

1 序	5
2 802.11中的安全方法	6
2.1 认证	6
2.1.1 认证方法	6
2.1.2 两种认证方法的区别	7
2.1.3 认证配置	8
2.2 WEP加密	8
2.2.1 加密算法	8
2.2.2 数据结构	9
2.2.3 WEP加密（认证）图解	11
2.3 流加密和块加密	11
3 802.11i-认证	12
3.1 802.11i的认证体系	12
3.2 PSK认证	14
3.2.1 配置	14
3.2.2 认证过程	15
3.2.3 PSK认证图解	16
3.3 802.1x认证	16
3.3.1 802.1x的认证机制	16
3.3.2 802.1x的体系结构	17
3.3.3 EAP	18
3.3.4 RADIUS	20
3.3.5 802.1x认证过程	22
3.3.6 802.1x认证图解	26
3.3.7 配置	26
3.3.8 认证终端	26
3.3.9 DOMAIN-ISP	27

3.3.10	RADIUS SCHEME	27
4	802.11i-加密	28
4.1	中间安全机制-WPA	28
4.2	TKIP	28
4.2.1	TKIP MPDU formats	29
4.2.2	TKIP加密与接收	30
4.2.3	TKIP MIC和countermeasures procedures	30
4.3	CCMP : CTR with CBC-MAC Protocol	31
4.3.1	CCMP MPDU formats	32
4.3.2	CCMP的加密与接收	32
5	802.11i-密钥协商与分发	33
5.1	802.11i中的各种key	33
5.1.1	PMK	33
5.1.2	PMK的作用	34
5.1.3	PTK, KEK, KCK	35
5.1.4	MIC	37
5.2	密钥分发	37
5.2.1	PTK分发	37
5.2.2	GTK 分发	38
5.3	4-Way Handshake	38
5.3.1	4-Way Handshake: STEP 1	38
5.3.2	4-Way Handshake: STEP 2	39
5.3.3	4-Way Handshake: STEP 3	40
5.3.4	4-Way Handshake: STEP 4	41
图1 Open system authentication		6
图2 Shared key authentication		7

图3 Open system authentication	7
图4 Shared key authentication	8
图5 客户端设置.....	8
图6 WEP加密算法	9
图7 WEP加密数据结构	9
图8 WEP数据包	10
图9 WEP加密图解	11
图10 流加密.....	12
图11 块加密.....	12
图12 802.11i认证体系.....	13
图13 PSK认证客户端配置	14
图14 PSK认证过程	15
图15 PSK认证图解	16
图16 802.1x认证系统工作机制.....	17
图17 802.1x体系结构.....	18
图18 EAPOL报文结构.....	18
图19 EAP报文结构	19
图20 RADIUS报文结构.....	21
图21 RADIUS典型消息.....	21
图22 RADIUS计费报文.....	21
图23 RADIUS中的EAP属性	22
图24 802.1x认证过程.....	23
图25 密钥生成过程.....	24
图26 4次握手和6次握手.....	25
图27 6次握手中的GTK协商报文.....	25
图28 802.1x认证图解.....	26
图29 802.1x认证客户端配置.....	26
图30 TKIP密钥生成机制	29
图31 TKIP MPDU	30

图32 CCMP MPDU	32
图33 CCMP加密过程.....	33
图34 Key cache 漫游过程	34
图35 PMKID 1.....	35
图36 PMKID 2.....	35
图37 加密的key data域	36
图38 非加密的key data域	36
图39 组播密钥协商.....	37
图40 PTK 分发.....	37
图41 GTK分发	38
图42 4步握手1.....	39
图43 4步握手2.....	40
图44 4步握手3.....	41
图45 4步握手4.....	42
图46 组播密钥单独协商.....	42

1 序

Wlan设备，即基于802.11的设备使用radio frequencies 通信，RFs，射频。发送者广播数据希望在RF范围内的接收者可以收到，但同时，任何处于范围内的station都能接受到这个信息。

所以，我们需要一个安全机制来保证数据的私密性；

在wlan网络里面为了提供最低限度的安全机制需要两个部分：

- 1，一种方法，它来决定谁可以使用这个WLAN网络，通过认证来实现
- 2，一种方法，它来保证数据的私密性，通过加密机制来解决

WLAN安全=认证+加密

2 802.11 中的安全方法

2.1 认证

2.1.1 认证方法

认证方法有两种：Open system authentication 与 Shared key authentication；这两种方法在802.11中都有比较详细的讲述；

(1) Open system authentication: 开放系统认证是缺省使用的认证机制，即不认证。如果认证类型设置为开放系统认证，则所有请求认证的客户端都会通过认证。开放系统认证包括两个步骤：第一步是请求认证，第二步是返回认证结果。如果认证结果为“成功”，那么客户端和AP就通过双向认证。

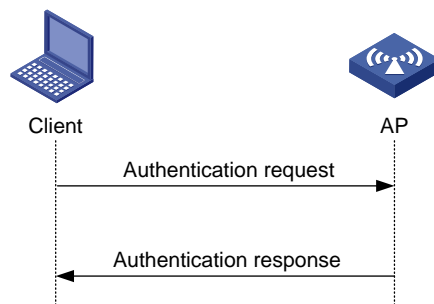


图1 Open system authentication

(2) Shared key authentication: 用于pre-RSN设备的认证，pre-RSN设备为不支持RSNAs (Robust Security Network Associations，健壮安全网络连接) 的设备；

在802.11i中规定：

Shared Key authentication can be used if and only if WEP has been selected.

所以除了WEP认证，其余的认证方式（802.1X与MAC）都不会选择Shared key的方式，只能选择Open system方式；

Shared key方式在认证前STA和AC上都预先设置好密码，可以称为预设密钥；

它的认证过程是4步；

第一步： 认证请求；

第二步： 请求响应；明文发送挑战报文；它的作用是什么？802.11i的描述：

Before sending the second frame in the Shared Key authentication sequence, the responder shall

use WEP to generate a string of octets to be used as the authentication challenge text.

Authentication algorithm dependent information = The authentication result

注意：如果第二步的认证结果是失败的，则不会有以下两步的交互；如果认证结果是成功的，则会有下面额外的信息：

Authentication algorithm dependent information = The challenge text

第三步：STA用key加密明文发送；AC收到密文数据包之后解密与明文对照；

Authentication algorithm dependent information = The challenge text from the second frame

第四步：认证结果；

Authentication algorithm dependent information = The authentication result

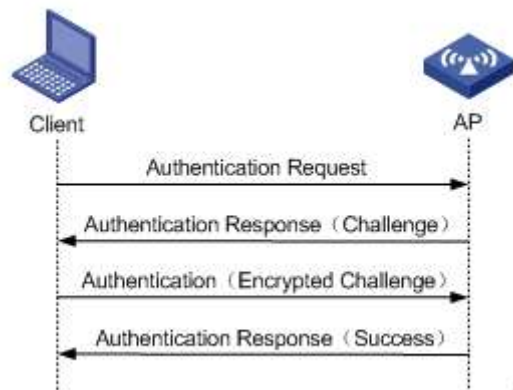


图2 Shared key authentication

2.1.2 两种认证方法的区别

如下：

前者是Open system 认证，特点是即使双预先配置的key是不同的，用户也可以通过认证，但是数据报文是不能交互的，在AP收到数据报文之后检查发现key是错误的，报文将被丢弃。

后者是Shared key 认证，特点是双方配置的密钥必须是一致的，否则不能通过认证。

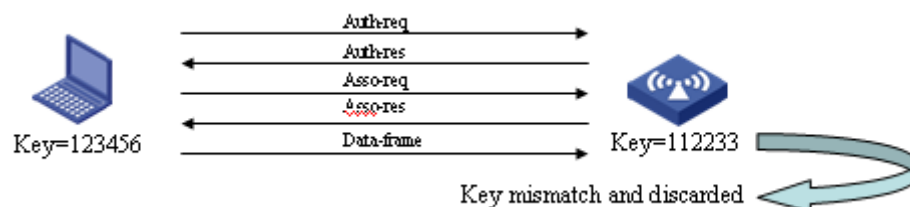


图3 Open system authentication

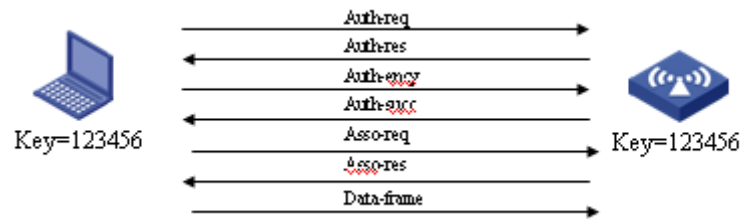


图4 Shared key authentication

2.1.3 认证配置

两边都需要预先知道静态key，AC上在模板中配置，客户端上线时预先设置，当然也可以不设置，默认取第一个；

客户端上线的时候也要输入：

网络名 (SSID) (N): nsw-wep40

无线网络密钥

此网络要求下列密钥：

网络验证 (A): 开放式

数据加密 (Q): WEP

网络密钥 (K): *****

确认网络密钥 (Q): *****

密钥索引 (高级) (X): 3

图5 客户端设置

认证的时候客户端与AC上的配置密钥要对应；

2.2 WEP 加密

2.2.1 加密算法

WEP: Wired Equivalent Privacy, 有线等效加密

WEP的特点：

- 采用基于RC4对称流加密算法的WEP加密
- STA和AP需要预先配置相同的静态Key，Key的长度为40 bit或104bit
- 每次对数据加密的Key=静态Key+24bit的IV值（IV值为动态生成）

