802.11i Authentication and Key Management (AKM)

White Paper

May 2005

Planet3 Wireless, Inc.
Devin Akin, CTO
Devin@cwnp.com

© Copyright 2005 The CWNP® Program

Introduction

The main problem in coming to grips with all of the facets of 802.11i authentication and key management is the plethora of standards and RFCs that come into play: everything references something else. When a standard assumes the reader already knows details and language from previous standards and RFCs, it makes it all the more difficult to piece together the big picture. This whitepaper should make this process easier to understand, as it can play an important role for wireless security and analysis professionals.

The 802.11i amendment describes the generic process for 802.1X/EAP authentication and key management processes, which includes both generation and distribution of AAA, pairwise, and group keys. There are five specific key types that are of particular interest in the 802.11i amendment. These keys (and their 802.11i definitions) are:

> *Authentication, Authorization, and Accounting (AAA) Key* - Key information that is jointly negotiated between the Supplicant and the Authentication Server (AS). This key information is transported via a secure channel from the AS to the Authenticator. The pairwise master key (PMK) may be derived from the AAA key.
>
> *Pairwise Master Key (PMK)* – The highest order key used within the 802.11i amendment. The PMK may be derived from an Extensible Authentication Protocol (EAP) method or may be obtained directly from a Preshared key (PSK).
>
> *Pairwise Transient Key (PTK)* - A value that is derived from the pairwise master key (PMK), Authenticator address (AA), Supplicant address (SPA), Authenticator nonce (ANonce), and Supplicant nonce (SNonce) using the pseudo-random function (PRF) and that is split up into as many as five keys, i.e., temporal encryption key, two temporal message integrity code (MIC) keys, EAPOL-Key encryption key (KEK), EAPOL-Key confirmation key (KCK).
>
> *Group Master Key (GMK)* - An auxiliary key that may be used to derive a group temporal key (GTK).
>
> *Group Temporal Key (GTK)* - A random value, assigned by the broadcast/multicast source, which is used to protect broadcast/multicast medium access control (MAC) protocol data units (MPDUs) from that source. The GTK may be derived from a group master key (GMK).

These definitions can be confusing unless some reasonable analogy is used to explain their functions in the overall key management scheme of 802.11i. The old adage that a picture is worth a thousand words is true here, so this white paper will provide a visual representation and explanation of this concept..

© Copyright 2005 The CWNP® Program

Key Generation and Distribution

Starting at the highest order in the key hierarchy is the AAA Key. RFC 3748 (2004) gives crucial information in sections 7.2(d), 7.10, and 7.13 about where and when specific keys may be generated and derived.

RFC 3748, section 7.2(d) says:

> *Description of key hierarchy. EAP methods deriving keys MUST either provide a reference to a key hierarchy specification, or describe how Master Session Keys (MSKs) and Extended Master Session Keys (EMSKs) are to be derived.*

RFC 3748, section 7.10 says:

> *The AAA Key is derived from the keying material exported by the EAP method (MSK and EMSK). This derivation occurs on the AAA server. In many existing protocols that use EAP, the AAA Key and MSK are equivalent, but more complicated mechanisms are possible.*

RFC 3748, section 7.13 says:

> *It is possible for the EAP peer and EAP server to mutually authenticate and derive a AAA Key for a ciphersuite used to protect subsequent data traffic. This does not present an issue on the peer, since the peer and EAP client reside on the same machine; all that is required is for the client to derive the AAA Key from the MSK and EMSK exported by the EAP method, and to subsequently pass a Transient Session Key (TSK) to the ciphersuite module.*

Section 7.2(d) tells us that EAP RFCs must either provide some other RFC or standard on which to base a key hierarchy or it must describe how the MSK is to be derived. The EMSK is reserved for future use and is presently not used by any EAP type. At this point, you might ask yourself, "what does an MSK have to do with an AAA Key? Were we not just talking about AAA keys?" Section 7.10 answers that question by explaining that the AAA Key is derived from the keying material exported by (or you could say "generated by") the EAP method. This keying material (or you could say "key") is the MSK. Notice that next it says that in many existing protocols that use EAP, the AAA Key and MSK are equivalent. For the purposes of 802.11i, this is true. Section 8.4.8 of the 802.11i standard says:

> *… the derivation of the PMK from the MSK is EAP-method-specific.*

Section 8.5.1.2 of the 802.11i standard says:

> *When not using a PSK, the PMK is derived from the AAA key. The PMK shall be computed as the first 256 bits (bits 0–255) of the AAA key: PMK ← L(PTK, 0, 256). When this derivation is used, the AAA key must consist of at least 256 bits.*

You can now see how the terms MSK and AAA Key are used synonymously in the 802.11i standard. "MSK" appears only this one time in the standard; other times, the term AAA Key is used. While we know that the PMK is the first 256 bits of the AAA Key (as shown in 802.11i, section 8.5.1.2 above), this leaves the unanswered question, "on what device(s) is/are the PMKs derived?" Again look back at RFC 3748, sections 7.10 and 7.13. They are each saying different things.

© Copyright 2005 The CWNP® Program

Section 7.10 says that the derivation of the PMK from the AAA key occurs on the AAA server (the authentication server). Section 7.13 says that both sides (supplicant and authentication server) can both derive the PMK. These sound like conflicting statements. Fortunately, section 7.2(d) spells it out when it says that it is the EAP method's responsibility to determine who gets a copy of the MSK when it is exported by the EAP process. The MSK and AAA Key are the same in 802.11i, and the PMK is derived from the AAA Key – voilá! All that to figure out what an AAA Key is and where it comes from.

Some EAP types may have the authentication server derive the AAA Key and subsequently send it to the supplicant through a secure channel. Other EAP types may have both the authentication server and the supplicant derive the PMK from the AAA Key individually. 802.11i, section 5.9.2.1 (c) says:

> *The Supplicant and AS authenticate each other and generate a PMK. The PMK is sent from the AS to the Authenticator over the secure channel.*

This leads you to believe that the 802.11i standard designers pretty much expect the AAA Key to exist both on the supplicant and authentication server (though it can certainly be done another way according to RFC 3748). For this reason, we will make this assumption in our illustration.

Consider that the Pairwise Master Key (PMK) gets its name because 1) it is only used between two (hence, a pair) stations – which are an AP and a client station – and 2) it is a "master key." The PMK is not used to encrypt data, but rather it is only used to produce PTKs. The Group Master Key (GMK) likewise gets its name because 1) it is a "group key" – meaning used for the benefit of the AP and more than one client station – and 2) because it's a "master key." Likewise, the GMK is not used to encrypt data, but only to produce GTKs.

