Méthodes de vérification pour le WLAN 802.11 et l'itinérance Fast-Secure sur CUWN

Table des matières

Introduction

Ce document décrit les types d'itinérance sans fil et à sécurité rapide disponibles pour les réseaux locaux sans fil (WLAN) IEEE 802.11 sur le réseau sans fil unifié (CUWN).

Conditions préalables

Exigences

Cisco vous recommande de prendre connaissance des rubriques suivantes :

- Principes fondamentaux du WLAN IEEE 802.11
- Sécurité WLAN IEEE 802.11
- Notions de base sur IEEE 802.1X/EAP

Composants utilisés

Les informations contenues dans ce document sont basées sur la version 7.4 du logiciel du contrôleur WLAN Cisco.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. Si votre réseau est en ligne, assurez-vous de bien comprendre l’incidence possible des commandes.

Informations générales

Les informations contenues dans ce document sont basées sur la version 7.4 du logiciel du contrôleur WLAN Cisco, mais la plupart des résultats et des comportements de débogage décrits peuvent s'appliquer à n'importe quelle version du logiciel qui prend en charge les méthodes décrites. Les spécifications de toutes les méthodes expliquées ici restent les mêmes sur les codes de contrôleur WLAN Cisco ultérieurs (jusqu'à la version 8.3 au moment de la mise à jour de cet article).

Ce document décrit les différents types d'itinérance sans fil et les méthodes d'itinérance rapide et sécurisée disponibles pour les réseaux locaux sans fil (WLAN) IEEE 802.11 pris en charge sur le réseau sans fil unifié Cisco (CUWN).

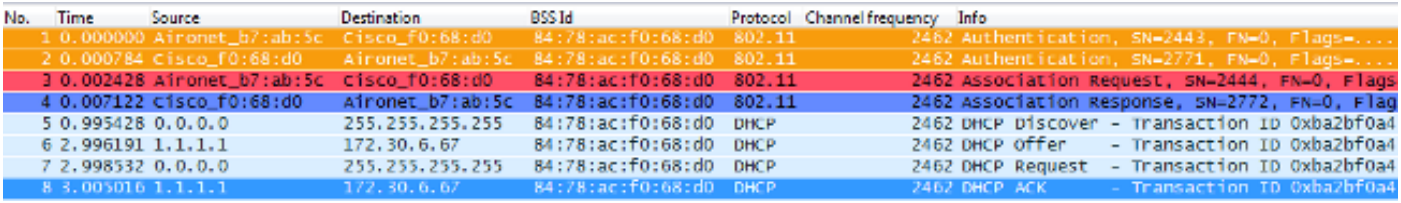
Le document ne fournit pas tous les détails sur la façon dont chaque méthode fonctionne ou comment ils sont configurés. Le but principal de ce document est de décrire les différences entre les différentes techniques disponibles, leurs avantages et limitations, et l'échange de trames sur chaque méthode. Des exemples de débogages de contrôleur WLAN (WLC) sont fournis, et des images de paquets sans fil sont utilisées afin d'analyser et d'expliquer les événements qui se produisent pour chaque méthode d'itinérance décrite.

Avant de donner une description des différentes méthodes d'itinérance rapide et sécurisée disponibles pour les WLAN, il est important de comprendre comment fonctionne le processus d'association WLAN et comment un événement d'itinérance régulier se produit lorsqu'aucune sécurité n'est configurée sur le SSID (Service Set Identifier).

Lorsqu'un client sans fil 802.11 se connecte à un point d'accès (AP), avant de commencer à transmettre le trafic (trames de données sans fil), il doit d'abord passer le processus d'authentification 802.11 Open System de base. Ensuite, le processus d'association doit être terminé. Le processus d'authentification Open System est semblable à une connexion par câble sur le point d'accès que le client sélectionne. Il s'agit d'un point très important, car c'est toujours le client sans fil qui sélectionne le point d'accès préféré et fonde la décision sur plusieurs facteurs qui varient selon les fournisseurs. C'est pourquoi le client commence ce processus en envoyant la trame d'authentification au point d'accès sélectionné, comme indiqué plus loin dans ce document. Le point d'accès ne peut pas vous demander d'établir une connexion.

Une fois que le processus d'authentification Open System est terminé avec succès avec une réponse du point d'accès (« câble connecté »), le processus d'association termine essentiellement la négociation de couche 2 (L2) 802.11 qui établit la liaison entre le client et le point d'accès. Le point d'accès attribue un ID d'association au client si la connexion réussit, et le prépare afin de transmettre le trafic ou d'effectuer une méthode de sécurité de niveau supérieur si elle est configurée sur le SSID. Le processus d'authentification Open System se compose de deux trames de gestion ainsi que du processus d'association. Les trames d'authentification et d'association sont des trames de gestion sans fil, et non des trames de données, qui sont essentiellement celles utilisées pour le processus de connexion avec le point d'accès.

Voici une image des trames sans fil transmises par liaison radio pour ce processus :



No.	Time	Source	Destination	BSS Id	Protocol	Channel frequency	Info
1	0.000000	Aironet_b7:ab:5c	Cisco_f0:68:d0	84:78:ac:f0:68:d0	802.11	2462	Authentication, SN=2443, FN=0, Flags=....
2	0.000784	Cisco_f0:68:d0	Aironet_b7:ab:5c	84:78:ac:f0:68:d0	802.11	2462	Authentication, SN=2771, FN=0, Flags=....
3	0.002428	Aironet_b7:ab:5c	Cisco_f0:68:d0	84:78:ac:f0:68:d0	802.11	2462	Association Request, SN=2444, FN=0, Flags
4	0.007122	Cisco_f0:68:d0	Aironet_b7:ab:5c	84:78:ac:f0:68:d0	802.11	2462	Association Response, SN=2772, FN=0, Flag
5	0.995428	0.0.0.0	255.255.255.255	84:78:ac:f0:68:d0	DHCP	2462	DHCP Discover - Transaction ID 0xba2bf0a4
6	2.996191	1.1.1.1	172.30.6.67	84:78:ac:f0:68:d0	DHCP	2462	DHCP Offer - Transaction ID 0xba2bf0a4
7	2.998532	0.0.0.0	255.255.255.255	84:78:ac:f0:68:d0	DHCP	2462	DHCP Request - Transaction ID 0xba2bf0a4
8	3.005016	1.1.1.1	172.30.6.67	84:78:ac:f0:68:d0	DHCP	2462	DHCP ACK - Transaction ID 0xba2bf0a4

Remarque : si vous souhaitez en savoir plus sur l'analyse sans fil 802.11 et sur les filtres/couleurs utilisés sur Wireshark pour les images qui apparaissent dans ce document, consultez le billet de la communauté d'assistance Cisco intitulé 802.11 Sniffer image Analysis.

Le client sans fil commence par la trame d'authentification, et le point d'accès répond par une autre trame d'authentification. Le client envoie alors la trame de demande d'association, et le point d'accès termine dans une réponse avec la trame de réponse d'association. Comme indiqué dans les paquets DHCP, une fois que les processus d'authentification et d'association du système ouvert 802.11 sont passés, le client commence à transmettre des trames de données. Dans ce cas, aucune méthode de sécurité n'est configurée sur le SSID, de sorte que le client commence immédiatement à envoyer des trames de données (dans ce cas, DHCP) qui ne sont pas chiffrées.

Comme indiqué plus loin dans ce document, si la sécurité est activée sur le SSID, il existe des trames d'authentification et de connexion de chiffrement de niveau supérieur pour la méthode de

sécurité spécifique, juste après la réponse d'association et avant l'envoi de trames de données de trafic client, telles que DHCP, le protocole ARP (Address Resolution Protocol) et les paquets d'applications, qui sont chiffrés. Les trames de données ne peuvent être envoyées que jusqu'à ce que le client soit entièrement authentifié et que les clés de chiffrement soient négociées, en fonction de la méthode de sécurité configurée.

D'après l'image précédente, voici les messages que vous voyez dans les sorties de la commande WLC debug client quand le client sans fil commence une nouvelle association au WLAN :

<#root>

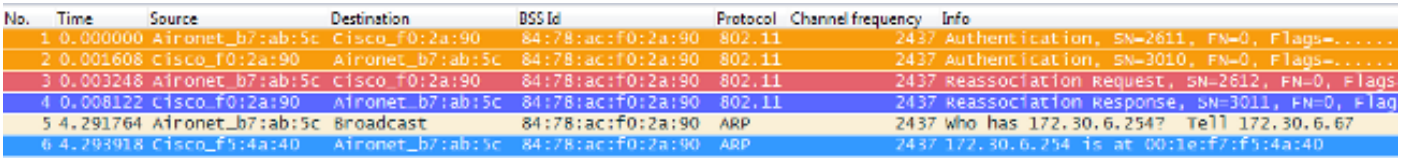
```
*apfMsConnTask_0: Jun 21 18:55:14.221: 00:40:96:b7:ab:5c
  Association received from mobile on BSSID 84:78:ac:f0:68:d0

!--- This is the Association Request from the wireless client
     to the selected AP

.

*apfMsConnTask_0: Jun 21 18:55:14.222: 00:40:96:b7:ab:5c
  Sending Assoc Response to station on BSSID 84:78:ac:f0:68:d0
  (status 0) ApVapId 1 Slot 0

!--- This is the Association Response from the AP to the client

.
```

Remarque : le débogage WLC utilisé pour les sorties montrées dans ce document est la commande debug client, et les exemples ne montrent que quelques messages pertinents, pas la sortie entière. Pour plus de détails sur cette commande debug, référez-vous au document intitulé Comprendre le client de débogage sur les contrôleurs de réseau local sans fil (WLC).

Ces messages montrent les trames de requête et de réponse d'association ; les trames d'authentification initiales ne sont pas consignées au niveau du WLC parce que cette connexion se produit rapidement au niveau du point d'accès sur le CUWN.

Quelles informations s’affichent lorsque le client est en itinérance ? Le client échange toujours quatre trames de gestion lors de l'établissement d'une connexion à un point d'accès, qui est due soit à l'établissement d'une association par le client, soit à un événement d'itinérance. Le client n'a qu'une seule connexion établie à un seul point d'accès à la fois. La seule différence dans l'échange de trames entre une nouvelle connexion à l'infrastructure WLAN et un événement d'itinérance est que les trames d'association d'un événement d'itinérance sont appelées des trames de réassociation, qui indiquent que le client est réellement en itinérance à partir d'un autre AP sans aucune tentative d'établir une nouvelle association au WLAN. Ces trames peuvent contenir différents éléments qui sont utilisés afin de négocier l'événement d'itinérance ; cela dépend de la configuration, mais ces détails sont hors de la portée de ce document.

Voici un exemple d’échange de trames :



No.	Time	Source	Destination	BSS Id	Protocol	Channel frequency	Info
1	0.000000	Aironet_b7:ab:5c	Cisco_f0:2a:90	84:78:ac:f0:2a:90	802.11	2437	Authentication, SN=2611, FN=0, Flags=.....
2	0.001608	Cisco_f0:2a:90	Aironet_b7:ab:5c	84:78:ac:f0:2a:90	802.11	2437	Authentication, SN=3010, FN=0, Flags=.....
3	0.003248	Aironet_b7:ab:5c	Cisco_f0:2a:90	84:78:ac:f0:2a:90	802.11	2437	Reassociation Request, SN=2612, FN=0, Flags
4	0.008122	Cisco_f0:2a:90	Aironet_b7:ab:5c	84:78:ac:f0:2a:90	802.11	2437	Reassociation Response, SN=3011, FN=0, Flag
5	4.291764	Aironet_b7:ab:5c	Broadcast	84:78:ac:f0:2a:90	ARP	2437	Who has 172.30.6.254? Tell 172.30.6.67
6	4.293918	Cisco_f5:4a:40	Aironet_b7:ab:5c	84:78:ac:f0:2a:90	ARP	2437	172.30.6.254 is at 00:1e:f7:f5:4a:40

Ces messages apparaissent dans le résultat du débogage :

<#root>

```
*apfMsConnTask_2: Jun 21 19:02:19.709: 00:40:96:b7:ab:5c
  Reassociation received from mobile on BSSID 84:78:ac:f0:2a:90

!--- This is the Reassociation Request from the wireless client
     to the selected AP

.

*apfMsConnTask_2: Jun 21 19:02:19.710: 00:40:96:b7:ab:5c
  Sending Assoc Response to station on BSSID 84:78:ac:f0:2a:90
  (status 0) ApVapId 1 Slot 0

!--- This is the Reassociation Response from the AP to the client

.
```

Comme indiqué, le client exécute avec succès un événement d'itinérance après l'envoi de la demande de réassociation au nouveau point d'accès et reçoit la réponse de réassociation du point d'accès. Comme le client possède déjà une adresse IP, les premières trames de données sont destinées aux paquets ARP.