WEP的不足之处：

- 所有的STA共用相同的静态Key：一个用户的密钥泄漏，所有用户的安全都被威胁；静态Key泄漏被发现前，网络存在安全隐患
- IV值太短：很容易被破解

802.11i中描述的WEP加密的算法图：

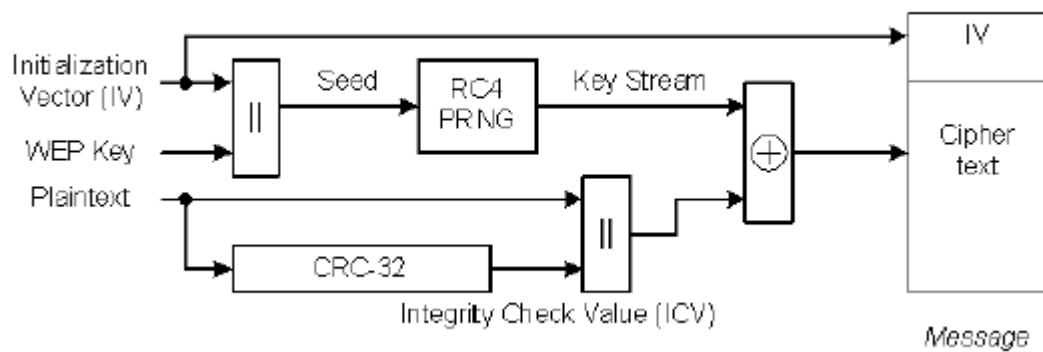


图6 WEP 加密算法

这个图不难明白，我们可以把它抽象出来就是一句话：

WEP加密，就是STA和AC需要预先配置相同的静态Key，Key的长度为40 bit或104bit，每次对数据加密的Key=静态Key+24bit的IV值（IV值为动态生成）；

静态WEP有缺陷，动态WEP可弥补；但是我们不支持动态WEP；

2.2.2 数据结构

下面的这张图表给出了使用WEP加密的数据结构：

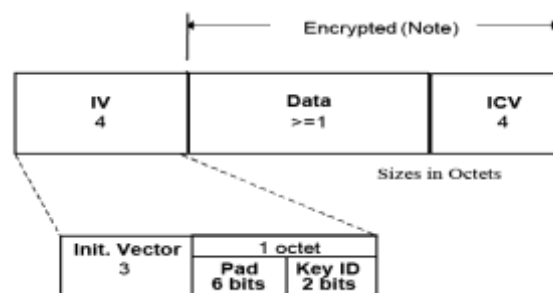
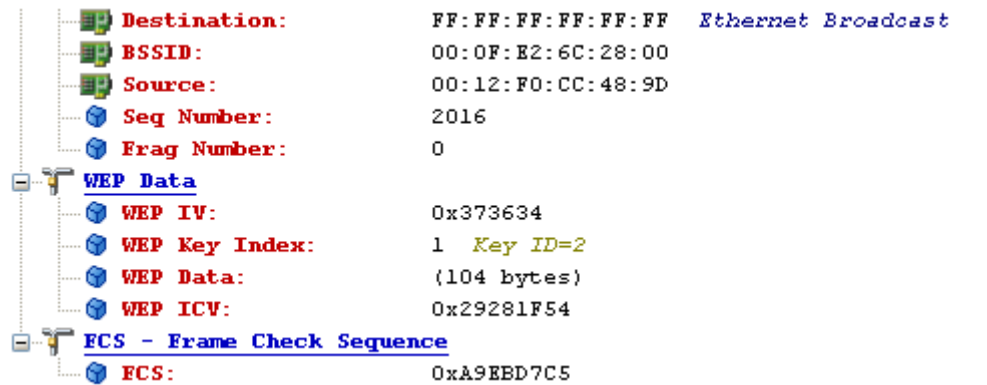


图7 WEP 加密数据结构

几点需要说明的:

- (1) IV是明文;
- (2) Pad字段不使用, 全0; Key字段标明使用的是哪个key;
- (3) 加密的部分是Data与ICV;

看一个具体的数据包:



Destination:	FF:FF:FF:FF:FF:FF	Ethernet Broadcast
BSSID:	00:0F:E2:6C:28:00	
Source:	00:12:F0:CC:48:9D	
Seq Number:	2016	
Frag Number:	0	
WEP Data		
WEP IV:	0x373634	
WEP Key Index:	1	Key ID=2
WEP Data:	{104 bytes}	
WEP ICV:	0x29281F54	
FCS - Frame Check Sequence		
FCS:	0xA9EBD7C5	

图8 WEP 数据包

WEP加密IV起到很重要的作用:

如果没有IV或者固定IV, 结果是:

The input plaintext always produces the same ciphertext.

所以要对IV有要求, 才能实现

The input plaintext doest not alway produces the same ciphertext

IV是3bytes, 24bits, 总会出现重复; 解决办法: 更高级的加密方法, 802.11i的内容

WEP解密:

- (1) WEP解密得出明文和ICV
- (2) 重新计算ICV
- (3) 比较是否一致

解密的初始seed得出是密钥与IV, 需要明文IV。

2.2.3 WEP 加密（认证）图解

WEP加密（认证）的图解：

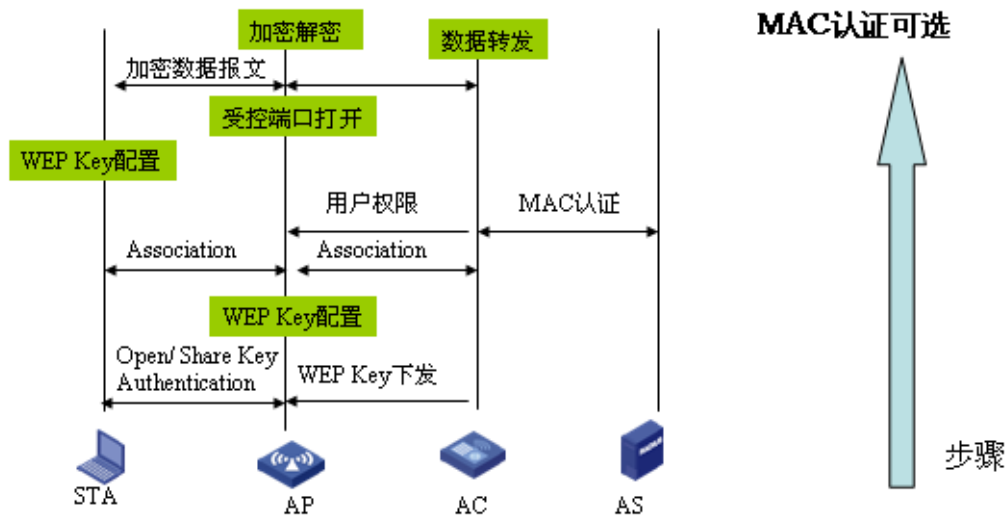


图9 WEP 加密图解

上图是从下向上进行，可以分为以下过程：

- (1) 链路认证过程，wep key的下发。
- (2) 获得用户权限。
- (3) 数据报文交互。

2.3 流加密和块加密

块加密和流加密统称为Electronic Code Book (ECB)方式，其特征是相同的明文将产生相同的加密结果。

802.11协议的加密机制采用RC4进行加密（WEP），RC4是流(stream)加密，简单的说就是通过将Key stream和明文流XOR得到密文；而块(block)加密是将明文分割为多个block，再和Key stream XOR，后面要和说的AES算法即是块加密，它在802.11i中被详细讲述。

如下所示，两种加密方法的图解。

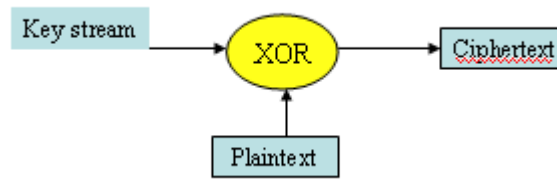


图10 流加密

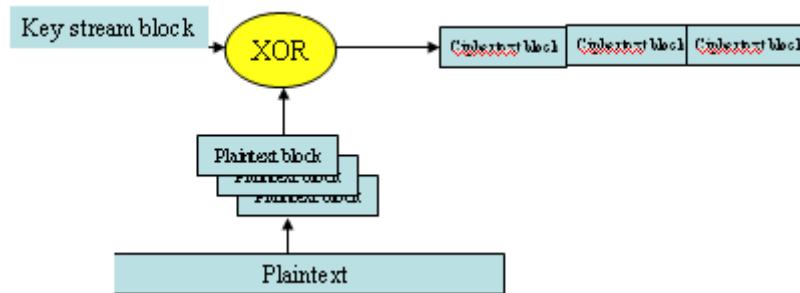


图11 块加密

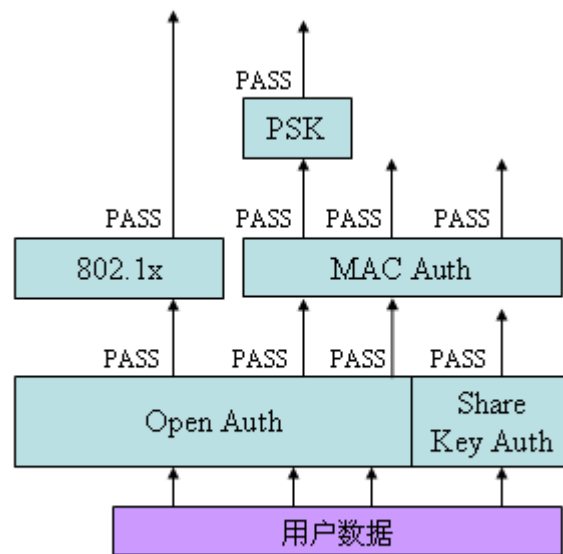
流加密：从密钥生成一个密钥串流。流加密使用不同长度的密钥串流来加密一个数据帧，密钥流的长度决定于需要加密的报文的长度。例如：报文有15个bit，就产生15个bit的加密密钥，报文有2000个bit，就产生2000个bit的加密密钥流。这些密钥流都从初始的64位来生成。

块加密：采用长度固定的密钥串流。该加密方法是把明文分成固定长度的数据块，每个数据块分别和密钥串流进行加密。分块的时候如果明文的数据块小于密钥串流，那么可以在明文数据块后面填充空白数据（就是填充一段0）。

3 802.11i-认证

3.1 802.11i 的认证体系

如下图，给出了802.11i协议中的认证层次和关系。



认证层次和关系

图12 802.11i 认证体系

802.11i中关于认证的描述：

同一SSID下同时支持不同的方式

- 802.11i Pre-RSNA方式
- 802.11i RSNA方式
- WAPI方式
- 基于MAC认证

802.11i

- 支持802.1x的Authenticator功能
- 支持802.1x中的TLS、PEAP、TTLS
- 支持Key的4次握手协商
- 支持Pre-share key认证
- 支持用户加密Key的生成、管理和下发
- WEP、TKIP、AES硬件加密

为了弥补802.11中只有链路层的认证的局限性，在802.11i中引入了用户接入认证方法，包括PSK认证，802.1x认证和mac认证。

(1) PSK认证

PSK认证需要实现在无线客户端和设备端配置相同的预共享密钥，如果密钥相同，PSK接入认证成功；如果密钥不同，PSK接入认证失败。

(2) 802.1x认证

802.1x协议是一种基于端口的网络接入控制协议（port based network access control protocol）。“基于端口的网络接入控制”是指在WLAN接入设备的端口这一级对所接入的用户设备进行认证和控制。连接在端口上的用户设备如果能通过认证，就可以访问WLAN中的资源；如果不能通过认证，则无法访问WLAN中的资源。

