our database. Whenever a station issues the authentication, the AP can pick up a responsible individual-key to authorize it and distribute group-key by station's MAC address.

3.1.3 Clock Synchronization



The IEEE 802.11 relies on the synchronized clock to coordinate the actions between access point and mobile stations. Especially, in the Frequency-Hopping networks, it changes the radio channel in different time slot frequently.

It is like the Power Management, the access point centrally maintains the networks timer, and each station has to listen and copy the timing information from the clock distribution frame. Both the Beacon and Probe-Response frame contains the timing information, called Timestamp. The Timestamp field is long to 64-bit, microsecond precision counter. When the station receives the timing information, it

saves it and adds a small offset, propagation and processing time, to the giving time value. Although the Beacon frame is a most frequent frame in the wireless networks, the radio could have unexpected interference in the air. Therefore, the mobile station needs to maintain its local timer to keep the synchronized clock roughly, even it misses timing information frame.

It is important that the timer synchronization has to finish before any association and authentication. In our periodic key updated method, we still greatly depend on this clock synchronization issue to synchronize the re-key timing precisely.

3.1.4 Hash Function



Hash function is a one-way secure function. This function produces a fix-length output by a variable-length input in the forward direction, but it is computationally infeasible to find the input data with the fix-length output. These properties make it becomes a great deal on the message signature and authentication protocol. There are some hash functions to be used commonly, such as MD4 [21], MD5, and SHA-1 [22].

In the IEEE 802.11, the WEP protocol does not use any hash function to

enhance the message validation and integrity. In our proposed method, the hash function is required to generate a forward direction random nonce. Each access point and mobile station use the same hash function and generate the same nonce.


3.1.5 The Challenge Text Of Authentication

The IEEE 802.11 authentication is a four-way handshake protocol. The AP uses the challenge text, which is contained in the authentication frames, to authenticate stations. The station responds the encryption of challenge text to prove it shares the same secret key with the AP. The details of the authentication protocol will be described in section 3.2.

The challenge text is a 128-byte of nonce. In the IEEE 802.11, it recommends that the challenge text should be a random nonce and must not be a constant byte repeated 128 times. Besides the restriction, the challenge text is essentially very flexible. Therefore, we can put more semantics on the challenge text. This ideal, semantic challenge text, was proposed in [8]. For example, the AP can distribute the current group-key to the station. Note that most semantics are confidential, and they must be encrypted with the individual secret key. In our proposed method, we will use the technique to transport useful information.

3.2 Authentication Management

In the wireless LANs, the authentication is a major component of maintaining network security. IEEE 802.11 specifies two approaches: open-system authentication and shared-key authentication. In open-system authentication, the AP actually accepts any station without validation. That is why it is named “open”. Shared-key authentication requires WEP in both AP and station sides, thus a shared key must be distributed to stations before authentication started. Share-key authentication consists of four steps, as shown in figure 3-1. Each step has a sequence number to identify the message. The details of shared-key authentication are described as follows.

- 
- Step 1: The station issues an authentication request that contains the authentication algorithm identification, which is set to 1 for shared-key authentication.
 - Step 2: If the AP accepts the authentication request, it responses a frame that is composed of status code, challenge text, etc. The status code is set to 0 for success, and the challenge text is 128 bytes of random nonce.
 - Step 3: The station returns the encryption of full frame, which includes the challenge text. The encryption uses WEP with shared-key. This step attempts to prove the station shares the same shared-key with the AP.
 - Step 4: If the encryption frame is verified by the AP, it responses the status code of successful. Then, the authentication is finished.