Si vous attendez un événement d'itinérance, mais que le client envoie une demande d'association au lieu d'une demande de réassociation (que vous pouvez confirmer à partir de certaines images et débogages similaires à ceux expliqués précédemment dans ce document), alors le client n'est pas vraiment en itinérance. Le client commence une nouvelle association au WLAN comme si une déconnexion avait eu lieu et tente de se reconnecter de zéro. Cela peut se produire pour plusieurs raisons, par exemple lorsqu'un client s'éloigne des zones de couverture et trouve ensuite un point d'accès avec une qualité de signal suffisante pour démarrer une association, mais cela indique normalement un problème client où le client ne lance pas d'événement d'itinérance en raison de problèmes de pilotes, de microprogrammes ou de logiciels.

Remarque : vous pouvez vérifier auprès du fournisseur du client sans fil afin de déterminer la cause du problème.

Itinérance avec sécurité de niveau supérieur

Lorsque le SSID est configuré avec une sécurité de niveau 2 supérieur en plus de l'authentification 802.11 Open System de base, davantage de trames sont nécessaires pour l'association initiale et lors de l'itinérance. Les deux méthodes de sécurité les plus courantes, normalisées et mises en

oeuvre pour les WLAN 802.11, sont décrites dans ce document :

- WPA/WPA2-PSK (Pre-Shared Key) : authentification des clients à l'aide d'une clé prépartagée.
- WPA/WPA2-EAP (Extensible Authentication Protocol) : authentification des clients à l'aide d'une méthode 802.1X/EAP afin de valider des informations d'identification plus sécurisées grâce à l'utilisation d'un serveur d'authentification, tel que des certificats, un nom d'utilisateur et un mot de passe, et des jetons.

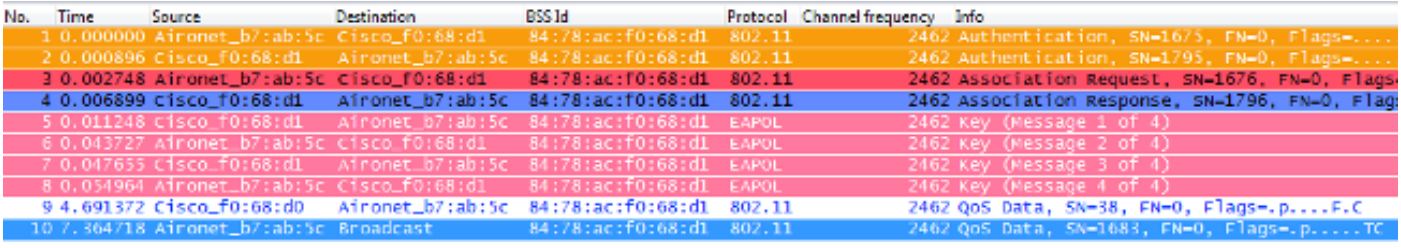
Il est important de savoir que, même si ces deux méthodes (PSK et EAP) authentifient/valident les clients de différentes manières, les deux utilisent essentiellement les mêmes règles WPA/WPA2 pour le processus de gestion des clés. Que la sécurité soit WPA/WPA2-PSK ou WPA/WPA2-EAP, le processus connu sous le nom de connexion en 4 étapes WPA/WPA2 lance la négociation de clé entre le WLC/AP et le client avec une clé de session principale (MSK) comme matériel de clé d'origine une fois que le client est validé avec la méthode d'authentification spécifique utilisée.

Voici un résumé du processus :

1. Une clé MSK est dérivée de la phase d'authentification EAP lorsque la sécurité 802.1X/EAP est utilisée, ou de la clé PSK lorsque la méthode de sécurité WPA/WPA2-PSK est utilisée.
2. À partir de ce MSK, le client et le WLC/AP dérivent la clé principale par paire (PMK), et le WLC/AP génère une clé principale de groupe (GMK).
3. Une fois que ces deux clés principales sont prêtes, le client et le WLC/AP lancent la connexion en 4 étapes WPA/WPA2 (qui est illustrée plus loin dans ce document avec quelques images d'écran et des débogages) avec les clés principales comme valeurs de départ pour la négociation des clés de chiffrement réelles.
4. Ces dernières sont appelées PTK (Pairwise Transient Key) et GTK (Group Transient Key). Le PTK est dérivé du PMK et utilisé afin de chiffrer les trames de monodiffusion avec le client. La GTK (Group Transient Key) est dérivée de la GMK et est utilisée pour chiffrer la multidiffusion/diffusion sur ce SSID/AP spécifique.

WPA/WPA2-PSK

Lorsque WPA-PSK ou WPA2-PSK est exécuté via le protocole TKIP (Temporal Key Integrity Protocol) ou AES (Advanced Encryption Standard) pour le cryptage, le client doit passer par le processus connu sous le nom de WPA 4-Way handshake pour l'association initiale et également lors de l'itinérance. Comme expliqué précédemment, il s'agit essentiellement du processus de gestion des clés utilisé pour que WPA/WPA2 dérive les clés de cryptage. Cependant, lorsque la clé PSK est exécutée, elle est également utilisée afin de vérifier que le client dispose d'une clé pré-partagée valide pour se connecter au WLAN. Cette image montre le processus d'association initial lorsque WPA ou WPA2 avec PSK est exécuté :



Comme indiqué, après le processus d'authentification et d'association du système ouvert 802.11, il y a quatre trames EAPOL de la connexion en 4 étapes WPA, qui sont initiées par l'AP avec message-1, et terminées par le client avec message-4. Après une connexion réussie, le client commence à transmettre des trames de données (telles que DHCP), qui dans ce cas sont chiffrées avec les clés dérivées de la connexion en 4 étapes (c'est pourquoi vous ne pouvez pas voir le contenu réel et le type de trafic des images sans fil).

Remarque : les trames EAPOL sont utilisées afin de transporter toutes les trames de gestion des clés et les trames d'authentification 802.1X/EAP par liaison radio entre le point d'accès et le client ; elles sont transmises sous forme de trames de données sans fil.

Ces messages apparaissent dans les résultats du débogage :

<#root>

```
*apfMsConnTask_0: Jun 21 19:30:05.172: 00:40:96:b7:ab:5c
  Association received from mobile on BSSID 84:78:ac:f0:68:d1
*apfMsConnTask_0: Jun 21 19:30:05.173: 00:40:96:b7:ab:5c
  Sending Assoc Response to station on BSSID 84:78:ac:f0:68:d1
  (status 0) ApVapId 2 Slot 0

!--- The Association handshake is finished.



*dot1xMsgTask: Jun 21 19:30:05.178: 00:40:96:b7:ab:5c
  Sending EAPOL-Key Message to mobile 00:40:96:b7:ab:5c
  state INITPMK (message 1), replay counter
  00.00.00.00.00.00.00.00

!--- Message-1 of the WPA/WPA2 4-Way handshake is sent
     from the WLC/AP to the client.



*Dot1x_NW_MsgTask_4: Jun 21 19:30:05.289: 00:40:96:b7:ab:5c
  Received EAPOL-Key from mobile 00:40:96:b7:ab:5c
*Dot1x_NW_MsgTask_4: Jun 21 19:30:05.289: 00:40:96:b7:ab:5c
  Received EAPOL-key in PTK_START state (message 2)
  from mobile 00:40:96:b7:ab:5c

!--- Message-2 of the WPA/WPA2 4-Way handshake is successfully
     received from the client.



*Dot1x_NW_MsgTask_4: Jun 21 19:30:05.290: 00:40:96:b7:ab:5c
  Sending EAPOL-Key Message to mobile 00:40:96:b7:ab:5c
  state PTKINITNEGOTIATING (message 3), replay counter
  00.00.00.00.00.00.00.01

!--- Message-3 of the WPA/WPA2 4-Way handshake is sent
     from the WLC/AP to the client.



*Dot1x_NW_MsgTask_4: Jun 21 19:30:05.309: 00:40:96:b7:ab:5c
  Received EAPOL-Key from mobile 00:40:96:b7:ab:5c
```

```
*Dot1x_NW_MsgTask_4: Jun 21 19:30:05.310: 00:40:96:b7:ab:5c
  Received EAPOL-key in PTKINITNEGOTIATING state (message 4)
  from mobile 00:40:96:b7:ab:5c

!--- Message-4 (final message) of the WPA/WPA2 4-Way handshake
     is successfully received from the client, which confirms
     the installation of the derived keys. They can now be used in
     order to encrypt data frames with current AP.
```

En itinérance, le client suit essentiellement le même échange de trames, où la connexion WPA en 4 étapes est requise pour dériver de nouvelles clés de cryptage avec le nouveau point d'accès. Cela est dû à des raisons de sécurité établies par la norme, et au fait que le nouveau point d'accès ne connaît pas les clés d'origine. La seule différence est qu'il y a des trames de réassociation au lieu de trames d'association, comme le montre cette image :

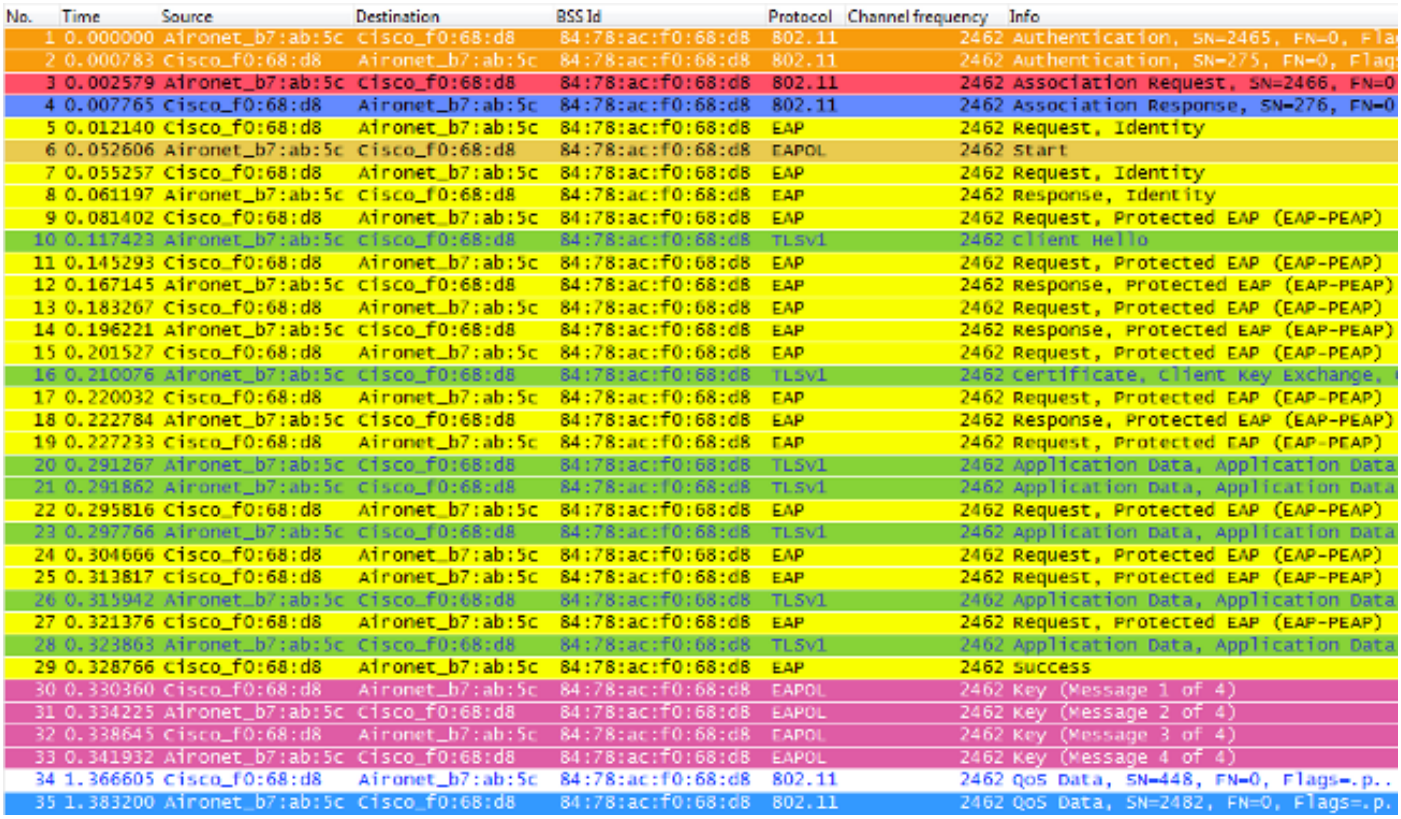
No.	Time	Source	Destination	BSS Id	Protocol	Channel frequency	Info
1	0.000000	Aironet_b7:ab:5c	Cisco_f0:2a:91	84:78:ac:f0:2a:91	802.11	2437	Authentication, SN=2356, FN=0, Flags=.....
2	0.000846	Cisco_f0:2a:91	Aironet_b7:ab:5c	84:78:ac:f0:2a:91	802.11	2437	Authentication, SN=3694, FN=0, Flags=.....
3	0.004296	Aironet_b7:ab:5c	Cisco_f0:2a:91	84:78:ac:f0:2a:91	802.11	2437	Reassociation Request, SN=2357, FN=0, Flags
4	0.010867	Cisco_f0:2a:91	Aironet_b7:ab:5c	84:78:ac:f0:2a:91	802.11	2437	Reassociation Response, SN=3695, FN=0, Flag
5	0.013109	Cisco_f0:2a:91	Aironet_b7:ab:5c	84:78:ac:f0:2a:91	EAPOL	2437	Key (Message 1 of 4)
6	0.034339	Aironet_b7:ab:5c	Cisco_f0:2a:91	84:78:ac:f0:2a:91	EAPOL	2437	Key (Message 2 of 4)
7	0.041124	Cisco_f0:2a:91	Aironet_b7:ab:5c	84:78:ac:f0:2a:91	EAPOL	2437	Key (Message 3 of 4)
8	0.056241	Aironet_b7:ab:5c	Cisco_f0:2a:91	84:78:ac:f0:2a:91	EAPOL	2437	Key (Message 4 of 4)
9	0.695758	Aironet_b7:ab:5c	Broadcast	84:78:ac:f0:2a:91	802.11	2437	QoS Data, SN=2360, FN=0, Flags=.p..R..TC
10	0.698357	Cisco_f5:4a:40	Aironet_b7:ab:5c	84:78:ac:f0:2a:91	802.11	2437	QoS Data, SN=42, FN=0, Flags=.p....F.C