(3) MAC接入认证

MAC地址认证是一种基于端口和MAC地址对用户的网络访问权限进行控制的认证方法，它不需要用户安装任何客户端软件。设备在首次检测到用户的MAC地址以后，即启动对该用户的认证操作。

3.2 PSK 认证

PSK在802.11i中的解释：

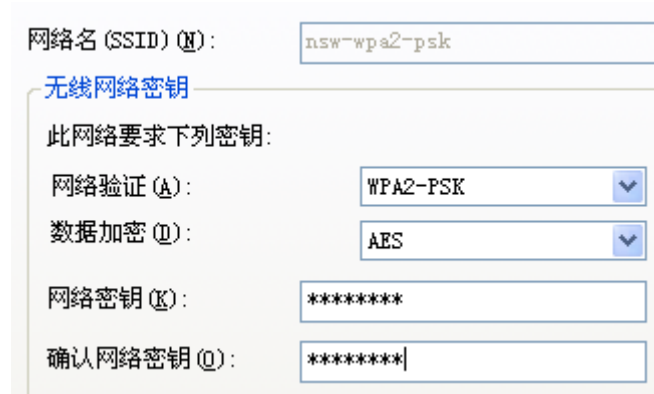
preshared key (PSK): A static key that is distributed to the units in the system by a method outside the scope of this amendment, always by some out-of-band means.

不难理解，PSK是一个静态的key，也就是说我们事先配置好的；

3.2.1 配置

下面是PSK配置的接口和模板：

STA上的配置：



网络名 (SSID) (N): nsw-wpa2-psk

无线网络密钥

此网络要求下列密钥：

网络验证 (A): WPA2-PSK

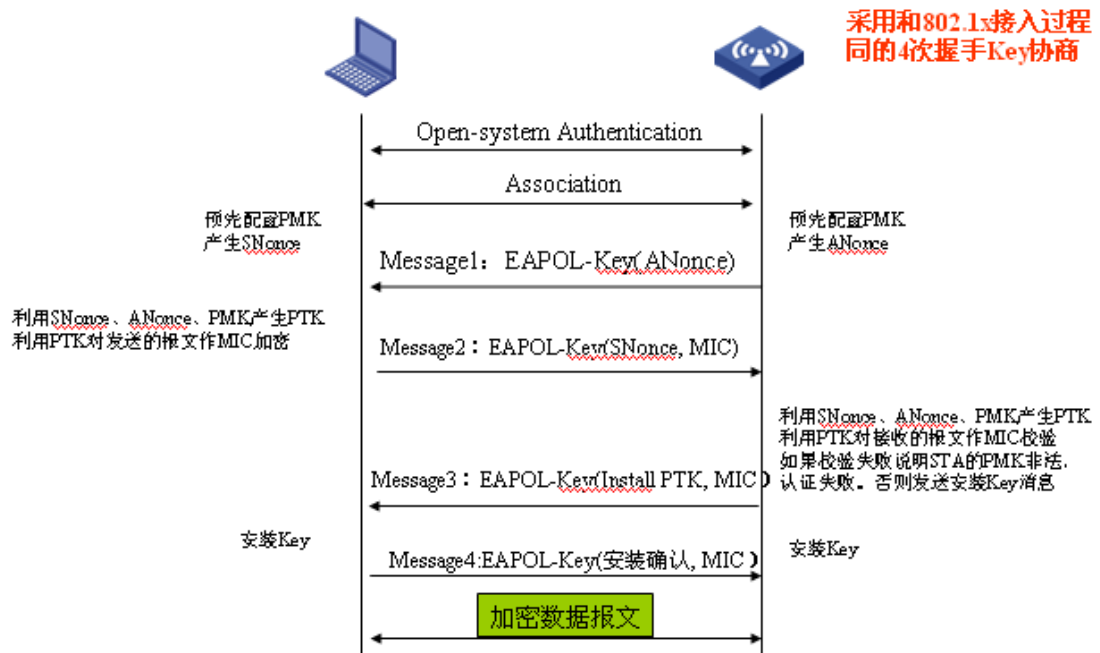
数据加密 (Q): AES

网络密钥 (K): *****

确认网络密钥 (Q): *****

图13 PSK 认证客户端配置

3.2.2 认证过程



认证过程;

- (1) AUTHENTICATION 和 ASSOCIATION是open-system认证和关联，属于链路层认证。
- (2) EAPOL-KEY的协商，即4 way handshake；这个在后面有专门的讲述；关键一点的理解是，预先配置的密码作什么用的？[产生在4way协商中需要的PMK](#)；

3.2.3 PSK 认证图解

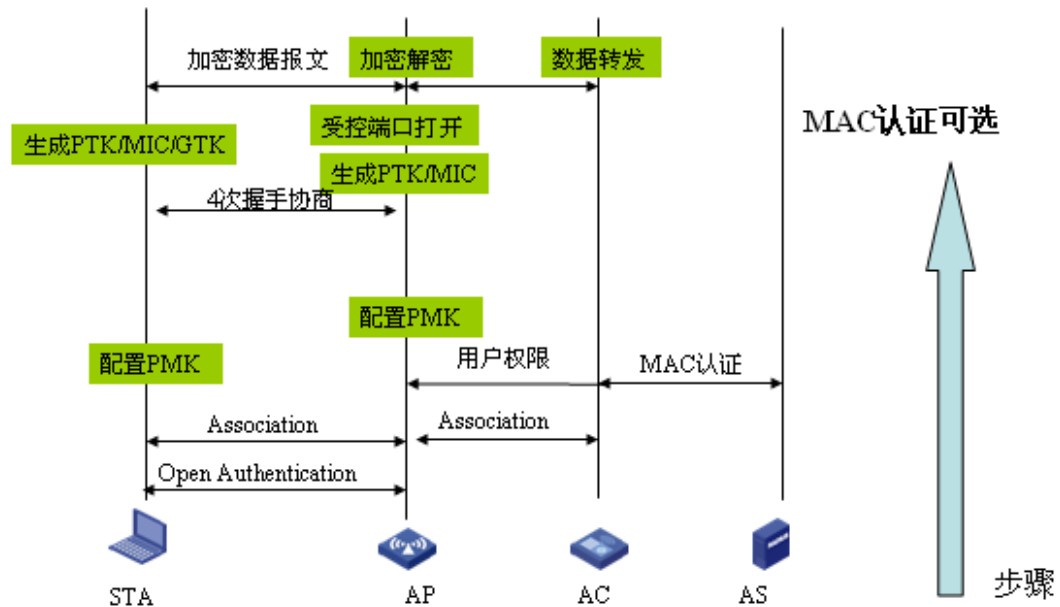


图15 PSK 认证图解

上图是从下向上进行，可以分为以下过程：

- (1) 链路认证过程
- (2) 静态配置密钥，产生PMK
- (3) 密钥协商，产生机密密钥
- (4) 数据报文交互

3.3 802.1x 认证

在无线中，802.1x认证和MAC认证被集成在端口安全模块，用于无线口的接入认证；当然，802.1x和MAC认证也存在与独立的模块，用于以太口的认证，但是在无线控制器上，这个认证没有什么实际意义，所以下面我们要说的都是端口安全下的对于无线接口的认证方法；

3.3.1 802.1x 的认证机制

其工作机制如下：

IEEE 802.1x认证系统利用EAP（Extensible Authentication Protocol，可扩展认证协议），作为在客户端、设备端和认证服务器之间交换认证信息的手段。

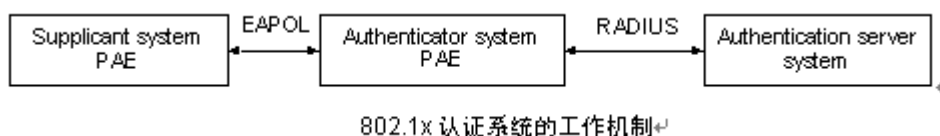


图16 802.1x 认证系统工作机制

（1）在客户端PAE与设备端PAE之间，EAP协议报文使用EAPOL封装格式，直接承载于LAN环境中。

（2）在设备端PAE与RADIUS（Remote Authentication Dial-In User Service，远程认证拨号用户服务）服务器之间，EAP协议报文可以使用EAPOR封装格式（EAP over RADIUS），承载于RADIUS协议中；也可以由设备端PAE进行终结，在设备端PAE与RADIUS服务器之间传送包含PAP（Password Authentication Protocol，密码验证协议）或CHAP（Challenge Handshake Authentication Protocol，质询握手验证协议）属性的报文。

（3）认证服务器通常为RADIUS服务器，该服务器可以存储有关用户的信息。例如，用户名、密码以及用户所属的VLAN、CAR参数、优先级、用户的访问控制列表等。

（4）当用户通过认证后，认证服务器会把用户的相关信息传递给设备端，设备端PAE根据RADIUS服务器的指示（Accept或Reject）决定受控端口的授权/非授权状态。

3.3.2 802.1x 的体系结构

使用802.1x的系统为典型的Client/Server体系结构，包括三个实体，分别为Supplicant system（客户端-STA）、Authenticator system（设备端-AC）以及Authentication server system（认证服务器-RADIUS）。

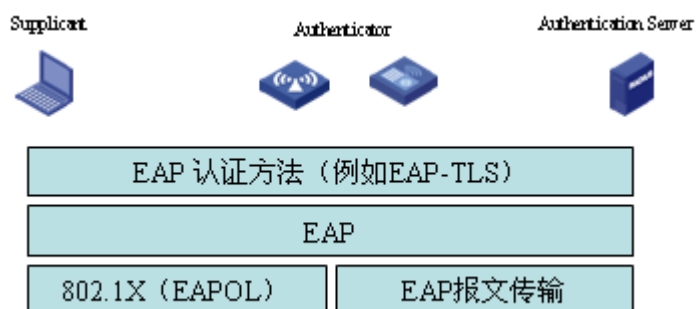


图17 802.1x 体系结构

EAP协议框架：作为认证协议框架，可以承载多种认证方法，而不限具体的认证方法，具有很好的扩展性。

EAP认证方法：定义了实际的认证过程和方法，代表性认证方法包括了EAP-TLS、EAP-PEAP等。

802.1x报文：显然，EAP报文(EAP认证方法)在特定的链路层协议传递时，需要一定的报文封装格式。这就是EAPOL（EAP over link）报文的作用。

EAP报文传输：EAPOL报文主要在supplicant和authenticator之间传送。由于认证是通过authentication server完成的，所以在认证过程中，authenticator将把EAPOL报文中的认证报文封装到Radius报文中，通过Radius报文和authentication server进行交互。

