

# WLAN Security

## 7. WPA

(C) Herbert Haas 2006/4/1

### Content

In this chapter a detailed overview about today's WLAN security problems and solutions are presented.

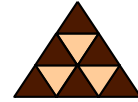
This subchapter provides an overview about the WPA procedures.

### Objective

After completing this chapter the following tasks could be solved:

- Emphasize the basic vulnerabilities of WLAN
- Explain why WEP is insecure and give a mathematical example
- Compare WEP and TKIP/MIC
- Highlight the design flaws of the WLAN standard authentication
- Explain the design idea of 802.1x
- Compare EAP-TLS, LEAP, PEAP, EAP-TTLS, EAP-FAST with each other and emphasize important security features
- Explain the design concept of WPA and WPA2
- Implement a reliable 802.1x infrastructure over a WAN connection
- List important issues to be considered when choosing a VPN design
- Explain PSPF

# Introduction



- **802.1x alone does not (need to) provide key management**
  - ♦ Often 802.1x is simply combined with WEP
  - ♦ Even 802.1x with TKIP would always start with same base key
- **Basic Idea of WPA:**
  - ♦ Strong per-user, per-session, per-packet keying (TKIP and MIC)
  - ♦ Use 802.1x and dynamical transient key management
  - ♦ Alternatively pre-shared keys (SOHO apps.) instead of 802.1x
- **WPA starts with a security capability negotiation**
  - ♦ Therefore cipher suites must be configured on AP
  - ♦ APs advertises capabilities in beacon and in probe-response frames
    - "Cipher Suite" = Auth. Method + Encryption Method
  - ♦ Client can select the desired method during association request

The basic idea of WPA is to **combine 802.1x authentication with TKIP and MIC**. Furthermore, dynamically established **master keys** should be the basis to calculate dynamic per-user, per-session, and per-packet keys, using TKIP.

Key management can be performed either through RADIUS (like 802.1x is doing, and then it is called "WPA-EAP") or alternatively via **pre-shared keys** without any additional servers. Both mechanisms will generate a master session key for the Authenticator (AP) and Supplicant (client station).

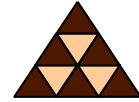
WPA allows to configure "**cipher suites**" on the AP, while the clients may select the most appropriate one during the association process.

The master key is calculated "pair-wise", that is on the AP as well as on the client device, either based on 802.1x authentication states or on a Pre-Shared-Key (PSK). The **WPA-PSK** method is only used, when there is no Authentication Server available, typically in home installations.

**Note:** WPA-PSK is not supported by Cisco ADU or ACU.

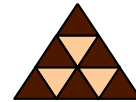
**Note:** When using WPA encryption on an access point, encryption key 1 must not be used as the WPA key negotiation mechanism uses this key position in the AP to transfer authentication data to the client.

# WPA/WPA-2



- **Certified EAP Methods**
  - ◆ EAP-TLS (originally the only one)
  - ◆ EAP-TTLS/MSCHAPv2
  - ◆ PEAPv0/EAP-MSCHAPv2
  - ◆ PEAPv1/EAP-GTC
  - ◆ EAP-SIM
- **Native OS support**
  - ◆ Windows XP with Service Pack 2 and WPA2 patch
  - ◆ No support for Win2k
  - ◆ Linux: *wpa\_supplicant* (large feature set)

# WPA Concepts



- **1) Pairwise Master Key (PMK) is negotiated between client and AS**
  - Based on 802.1x credentials or based on a PSK in home environments
  - PMK is designed to last the entire session
  - Should be exposed as little as possible (therefore PTK needed)
- **2) PMK is pushed from AS to AP**
  - Via RADIUS-Access-Accept message
- **3) AP generates Pairwise Transient Key (PTK)**
  - Negotiated via Four-Way Handshake to client
  - $PTK = \text{HASH}(PMK, AP\_nonce, STA\_nonce, AP\_MAC, STA\_MAC)$
  - From PTK, other working keys are generated (KCK, KEK, TK)
- **4) AP also derives a Group Temporal Key (GTK)**
  - To decrypt multicast and broadcast traffic
  - Must be the same on all clients (!)
  - Need to be updated periodically (e. g. when a device leaves the network)
  - AP sends new GTK to each client, encrypted with client's PTK
  - Each client must acknowledge the new GTK