Vous voyez les mêmes messages dans les sorties de débogage, mais le premier paquet du client est une réassociation au lieu d'une association, comme indiqué et expliqué précédemment.

WPA/WPA2-EAP

Lorsqu'une méthode 802.1X/EAP est utilisée afin d'authentifier les clients sur un SSID sécurisé, il y a encore plus de trames requises avant que le client commence à transmettre le trafic. Ces trames supplémentaires sont utilisées afin d'authentifier les informations d'identification du client, et selon la méthode EAP, il peut y avoir entre quatre et vingt trames. Elles se produisent après l'association/la réassociation, mais avant la connexion en 4 étapes WPA/WPA2, car la phase d'authentification dérive le MSK utilisé comme valeur de départ pour la génération finale de la clé de chiffrement dans le processus de gestion des clés (connexion en 4 étapes).

Cette image montre un exemple des trames échangées par radio entre le point d'accès et le client sans fil lors de l'association initiale lorsque WPA avec PEAPv0/EAP-MSCHAPv2 est exécuté :



No.	Time	Source	Destination	BSS Id	Protocol	Channel frequency	Info
1	0.000000	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	802.11	2462	Authentication, SN=2465, FN=0, Fla
2	0.000783	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	802.11	2462	Authentication, SN=275, FN=0, Flag
3	0.002579	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	802.11	2462	Association Request, SN=2466, FN=0
4	0.007765	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	802.11	2462	Association Response, SN=276, FN=0
5	0.012140	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Request, Identity
6	0.052606	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	EAPOL	2462	Start
7	0.055257	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Request, Identity
8	0.061197	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	EAP	2462	Response, Identity
9	0.081402	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Request, Protected EAP (EAP-PEAP)
10	0.117423	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	TLSv1	2462	Client Hello
11	0.145293	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Request, Protected EAP (EAP-PEAP)
12	0.167145	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	EAP	2462	Response, Protected EAP (EAP-PEAP)
13	0.183267	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Request, Protected EAP (EAP-PEAP)
14	0.196221	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	EAP	2462	Response, Protected EAP (EAP-PEAP)
15	0.201527	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Request, Protected EAP (EAP-PEAP)
16	0.210076	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	TLSv1	2462	Certificate, Client Key Exchange,
17	0.220032	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Request, Protected EAP (EAP-PEAP)
18	0.222784	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	EAP	2462	Response, Protected EAP (EAP-PEAP)
19	0.227233	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Request, Protected EAP (EAP-PEAP)
20	0.291267	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	TLSv1	2462	Application Data, Application Data
21	0.291862	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	TLSv1	2462	Application Data, Application Data
22	0.295816	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Request, Protected EAP (EAP-PEAP)
23	0.297766	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	TLSv1	2462	Application Data, Application Data
24	0.304666	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Request, Protected EAP (EAP-PEAP)
25	0.313817	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Request, Protected EAP (EAP-PEAP)
26	0.315942	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	TLSv1	2462	Application Data, Application Data
27	0.321376	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Request, Protected EAP (EAP-PEAP)
28	0.323863	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	TLSv1	2462	Application Data, Application Data
29	0.328766	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAP	2462	Success
30	0.330360	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAPOL	2462	Key (Message 1 of 4)
31	0.334225	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	EAPOL	2462	Key (Message 2 of 4)
32	0.338645	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	EAPOL	2462	Key (Message 3 of 4)
33	0.341932	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	EAPOL	2462	Key (Message 4 of 4)
34	1.366605	Cisco_f0:68:d8	Aironet_b7:ab:5c	84:78:ac:f0:68:d8	802.11	2462	QoS Data, SN=448, FN=0, Flags=.p..
35	1.383200	Aironet_b7:ab:5c	Cisco_f0:68:d8	84:78:ac:f0:68:d8	802.11	2462	QoS Data, SN=2482, FN=0, Flags=.p.

Parfois, cet échange montre plus ou moins de trames, ce qui dépend de plusieurs facteurs, tels que la méthode EAP, les retransmissions dues à des problèmes, le comportement du client (comme les deux demandes d'identité dans cet exemple, parce que le client envoie un EAPOL START après que l'AP envoie la première demande d'identité), ou si le client a déjà échangé le certificat avec le serveur. Chaque fois que le SSID est configuré pour une méthode 802.1X/EAP, il y a plus de trames (pour l'authentification) et, par conséquent, il faut plus de temps avant que le client commence à envoyer des trames de données.

Voici un résumé des messages de débogage :

<#root>

```
*apfMsConnTask_0: Jun 21 23:41:19.092: 00:40:96:b7:ab:5c
  Association received from mobile on BSSID 84:78:ac:f0:68:d8
*apfMsConnTask_0: Jun 21 23:41:19.094: 00:40:96:b7:ab:5c
  Sending Assoc Response to station on BSSID 84:78:ac:f0:68:d8
  (status 0) ApVapId 9 Slot 0
```

!--- The Association handshake is finished.

```
*dot1xMsgTask: Jun 21 23:41:19.098: 00:40:96:b7:ab:5c
  Sending EAP-Request/Identity to mobile 00:40:96:b7:ab:5c
  (EAP Id 1)
```

**!--- The EAP Identity Request is sent to the client once it is
associated in order to begin the higher-level authentication
process. This informs the client that an identity to start
this type of 802.1X/EAP authentication must be provided.**

```
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.226: 00:40:96:b7:ab:5c
  Received EAPOL START from mobile 00:40:96:b7:ab:5c
```

!--- The wireless client decides to start the EAP authentication process, and informs the AP with an EAPOL START data frame.

```
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.227: 00:40:96:b7:ab:5c
  Sending EAP-Request/Identity to mobile 00:40:96:b7:ab:5c
  (EAP Id 2)
```

!--- WLC/AP sends another EAP Identity Request to the client.

```
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.235: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.235: 00:40:96:b7:ab:5c
  Received Identity Response (count=2) from mobile 00:40:96:b7:ab:5c
```

!--- The client responds with an EAP Identity Response on an EAPOL frame.

```
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.301: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.301: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 3)
```

!--- Once the WLC/AP sends the client response to the Authentication Server on a RADIUS Access-Request packet, the server responds with a RADIUS Access-Challenge in order to officially start the EAP negotiation, handshake, and authentication with the client (sometimes with mutual authentication, dependent upon the EAP method). This response received by the WLC/AP is sent to the client.

```
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.344: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.344: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 3, EAP Type 25)
```

!--- The client responds with an EAP Response on an EAPOL frame, which is sent to the Authentication Server on a RADIUS Access-Request packet. The server responds with another RADIUS Access-Challenge. This process continues, dependent upon the EAP method (the exchange of certificates when used, the building of TLS tunnels, validation of client credentials, client validation of server identity when applicable). Hence, the next few messages are basically the same on the WLC/AP side, as this acts as a "proxy" between the client and the Authentication Server exchanges.

```
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.347: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.347: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
```

```
  (EAP Id 4)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.375: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.375: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 4, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.377: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.377: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 5)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.403: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.403: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 5, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.404: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.404: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 6)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.414: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.414: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 6, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.421: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.421: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 7)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.425: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.425: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 7, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.427: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.427: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 8)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.434: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.434: 00:40:96:b7:ab:5c
```

```
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 8, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.436: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.436: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 9)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.440: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.440: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 9, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.442: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.442: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 10)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.449: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.449: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 10, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.452: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.452: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 11)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.457: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.457: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 11, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.459: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.459: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 13)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.469: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.469: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 13, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.472: 00:40:96:b7:ab:5c
  Processing Access-Accept for mobile 00:40:96:b7:ab:5c
```

!--- The authentication finishes and is successful for this client,

so the RADIUS Server sends a RADIUS Access-Accept to the WLC/AP. This RADIUS Access-Accept comes with the special attributes that are assigned to this client (if any are configured on the Authentication Server for this client). This Access-Accept also comes with the MSK derived with the client in the EAP authentication process, so the WLC/AP installs it in order to initiate the WPA/WPA2 4-Way handshake with the wireless client.

```
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.473: 00:40:96:b7:ab:5c
  Sending EAP-Success to mobile 00:40:96:b7:ab:5c
  (EAP Id 13)
```

!--- The accept/pass of the authentication is sent to the client as an EAP-Success message.

```
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.473: 00:40:96:b7:ab:5c
  Sending EAPOL-Key Message to mobile 00:40:96:b7:ab:5c
  state INITPMK (message 1), replay counter
  00.00.00.00.00.00.00.00
```

!--- Message-1 of the WPA/WPA2 4-Way handshake is sent from the WLC/AP to the client.

```
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.481: 00:40:96:b7:ab:5c
  Received EAPOL-Key from mobile 00:40:96:b7:ab:5c
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.481: 00:40:96:b7:ab:5c
  Received EAPOL-key in PTK_START state (message 2)
  from mobile 00:40:96:b7:ab:5c
```

!--- Message-2 of the WPA/WPA-2 4-Way handshake is successfully received from the client.

```
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.481: 00:40:96:b7:ab:5c
  Sending EAPOL-Key Message to mobile 00:40:96:b7:ab:5c
  state PTKINITNEGOTIATING (message 3), replay counter
  00.00.00.00.00.00.00.01
```

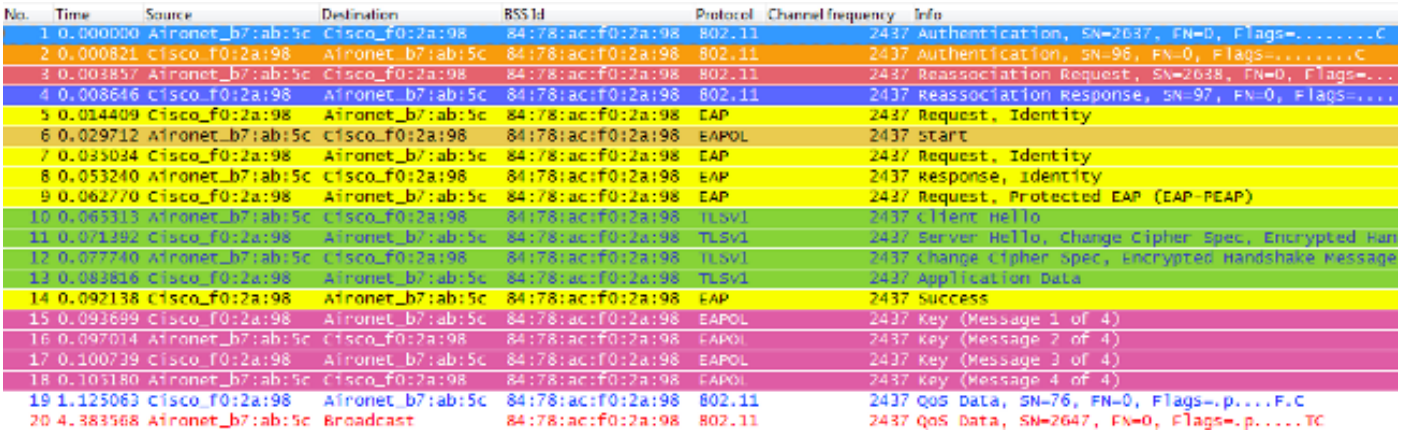
!--- Message-3 of the WPA/WPA2 4-Way handshake is sent from the WLC/AP to the client.

```
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.487: 00:40:96:b7:ab:5c
  Received EAPOL-Key from mobile 00:40:96:b7:ab:5c
*Dot1x_NW_MsgTask_4: Jun 21 23:41:19.487: 00:40:96:b7:ab:5c
  Received EAPOL-key in PTKINITNEGOTIATING state (message 4)
  from mobile 00:40:96:b7:ab:5c
```

!--- Message-4 (final message) of the WPA/WPA2 4-Way handshake is successfully received from the client, which confirms the installation of the derived keys. They can now be used in order to encrypt data frames with the current AP.