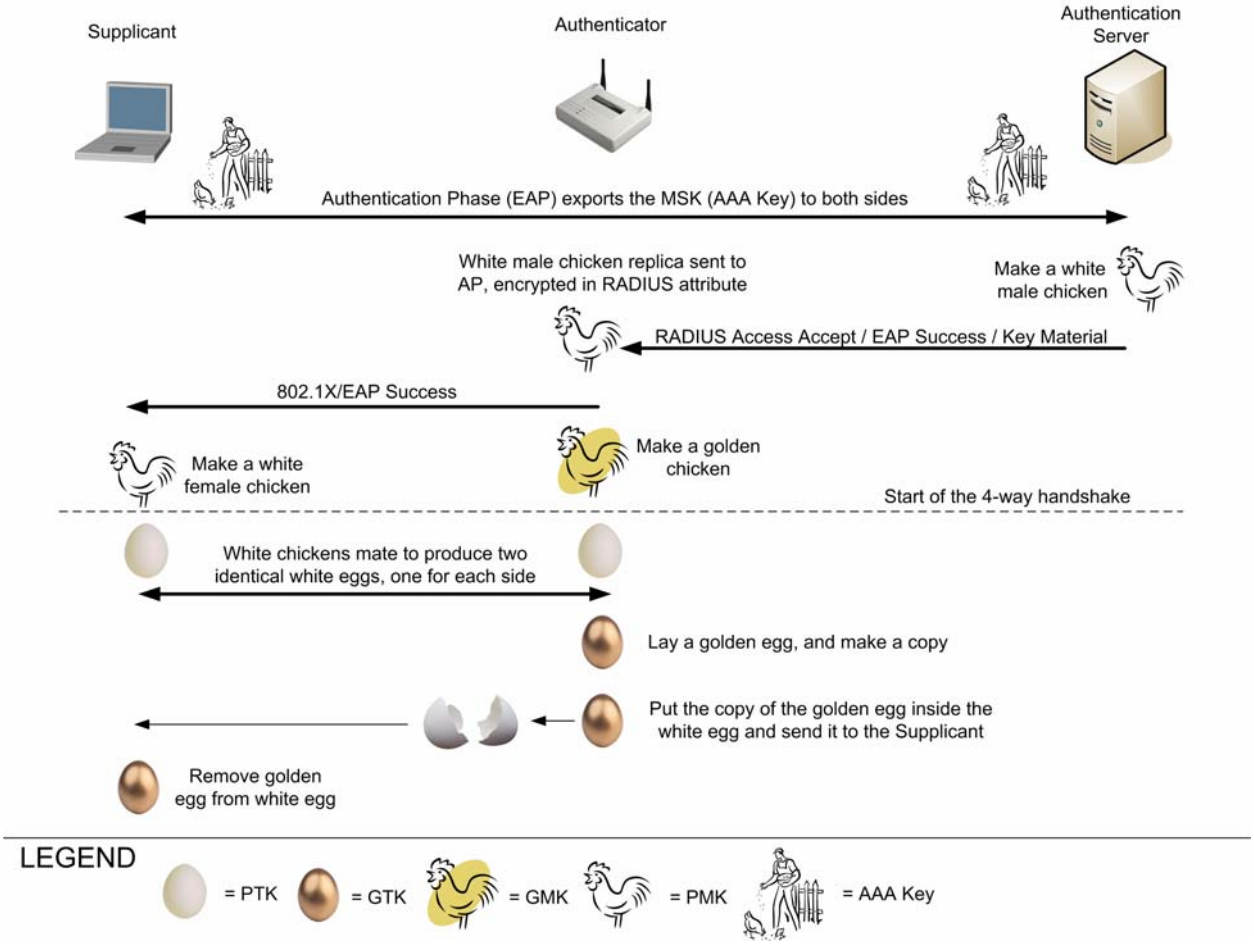
Master keys (whether pairwise or group) are not used to encrypt data, but rather to produce encryption keys (called "temporal keys") that are used to encrypt data. PTKs are used to encrypt unicast data between a client station and an AP, and GTKs are used to encrypt broadcast/multicast data between a client station and an AP. Now we have to get to the part where PMKs and GMKs are generated and subsequently PTKs and GTKs are derived. This is where an illustration is needed for better understanding

For the sake of verbalizing the illustrations, we will refer to the AAA Key as a chicken farmer, the PMK as a white chicken, the GMK as a golden chicken, the PTK as a white egg, and the GTK as a golden egg. It is important to know where each farmer lives, where each chicken type is created, and when/where each egg is produced.

Refer to Figure 1, and follow it from top to bottom to understand the flow of farmer, chicken, and egg generation and distribution. Keep in mind that the EAP authentication may give the AAA Key to the authentication server only or possibly to the authentication server and supplicant. Our example uses an EAP type that allows the AAA Key to reside on both the authentication server and supplicant after EAP authentication completes. A detailed study on EAP types is beyond the scope of this whitepaper. Figure 1 has a legend at the bottom for ease of understanding. Figure 1 only addresses key management when using 802.1X/EAP authentication.

© Copyright 2005 The CWNP® Program

FIGURE 1



Once the chicken farmer exists at the supplicant and the authentication server, the authentication server uses the chicken farmer to make a white chicken. A copy of the white chicken is sent in a RADIUS Access Accept frame to the authenticator. Next, the EAP Success message is removed from the RADIUS Access Accept frame and sent across the wireless medium to the supplicant to let it know that its EAP authentication was successful. The supplicant now knows that it is ok to have the chicken farmer produce a white chicken.

For example, lets say that the supplicant produces a female white chicken and the authentication server produces a male white chicken, which is sent to the authenticator.

Now the authenticator must make a golden chicken. This ends the chicken making process. Next are the processes of laying eggs, which in 802.11i terms are called the, "4-Way Handshake" and "Group Key Handshake."

The 4-Way Handshake is used by the supplicant and authenticator to produce a golden egg, produce two identical white eggs, and to get a copy of the golden egg from the authenticator to the supplicant. (There is more cryptographic detail to it than that going on behind the scenes).