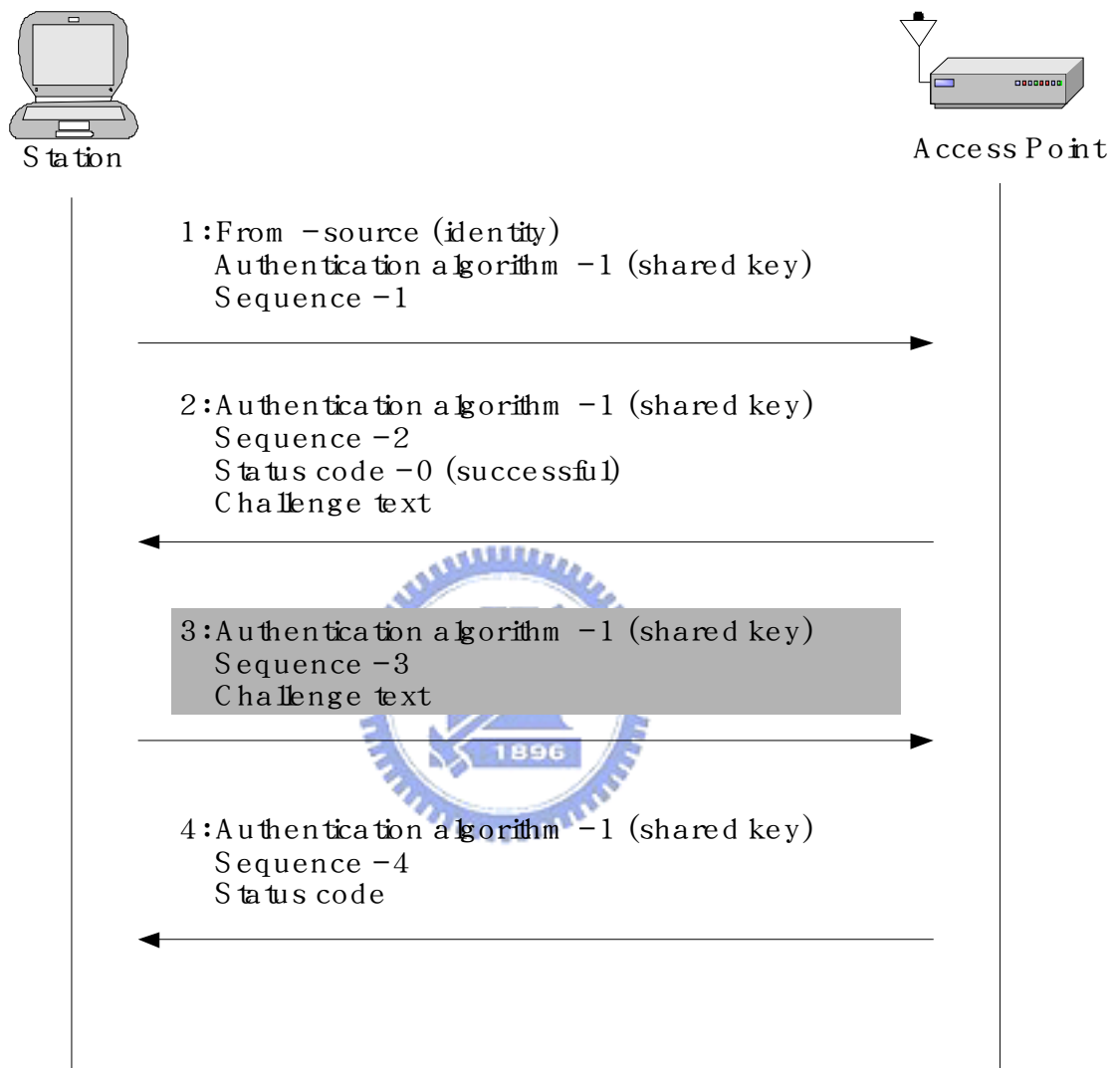


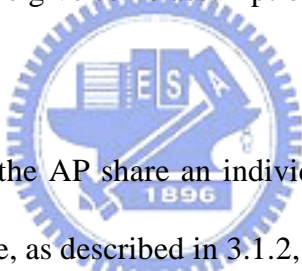
Figure 3-1. The shared-key authentication

Note that the shared-key authentication and data encryption use the same WEP key. In IEEE 802.11, the WEP key is a long-term key because IEEE 802.11 lacks of

key distribution mechanism. It makes the standard suffering from WEP weakness. In our proposed method, we use a long-term individual key and a session (temporary) key for authentication and data encryption, respectively. In addition, we propose a re-key mechanism for the session key. The re-key mechanism requires some synchronized semantics between AP and station. The content of challenge text, as described in section 3.1.5, gives a good way to carry these semantics.