Lorsque le client sans fil effectue une itinérance normale ici (comportement normal, sans

implémentation d'une méthode d'itinérance rapide et sécurisée), le client doit suivre exactement le même processus et effectuer une authentification complète sur le serveur d'authentification, comme illustré dans les images. La seule différence est que le client utilise une demande de réassociation afin d'informer le nouveau point d'accès qu'il est réellement en itinérance à partir d'un autre point d'accès, mais le client doit encore passer par une validation complète et une nouvelle génération de clé :



Comme indiqué, même lorsqu'il y a moins de trames que dans l'authentification initiale (qui est causée par plusieurs facteurs, comme mentionné précédemment), lorsque le client se déplace vers un nouveau point d'accès, les processus d'authentification EAP et de gestion de clé WPA doivent toujours être terminés afin de continuer à transmettre des trames de données (même si le trafic a été envoyé activement avant l'itinérance). Par conséquent, si le client dispose d'une application active sensible aux retards (par exemple, des applications de trafic vocal ou des applications sensibles aux dépassements de délai), l'utilisateur peut percevoir des problèmes lors de l'itinérance, tels que des interruptions audio ou des déconnexions d'application. Cela dépend de la durée nécessaire au processus pour que le client continue à envoyer/recevoir des trames de données. Ce délai peut être plus long, selon : l'environnement RF, la quantité de clients, le temps aller-retour entre le WLC et les LAP et avec le serveur d'authentification, et d'autres raisons.

Voici un résumé des messages de débogage pour cet événement d'itinérance (fondamentalement les mêmes que les précédents, donc ces messages ne sont pas décrits plus en détail) :

```
*apfMsConnTask_2: Jun 21 23:47:54.872: 00:40:96:b7:ab:5c
  Reassociation received from mobile on BSSID 84:78:ac:f0:2a:98

*apfMsConnTask_2: Jun 21 23:47:54.874: 00:40:96:b7:ab:5c
  Sending Assoc Response to station on BSSID 84:78:ac:f0:2a:98
  (status 0) ApVapId 9 Slot 0

*dot1xMsgTask: Jun 21 23:47:54.879: 00:40:96:b7:ab:5c
  Sending EAP-Request/Identity to mobile 00:40:96:b7:ab:5c
  (EAP Id 1)

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.895: 00:40:96:b7:ab:5c
  Received EAPOL START from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.895: 00:40:96:b7:ab:5c
  dot1x - moving mobile 00:40:96:b7:ab:5c into Connecting state
```

```
*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.895: 00:40:96:b7:ab:5c
  Sending EAP-Request/Identity to mobile 00:40:96:b7:ab:5c
  (EAP Id 2)

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.922: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.922: 00:40:96:b7:ab:5c
  Received Identity Response (count=2) from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.929: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.929: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 3)

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.941: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.941: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 3, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.943: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.943: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 4)

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.954: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.954: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 4, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.956: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.957: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 7)

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.976: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.976: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 7, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.978: 00:40:96:b7:ab:5c
  Processing Access-Accept for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.978: 00:40:96:b7:ab:5c
  Sending EAP-Success to mobile 00:40:96:b7:ab:5c
  (EAP Id 7)

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.978: 00:40:96:b7:ab:5c
  Sending EAPOL-Key Message to mobile 00:40:96:b7:ab:5c
  state INITPMK (message 1), replay counter
```

```
  00.00.00.00.00.00.00.00

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.995: 00:40:96:b7:ab:5c
  Received EAPOL-Key from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.995: 00:40:96:b7:ab:5c
  Received EAPOL-key in PTK_START state (message 2)
  from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:54.995: 00:40:96:b7:ab:5c
  Sending EAPOL-Key Message to mobile 00:40:96:b7:ab:5c
  state PTKINITNEGOTIATING (message 3), replay counter
  00.00.00.00.00.00.00.01

*Dot1x_NW_MsgTask_4: Jun 21 23:47:55.005: 00:40:96:b7:ab:5c
  Received EAPOL-Key from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 23:47:55.005: 00:40:96:b7:ab:5c
  Received EAPOL-key in PTKINITNEGOTIATING state (message 4)
  from mobile 00:40:96:b7:ab:5c
```

C'est ainsi que fonctionnent la norme 802.1X/EAP et le cadre de sécurité WPA/WPA2. Afin d'éviter l'impact de l'application/du service sur les retards d'un événement d'itinérance régulier, plusieurs méthodes d'itinérance rapide et sécurisée sont développées et mises en oeuvre par l'industrie WiFi afin d'accélérer le processus d'itinérance lorsque la sécurité est utilisée sur le WLAN/SSID. Les clients sont confrontés à une certaine latence lorsqu'ils continuent à transmettre du trafic tout en se déplaçant entre les points d'accès via le déploiement d'une sécurité de haut niveau sur le WLAN. Cela est dû à l'authentification EAP et aux échanges de trames de gestion de clé requis par la configuration de la sécurité, comme expliqué précédemment.

Il est important de comprendre que l'itinérance rapide et sécurisée est le seul terme utilisé par le secteur en référence à la mise en oeuvre d'une méthode/d'un schéma qui accélère le processus d'itinérance lorsque la sécurité est configurée sur le WLAN. Les différentes méthodes/schémas d’itinérance rapide et sécurisée disponibles pour les réseaux locaux sans fil et pris en charge par le CUWN sont expliqués dans la section suivante.

Itinérance rapide et sécurisée avec CCKM

Cisco Centralized Key Management (CCKM) est la première méthode d'itinérance rapide et sécurisée développée et mise en oeuvre sur les WLAN d'entreprise. Elle a été créée par Cisco en tant que solution utilisée afin de réduire les délais expliqués jusqu'à présent, lorsque la sécurité 802.1X/EAP est utilisée sur le WLAN. Comme il s'agit d'un protocole propriétaire de Cisco, il est uniquement pris en charge par les périphériques d'infrastructure WLAN et les clients sans fil Cisco (de plusieurs fournisseurs) compatibles CCX (Cisco Compatible Extension) pour CCKM.

CCKM peut être mis en oeuvre avec toutes les différentes méthodes de cryptage disponibles pour les WLAN, notamment : WEP, TKIP et AES. Elle est également prise en charge avec la plupart des méthodes d'authentification 802.1X/EAP utilisées pour les WLAN, en fonction de la version CCX prise en charge par les périphériques.

Remarque : pour obtenir une présentation du contenu des fonctionnalités prises en charge par les différentes versions de la spécification CCX (y compris les méthodes EAP prises en charge), consultez le document Versions et fonctionnalités CCX, puis vérifiez la version exacte de CCX prise en charge par vos clients sans fil (si elles sont compatibles CCX), afin de vérifier si la méthode de sécurité que vous souhaitez utiliser avec CCKM peut être implémentée.

Cette image sans fil fournit un exemple des trames échangées lors de l'association initiale lorsque vous exécutez CCKM avec TKIP comme méthode de cryptage et PEAPv0/EAP-MSCHAPv2 comme méthode 802.1X/EAP. Il s'agit fondamentalement du même échange que si WPA/TKIP avec PEAPv0/EAP-MSCHAPv2 est effectué, mais cette fois CCKM entre le client et l'infrastructure est négocié de sorte qu'ils utilisent différentes hiérarchies de clés et méthodes de cache afin d'effectuer l'itinérance sécurisée rapide lorsque le client doit se déplacer :

No.	Time	Source	Destination	BSS Id	Protocol	Channel frequency	Info
1	0.000000	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	802.11	2462	Authentication, SN=2518, FN=0, Flag
2	0.000906	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	802.11	2462	Authentication, SN=3096, FN=0, Flag
3	0.002675	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	802.11	2462	Association Request, SN=2519, FN=0,
4	0.007562	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	802.11	2462	Association Response, SN=3097, FN=0
5	0.013614	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAP	2462	Request, Identity
6	0.032754	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	EAPOL	2462	Start
7	0.042974	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	EAP	2462	Response, Identity
8	0.046855	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	EAP	2462	Response, Identity
9	0.054287	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAP	2462	Request, Protected EAP (EAP-PEAP)
10	0.090265	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	TLSv1	2462	Client Hello
11	0.107247	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAP	2462	Request, Protected EAP (EAP-PEAP)
12	0.124080	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	EAP	2462	Response, Protected EAP (EAP-PEAP)
13	0.140385	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAP	2462	Request, Protected EAP (EAP-PEAP)
14	0.154095	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	EAP	2462	Response, Protected EAP (EAP-PEAP)
15	0.158341	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAP	2462	Request, Protected EAP (EAP-PEAP)
16	0.176346	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	TLSv1	2462	Certificate, Client Key Exchange, C
17	0.186458	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAP	2462	Request, Protected EAP (EAP-PEAP)
18	0.195391	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	EAP	2462	Response, Protected EAP (EAP-PEAP)
19	0.201648	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAP	2462	Request, Protected EAP (EAP-PEAP)
20	0.298860	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	TLSv1	2462	Application Data, Application Data
21	0.310941	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	TLSv1	2462	Application Data, Application Data
22	0.315574	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAP	2462	Request, Protected EAP (EAP-PEAP)
23	0.318255	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	TLSv1	2462	Application Data, Application Data
24	0.324589	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAP	2462	Request, Protected EAP (EAP-PEAP)
25	0.332059	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAP	2462	Request, Protected EAP (EAP-PEAP)
26	0.339778	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAP	2462	Success
27	0.341365	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAPOL	2462	Key (Message 1 of 4)
28	0.354695	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	EAPOL	2462	Key (Message 2 of 4)
29	0.358951	Cisco_f0:68:d3	Aironet_b7:ab:5c	84:78:ac:f0:68:d3	EAPOL	2462	Key (Message 3 of 4)
30	0.362866	Aironet_b7:ab:5c	Cisco_f0:68:d3	84:78:ac:f0:68:d3	EAPOL	2462	Key (Message 4 of 4)

Voici un résumé des messages de débogage (avec quelques échanges EAP supprimés afin de réduire le résultat) :

<#root>

```
*apfMsConnTask_0: Jun 25 15:41:41.507: 00:40:96:b7:ab:5c
  Association received from mobile on BSSID 84:78:ac:f0:68:d3

!--- This is the Association Request from the client.



*apfMsConnTask_0: Jun 25 15:41:41.507: 00:40:96:b7:ab:5c
```

```
  Processing WPA IE type 221, length 22 for mobile
  00:40:96:b7:ab:5c
*apfMsConnTask_0: Jun 25 15:41:41.507: 00:40:96:b7:ab:5c
  CCKM: Mobile is using CCKM
```

!--- The WLC/AP finds an Information Element that claims CCKM support on the Association request that is sent from the client.

```
*apfMsConnTask_0: Jun 25 15:41:41.507: 00:40:96:b7:ab:5c
  Setting active key cache index 8 ---> 8
```