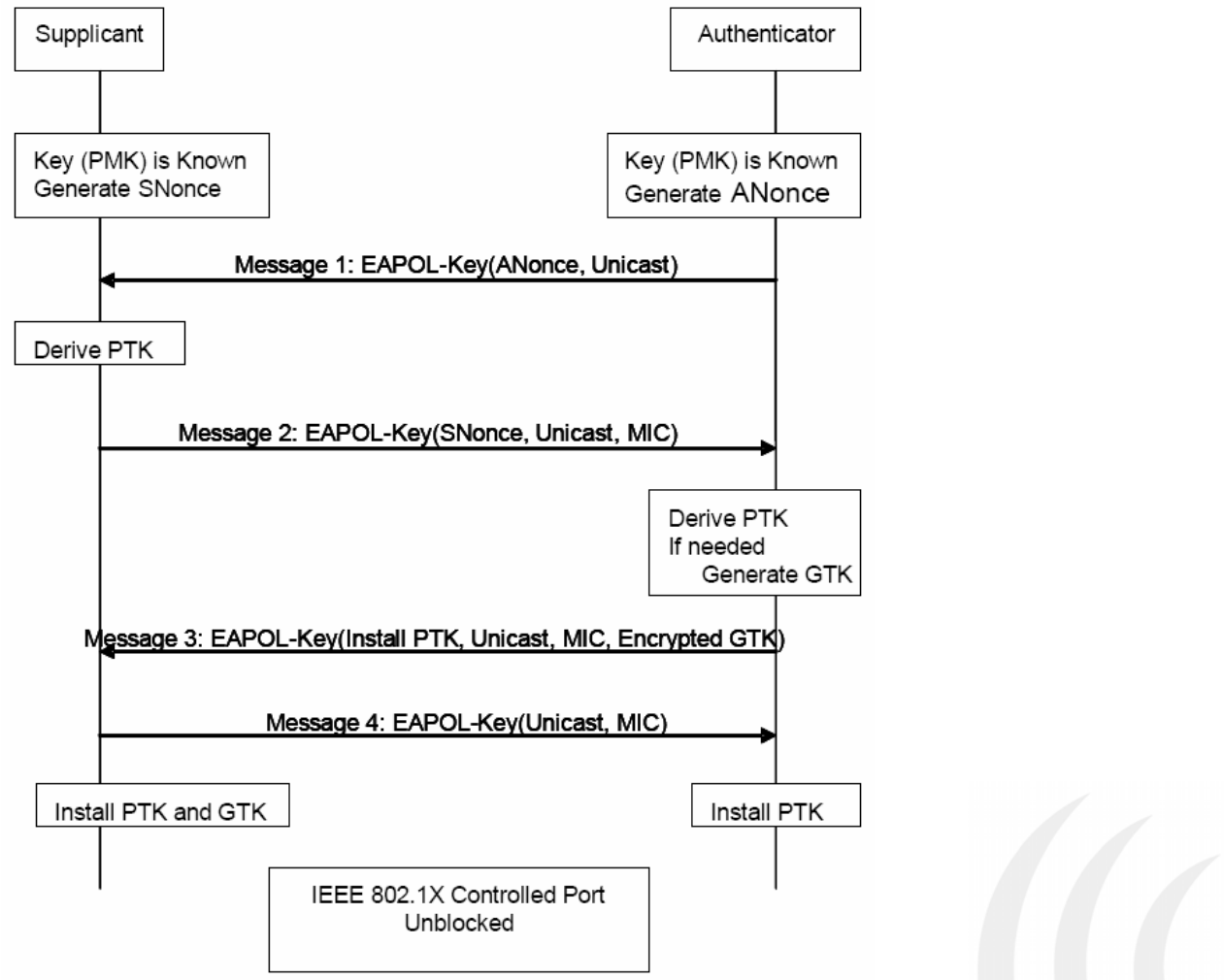
The 802.11i standard says this about the 4-Way Handshake:

A 4-Way Handshake utilizing EAPOL-Key frames is initiated by the Authenticator to do the following:

- Confirm that a live peer holds the PMK.
- Confirm that the PMK is current.
- Derive a fresh pairwise transient key (PTK) from the PMK. Install the pairwise encryption and integrity keys into IEEE 802.11.
- Transport the group temporal key (GTK) and GTK sequence number from Authenticator to Supplicant and install the GTK and GTK sequence number in the STA and, if not already installed, in the AP.
- Confirm the cipher suite selection.

The 802.11i process flow chart for the 4-Way Handshake is shown in Figure 2.