The proposed authentication is modified from the shared-key authentication. We make some changes on the Steps 2 and 3. Before stating the operation of proposed authentication, we give some assumptions and notations.

Assumptions:

- 
1. Each station and the AP share an individual secret key K_{div} . The AP has a key-mapping table, as described in 3.1.2, for key loop-up.
 2. There is already a session WEP key $K_{sess:p}$ in the AP side, p is the sequence number of the session key currently in use.

Notations:

- MAC^{AP} and MAC_{sta} : The MAC address of the AP and station respectively.
- TS_{AP} : The timestamp of the AP.

- T_{rekey} : The period of session key updated.
- T_{sess} : The left lifetime of current $K_{sess:p}$. This value must be less than or equal the T_{rekey} .
- SEED : This is a string of characters, alphanumeric characters. It is one of inputs of WEP key generator.
- $E_{Kdiv}[M]$: It is the RC4 encryption of message M by the individual key K_{div} .

Note that there is no IV been used.

- $Seq1, \dots$ and $Seq4$: These are the packet frames of four-step authentication respectively.



3.2.1 The Operation Of Proposed Authentication

In our proposed authentication, the Steps 1 and 4 are the same as well as normal IEEE 802.11, and the Steps 2 and 3 are described as follows.

Step 2: When an authentication frame with sequence no.1 arrives, the AP validates the MAC_{sta} from the frame header. If the station is allowed, the AP finds the station's individual key from the key-mapping table.

The AP prepares a message M , as shown in figure 3-2. The SIG is a hash digest for integrity of the semantic part of M . The keylen field is the length of $K_{sess:p}$. In this thesis, we assume the length of key $K_{sess:p}$ is 40-bit. The generation of $K_{sess:p}$ and SEED will be discussed in section 3.3.1.

The AP first fills MAC_{AP} , MAC_{sta} , TS_{AP} , T_{rekey} , T_{sess} , SEED, keylen, and $K_{sess:p}$ fields. The AP then computes the digest of those fields by hash function, and fills the SIG field within the digest. The hash function can be the SHA-1, which generates a 160-bit output. The message M leaves 61-byte unused padding, we fill a random nonce in it. Finally, the message M is encrypted by WEP and K_{div} , the encrypted M is thought of the challenge text.

The AP then sends the authentication frame with sequence no. 2, which contains the challenge text, to the station. Note that there is no necessary to mention the change of challenge text on the message $Seq2$, because the station can sense it.

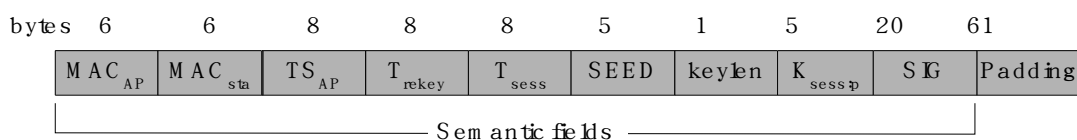


Figure 3-2. The format of message M , which contains some semantics

Step 3: Upon receipt of message $Seq2$, the station first decrypts the challenge text using the key K_{div} for the message M . The station then compares the MAC_{AP} and MAC_{sta} fields with the MAC address of AP and station, respectively. If these fields are not matched, the station will treat the AP as a normal 802.11 device. If matched, the station concludes that the AP is using the proposed method.

The station starts to compute the message digest as well as the AP did. It checks whether the computed digest is equal to the SIG field of message M or not. If the test failed, the station will abort the authentication. If passed, the station installs the $K_{sess:p}$ from the message M into its WEP key-set, which is used to encrypt the data traffic. The other fields of message M will be kept for the generation of session key.

The generation of message $Seq3$ is the same as the normal 802.11 authentication protocol did. The station uses the key, which is a random nonce IV concatenates the K_{div} , to encrypt the authentication frame whose payload is the challenge text. The station then sends the message $Seq3$ back to the AP, and the Step 3 is done.

Finally, the comparison between our proposed authentication and IEEE 802.11 is summarize in Table 3-1 where N is the challenge text, M is a message within some

semantics, and $WEP[M]$ is the WEP encryption of message M . In fact, this change can still be identified by normal IEEE 802.11 authentication, because the main modification is the content of the challenge text.