!--- This is the key cache index for this client, which is set temporally.

```
*apfMsConnTask_0: Jun 25 15:41:41.508: 00:40:96:b7:ab:5c
  Sending Assoc Response to station on BSSID 84:78:ac:f0:68:d3
  (status 0) ApVapId 4 Slot 0
```

!--- The Association Response is sent to the client.

```
*dot1xMsgTask: Jun 25 15:41:41.513: 00:40:96:b7:ab:5c
  Sending EAP-Request/Identity to mobile 00:40:96:b7:ab:5c
  (EAP Id 1)
```

!--- An EAP Identity Request is sent to the client once it is associated in order to begin the higher-level authentication process. This informs the client that an identity to start this type of 802.1X/EAP authentication must be provided. Further EAP messages are not described, as they are basically the same as the ones previously-explained.

```
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.536: 00:40:96:b7:ab:5c
  Received EAPOL START from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.536: 00:40:96:b7:ab:5c
  Sending EAP-Request/Identity to mobile 00:40:96:b7:ab:5c
  (EAP Id 2)

*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.546: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.546: 00:40:96:b7:ab:5c
  Received EAP Response packet with mismatching id
  (currentid=2, eapid=1) from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.550: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.550: 00:40:96:b7:ab:5c
  Received Identity Response (count=2) from mobile
  00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.555: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.555: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
```

```
  (EAP Id 3)

*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.594: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.594: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 3, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.840: 00:40:96:b7:ab:5c
  Processing Access-Accept for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.841: 00:40:96:b7:ab:5c
  Creating a PKC PMKID Cache entry for station 00:40:96:b7:ab:5c
  (RSN 0)<br/ >
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.841: 00:40:96:b7:ab:5c
  Setting active key cache index 8 ---> 8
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.841: 00:40:96:b7:ab:5c
  Setting active key cache index 8 ---> 0
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.841: 00:40:96:b7:ab:5c
  CCKM: Create a global PMK cache entry
```

!--- WLC creates a global PMK cache entry for this client, which is for CCKM in this case.

```
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.841: 00:40:96:b7:ab:5c
  Sending EAP-Success to mobile 00:40:96:b7:ab:5c
  (EAP Id 13)
```

!--- The client is informed of the successful EAP authentication.

```
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.841: 00:40:96:b7:ab:5c
  Sending EAPOL-Key Message to mobile 00:40:96:b7:ab:5c state
  INITPMK(message 1),replay counter 00.00.00.00.00.00.00.00
```

!--- Message-1 of the initial 4-Way handshake is sent from the WLC/AP to the client.

```
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.858: 00:40:96:b7:ab:5c
  Received EAPOL-Key from mobile 00:40:96:b7:ab:5c
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.858: 00:40:96:b7:ab:5c
  Received EAPOL-key in PTK_START state (message 2) from mobile
  00:40:96:b7:ab:5c
```

!--- Message-2 of the initial 4-Way handshake is received successfully from the client.

```
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.858: 00:40:96:b7:ab:5c
  CCKM: Sending cache add
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.858: CCKM: Sending CCKM PMK
  (Version_1) information to mobility group
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.858: CCKM: Sending CCKM PMK
  (Version_2) information to mobility group
```

!--- The CCKM PMK cache entry for this client is shared with the WLCs on the mobility group.

```
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.858: 00:40:96:b7:ab:5c
  Sending EAPOL-Key Message to mobile 00:40:96:b7:ab:5c
  state PTKINITNEGOTIATING (message 3), replay counter
  00.00.00.00.00.00.00.01
```

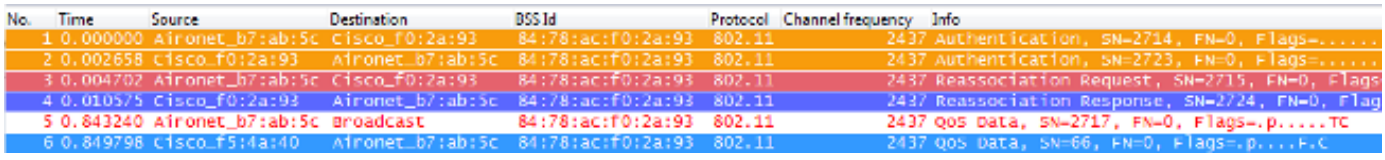
!--- Message-3 of the initial 4-Way handshake is sent from the WLC/AP to the client.

```
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.866: 00:40:96:b7:ab:5c
  Received EAPOL-Key from mobile 00:40:96:b7:ab:5c
*Dot1x_NW_MsgTask_4: Jun 25 15:41:41.866: 00:40:96:b7:ab:5c Received
  EAPOL-key in PTKINITNEGOTIATING state (message 4) from mobile
  00:40:96:b7:ab:5c
```

!--- Message-4 (final message) of this initial 4-Way handshake is received successfully from the client, which confirms the installation of the derived keys. They can now be used in order to encrypt data frames with the current AP.

Avec CCKM, l'association initiale au WLAN est similaire à la norme WPA/WPA2, où un MSK (également appelé NSK) est mutuellement dérivé avec le client et le serveur RADIUS. Cette clé primaire est envoyée du serveur au WLC après une authentification réussie, et est mise en cache comme base pour la dérivation de toutes les clés suivantes pendant la durée de vie de l'association du client avec ce WLAN. À partir de là, le WLC et le client dérivent les informations de départ qui sont utilisées pour l'itinérance rapide et sécurisée basée sur CCKM, cela passe par une connexion en 4 étapes similaire à celle de WPA/WPA2, afin de dériver les clés de cryptage unicast (PTK) et multicast/broadcast (GTK) avec le premier AP.

La grande différence se remarque lors de l'itinérance. Dans ce cas, le client CCKM envoie une seule trame de demande de réassociation au point d'accès/WLC (qui inclut un MIC et un numéro aléatoire séquentiellement incrémenté), et fournit suffisamment d'informations (qui inclut la nouvelle adresse MAC AP -BSSID-) afin de dériver le nouveau PTK. Avec cette demande de réassociation, le WLC et le nouveau AP ont également assez d'informations pour dériver le nouveau PTK, donc ils répondent simplement avec une réponse de réassociation. Le client peut maintenant continuer à transmettre le trafic, comme illustré dans cette image :



No.	Time	Source	Destination	BSS Id	Protocol	Channel frequency	Info
1	0.000000	Aironet_b7:ab:5c	Cisco_f0:2a:93	84:78:ac:f0:2a:93	802.11	2437	Authentication, SN=2714, FN=0, Flags=.....
2	0.002658	Cisco_f0:2a:93	Aironet_b7:ab:5c	84:78:ac:f0:2a:93	802.11	2437	Authentication, SN=2723, FN=0, Flags=.....
3	0.004702	Aironet_b7:ab:5c	Cisco_f0:2a:93	84:78:ac:f0:2a:93	802.11	2437	Reassociation Request, SN=2715, FN=0, Flags
4	0.010575	Cisco_f0:2a:93	Aironet_b7:ab:5c	84:78:ac:f0:2a:93	802.11	2437	Reassociation Response, SN=2724, FN=0, Flag
5	0.843240	Aironet_b7:ab:5c	Broadcast	84:78:ac:f0:2a:93	802.11	2437	QoS Data, SN=2717, FN=0, Flags=.p.....TC
6	0.849798	Cisco_f5:4a:40	Aironet_b7:ab:5c	84:78:ac:f0:2a:93	802.11	2437	QoS Data, SN=66, FN=0, Flags=.p....F.C

Voici un résumé des débogages WLC pour cet événement d'itinérance :

<#root>

```
*apfMsConnTask_2: Jun 25 15:43:33.749: 00:40:96:b7:ab:5c
  CCKM: Received REASSOC REQ IE
*apfMsConnTask_2: Jun 25 15:43:33.749: 00:40:96:b7:ab:5c
  Reassociation received from mobile on BSSID
```

```
  84:78:ac:f0:2a:93
*apfMsConnTask_2: Jun 25 15:43:33.750: 00:40:96:b7:ab:5c
  Processing WPA IE type 221, length 22 for mobile
  00:40:96:b7:ab:5c
*apfMsConnTask_2: Jun 25 15:43:33.750: 00:40:96:b7:ab:5c
  CCKM: Mobile is using CCKM
```

!--- The Reassociation Request is received from the client, which provides the CCKM information needed in order to derive the new keys with a fast-secure roam.

```
*apfMsConnTask_2: Jun 25 15:43:33.750: 00:40:96:b7:ab:5c
  Setting active key cache index 0 ---> 8

*apfMsConnTask_2: Jun 25 15:43:33.750: 00:40:96:b7:ab:5c
  CCKM: Processing REASSOC REQ IE

*apfMsConnTask_2: Jun 25 15:43:33.750: 00:40:96:b7:ab:5c
  CCKM: using HMAC MD5 to compute MIC
```

!--- WLC computes the MIC used for this CCKM fast-roaming exchange.

```
*apfMsConnTask_2: Jun 25 15:43:33.750: 00:40:96:b7:ab:5c
  CCKM: Received a valid REASSOC REQ IE

*apfMsConnTask_2: Jun 25 15:43:33.751: 00:40:96:b7:ab:5c
  CCKM: Initializing PMK cache entry with a new PTK
```

!--- The new PTK is derived.

```
*apfMsConnTask_2: Jun 25 15:43:33.751: 00:40:96:b7:ab:5c
  Setting active key cache index 8 ---> 8

*apfMsConnTask_2: Jun 25 15:43:33.751: 00:40:96:b7:ab:5c
  Setting active key cache index 8 ---> 8

*apfMsConnTask_2: Jun 25 15:43:33.751: 00:40:96:b7:ab:5c
  Setting active key cache index 8 ---> 0

*apfMsConnTask_2: Jun 25 15:43:33.751: 00:40:96:b7:ab:5c
  Creating a PKC PMKID Cache entry for station
  00:40:96:b7:ab:5c (RSN 0) on BSSID 84:78:ac:f0:2a:93
```

!--- The new PMKID cache entry is created for this new AP-to-client association.

```
*apfMsConnTask_2: Jun 25 15:43:33.751: 00:40:96:b7:ab:5c
  CCKM: using HMAC MD5 to compute MIC
*apfMsConnTask_2: Jun 25 15:43:33.751: 00:40:96:b7:ab:5c
  Including CCKM Response IE (length 62) in Assoc Resp to mobile
*apfMsConnTask_2: Jun 25 15:43:33.751: 00:40:96:b7:ab:5c
  Sending Assoc Response to station on BSSID 84:78:ac:f0:2a:93
  (status 0) ApVapId 4 Slot 0
```

!--- The Reassociation Response is sent from the WLC/AP to the client, which includes the CCKM information required

```
     in order to confirm the new fast-roam and key derivation.


*dot1xMsgTask: Jun 25 15:43:33.757: 00:40:96:b7:ab:5c
  Skipping EAP-Success to mobile 00:40:96:b7:ab:5c

!--- EAP is skipped due to the fast roaming, and CCKM does not
     require further key handshakes. The client is now ready to
     pass encrypted data frames on the new AP.
```

Comme illustré, l'itinérance sécurisée rapide est effectuée tandis que les trames d'authentification EAP sont évitées et encore plus d'échanges en quatre étapes, car les nouvelles clés de cryptage sont toujours dérivées, mais basées sur le schéma de négociation CCKM. Ceci est complété avec les trames de réassociation d'itinérance et les informations précédemment mises en cache par le client et le WLC.

FlexConnect avec CCKM

- L'authentification centrale est prise en charge. Cela inclut la commutation de données locale et centrale. Les points d'accès doivent faire partie du même groupe FlexConnect.
- L'authentification locale flexible est prise en charge. En mode connecté, le cache peut être distribué du point d'accès au contrôleur, puis au reste des points d'accès du groupe FlexConnect.
- Le mode autonome est pris en charge. Si le cache est déjà présent sur l'AP (en raison de la distribution précédente), l'itinérance rapide doit fonctionner. La nouvelle authentification en mode autonome ne prend pas en charge l'itinérance rapide et sécurisée.

Avantages de CCKM

- CCKM est la méthode d'itinérance rapide et sécurisée la plus rapide, principalement déployée sur les WLAN d'entreprise. Les clients n'ont pas besoin de passer par une connexion de gestion de clé pour dériver de nouvelles clés lorsqu'un déplacement a lieu entre les points d'accès, et ne sont plus jamais obligés d'effectuer une authentification 802.1X/EAP complète avec de nouveaux points d'accès pendant la durée de vie du client sur ce WLAN.
- CCKM prend en charge toutes les méthodes de cryptage disponibles dans la norme 802.11 (WEP, TKIP et AES), en plus de certaines méthodes propriétaires Cisco traditionnelles encore utilisées sur les clients existants.