3.3.3 EAP

802.1x利用了EAP（Extensible Authentication Protocol ）协议来实现了对多种认证方法（如EAP-TLS）的支持。注：EAP是用来传送认证消息的协议，本身并不是认证方法。

认证过程以EAPOL报文开始

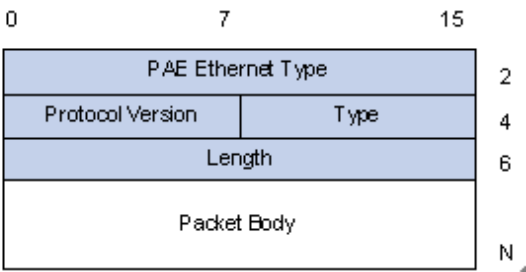


图18 EAPOL 报文结构

EAPOL是802.1x协议定义的一种链路层报文封装格式，主要用于在客户端和设备端之间传送EAP协议报文。

PAE Ethernet Type：表示协议类型，为0x888E。

Protocol Version：表示EAPOL帧的发送方所支持的协议版本号。

Type：

EAP-Packet（值为0x00），认证信息帧，用于承载认证信息；

EAPOL-Start（值为0x01），认证发起帧；

EAPOL-Logoff（值为0x02），退出请求帧；

EAPOL-Key（值为0x03），密钥信息帧；

Length: 表示数据长度，也就是“Packet Body”字段的长度，单位为字节。如果为0，则表示没有后面的数据域。

Packet Body: 根据不同的Type有不同的格式。

其中，EAPOL-Start，EAPOL-Logoff和EAPOL-Key仅在客户端和设备端之间存在；在设备端和认证服务器之间，EAP-Packet报文重新封装承载于RADIUS协议上，以便穿越复杂的网络到达认证服务器。

EAP的报文结构如下，EAP最初规格RFC2284,最新在RFC3748中被描述；

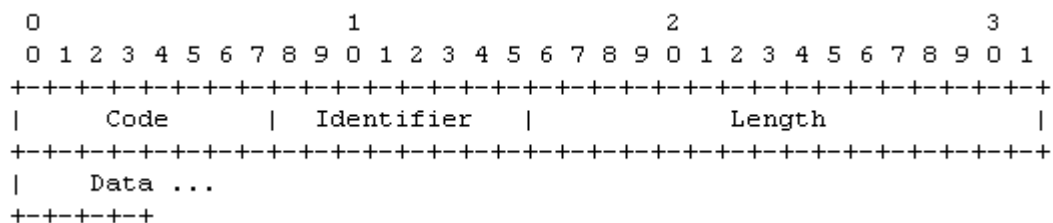


图19 EAP 报文结构

Code用来标识EAP报文的类型，有4种类型被定义：

1 Request 2 Response 3Success 4Failure

Identifier字段是一个无符号的整数，用来标识请求与响应；根据这个字段可以判断消息是否被重传；重传消息具有相同的identifier值，信的消息具有不同的identifier值；

Length字段标识了所有字段的总长度，并且包括自己所占用的两个字节；

Data字段要注意一点，它实际上是包含类型（type）的，用来标识是请求和响应的类型；

所以这个字段应该被分为两部分：type和type-data；

EAP并没有定义具体的认证方法，而是通过EAP-Request 和Response承载其他的认证方法的报文；EAP支持的认证协议包括：EAP-TLS,PEAP,MD5-CHALLENGE,TTLS等。

我们常用的是EAP-TLS和PEAP

EAP-TLS: EAP-Transport Layer Security（简称EAP-TLS）

➤ 协议原本就是设计来用在易遭窥视的链路层上。TLS的前身是保护网际网络交易安

全的协议Secure Socket Layer（简称SSL）。

- 从许多方面来看，无线局域网的使用案例（use case）与网际网络类似。数据必须在完全不可信赖且攻击者立的网络环境中进行传送。
- TLS 的目的，就是在不可信赖的网络环境中建立一条可信赖的沟通管道。
- TLS 通过凭证交换来进行相互认证。使用者必须将数位凭证送交认证服务器以进行验证，但是认证服务器也必须提供本身的凭证。通过可信赖的凭证发行机构验证服务器的凭证真伪，用户端就可以确定所连接的网络经过凭证发行机构授权无误。

EAP-PEAP: “Protected” EAP, PEAP 的运作方式是协议首先使用类似EAP-TLS 的方式建立起一个TLS 管道。进行下一个步骤之前，会先使用认证服务器的数位凭证来验证此网络是否可受信赖。

- 第二个步骤是使用TLS 管道为旧式的身份认证协议加密，然后以之验证使用者身份。
- 有时候第一个步骤也称为「外层」（outer）身份认证，因为它是用来保护第二个或者「内层」（inner）身份认证的管道。
- 认证还是免不了，但只有外层身份认证需要用到。
- PEAP 则是在管道内进行第二次EAP 交换程序。

EAP-TLS: 要求客户端和服务端证书

EAP-PEAP: 使用基于证书的服务器和客户端密码

从以上分析我们可以用最简单的话总结出对这两种方式的最简单的描述：**PEAP**是用密码来认证，而**TLS**是用证书来认证

3.3.4 RADIUS

前面的认证接入过程，用到了RADIUS消息；标注RFC2865

RADIUS:

远程认证拨入用户服务（Remote Authentication Dial-In User Service），RADIUS是用于网络访问服务器（NAS）和AAA服务器间通信的一种协议，实现对用户的认证，计费 and 授权。

RADIUS协议应用系统主要是由3个部分组成：用户，Radius客户端（NAS），Radius服务

器；Radius客户端和服务端之间基于UDP协议（端口号1812）通讯。

RADIUS报文中字段有两种编码方式，前4个字段的格式是固定的V格式，字段中只包含属性的值；报文中所带的众多属性（Attributes）使用的是TLV的编码格式

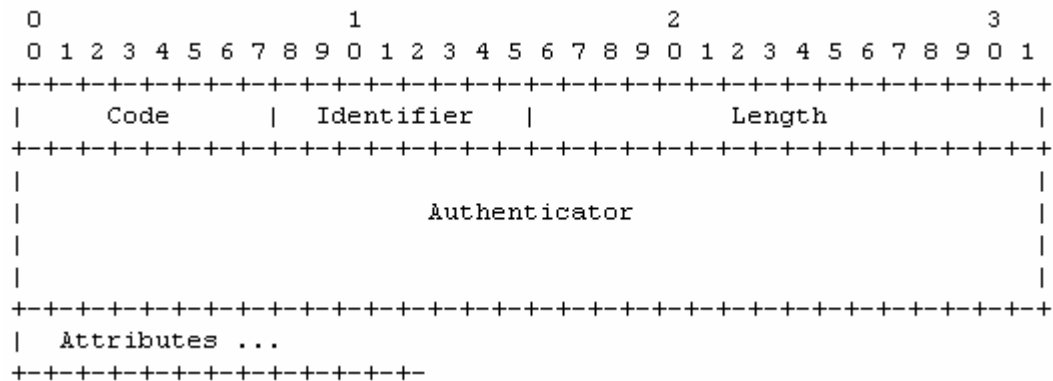


图20 RADIUS 报文结构

认证过程：

AC和服务端需要交互的4个主要的RADIUS消息，分别是认证请求/认证接受/计费请求/计费接受；

CODE：标识消息的作用，1/2/4/5分别对应下面的四种消息；

162.105.34.2	162.105.34.3	RADIUS Access-Request(1) (id=94, l=266)
162.105.34.3	162.105.34.2	RADIUS Access-Accept(2) (id=94, l=224)
162.105.34.2	162.105.34.3	RADIUS Accounting-Request(4) (id=130, l=232)
162.105.34.3	162.105.34.2	RADIUS Accounting-Response(5) (id=130, l=32)
Hangzhou_32:41:06	IntelCor_cc:f5:30	EAP Success

图21 RADIUS 典型消息

CODE=4的计费请求消息中的属性40：1计费开始，2计费结束；

```

Attribute value Pairs
+ AVP: l=6 t=Tunnel-Type(64) Tag=0x00: VLAN(13)
+ AVP: l=6 t=Tunnel-Medium-Type(65) Tag=0x00: IEEE-802(6)
+ AVP: l=6 t=Tunnel-Private-Group-Id(81) Tag=0x00: \000\000\202
+ AVP: l=10 t=User-Name(1): nsw@cams
+ AVP: l=8 t=NAS-Identifier(32): H3C-AC
+ AVP: l=6 t=NAS-Port(5): 33235074
+ AVP: l=39 t=NAS-Port-Id(87): slot=1;subslot=15;port=178;vlanid=130
+ AVP: l=6 t=NAS-Port-Type(61): wireless-802.11(19)
+ AVP: l=16 t=Calling-Station-Id(31): 0012-f0cc-f530
+ AVP: l=6 t=Acct-Status-Type(40): Start(1)

```

图22 RADIUS 计费报文

RADIUS为支持EAP认证增加了两个属性：EAP-Message（EAP消息）和 Message-Authenticator（消息认证码）。Radius使用Access-request 来承载EAP message (from NAS to AS);使用Access-Challenge 来承载EAP message (from AS to NAS);

EAP-Message：用来封装EAP数据包，类型代码为79，String域最长253字节，如果EAP数据包长度大于253字节，可以对其进行分片，依次封装在多个EAP-Message属性中

Message-Authenticator：用于在使用EAP、CHAP等认证方法的过程中，避免接入请求包被窃听。在含有EAP-Message属性的数据包中，必须同时也包含Message-Authenticator，否则该数据包会被认为无效而被丢弃。