Table 3-1. The comparison of 802.11 and proposed authentication

Seq	Direction	802.11	RRKM	Description
1	Host AP	Algorithm - 1	Algorithm - 1	Request with algorithm ID
2	Host AP	N	$WEP[M]$	Challenge text
3	Host AP	$WEP[N]$	$WEP[WEP[M]]$	Encrypt Challenge text
4	Host AP	(Un) Successful	(Un)Successful	Return code

3.3 Periodic Key Updated

In the proposed method, our major target aims at the group key updated to strength the vulnerable WEP protocol. This is, because we believe the fact that the attacks have to take between 1,000,000 and 2,000,000 packets to recover a 128-bit WEP key [23]. There have been many proposals addressing on this issue in a similar

manner, which distributes the key frame periodically. Instead of distributing the key frame, our proposed method proposes the AP and stations process the key updated action independently. This operation uses hash function and clock synchronization, as described in designed issues, to generate and synchronize the session key respectively.

This section will describe the generation and re-key mechanism of session key. The mechanism consists of three steps as follows.

Initial step: The first session key is generated and distributed.

Synchronous step: The session key is updated periodically.

Revocation step: A re-authentication is requested.

The details of mechanism are described as in the following. Besides, an example of key updated is given in the final. Figure 3-3 shows the key updated flow.

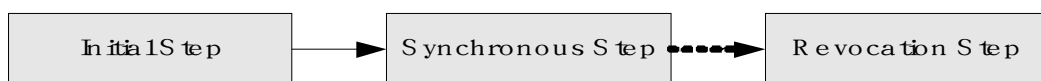


Figure 3-3. The key updated flow chart.

3.3.1 Initial Step

Note that the first session key is always coming from the AP side. When the AP is turned on, it produces two 40-bit random numbers, the first session key $K_{sess:0}$ and SEED. The SEED will be used to generate another session key, and it can safely recycle the same session key by different SEED. In the meantime, the timer of session key is initiated $T_{sess} = T_{rekey}$, since $K_{sess:0}$ is beginning its lifetime. The AP then starts to accept the authentication request until it is turned off. The AP transfers the authentication frame to the station in which the frame is composed of the current session key $K_{sess:p}$ and SEED for the key generation, and the TS_{AP} , T_{rekey} , and T_{sess} for the key synchronization. The detailed operation of authentication has been described in section 3.2.1.



3.3.2 Synchronous Step

Both AP and station use the current session key to communicate by each other until the timer T_{sess} is counting down to zero. The AP and station then have to generate a new session key $K_{sess:p+1}$. They first concatenate the $K_{sess:p}$ with the SEED. The concatenated result then is passed through the hash function. The operation can be presented by $K_{sess:p+1} = Hash(K_{sess:p}, SEED)$, for $p \geq 0$. Finally, the first 40-bit output of hash function is taken to be the new session key. The current session key is replaced by the new one $K_{sess:p+1} \rightarrow K_{sess:p}$ and the timer is reset T_{sess}

= T_{rekey} . Therefore, both AP and station can keep their communication using the updated session key. The session key life between AP and stations is showed in figure 3-4.