Contre avec CCKM

- CCKM est une méthode propriétaire de Cisco qui limite la mise en oeuvre et la prise en charge de l'infrastructure WLAN de Cisco et des clients sans fil CCX.
- CCX version 5 n'est pas largement adopté, de sorte que CCKM avec WPA2/AES n'est pas pris en charge par de nombreux clients sans fil CCX (principalement parce que la plupart d'entre eux prennent déjà en charge CCKM avec WPA/TKIP, qui est encore très sécurisé).

Itinérance rapide et sécurisée avec mise en cache PMKID / mise en cache des clés rémanentes

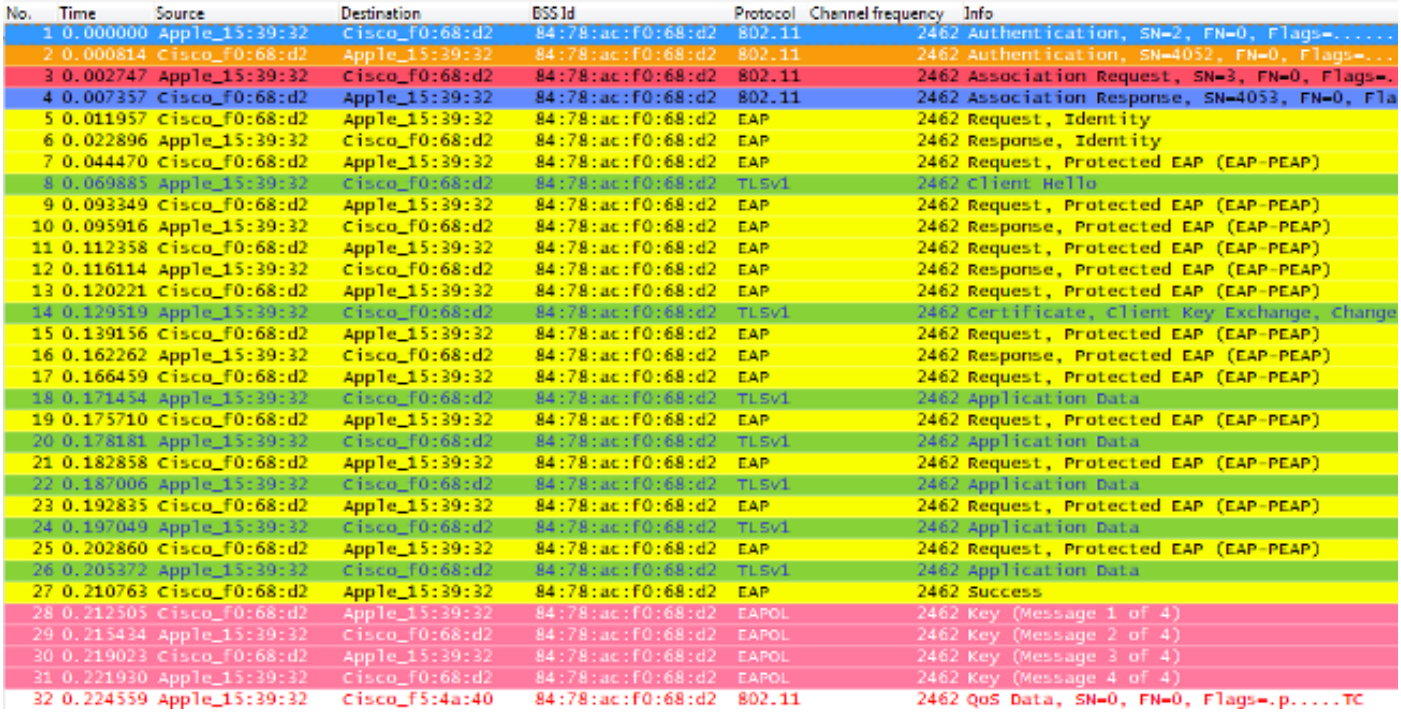
Pairwise think Key ID (PMKID) caching, ou Sticky Key Caching (SKC), est la première méthode d'itinérance sécurisée rapide suggérée par la norme IEEE 802.11 dans la modification de sécurité 802.11i, où le but principal est de standardiser un niveau élevé de sécurité pour les WLAN. Cette technique d'itinérance rapide et sécurisée a été ajoutée en tant que méthode facultative pour les périphériques WPA2 afin d'améliorer l'itinérance lors de la mise en oeuvre de cette sécurité.

Cela est possible parce que, chaque fois qu'un client est entièrement authentifié EAP, le client et le serveur d'authentification dérivent un MSK, qui est utilisé afin de dériver le PMK. Cette méthode est utilisée comme amorce pour la connexion en 4 étapes WPA2 afin de dériver la clé de chiffrement de monodiffusion finale (PTK) qui est utilisée pour la session (jusqu'à ce que le client se déplace vers un autre AP ou que la session expire) ; par conséquent, cette méthode empêche la phase d'authentification EAP lors de l'itinérance parce qu'elle réutilise la clé PMK d'origine mise en cache par le client et l'AP. Le client doit uniquement passer par la connexion en 4 étapes WPA2 pour dériver de nouvelles clés de chiffrement.

Cette méthode n'est pas largement déployée comme la méthode d'itinérance 802.11 standard Fast-Secure recommandée, principalement pour les raisons suivantes :

- Cette méthode est facultative et n'est pas prise en charge par tous les périphériques WPA2, car l'objectif de la modification 802.11i ne concerne pas l'itinérance sécurisée rapide, et l'IEEE a déjà travaillé sur une autre modification visant à normaliser l'itinérance sécurisée rapide pour les réseaux locaux sans fil (802.11r, qui est traitée plus loin dans ce document).
- Cette méthode a une grande limitation dans sa mise en oeuvre : les clients sans fil peuvent seulement effectuer une itinérance rapide et sécurisée lorsqu'ils reviennent à un point d'accès où ils étaient précédemment authentifiés/connectés.

Avec cette méthode, l'association initiale à n'importe quel point d'accès est semblable à une première authentification régulière au WLAN, où l'authentification 802.1X/EAP complète par rapport au serveur d'authentification et la connexion en 4 étapes pour la génération de clé doivent avoir lieu avant que le client puisse envoyer des trames de données, comme illustré dans cette image d'écran :



No.	Time	Source	Destination	BSS Id	Protocol	Channel frequency	Info
1	0.000000	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	802.11	2462	Authentication, SN=2, FN=0, Flags=.....
2	0.000814	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	802.11	2462	Authentication, SN=4052, FN=0, Flags=...
3	0.002747	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	802.11	2462	Association Request, SN=3, FN=0, Flags=.
4	0.007357	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	802.11	2462	Association Response, SN=4053, FN=0, Fla
5	0.011957	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAP	2462	Request, Identity
6	0.022896	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAP	2462	Response, Identity
7	0.044470	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
8	0.069885	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Client Hello
9	0.093349	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
10	0.095916	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAP	2462	Response, Protected EAP (EAP-PEAP)
11	0.112358	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
12	0.116114	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAP	2462	Response, Protected EAP (EAP-PEAP)
13	0.120221	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
14	0.129519	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Certificate, Client Key Exchange, Change
15	0.139156	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
16	0.162262	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAP	2462	Response, Protected EAP (EAP-PEAP)
17	0.166459	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
18	0.171454	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Application Data
19	0.175710	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
20	0.178181	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Application Data
21	0.182858	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
22	0.187006	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Application Data
23	0.192835	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
24	0.197049	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Application Data
25	0.202860	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
26	0.205372	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Application Data
27	0.210763	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAP	2462	Success
28	0.212505	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAPOL	2462	Key (Message 1 of 4)
29	0.215434	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAPOL	2462	Key (Message 2 of 4)
30	0.219023	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAPOL	2462	Key (Message 3 of 4)
31	0.221930	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAPOL	2462	Key (Message 4 of 4)
32	0.224559	Apple_15:39:32	Cisco_f5:4a:40	84:78:ac:f0:68:d2	802.11	2462	QoS Data, SN=0, FN=0, Flags=.p.....TC

Les débogages révèlent le même échange de trames d'authentification EAP que les autres méthodes lors de l'authentification initiale sur le WLAN, avec quelques sorties ajoutées en ce qui concerne les techniques de mise en cache des clés utilisées ici. Ces sorties de débogage sont coupées afin d'afficher principalement les nouvelles informations, pas l'échange de trame EAP entier, parce que fondamentalement les mêmes informations sont échangées à chaque fois pour l'authentification du client par rapport au serveur d'authentification. Ceci est démontré jusqu'à présent, et corrélé avec les trames d'authentification EAP montrées dans les images de paquets, de sorte que la plupart des messages EAP sont supprimés des sorties de débogage pour simplifier :

<#root>

```
*apfMsConnTask_0: Jun 22 00:23:15.097: ec:85:2f:15:39:32
  Association received from mobile on BSSID 84:78:ac:f0:68:d2

!--- This is the Association Request from the client.



*apfMsConnTask_0: Jun 22 00:23:15.098: ec:85:2f:15:39:32
  Processing RSN IE type 48, length 20 for mobile ec:85:2f:15:39:32

!--- The WLC/AP finds an Information Element that claims PMKID
     Caching support on the Association request that is sent
     from the client.



*apfMsConnTask_0: Jun 22 00:23:15.098: ec:85:2f:15:39:32
  Received RSN IE with 0 PMKIDs from mobile ec:85:2f:15:39:32

!--- Since this is an initial association, the Association
     Request comes without any PMKID.
```

```
*apfMsConnTask_0: Jun 22 00:23:15.098: ec:85:2f:15:39:32
  Setting active key cache index 8 ---> 8

*apfMsConnTask_0: Jun 22 00:23:15.099: ec:85:2f:15:39:32
  Sending Assoc Response to station on BSSID 84:78:ac:f0:68:d2
  (status 0) ApVapId 3 Slot 0
```

!--- The Association Response is sent to the client.

```
*dot1xMsgTask: Jun 22 00:23:15.103: ec:85:2f:15:39:32
  Sending EAP-Request/Identity to mobile ec:85:2f:15:39:32
  (EAP Id 1)

*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.118: ec:85:2f:15:39:32
  Received EAPOL EAPPKT from mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.118: ec:85:2f:15:39:32
  Received Identity Response (count=1) from mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.126: ec:85:2f:15:39:32
  Processing Access-Challenge for mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.126: ec:85:2f:15:39:32
  Sending EAP Request from AAA to mobile ec:85:2f:15:39:32
  (EAP Id 2)

*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.146: ec:85:2f:15:39:32
  Received EAPOL EAPPKT from mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.146: ec:85:2f:15:39:32
  Received EAP Response from mobile ec:85:2f:15:39:32
  (EAP Id 2, EAP Type 25)

*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.274: ec:85:2f:15:39:32
  Processing Access-Accept for mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.274: ec:85:2f:15:39:32
  Creating a PKC PMKID Cache entry for station ec:85:2f:15:39:32
  (RSN 2)
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.274: ec:85:2f:15:39:32
  Setting active key cache index 8 ---> 8
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.274: ec:85:2f:15:39:32
  Setting active key cache index 8 ---> 0
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.274: ec:85:2f:15:39:32
  Adding BSSID 84:78:ac:f0:68:d2 to PMKID cache at index 0
  for station ec:85:2f:15:39:32
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.274:
  New PMKID: (16)
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.274:
  [0000] c9 4d 0d 97 03 aa a9 0f 1b c8 33 73 01 f1 18 f5
```

!--- WLC creates a PMK cache entry for this client, which is used for SKC in this case, so the PMKID is computed with the AP MAC address (BSSID 84:78:ac:f0:68:d2).

```
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.274: ec:85:2f:15:39:32
  Sending EAP-Success to mobile ec:85:2f:15:39:32
```

```
  (EAP Id 12)

*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.275:
  Including PMKID in M1  (16)
```

!--- The hashed PMKID is included on the Message-1 of the WPA/WPA2 4-Way handshake.

```
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.275:
  [0000] c9 4d 0d 97 03 aa a9 0f 1b c8 33 73 01 f1 18 f5
```

!--- This is the hashed PMKID.

```
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.275: ec:85:2f:15:39:32
  Sending EAPOL-Key Message to mobile ec:85:2f:15:39:32
  state INITPMK (message 1), replay counter
  00.00.00.00.00.00.00.00
```

!--- Message-1 of the WPA/WPA2 4-Way handshake is sent from the WLC/AP to the client.

```
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.284: ec:85:2f:15:39:32
  Received EAPOL-Key from mobile ec:85:2f:15:39:32
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.284: ec:85:2f:15:39:32
  Received EAPOL-key in PTK_START state (message 2) from mobile
  ec:85:2f:15:39:32
```

!--- Message-2 of the WPA/WPA-2 4-Way handshake is successfully received from the client.