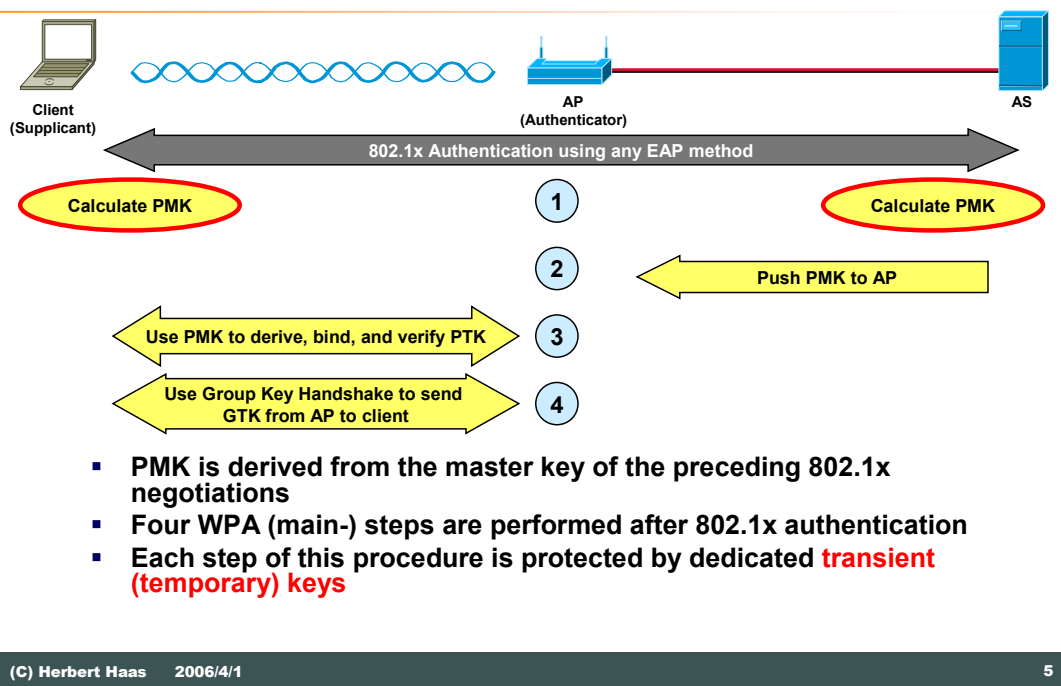
Unlike WEP, which uses a single key for unicast data encryption and typically a separate key for multicast and broadcast data encryption, WPA uses a set of four different keys for each wireless client-wireless AP pair (known as the pairwise temporal keys) and a set of two different keys for multicast and broadcast traffic.

This set of messages exchanges the values needed to determine the pairwise temporal keys, verifies that each wireless peer has knowledge of the PMK (by verifying the value of the MIC), and indicates that each wireless peer is ready to begin encrypting and providing message integrity protection for subsequent unicast data frames and EAPOL-Key messages.

For multicast and broadcast traffic, the wireless AP derives a 128-bit group encryption key and a 128-bit group integrity key and sends these values to the wireless client using an EAPOL-Key message, encrypted with the EAPOL-Key encryption key and integrity-protected with the EAPOL-Key integrity key. The wireless client acknowledges the receipt of the EAPOL-Key message with an EAPOL-Key message.

When a device leaves the network, the GTK also needs to be updated to prevent the device from receiving any more multicast or broadcast messages.