9	6.775008	162.105.34.2	162.105.34.3	RADIUS Access-Request(1) (Id=88, L=340)
10	6.778106	162.105.34.3	162.105.34.2	RADIUS Access-challenge(11) (Id=88, L=1100)
11	6.780829	Hangzhou 32:41:06 IntelCor	cc:f5:30	EAP Request. PEAP [Pakekar]
Frame 9 (382 bytes on wire (382 bytes captured))				
Ethernet II, Src: Hangzhou_59:47:7c (00:0f:e2:59:47:7c), Dst: D-Link_80:4e:69 (00:0f:3d:80:4e:69)				
Internet Protocol, Src: 162.105.34.2 (162.105.34.2), Dst: 162.105.34.3 (162.105.34.3)				
User Datagram Protocol, Src Port: 1024 (1024), Dst Port: radius (1812)				
Radius Protocol				
Code: Access-Request (1)				
Packet identifier: 0x58 (88)				
Length: 340				
Authenticator: 000061CB0000708500000D2A00007C46				
Attribute Value Pairs				
AVP: l=10 t=User-Name(1): nsw@cams				
AVP: l=6 t=Framed-MTU(12): 1450				
AVP: l=114 t=EAP-Message(79) Last Segment [1]				
AVP: l=18 t=Message-Authenticator(80): EB8CE276ED1235BB0C77AD268CF0149A				

图23 RADIUS 中的 EAP 属性

3.3.5 802.1x 认证过程

802.1x认证的过程简单的说就是一个验证无线接口是否可用并且协商数据加密key的过程；

下面看一下802.1X认证的过程

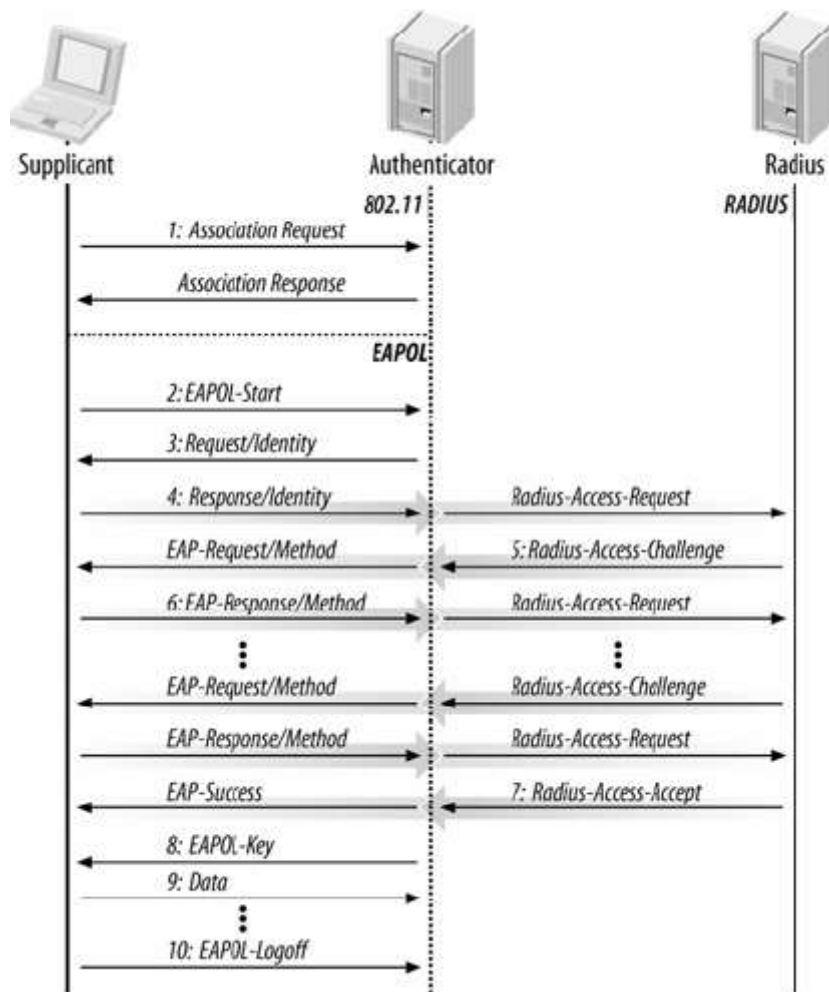


图24 802.1x 认证过程

以EAP-PEAP的方式为例：

（1）**认证开始**：认证过程可以由用户主动发起，也可以由认证系统发起；消息调试中看第一个消息的方向是怎么样的，是否是EAPOL-START消息；

（2）**步骤2和步骤3是请求用户名**：这个用户名就是我们在认证开始的时候，在STA（windows）弹出来的对话框中输入的用户名和密码中的用户名部分；

（3）**步骤4到步骤7是认证接入过程**：Authenticator收到Supplicant的用户名之后进行RADIUS封包发送到radius服务器；服务器收到之后查找此用户名对应的密码，先加密，然后使用challenge消息讲加密所使用的key送过来；认证申请者使用这个key讲密码加密发送；服务器收到密码之后与自己找到的加密的密码比较，确定用户身份；

到这里认证已经结束，会收到成功或者失败消息；

注意：我们认为计费是认证的必需的一部分，所以在认证通过之后还要进行计费的协商，

如果不能通过，整个认证都是失败的；

（4）步骤8是11key的协商；

11Key的密钥协商为802.1x的EAPOL-Key协商的一种，11Key完成载荷组装后，通过802.1x完成EAPOL-Key报文头、EAPOL 报文头以及802.3报文头后发送；

11key的协商到底作了些什么？简单的说就是安装用于数据加密的PTK；而这个过程又是很复杂很繁琐的过程；在后面有专门的章节讲述它的协商过程；这里先简单的看下：

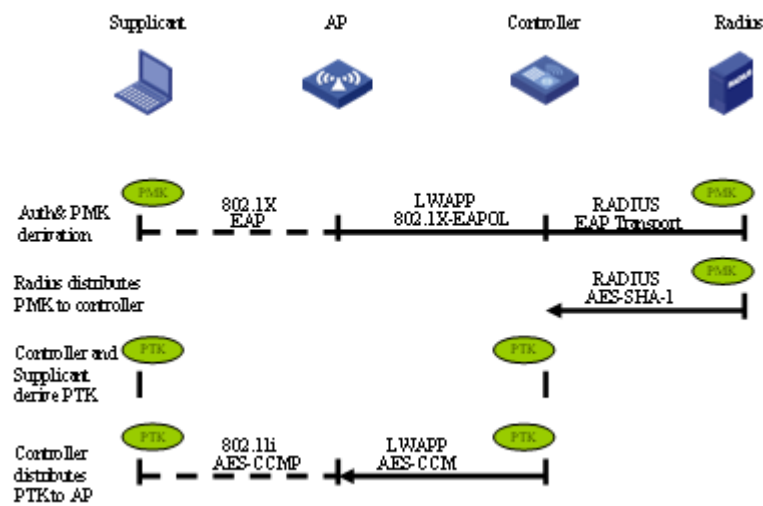


图25 密钥生成过程

从上面的图中可以看出大致可以分几个步骤：

- 通过一些方法预先准备好PMK；这些方法在后面介绍；
- 在与服务器交换消息后AC和认证请求者分别生成PTK
- AC下发PTK给AP；

上面中有一些专有名词，在后面会统一解释一下；

11key的协商有4次握手或者6次握手；

下面是4次和6次握手报文的截图；

450	48.699893	Hangzhou_32:41:06	IntelCor_cc:f5:30	EAP	Success
451	48.700197	Hangzhou_32:41:06	IntelCor_cc:f5:30	EAPOL	Key
452	48.736203	IntelCor_cc:f5:30	Hangzhou_32:41:06	EAPOL	Key
453	48.742813	Hangzhou_32:41:06	IntelCor_cc:f5:30	EAPOL	Key
454	48.752146	IntelCor_cc:f5:30	Hangzhou_32:41:06	EAPOL	Key
463	48.762346	Hangzhou_32:41:06	IntelCor_cc:f5:30	EAP	Success

80	4.606853	Hangzhou_32:41:02	IntelCor_cc:f5:30	EAP	Success
81	4.607124	Hangzhou_32:41:02	IntelCor_cc:f5:30	EAPOL	Key
82	4.625481	IntelCor_cc:f5:30	Hangzhou_32:41:02	EAPOL	Key
83	4.629644	Hangzhou_32:41:02	IntelCor_cc:f5:30	EAPOL	Key
84	4.634499	IntelCor_cc:f5:30	Hangzhou_32:41:02	EAPOL	Key
87	4.638860	Hangzhou_32:41:02	IntelCor_cc:f5:30	EAPOL	Key
88	4.651122	IntelCor_cc:f5:30	Hangzhou_32:41:02	EAPOL	Key
89	4.651885	Hangzhou_32:41:02	IntelCor_cc:f5:30	EAP	Success

图26 4次握手和6次握手

如果是6次握手，与4次握手比较是将GTK的协商单独进行；

6次握手中单独进行GTK协商的报文：

```

Key Information: 0x03d1
.... .... .001 = Key Descriptor Version: HMAC-MD5 for M
.... .... 0... = Key Type: Group key
.... .... ..01 .... = Key Index: 1
.... .... .1.. .... = Install flag: Set
.... .... 1... .... = Key Ack flag: Set
.... ...1 .... .... = Key MIC flag: Set
.... ..1. .... .... = Secure flag: Set
.... .0.. .... .... = Error flag: Not set
.... 0... .... .... = Request flag: Not set
.... 0 .... .... .... = Encrypted Key Data flag: Not set
Key Length: 32
Replay Counter: 3
Nonce: 184D9426A695A3456E1B821ADE253C491F2BC3D6A3C2909E...