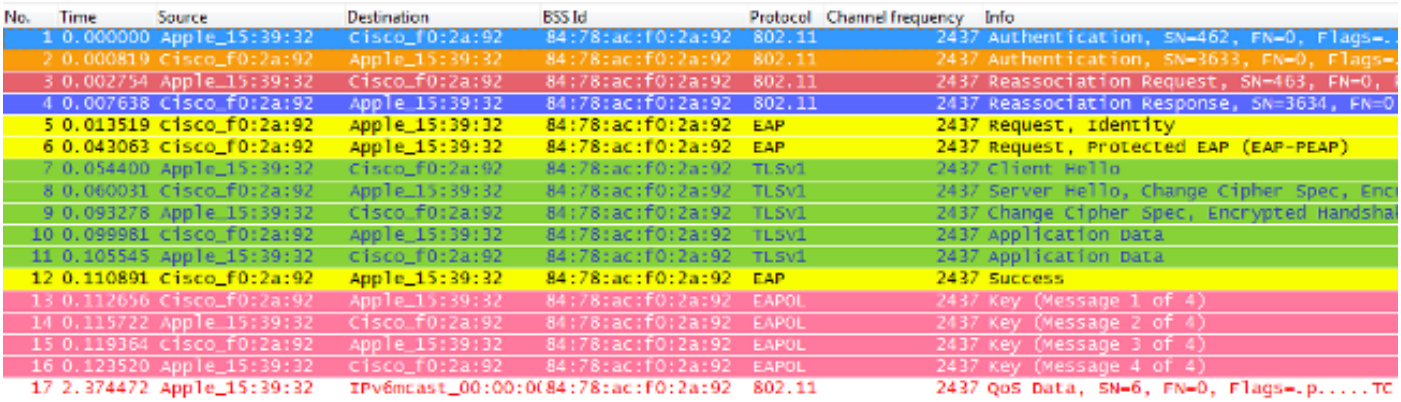
```
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.284: ec:85:2f:15:39:32
  PMK: Sending cache add

*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.285: ec:85:2f:15:39:32
  Sending EAPOL-Key Message to mobile ec:85:2f:15:39:32
  state PTKINITNEGOTIATING (message 3), replay counter
  00.00.00.00.00.00.00.01
```

!--- Message-3 of the WPA/WPA2 4-Way handshake is sent from the WLC/AP to the client.

```
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.291: ec:85:2f:15:39:32
  Received EAPOL-Key from mobile ec:85:2f:15:39:32
*Dot1x_NW_MsgTask_2: Jun 22 00:23:15.291: ec:85:2f:15:39:32
  Received EAPOL-key in PTKINITNEGOTIATING state (message 4)
  from mobile ec:85:2f:15:39:32
```

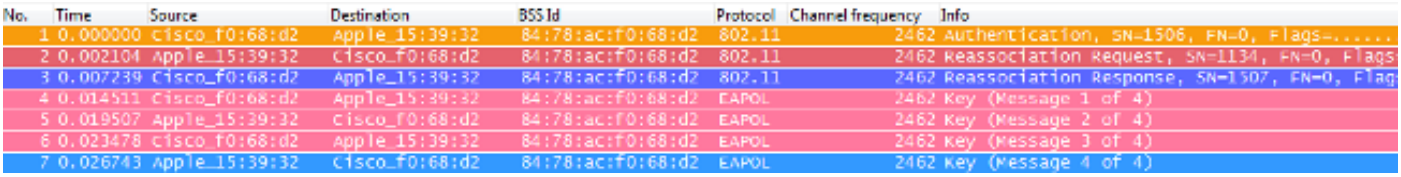
!--- Message-4 (final message) of this initial WPA/WPA2 4-Way handshake is successfully received from the client, which confirms the installation of the derived keys. They can now be used in order to encrypt data frames with the current AP.

Avec cette méthode, le point d'accès et le client sans fil mettent en cache les PMK des associations sécurisées déjà établies. Par conséquent, si le client sans fil se déplace vers un nouveau point d'accès auquel il n'a jamais été associé, le client doit à nouveau effectuer une authentification EAP complète, comme illustré dans cette image où le client se déplace vers un nouveau point d'accès :



No.	Time	Source	Destination	BSS Id	Protocol	Channel frequency	Info
1	0.000000	Apple_15:39:32	Cisco_f0:2a:92	84:78:ac:f0:2a:92	802.11	2437	Authentication, SN=462, FN=0, Flags=.
2	0.000819	Cisco_f0:2a:92	Apple_15:39:32	84:78:ac:f0:2a:92	802.11	2437	Authentication, SN=3633, FN=0, Flags=
3	0.002754	Apple_15:39:32	Cisco_f0:2a:92	84:78:ac:f0:2a:92	802.11	2437	Reassociation Request, SN=463, FN=0,
4	0.007638	Cisco_f0:2a:92	Apple_15:39:32	84:78:ac:f0:2a:92	802.11	2437	Reassociation Response, SN=3634, FN=0
5	0.013519	Cisco_f0:2a:92	Apple_15:39:32	84:78:ac:f0:2a:92	EAP	2437	Request, Identity
6	0.043063	Cisco_f0:2a:92	Apple_15:39:32	84:78:ac:f0:2a:92	EAP	2437	Request, Protected EAP (EAP-PEAP)
7	0.054400	Apple_15:39:32	Cisco_f0:2a:92	84:78:ac:f0:2a:92	TLSv1	2437	Client Hello
8	0.060031	Cisco_f0:2a:92	Apple_15:39:32	84:78:ac:f0:2a:92	TLSv1	2437	Server Hello, Change Cipher Spec, Enc
9	0.093278	Apple_15:39:32	Cisco_f0:2a:92	84:78:ac:f0:2a:92	TLSv1	2437	Change Cipher Spec, Encrypted Handsha
10	0.099981	Cisco_f0:2a:92	Apple_15:39:32	84:78:ac:f0:2a:92	TLSv1	2437	Application Data
11	0.105545	Apple_15:39:32	Cisco_f0:2a:92	84:78:ac:f0:2a:92	TLSv1	2437	Application Data
12	0.110891	Cisco_f0:2a:92	Apple_15:39:32	84:78:ac:f0:2a:92	EAP	2437	Success
13	0.112656	Cisco_f0:2a:92	Apple_15:39:32	84:78:ac:f0:2a:92	EAPOL	2437	Key (Message 1 of 4)
14	0.115722	Apple_15:39:32	Cisco_f0:2a:92	84:78:ac:f0:2a:92	EAPOL	2437	Key (Message 2 of 4)
15	0.119364	Cisco_f0:2a:92	Apple_15:39:32	84:78:ac:f0:2a:92	EAPOL	2437	Key (Message 3 of 4)
16	0.123520	Apple_15:39:32	Cisco_f0:2a:92	84:78:ac:f0:2a:92	EAPOL	2437	Key (Message 4 of 4)
17	2.374472	Apple_15:39:32	IPv6mcast_00:00:0	84:78:ac:f0:2a:92	802.11	2437	QoS Data, SN=6, FN=0, Flags=.p.....TC

Cependant, si le client sans fil revient vers un point d'accès où une association/authentification précédente a eu lieu, alors le client envoie une trame de demande de réassociation qui répertorie plusieurs PMKID, qui informe le point d'accès des PMK mis en cache à partir de tous les points d'accès où le client a précédemment authentifié. Par conséquent, puisque le client est en itinérance vers un AP qui a également un PMK mis en cache pour ce client, alors le client n'a pas besoin de ré-authentifier par EAP afin de dériver un nouveau PMK. Le client passe simplement par la connexion en 4 étapes WPA2 afin de dériver les nouvelles clés de chiffrement transitoires :



No.	Time	Source	Destination	BSS Id	Protocol	Channel frequency	Info
1	0.000000	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	802.11	2462	Authentication, SN=1506, FN=0, Flags=.....
2	0.002104	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	802.11	2462	Reassociation Request, SN=1134, FN=0, Flags
3	0.007239	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	802.11	2462	Reassociation Response, SN=1507, FN=0, Flag
4	0.014511	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAPOL	2462	Key (Message 1 of 4)
5	0.019507	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAPOL	2462	Key (Message 2 of 4)
6	0.023478	Cisco_f0:68:d2	Apple_15:39:32	84:78:ac:f0:68:d2	EAPOL	2462	Key (Message 3 of 4)
7	0.026743	Apple_15:39:32	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAPOL	2462	Key (Message 4 of 4)

Remarque : cette image n'affiche pas la première trame d'authentification 802.11 Open System du client, mais cela n'est pas dû à la méthode implémentée, car cette trame est toujours requise. La raison en est que cette trame spécifique n'est pas imagée par l'adaptateur ou le logiciel d'image de paquets sans fil utilisé afin de renifler les trames en direct pour cet exemple, mais elle est laissée comme ceci sur l'exemple à des fins éducatives. Sachez qu'il est possible que cela se produise lorsque vous exécutez des images de paquets en direct ; certaines trames peuvent être manquées par l'image, mais sont en fait échangées entre le client et le point d'accès. Sinon, l'itinérance ne commence jamais dans cet exemple.

Voici un résumé des débogages WLC pour cette méthode d'itinérance rapide et sécurisée :

```
*apfMsConnTask_0: Jun 22 00:26:40.787: ec:85:2f:15:39:32
  Reassociation received from mobile on BSSID
  84:78:ac:f0:68:d2
```

!--- This is the Reassociation Request from the client.

```
*apfMsConnTask_0: Jun 22 00:26:40.787: ec:85:2f:15:39:32
  Processing RSN IE type 48, length 38 for mobile
  ec:85:2f:15:39:32
```

**!--- The WLC/AP finds an Information Element that claims PMKID
     Caching support on the Association request that is sent
     from the client.**

```
*apfMsConnTask_0: Jun 22 00:26:40.787: ec:85:2f:15:39:32
  Received RSN IE with 1 PMKIDs from mobile
  ec:85:2f:15:39:32
```

**!--- The Reassociation Request from the client comes with
     one PMKID.**

```
*apfMsConnTask_0: Jun 22 00:26:40.787:
  Received PMKID:  (16)
*apfMsConnTask_0: Jun 22 00:26:40.788:
  [0000] c9 4d 0d 97 03 aa a9 0f 1b c8 33 73 01 f1 18 f5
```

!--- This is the PMKID that is received.

```
*apfMsConnTask_0: Jun 22 00:26:40.788: ec:85:2f:15:39:32
  Searching for PMKID in MSCB PMKID cache for mobile
  ec:85:2f:15:39:32
```

!--- WLC searches for a matching PMKID on the database.

```
*apfMsConnTask_0: Jun 22 00:26:40.788: ec:85:2f:15:39:32
  Found an cache entry for BSSID 84:78:ac:f0:68:d2 in
  PMKID cache at index 0 of station ec:85:2f:15:39:32

*apfMsConnTask_0: Jun 22 00:26:40.788: ec:85:2f:15:39:32
  Found a valid PMKID in the MSCB PMKID cache for mobile
  ec:85:2f:15:39:32
```

**!--- The WLC validates the PMKID provided by the client,
     and confirms that it has a valid PMK cache for this
     client-and-AP pair.**

```
*apfMsConnTask_0: Jun 22 00:26:40.788: ec:85:2f:15:39:32
  Setting active key cache index 1 ---> 0

*apfMsConnTask_0: Jun 22 00:26:40.788: ec:85:2f:15:39:32
  Sending Assoc Response to station on BSSID
  84:78:ac:f0:68:d2(status 0) ApVapId 3 Slot 0
```

!--- The Reassociation Response is sent to the client, which

validates the fast-roam with SKC.

```
*dot1xMsgTask: Jun 22 00:26:40.795: ec:85:2f:15:39:32
  Initiating RSN with existing PMK to mobile
  ec:85:2f:15:39:32
```

!--- WLC initiates a Robust Secure Network association with this client-and-AP pair based on the cached PMK found. Hence, EAP is avoided as per the next message.

```
*dot1xMsgTask: Jun 22 00:26:40.795: ec:85:2f:15:39:32
  Skipping EAP-Success to mobile ec:85:2f:15:39:32

*dot1xMsgTask: Jun 22 00:26:40.795: ec:85:2f:15:39:32
  Found an cache entry for BSSID 84:78:ac:f0:68:d2 in
  PMKID cache at index 0 of station ec:85:2f:15:39:32

*dot1xMsgTask: Jun 22 00:26:40.795: Including PMKID in M1(16)
```