FIGURE 2 (from IEEE 802.11i)

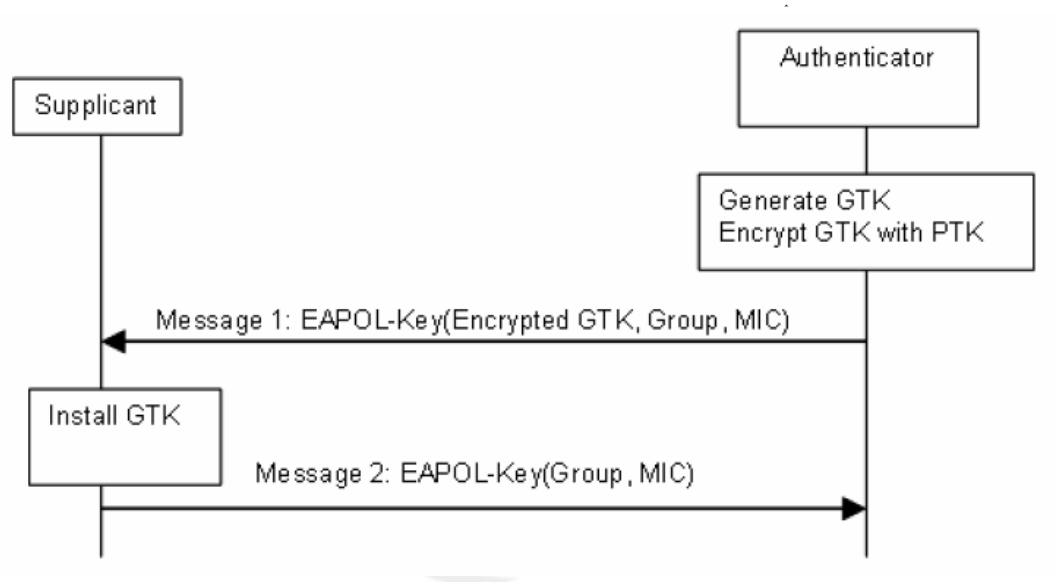


The Group Key Handshake could be considered a 2-way handshake. 802.11i states that it is used for the following purposes:

If the Authenticator later changes the GTK, it sends the new GTK and GTK sequence number to the Supplicant using the Group Key Handshake to allow the Supplicant to continue to receive broadcast/multicast messages and, optionally, to transmit and receive unicast frames. EAPOL-Key frames are used to carry out this exchange.

Without a separate Group Key Handshake, a new PTK would have to be generated and installed by the authenticator and every supplicant every time the GTK needed to be changed. This 2-way handshake lends simplicity and much less overhead to broadcast/multicast key generation and distribution to supplicants. The 802.11i process flow chart for the Group Key Handshake is shown in Figure 3.

FIGURE 3 (from IEEE 802.11i)



Let's step through the 4-Way Handshake using our little chicken/egg analogy. The first step is for the male and female chickens to work together (mate) to produce two identical white eggs (PTKs), and each chicken gets one. Second, the authenticator's golden chicken lays two golden eggs, and one is sent to the supplicant inside of a white egg (considered a secure channel).

It is important to note that only those stations holding a white egg can open anything wrapped in a white egg. Likewise only those stations holding a golden egg can open anything wrapped in a golden egg. When the supplicant opens the white egg, it receives a prized golden egg. Now both the supplicant and authenticator can communicate unicast and broadcast/multicast data securely. Now is the perfect time to remind the reader that only temporal keys (eggs) are used to encrypt data, and they are periodically changed by the authenticator and supplicant during either a 4-Way Handshake or the Group Key Handshake. The authenticator is the sole authority on when new golden eggs are produced because it holds the only golden chicken.

Reauthentication

The next question becomes, "What about reauthentication?" With 802.11i compliant systems, a supplicant only needs to authenticate one time – upon initial attachment to the wireless network. Once the supplicant is authenticated, the white chickens are produced by the chicken farmers at the supplicant and authentication server, and the white chicken is subsequently sent to the authenticator, the supplicant and authenticator distinguish the white chickens by their ID numbers (PMKIDs).