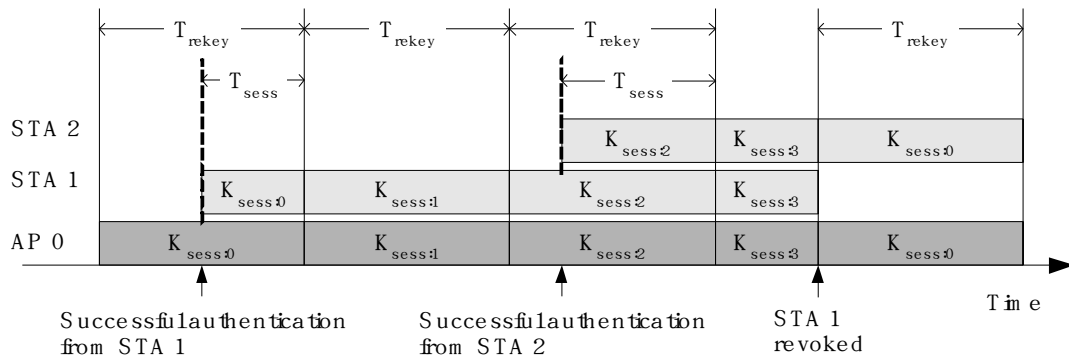


Figure 3-4. The session key life cycle



3.3.3 Revocation Step

In general, there are two conditions that need to enter this step. First, the session key and SEED could be leaking out. Second, the administrator tries to prevent a station from accessing the wireless LAN. If one of these conditions above is true, the AP will send a de-authentication message to every station and enter the initial step again. Therefore, the AP can revoke the illegitimate station and renew the session key and SEED. Figure 3-4 shows that the STA2 is revoked by the AP0 and the STA1 is re-authenticated successfully.

3.3.4 An Illustrated Example

Now, we take figure 3-5 as an example to demonstrate our key updated method. We assume that both STA1 and STA2 are covered under AP0 and T_{rekey} is 60 seconds.

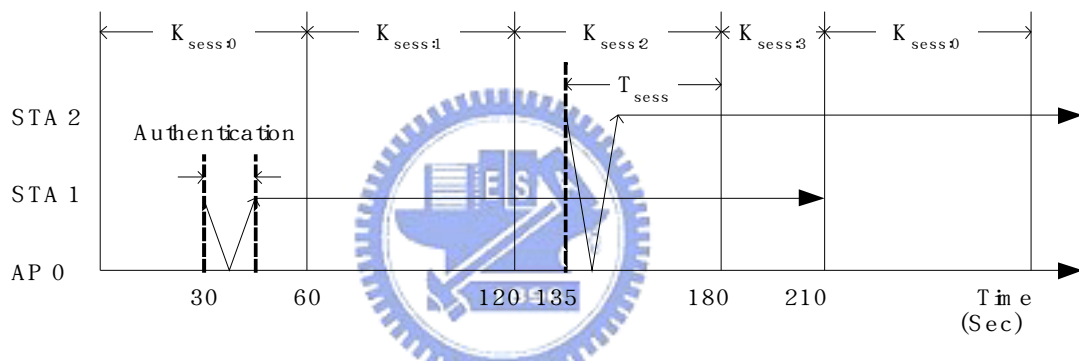


Figure 3-5. An illustrated example for the session key life cycle

Initial Step:

At the first, the AP0 generates two 40-bit random numbers:

$K_{sess:0} = 0x393c577a1d$

SEED = 0x67bc3691e9

Then, set $T_{sess} = T_{rekey} = 60 \text{ sec}$. These settings will be completed before the AP0 begins accepting an authentication request.

At 30 seconds, the STA1 is attempting to join the network. The STA1 starts to send the authentication request, and got the response from AP0 that contains $K_{sess:0}$, SEED, $T_{rekey} = 60 \text{ (sec)}$, and $T_{sess} = 30 \text{ (sec)}$. Upon successful authentication, the STA1 communicates with the AP0 using $K_{sess:0}$ until the T_{sess} counting down to zero. Note that the authentication usually takes few milliseconds (ms), the timer T_{sess} of STA1 minus the propagation and process delay.

Synchronous Step:

At 60 seconds, $T_{sess} = 0$, the STA1 produces a new session key $K_{sess:1}$ as well as AP in the same time. The generation is shown as follows.

$$\begin{aligned} K_{sess:1} &= \text{HASH}(K_{sess:0}, \text{SEED}) \\ &= \text{HASH}(0x393c577a1d, 0x67bc3691e9) \\ &= 0xad735693fb \end{aligned}$$

Then, the AP and STA1 can continue their communication using the new session key

$K_{sess:1}$ for next 60 seconds.

At 135 seconds, the STA2 joins the network successfully while the sequence number of session key is 2. The key generation and synchronization of STA2 is as well as the STA1.

Revocation Step:

At 210 seconds, the administrator considers the STA1 is expired. He first triggers the AP0 sending the de-authentication frame to every authenticated station. The AP0 then backs to the initiation step to produce the first session key $K_{sess:0}$ and reset timer T_{sess} . From now on, the STA1 is revoked and the STA2 joins the network again successfully.