```

图27 6次握手中的 GTK 协商报文

(5) 11key协商成功的话，就可以传送数据了；如果11key的协商失败，会切断前面已经通过认证的连接；

3.3.6 802.1x 认证图解

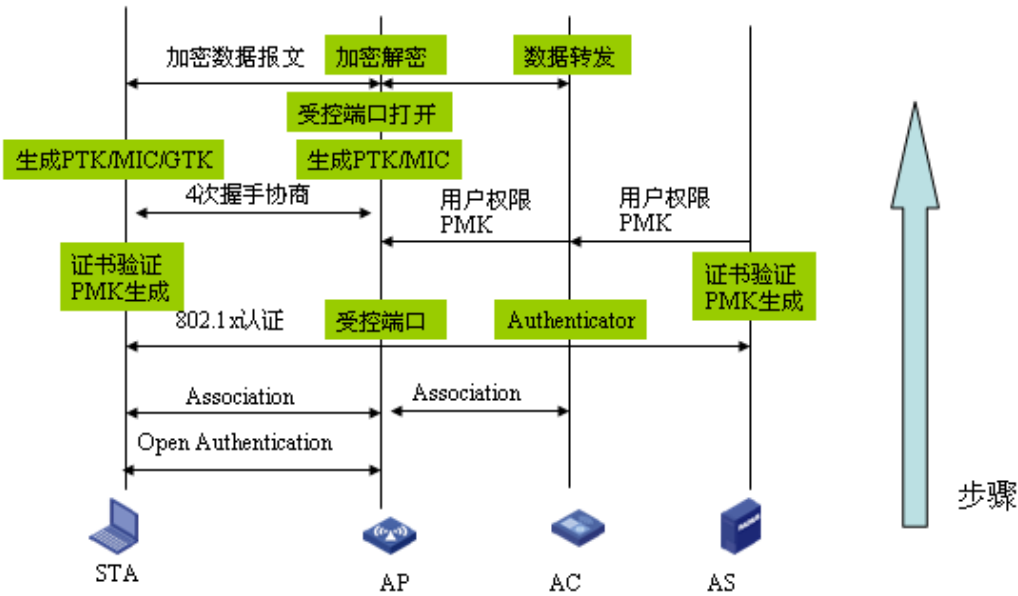


图28 802.1x 认证图解

上图是从下向上进行，可以分为以下过程：

- (1) 链路认证过程
- (2) 802.1x认证过程，产生PMK
- (3) 密钥协商，产生机密密钥
- (4) 数据报文交互

3.3.7 配置

#

3.3.8 认证终端

在终端进行认证的时候，如果配置了要求输入用户名和密码的时候，也有一点需要注意：

用户名 (U):	<input type="text"/>
密码 (P):	<input type="password"/>
登录域 (L):	<input type="text"/>

图29 802.1x 认证客户端配置

上面的是WINDOWS弹出的对话框，要求输入用户名和密码；如果我们使用CAMS作为远端服务器，“登录域”这里是不能填写的，因为CAMS的域配置是XX@domain的形式，所以只能在“用户名”里面填写XX@domain的形式；如果在“登录域”里面填写了登录的域，WINDOWS终端就会自动加在登录名字的前面，是domain\XX的形式，这个是给如果用WINDOWS2000\WINDOWS2003-SERVER作为远端服务器时候使用的；

3.3.9 DOMAIN-ISP

在认证的流程中，DOMAIN和RADIUS SCHEME两个模块参与其中，并且占有重要的地位；

认证用户的种类：

认证的用户可以分为两类：lan接入用户和login用户（不包含命令行用户）；在无线中lan接入用户是既支持认证、授权，也支持计费的；而login用户是只认证、授权，不计费的；

认证功能：

AAA提供了3种认证（授权、计费）方法：

不认证、本地认证、远端服务器认证；关于local方式，目前的实现是和none方式没有本质区别的，如果采用local方式，系统会查看该用户是否为active的，是否超过了连接数的限制，没有真正的认证过程；对于计费，也没有真正的计费过程，没有计费报文的发送；

如果选用了local方式，那么就要在设备上（AC）配置本地用户；

这里面一个很重要的配置是：IDLE-CUT，用户闲置切断；判断的依据是：在线客户端连接上来的接口数据统计单位时间内（秒）的数据流量小于100k/s；

3.3.10 RADIUS SCHEME

这里面配置一些设备和服务器交互的信息设置，认证/授权和计费的服务器是可以分离的；另外，我们的设备如果认证通过，计费失败也认为是失败；解决的方法是我们可以配置它不计费或者本地计费；

4 802.11i-加密

4.1 中间安全机制-WPA

WPA: Wi-Fi Protected Access. 无线网络最初采用的安全机制是**WEP**(有线等效私密),但是后来发现**WEP**是很不安全的

802.11组织着手制定新的安全标准,就是后来的802.11i协议;但是,标准的制定到最后的发布需要较长的时间,消费者不会因为为了网络的安全性而放弃原来的无线设备,所以Wi-Fi联盟在标准推出之前,在802.11i草案的基础上,制定了一种称为**WPA**的安全机制,它使用**TKIP**(临时密钥完整性协议),使用的加密算法还是**WEP**中使用的加密算法**RC4**,所以不需要修改原来无线设备的硬件, **WPA**针对**WEP**中存在的问题,通过软件升级的方法提高网络的安全性。

802.11i颁布之后, Wi-Fi联盟推出了**WPA2**,它支持**AES**(高级加密算法),因此它需要新的硬件支持,它使用**CCMP**(计数器模式密码块链消息完整码协议)。

所以说: **WPA**给用户在**WEP**和**WPA2**之间暂时提供一个完整的认证机制。

4.2 TKIP

TKIP: Temporal Key Integrity Protocol

TKIP是一种加密方法,用于增强pre-RSN硬件上的**WEP**协议的加密的安全性,其加密的安全性远远高于**WEP**; **WEP**主要的缺点在于,尽管**IV** (Initial Vector, 初始向量)改变但在所有的帧中使用相同的密钥,而且缺少密钥管理系统,不可靠; **TKIP**和**WEP**加密机制都是使用**RC4**算法,但是相比**WEP**加密机制, **TKIP**加密机制可以为**WLAN**服务提供更加安全的保护, **TKIP**对于**WEP**的改进如下:

- **TKIP**通过增长了算法的**IV**长度提高了**WEP**加密的安全性;将**IV size** 从 24 bits增加到 48 bits,减少了**IV**重用。
 - **TKIP**支持**MIC**和Countermeasure功能: 使用Michael来实现**MIC** (Message Integrity Code, 信息完整性校验)。结合**MIC**, **TKIP**采用了countermeasures 方式: 一旦发现了攻击 (**MIC Failure**), 就中止该用户接入。
 - **TKIP**支持密钥的动态协商, 解决了**WEP**加密需要静态配置密钥的限制: 使用了Per-Packet Key Mixing 方式来增加key的安全性。
-

TKIP的密钥算法是经过两次密钥mixing，如下图所示：

第一次，得到中介key（Intermediate key），称为TTAK（TKIP-mixed transmit address and key）。

TTAK = Phase1 (TK, TA, TSC)

第二次，得到WEP seed或者per-frame key。

WEP seed = Phase2 (TTAK, TK, TSC)

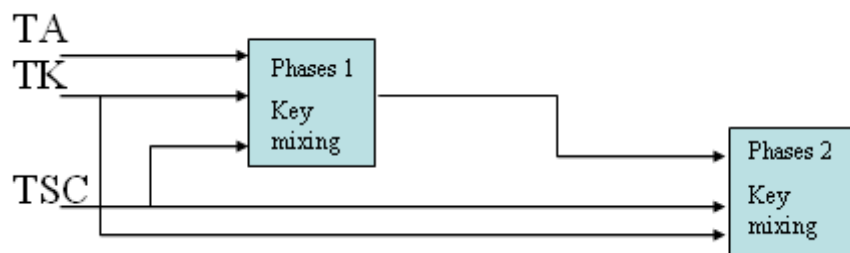


图30 TKIP 密钥生成机制

下面对上面所说的两次配钥详细说明：

Tkip的两次配钥：配钥的目的是使各个帧之间所使用的密钥彼此间存在差异。

第一次配钥：使用的输入量是传送端地址，序号的前32位以及128位的临时密钥，输出一个80位元的值。由于使用的都是一些简单的运算，所以整个计算量是很小的，这个为了符合硬件的需要；只要序号的前32位不变，第一次配钥的结果就是一个常量。因此只要65535个帧计算一次就可以了。

第二此配钥：这次的目的是针对每个帧。使用的输入量是第一次的结果，临时密钥和TSC的后16位，输出量是128位元的RC4密钥，作为wep的随即种子使用。这些量中，TSC是变化的，保证每个帧使用的密钥都不相同，16位TSC还用来产生WEP IV的一部分。变化的TSC的变化方式是经过严格定义的，TSC被传送到接受端以便解密。

4.2.1 TKIP MPDU formats

数据结构如下，有几点需要说明：

- （1）在数据域后面追加8个字节的MIC域，当作数据域的一部分。
- （2）被加密的是Data+MIC+ICV。

(3) IV/KeyID保留自WEP，但是具有不同的含义：前3个位元记载了部分的TKIP序号，以及目前使用的密钥编号。

(4) Extended IV：记载其余的序号。

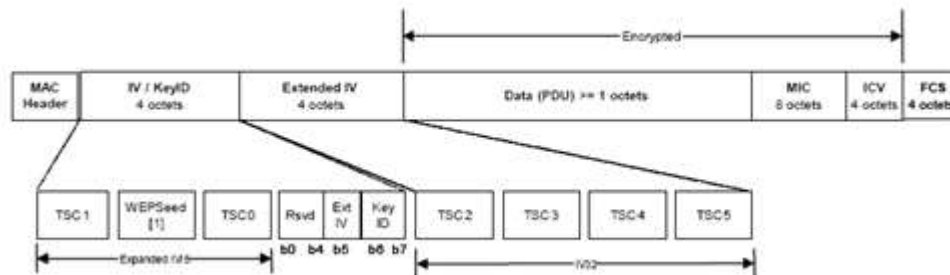


图31 TKIP MPDU

4.2.2 TKIP 加密与接收

加密过程：

- (1) 待传送的11数据帧。
- (2) 计算MIC；和明文一起是被加密的。
- (3) 为每个数据帧采用WEP密钥加密；通过上面的二次配钥，TKIP为每个帧产生WEP密钥。
- (4) 帧 +MIC+步骤三所得到的RC4 WEP密钥，一起交给WEP，由WEP 进行帧分封过程。

TKIP接收到被加密的数据帧，作如下动作：

- (1) 检查序号TSC，防止重放攻击。
- (2) 得到WEP加密的随机种子，使用传送端地址、临时密钥以及序号、接收端地址。
- (3) 解包，并且同时进行ICV校验。
- (4) 计算MIC，与数据报携带过来的MIC作比较。

4.2.3 TKIP MIC 和 countermeasures procedures

MIC: temporal message integrity code

- (1) 临时密钥的一部分被用作MIC key，来确定MAC层MSDUs或MPDUs的完整性。
- (2) Tkip MIC提供的保护功能是非常弱的，但是它是在大多数原有硬件上的最好的实

现。并且在不同的数据传输方向上使用不同的MIC key，这个在后面的密钥分发会讲到。

(3) MIC的目的：使对于数据的修改的主动攻击实施起来更加困难。在发送报文时MIC应用于MSDU，接受者也在MSDU阶段验证MIC，如果MIC检查失败，MSDU将被丢弃并且触发反制措施。