© Copyright 2005 The CWNP® Program

When the supplicant roams, it sends the white chicken number (PMKID) in the reassociation request frame, notifying the authenticator that it wishes to use this cached white chicken to produce fresh white eggs rather than go through the 802.1X/EAP reauthentication process to generate another one (which would take a long time). The 802.11i standard states it this way:

> *An AP can retain PMKs for STAs in the ESS to which it has previously performed a full IEEE 802.1X authentication. If a STA wishes to roam to an AP for which it has cached one or more PMKSAs, it can include one or more PMKIDs in the RSN information element of its Reassociation Request frame. An AP whose Authenticator has retained the PMK for one or more of the PMKIDs can skip the 802.1X authentication and proceed with the 4-Way Handshake. The AP shall include the PMKID of the selected PMK in Message 1 of the 4-Way Handshake. If none of the PMKIDs of the cached PMKSAs matches any of the supplied PMKIDs, then the Authenticator shall perform another IEEE 802.1X authentication. Similarly, if the STA fails to send a PMKID, the STA and AP must perform a full IEEE 802.1X authentication.*

Being able to use cached white chickens to produce fresh white eggs rather than generate new white chickens (and then subsequently have them lay fresh white eggs) shortens the process of roaming from hundreds of milliseconds to tens of milliseconds, yielding a roaming speed suitable for VoIP and other similar time/latency sensitive applications.

The next piece of the 802.11i key management puzzle is use of preshared keys (PSKs). When using PSKs, the 802.11i standard calls for a passphrase-to-PSK mapping scheme for the following reasons (from 802.11i)

> *The RSNA PSK consists of 256 bits, or 64 octets when represented in hex. It is difficult for a user to correctly enter 64 hex characters. Most users, however, are familiar with passwords and pass-phrases and feel more comfortable entering them than entering keys. A user is more likely to be able to enter an ASCII password or pass-phrase, even though doing so limits the set of possible keys. This suggests that the best that can be done is to introduce a pass-phrase to PSK mapping.*
>
> *This clause defines a pass-phrase–to–PSK mapping that is the recommended practice for use with RSNAs. This pass-phrase mapping was introduced to encourage users unfamiliar with cryptographic concepts to enable the security features of their WLAN.*
>
> *A pass-phrase typically has about 2.5 bits of security per character, so the pass-phrase mapping converts an n octet password into a key with about 2.5n + 12 bits of security. Hence, it provides a relatively low level of security, with keys generated from short passwords subject to dictionary attack. Use of the key hash is recommended only where it is impractical to make use of a stronger form of user authentication. A key generated from a pass-phrase of less than about 20 characters is unlikely to deter attacks.*

The formula used by the 802.11i standard for deriving a PSK from a passphrase is as follows:

> *The pass-phrase mapping defined in this subclause uses the PBKDF2 method from PKCS [B16]. PSK = PBKDF2(PassPhrase, ssid, ssidLength, 4096, 256)*
>
> *Here, the following assumptions apply:*

- A pass-phrase is a sequence of between 8 and 63 ASCII-encoded characters. The limit of 63 comes from the desire to distinguish between a pass-phrase and a PSK displayed as 64 hexadecimal characters.
- Each character in the pass-phrase must have an encoding in the range of 32 to 126 (decimal), inclusive.
- ssid is the SSID of the ESS or IBSS where this pass-phrase is in use, encoded as an octet string used in the Beacon and Probe Response frames for the ESS or IBSS.
- ssidLength is the number of octets of the ssid.
- 4096 is the number of times the pass-phrase is hashed.
- 256 is the number of bits output by the pass-phrase mapping.
- Section 5.9.2.2 of the 802.11i standard tells us that the PSK mapped from the passphrase is used as the PMK in both the authenticator and the supplicant.