# The Basic Steps

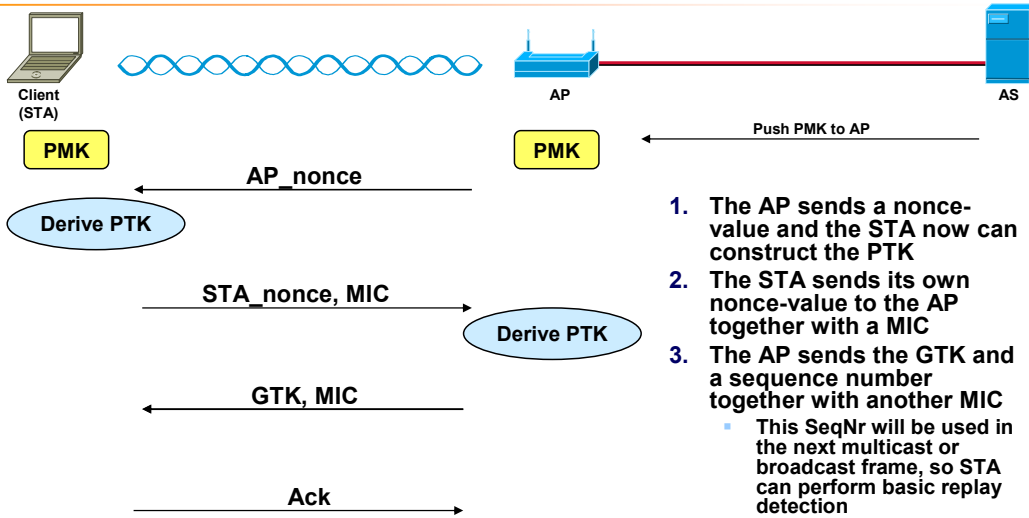
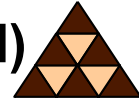


The **Pairwise Master Key (PMK)** is typically calculated using some authentication data which had been derived at the end of a preceding 802.1x/EAP negotiation. For example if EAP-TLS were used, then the PMK = PRF (MasterKey, clientHello.random, serverHello.random, "client EAP encryption")

WPA implements a new 4-Way Handshake and a Group Key Handshake for generating and exchanging data encryption keys between the Authenticator and Supplicant. This handshake is also used to verify that both Authenticator and Supplicant know the master session key.

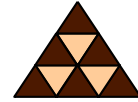
**Note:** WPA also includes the requirement to use open key authentication and to obsolete the flawed shared-key authentication. Like 802.11i, WPA capabilities are advertised in beacons, probe responses, association requests, and reassociation requests.

# WPA – Basic Handshake (Simplified)



1. The AP sends a nonce-value and the STA now can construct the PTK
2. The STA sends its own nonce-value to the AP together with a MIC
3. The AP sends the GTK and a sequence number together with another MIC
  - This SeqNr will be used in the next multicast or broadcast frame, so STA can perform basic replay detection
4. The STA sends a confirmation to the AP

# WPA Details – Transient Keys



- **The PTK (256 bit) is the basis to derive additional transient keys**
  - ♦ **Data Encryption Key (128 bit)**
    - For unicast frames
    - Aka Temporal Key (TK)
  - ♦ **Data Integrity Key (128 bit)**
    - For unicast MIC
  - ♦ **Key Encryption Key (KEK, 128 bit)**
    - To encrypt EAPoL key messages
  - ♦ **Key Integrity Key (KIK, 128 bit)**
    - To calculate the MIC for EAPoL key messages
- **The GTK (256 bit) is the basis to derive**
  - ♦ **A Group Encryption Key (GEK)**
  - ♦ **A Group Integrity Key (GIK)**

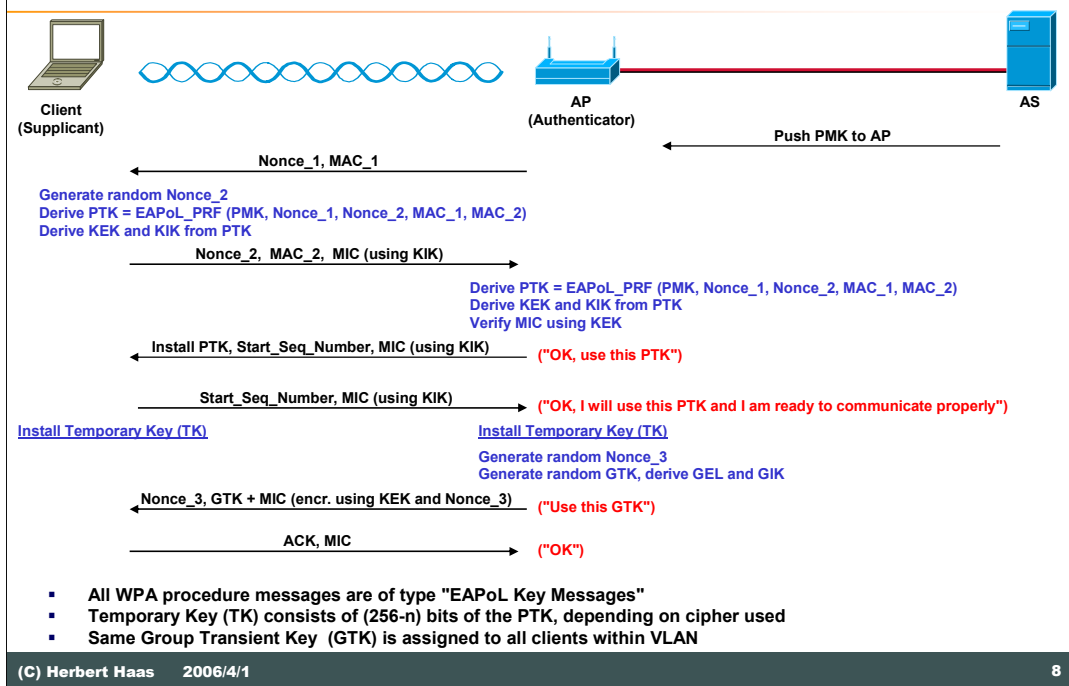
Based on the PTK, several temporary working keys are derived:

- A 128-bit **Data Encryption Key** for unicast transmission which is similar as a WEP key and consists of 256-n bits of the PTK key.
- A 128-bit **Data Integrity Key** for unicast MIC
- A 128-bit **EAPoL Key Encryption Key (KEK)** to encrypt EAPoL key messages. This key simply consists of the bits 128-255 of the PTK.
- A 128-bit **EAPoL Key Integrity Key (KIK)** to calculate the MIC for EAPoL key messages. This key is also called "Key Confirmation Key" (KCK) and consists of the bits 0-127 of the PTK.

Based on the GTK, these temporary working keys are derived:

- A 128-bit **Group Encryption Key (GEK)**, which is also known as Group Transient Key (GTK) to encrypt multicast and broadcast frames. This key simply consists of the bits 128-255 of the GTK.
- A 128-bit **Group Integrity Key (GIK)** to calculate the MIC for multicast and broadcast frames. This key simply consists of the bits 0-127 of the PTK.

# (WPA – Detailed)

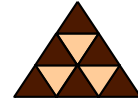


The temporary key exchange is **initiated by the AP** and consists of the following steps:

1. The AP sends an EAPoL key message including Nonce\_1 and MAC\_1. This message is not encrypted, and no MIC is possible at that stage.
2. Now, the client can calculate a "**Pairwise Transient Key**" (PTK) and derives the KEK and KIK.
3. The client sends an EAPoL key message including Nonce\_2 and MAC\_2 plus MIC. The MIC is calculated using the EAPoL-KIK.
4. Now, the AP can also derive the PTK and can verify the MIC.
5. The AP sends an EAPoL key message including a MIC and a start-sequence-number to indicate that the AP is now ready to send encrypted unicast frames as well as EAPoL key frames.
6. The client also sends an EAPoL key message including a MIC and a start-sequence-number to indicate that the client is now also ready to send unicast frames as well as EAPoL key frames.
7. The AP finally calculates a 128-bit Group Encryption Key (GEK) as well as a 128-bit Group Integrity Key (GIK) and transmits these values via an EAPoL key message (encrypted with EAPoL-KEK and protected by EAPoL-KIK) to this client.
8. The client acknowledges this message by sending a valid EAPoL key message.

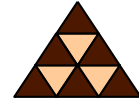
**Note:** The basic idea of all this is to use a PMK to generate "fresh" PTKs for encryption.

# GTK Issues



- **GTK is either**
  - ♦ A pseudo-random number chosen by AP
  - ♦ The first PTK that the AP uses
- **GTK Usage**
  - ♦ Cannot be used with sequence numbers because it is used for **ALL** clients
    - Distant clients might overhear some frames
  - ♦ So management and broadcast frames are encrypted via **WEP** only
    - Broadcast key rotation recommended

## WPA-2



- **WPA2 mandates both TKIP and AES capability**
- **PMK caching support**
  - ◆ AP caches credentials to allow fast reconnect
- **Pre-authentication support**
  - ◆ Allows a client to pre-authenticate with the AP toward which it is moving
  - ◆ But still maintains a connection to the AP it's moving away from
- **Roaming times below 100 ms**