(4) MIC不能单独提供完整的伪造保护，所以它不能防御重放攻击。所以，TKIP提供了TSC sequencing 和ICV进行重放保护。

TKIP countermeasures procedures

TKIP的反制措施完成以下事情：

- (1) 记录MIC错误事件日志，供管理员处理。即反制动作实在MIC检验时进行的。
- (2) MIC错误率一定要低于60s两次。如果在60s内检测到两次MIC错误，将切断接入用户。
- (3) 更新密钥。

TSC：序号计数器，即初始向量

- (1) 每个MPDU都有一个唯一的TKIP TSC值。
- (2) TSC用来防止重放攻击。TKIP总是保留最近的序号，接受到帧之后比较。
- (3) TSC初始为1（安装了新的主钥），然后作为一个递增的计数器。

在验证MIC之前，the receiver 将检查每个MPDU的FCS,ICV和TSC，因为有些错误可能不是真正的MIC错误。任何一个携带不合法FCS，不正确的ICV或者TSC值小于或者等于TSC Replay counter的MPDU将在MIC检查之前被丢弃。这样就避免了不必要的MIC错误事件。

4.3 CCMP : CTR with CBC-MAC Protocol

CCMP (Counter mode with CBC-MAC Protocol, [计数器模式]搭配[区块密码锁链—信息真实性检查码]协议)，此加密机制是基于AES（Advanced Encryption Standard，高级加密标准）加密机制的CCM（Counter-Mode/CBC-MAC，区块密码锁链—信息真实性检查码）方法，仅用于RSNA客户端。

CCM结合CTR（Counter mode，计数器模式）进行机密性校验，同时结合CBC-MAC（区块密码锁链—信息真实性检查码）进行认证和完整性校验。CCM可以保护MPDU数据段和IEEE 802.11首部中被选字段的完整性。AES-CCM为块加密，802.11i要求AES为128 bit，每block 128 bits.对于块加密，需要将待加密的消息转化为block，这个过程称为mode of

operation。

CCM中每个会话都需要一个新的临时密钥。对于每个通过给定的临时密钥加密的帧来说，CCM同样需要确定唯一的随机值（nonce）。CCMP使用48位的PN（packet number）来实现这个目的。对于同一个临时密钥，重复使用PN会使所有的安全保证无效。

除了数据加密，AES-CCM还使用cipher block chaining (CBC)来产生MIC，以实现数据的Integrity。

4.3.1 CCMP MPDU formats

如下图所示，有几点需要说明：

- （1）加密的是Data和MIC
- （2）简单描述过程：Block—变化的counter—AES得到密钥—与Block异或

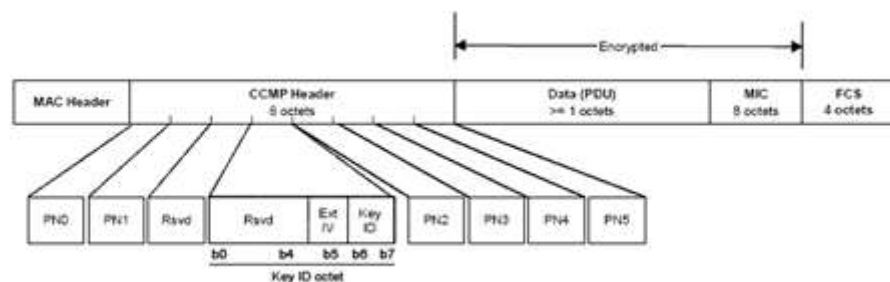


图32 CCMP MPDU

4.3.2 CCMP 的加密与接收

加密过程如下：

- （1）PN：48位，初始值随机，然后渐增1；使用相同的临时密钥加密的数据过程中不会重复。即每个MPDU加密使用的PN都是不同的，防止重放攻击。
- （2）AAD：the additional authentication data，额外认证数据。从MPDU的帧标头计算得到，用来保护协议版本，帧类型，顺序号等字段。AAD不能被加密，因为接收端会检验这些信息。
- （3）CCM Nonce：从PN，MPDU头部的地址2（即TA）组合而成，这样的目的是为了不同的客户端可以使用相同的PN。

- (6) 传送: MAC标头, CCMP标头+ (5) 得到待传送帧。

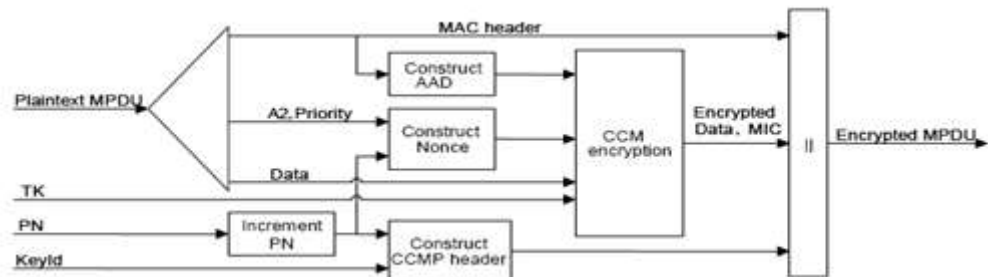


图33 CCMP 加密过程

CCMP数据的接收:

CCMP接收到被加密的数据帧，作如下动作：

- (1) 还原AAD中的信息，这个是没有经过加密的。
- (2) 得到 CCMP nonce，包含编号，地址等信息，也是从没有加密的标头中取得。
- (3) 通过临时密钥，nonce，AAD来解密数据包。
- (4) (2) 得到的包编号和保存的序号作比较，防止重放攻击。
- (5) 计算MIC，与数据报携带过来的MIC作比较。

5 802.11i-密钥协商与分发

5.1 802.11i 中的各种 key

在11key的协商中会有好多的key被用到,这里只是简单介绍一下;

5.1.1 PMK

PMK是很重要的；具体PMK是什么？先看802.11i的描述：

pairwise master key (PMK): The highest order key used within this amendment. The PMK may be derived from an Extensible Authentication Protocol (EAP) method or may be obtained directly from a preshared key (PSK).

翻译成中文叫做：成对主钥：

由以上描述我们可以得出**PMK的来源**有两个：an Extensible Authentication Protocol 或者 a preshared key (PSK)；这主要是由不同的认证方法得到的（具体的认证方法在下文讲述）；

如果是PSK的认证方式，由于不会到远端服务器上去认证，并且事先已经配置好了密码，所以得到比较简单；如果是到远端服务器上认证，则需要从消息的交互中得到，也就是说，它产生自远端服务器；

5.1.2 PMK 的作用

在我们实现的key cache方式的快速漫游中，PMK起到重要作用：

Key caching过程，图例在后面

1. STA第一次接入时（接入Old AP）采用正常的802.1x认证过程
2. 认证通过后STA把使用的PMK信息保存在Cache中
3. STA向New AP发起Reassociation时协商使用PMK Cache方式做认证
4. STA利用在Cache中保存的在Old AP中使用的PMK和New AP发起4次握手协商过程
5. 协商成功，STA开始传送数据报文

Key的保存：

- STA保存在cache中，以便于在AP切换时使用
- 有新用户上线AC都将在组内同步连接的STA的 Key cache

作用：有效解决快速漫游中的安全性

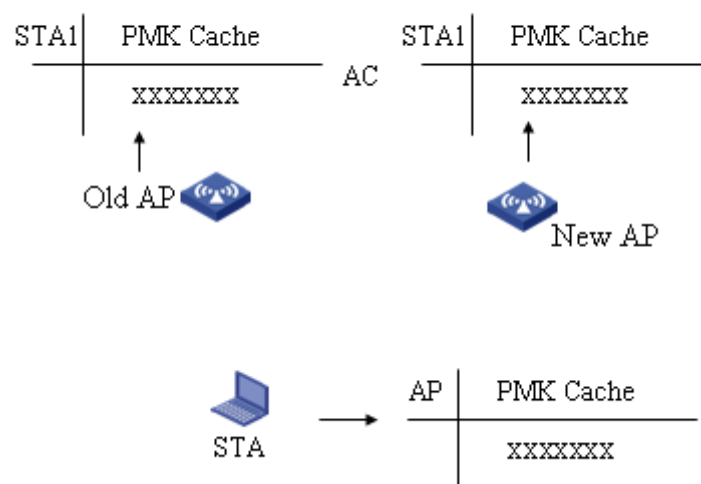


图34 Key cache 漫游过程

我们在漫游过程中可以看到一些相关消息中携带的PMKID信息；注：PMK是要保密的，所以不能在消息中携带，而携带了PMKID，通过与BSSID，MAC混合经过不可逆的算法可

以得到PMK。

(1) 在4 way handshake的mess1中，packet data的12个字节，再加上OUI的后4个字节，总共16个字节。

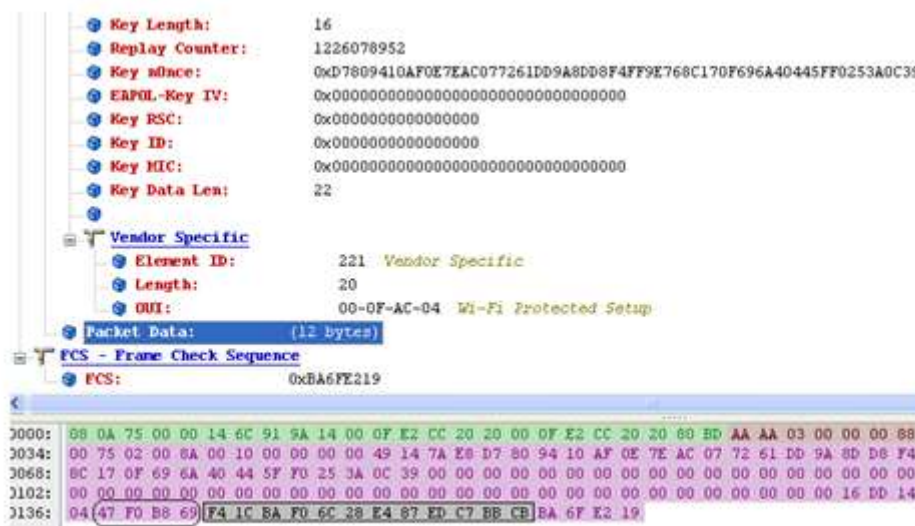


图35 PMKID 1

(2) STA发给FA的associate报文中携带PMKID，是快速漫游可以实现的关键。

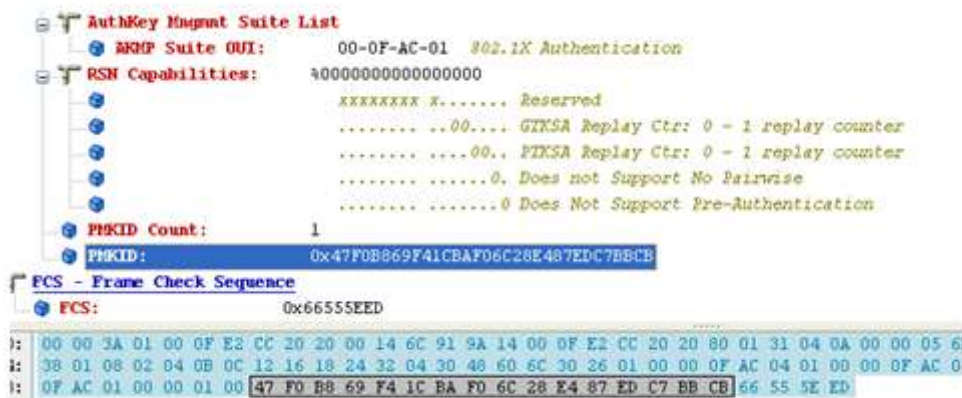


图36 PMKID 2

5.1.3 PTK, KEK, KCK

PTK:

说PMK重要，原因是它是生成PTK的材料；那么PTK是什么？还是先看协议的描述：

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).