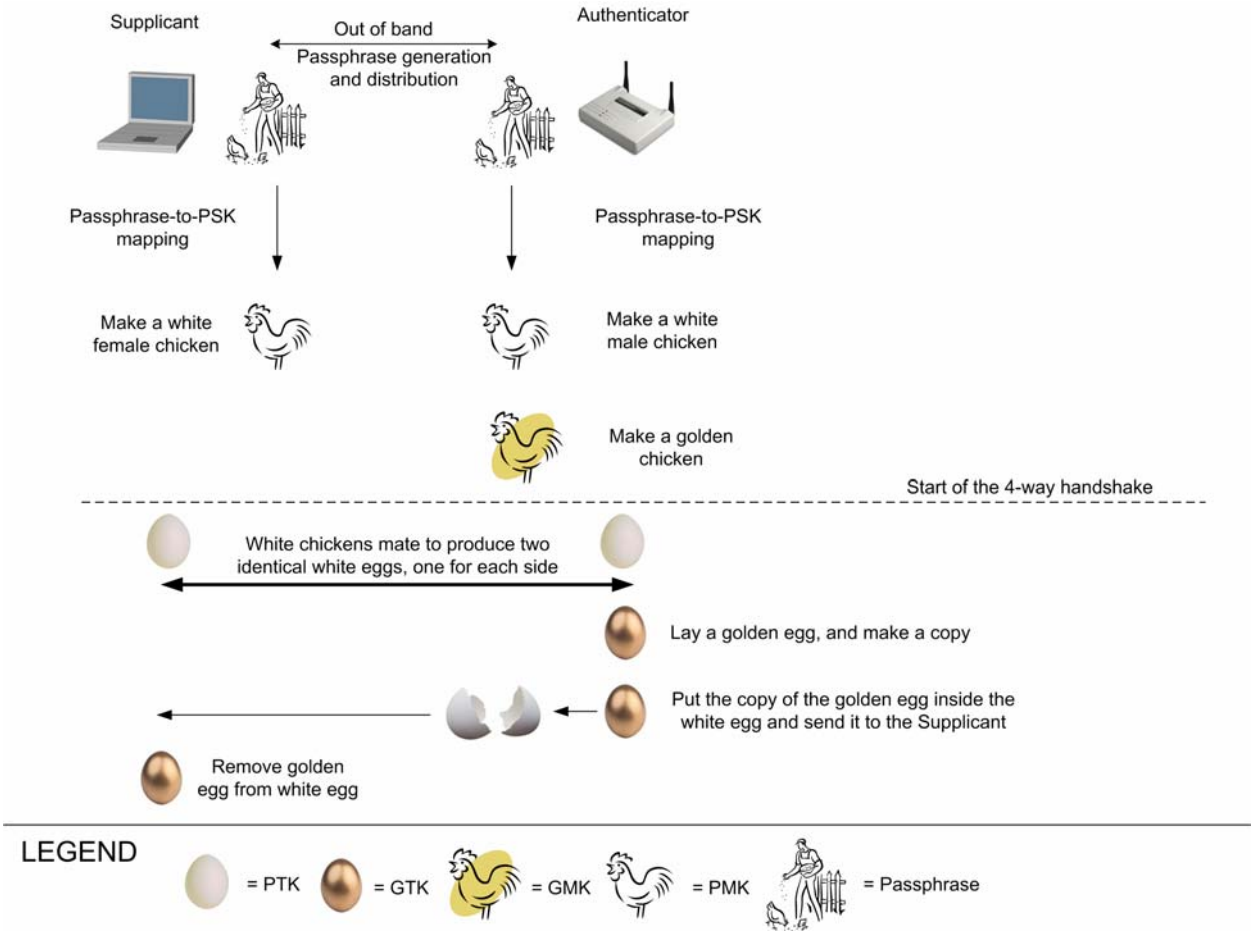
The following AKM operations are carried out when the PMK is a PSK:

- A STA discovers the AP's security policy through passively monitoring Beacon frames or through active probing. A STA associates with an AP and negotiates a security policy. The PMK is the PSK.
- The 4-Way Handshake using EAPOL-Key frames is used just as with IEEE 802.1X authentication, when an AS is present.
- The GTK and GTK sequence number are sent from the Authenticator to the Supplicant just as in the AS case.

To summarize, the passphrase should be thought of as an AAA Key (chicken farmer), even though the process of deriving a PMK from an AAA Key is completely different from deriving the PMK from the passphrase.

In both 802.1X/EAP and PSK authentication, once the PMKs (white chickens) are located on both the authenticator and the supplicant and the GMK (golden chicken) has been generated by the authenticator, the 4-Way Handshake can begin. From that point, the temporal key (egg) generation is the same for both authentication types. Figure 4 gives a chicken/egg graphical representation of the passphrase generation and distribution, passphrase-to-PSK mapping, and generation of PMKs and GMKs. This is followed by the 4-Way Handshake we have previously seen.

FIGURE 4



Summary

A wireless LAN administrator who typically performs tasks such as configuring access points, clients, WLAN switches, and enterprise wireless gateways will probably not need this level of information to perform their job. While a reasonable level of understanding of authentication should be in the administrator's repertoire, a detailed understanding of key management is usually left to security and analysis professionals. Key management is a difficult topic to understand with all of the standards and RFCs, and this white paper should help clarify how it works.