翻译成中文叫做：成对临时密钥；

由以上描述我们可以看出，PTK的生成需要5个条件：PMK，一对MAC地址加上一对乱数（nonce）；而由PTK生成的也是5个key：加密数据的PTK，两个MIC key，KEK与KCK；

KCK和KEK：

KEK： 加密key的key；

EAPOL-Key encryption key (KEK): A key used to encrypt the Key Data field in an EAPOL-Key frame.

KCK： 解密key的key；

EAPOL-Key confirmation key (KCK): A key used to integrity-check an EAPOL-Key frame.



图37 加密的 key data 域

如上，CCMP机密的第三个报文，key data域被加密；有且仅有这个域被加密。因为在这个域中包含组密钥的信息；AP用KEK来加密，STA接收到报文之后用KCK来解密。

如果是TKIP加密，我们就找不到被加密的key data域，因为TKIP的6步握手机制保证了组密钥的单独协商。



图38 非加密的 key data 域

如上TKIP key 交换的第三个报文，key data域是明文。TKIP在4 way handshake 中并没有协商组播加密密钥，所以需要单独协商组播密钥，如下：

46	2.672277	Hangzhou_fe:03:40	Netgear_91:9a:14	EAPOL Key
47	2.691108	Netgear_91:9a:14	Hangzhou_fe:03:40	EAPOL Key
48	2.693877	Hangzhou_fe:03:40	Netgear_91:9a:14	EAPOL Key
49	2.696328	Netgear_91:9a:14	Hangzhou_fe:03:40	EAPOL Key
50	2.698014	13.0.0.235	13.0.0.77	LWAPP CNTL Bad Type: 0xda
51	2.699001	13.0.0.77	13.0.0.235	IEEE 8 Unrecognized (Reserved)
52	2.703638	Hangzhou_fe:03:40	Netgear_91:9a:14	EAPOL Key
53	2.757115	Netgear_91:9a:14	Hangzhou_fe:03:40	EAPOL Key

图39 组播密钥协商

5.1.4 MIC

MIC：消息完整性校验,这个值在11key协商过程中是必须检查的字段；

message integrity code (MIC): A value generated by a symmetric key cryptographic function. If the input data are changed, a new value cannot be correctly computed without knowledge of the symmetric key. Thus, the secret key protects the input data from undetectable alteration. This is traditionally called a *message authentication code* (MAC), but the acronym MAC is already reserved for another meaning in this amendment.

5.2 密钥分发

5.2.1 PTK 分发

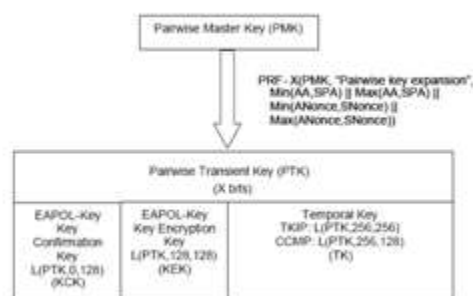


图40 PTK 分发

如上图PTK的0-127bit作为KCK，128-255bit作为KEK

如果是TKIP加密，从256bit开始的256bits供数据数据加解密使用；如果是CCMP加密，从256bit开始的128bits供数据数据加解密使用；称为TK。

TK：当STA 使用TKIP加密的时候

0-127bit：数据加密和解密使用；128-191：用作发送数据时的MIC key；192-255：用作接受数据时的MIC key

而CCMP只有一个密钥，即TK，加解密和MIC计算是一次完成的。

5.2.2 GTK 分发

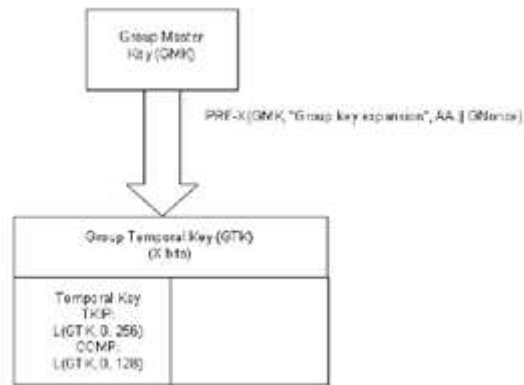


图41 GTK 分发

TKIP: 256bits中的前128bits用户数据的加解密；接下的64bits和64bits用作MIC key

CCMP: 128bits的key，所有计算一次完成。

混合加密：组播加密总是在寻找最弱的加密方法，如果是wep40/104，则使用GTK的前40/104bits作为TK；

5.3 4-Way Handshake

4-way handshake即11key协商，在认证成功之后进行；4 way handshake就是要得到数据加密的密钥，也可以说从PMK得到PTK的过程。

11Key的密钥协商为802.1x的EAPOL-Key协商的一种，11Key完成载荷组装后，通过802.1x完成EAPOL-Key报文头、EAPOL 报文头以及802.3报文头后发送；

11key的协商有4步或者6步协商，决定于采用哪种加密方法：

- (1) 如果是TKIP加密，使用6步握手协商，它的PTK协商和GTK协商是分开进行的；
- (2) 如果是CCMP加密，使用4步握手协商，在PTK协商的同时完成了GTK的协商。

下面以CCMP的密钥协商过程为例介绍4 way handshake

5.3.1 4-Way Handshake: STEP 1

方向：Authenticator - Supplicant 即AP发给STA

作用：传送ANonce

加密：无



图42 4步握手 1

STA:

- (1) 检查RC域。此值若小于等于当前本地使用的值，key1被丢弃。
- (2) 生成SNonce。
- (3) 生成PTK: ASnoce, SNonce, PMK
- (4) 构造key2

5.3.2 4-Way Handshake: STEP 2

方向： Supplicant - Authenticator 即STA发给AP

作用：传送SNonce，检查STA的PMK的合法性

加密：MIC，此报文是经过MIC加密的



图43 4步握手 2

AP:

- (1) 生成PTK: ASnoce, SNonce, PMK
- (2) MIC校验: 利用PTK对接收的报文作MIC校验如果校验失败说明STA的PMK非法, 认证失败
- (3) 检查RSN-INFO域: 检查STA关联AP时的ASSOCIATE消息是否匹配
- (4) 构造key3

5.3.3 4-Way Handshake: STEP 3

方向: Authenticator - Supplicant 即AP发给STA

作用: 安装key

加密: 加密key data域



图44 4步握手 3

STA:

- (1) 检查RC域。
- (2) 检查ANonce: 如果与key1中不一致, 丢弃key3
- (3) 检查RSN-INFO
- (4) MIC校验
- (5) 更新本地RC。为rekey做准备
- (6) 安装key
- (7) 构造key4
- (8) key data: 加密域, 传播组播密钥

5.3.4 4-Way Handshake: STEP 4

方向: Supplicant - Authenticator 即STA发给AP

作用: 安装key, 等于ACK

加密: MIC加密



图45 4步握手 4

AP:

- (1) 检查RC域。
- (2) MIC校验
- (3) 更新本地RC。
- (4) 安装key

注意:

- (1) 如果是CCMP加密, 4 way handshake完成后单播密钥和组播密钥都协商完成。
- (2) 如果是TKIP加密, 还需要两步组播密钥协商的过程。

如一下两图中的最后两步交换即为协商组播密钥。

:40	00:14:6C:91:9A:14	00:0F:E2:FE:03:40	7	80%	54.0	137	EAPOL-Key
:14	00:0F:E2:FE:03:40	00:0F:E2:FE:03:40	7	70%	1.0	159	EAPOL-Key
:40	00:14:6C:91:9A:14	00:0F:E2:FE:03:40	7	81%	54.0	163	EAPOL-Key
:14	00:0F:E2:FE:03:40	00:0F:E2:FE:03:40	7	67%	1.0	135	EAPOL-Key
:40	00:14:6C:91:9A:14	00:0F:E2:FE:03:40	U	7	81%	54.0	802.11 TKIP Data
:14	00:0F:E2:FE:03:40	00:0F:E2:FE:03:40	U	7	68%	1.0	802.11 TKIP Data

45	2.670599	Hangzhou_fe:03:40	Netgear_91:9a:14	EAP	Success
46	2.672277	Hangzhou_fe:03:40	Netgear_91:9a:14	EAPOL	Key
47	2.691108	Netgear_91:9a:14	Hangzhou_fe:03:40	EAPOL	Key
48	2.693677	Hangzhou_fe:03:40	Netgear_91:9a:14	EAPOL	Key
49	2.696328	Netgear_91:9a:14	Hangzhou_fe:03:40	EAPOL	Key
50	2.698014	13.0.0.235	13.0.0.77	LWAPP	CNTL Bad T
51	2.699001	13.0.0.77	13.0.0.235	IEEE 8	unrecogniz
52	2.703638	Hangzhou_fe:03:40	Netgear_91:9a:14	EAPOL	Key
53	2.757115	Netgear_91:9a:14	Hangzhou_fe:03:40	EAPOL	Key

图46 组播密钥单独协商

思考：为什么组播密钥协商的过程有的被加密了，而有的却没有加密呢？

被加密的协商过程中使用的加密密钥是什么？