!--- The hashed PMKID is included on the Message-1 of the WPA/WPA2 4-Way handshake.

```
*dot1xMsgTask: Jun 22 00:26:40.795:
  [0000] c9 4d 0d 97 03 aa a9 0f 1b c8 33 73 01 f1 18 f5
```

!--- The PMKID is hashed. The next messages are the same WPA/WPA2 4-Way handshake messages described thus far that are used in order to finish the encryption keys generation/installation.

```
*dot1xMsgTask: Jun 22 00:26:40.795: ec:85:2f:15:39:32
  Sending EAPOL-Key Message to mobile ec:85:2f:15:39:32 state
  INITPMK (message 1), replay counter 00.00.00.00.00.00.00.00

*Dot1x_NW_MsgTask_2: Jun 22 00:26:40.811: ec:85:2f:15:39:32
  Received EAPOL-Key from mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 22 00:26:40.812: ec:85:2f:15:39:32
  Received EAPOL-key in PTK_START state (message 2) from mobile
  ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 22 00:26:40.812: ec:85:2f:15:39:32
  PMK: Sending cache add

*Dot1x_NW_MsgTask_2: Jun 22 00:26:40.812: ec:85:2f:15:39:32
  Sending EAPOL-Key Message to mobile ec:85:2f:15:39:32 state
  PTKINITNEGOTIATING (message 3), replay counter
  00.00.00.00.00.00.00.01

*Dot1x_NW_MsgTask_2: Jun 22 00:26:40.820: ec:85:2f:15:39:32
  Received EAPOL-Key from mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 22 00:26:40.820: ec:85:2f:15:39:32
  Received EAPOL-key in PTKINITNEGOTIATING state (message 4)
  from mobile ec:85:2f:15:39:32
```

FlexConnect avec mise en cache PMKID / mise en cache des clés rémanentes

- Lorsque vous utilisez cette méthode sur une configuration FlexConnect, elle peut fonctionner et le comportement peut sembler similaire à ce qui a été expliqué précédemment si vous utilisez l'authentification centrale de nouveau au WLC (avec la commutation centrale ou locale) ; cependant, cette méthode SKC n'est pas prise en charge sur FlexConnect.
- Cette méthode n'est officiellement prise en charge que sur CUWN avec des points d'accès en mode local, pas sur FlexConnect ou d'autres modes.

Avantages de la mise en cache PMKID / Sticky Key Caching

Cette méthode peut être mise en oeuvre localement par des AP autonomes-indépendants, sans avoir besoin d'un dispositif centralisé pour gérer les clés mises en cache.

Inconvénients avec la mise en cache PMKID / Sticky Key Caching

- Comme mentionné précédemment dans ce document, la principale limitation de cette méthode est que le client peut seulement effectuer une itinérance rapide et sécurisée lors de l'itinérance de retour à un AP où il était précédemment associé/authentifié. En cas d'itinérance vers un nouveau point d'accès, le client doit effectuer à nouveau l'authentification EAP complète.
- Le client sans fil et les points d'accès doivent se souvenir de toutes les clés PMK dérivées de chaque nouvelle authentification, de sorte que cette fonctionnalité est normalement limitée à une certaine quantité de clés PMK qui sont mises en cache. Cette limite n'étant pas clairement définie par la norme, les fournisseurs peuvent définir différentes limites sur leurs implémentations de SKC. Par exemple, les contrôleurs WLAN Cisco peuvent actuellement mettre en cache les PMK d'un client pour un maximum de huit AP. Si un client se déplace vers plus de huit points d'accès par session, les points d'accès les plus anciens sont supprimés de la liste de cache afin de stocker les entrées nouvellement mises en cache.
- Cette méthode est facultative et n'est toujours pas prise en charge par de nombreux périphériques WPA2 ; par conséquent, elle n'est pas largement adoptée et déployée.
- SKC n'est pas pris en charge lorsque vous effectuez l'itinérance intercontrôleur, ce qui se produit lorsque vous passez entre des AP gérés par différents WLC, même s'ils sont sur le même groupe de mobilité.

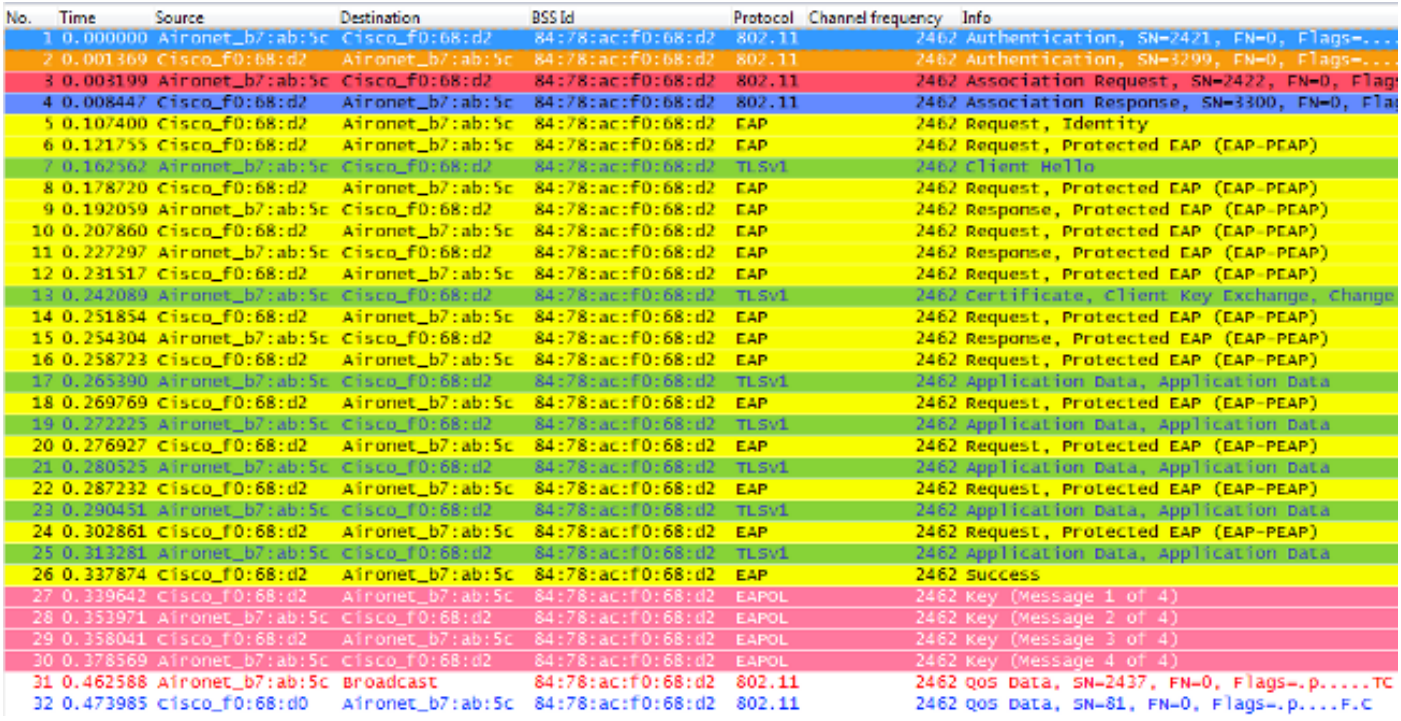
Itinérance rapide et sécurisée avec mise en cache des clés opportuniste

La mise en cache de clés opportuniste (OKC), également connue sous le nom de mise en cache de clés proactive (PKC) (ce terme est expliqué plus en détail dans une note qui suit), est fondamentalement une amélioration de la méthode de mise en cache WPA2 PMKID décrite précédemment, c'est pourquoi elle est également appelée mise en cache proactive/opportuniste PMKID. Par conséquent, il est important de noter qu'il ne s'agit pas d'une méthode d'itinérance

rapide et sécurisée définie par la norme 802.11 et qu'elle n'est pas prise en charge par de nombreux périphériques, mais tout comme la mise en cache PMKID, elle fonctionne avec WPA2-EAP.

Cette technique permet au client sans fil et à l'infrastructure WLAN de ne mettre en cache qu'une seule clé PMK pendant la durée de vie de l'association du client avec ce WLAN (dérivée de la clé MSK après l'authentification 802.1X/EAP initiale avec le serveur d'authentification), même en cas d'itinérance entre plusieurs points d'accès, car ils partagent tous la clé PMK d'origine utilisée comme valeur initiale sur toutes les connexions WPA2 à quatre voies. Cela reste nécessaire, tout comme dans SKC, afin de générer de nouvelles clés de chiffrement chaque fois que le client se réassocie aux AP. Pour que les AP puissent partager cette PMK originale de la session client, ils doivent tous être sous une sorte de contrôle administratif, avec un périphérique centralisé qui met en cache et distribue la PMK originale pour tous les AP. Ceci est similaire au CUWN, où le WLC effectue ce travail pour tous les LAP sous son contrôle, et utilise les groupes de mobilité afin de gérer ce PMK entre plusieurs WLC ; par conséquent, c'est une limitation sur les environnements d'AP autonomes.

Avec cette méthode, tout comme dans la mise en cache PMKID (SKC), l'association initiale à n'importe quel point d'accès est une première authentification régulière au WLAN, où vous devez effectuer l'authentification 802.1X/EAP complète sur le serveur d'authentification et la connexion en 4 étapes pour la génération de clé avant de pouvoir envoyer des trames de données. Voici une image d'écran qui illustre ceci :



No.	Time	Source	Destination	BSS Id	Protocol	Channel frequency	Info
1	0.000000	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	802.11	2462	Authentication, SN=2421, FN=0, Flags=...
2	0.001369	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	802.11	2462	Authentication, SN=3299, FN=0, Flags=...
3	0.003199	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	802.11	2462	Association Request, SN=2422, FN=0, Flag
4	0.008447	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	802.11	2462	Association Response, SN=3300, FN=0, Flag
5	0.107400	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAP	2462	Request, Identity
6	0.121755	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
7	0.162562	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Client Hello
8	0.178720	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
9	0.192059	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAP	2462	Response, Protected EAP (EAP-PEAP)
10	0.207860	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
11	0.227297	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAP	2462	Response, Protected EAP (EAP-PEAP)
12	0.231517	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
13	0.242089	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Certificate, Client Key Exchange, Change
14	0.251854	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
15	0.254304	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAP	2462	Response, Protected EAP (EAP-PEAP)
16	0.258723	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
17	0.265390	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Application Data, Application Data
18	0.269769	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
19	0.272225	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Application Data, Application Data
20	0.276927	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
21	0.280525	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Application Data, Application Data
22	0.287232	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
23	0.290451	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Application Data, Application Data
24	0.302861	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAP	2462	Request, Protected EAP (EAP-PEAP)
25	0.313281	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	TLSv1	2462	Application Data, Application Data
26	0.337874	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAP	2462	Success
27	0.339642	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAPOL	2462	Key (Message 1 of 4)
28	0.353971	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAPOL	2462	Key (Message 2 of 4)
29	0.358041	Cisco_f0:68:d2	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	EAPOL	2462	Key (Message 3 of 4)
30	0.378569	Aironet_b7:ab:5c	Cisco_f0:68:d2	84:78:ac:f0:68:d2	EAPOL	2462	Key (Message 4 of 4)
31	0.462588	Aironet_b7:ab:5c	Broadcast	84:78:ac:f0:68:d2	802.11	2462	QoS Data, SN=2437, FN=0, Flags=.p.....TC
32	0.473985	Cisco_f0:68:d0	Aironet_b7:ab:5c	84:78:ac:f0:68:d2	802.11	2462	QoS Data, SN=81, FN=0, Flags=.p....F.C

Les sorties de débogage montrent essentiellement le même échange de trames d'authentification EAP que les autres méthodes décrites dans ce document lors de l'authentification initiale au WLAN (comme illustré dans les images), avec l'ajout de certaines sorties qui concernent les techniques de mise en cache des clés utilisées par le WLC ici. Cette sortie de débogage est également coupée afin d'afficher uniquement les informations pertinentes :

<#root>

```
*apfMsConnTask_0: Jun 21 21:46:06.515: 00:40:96:b7:ab:5c
  Association received from mobile on BSSID
  84:78:ac:f0:68:d2

!--- This is the Association Request from the client.



*apfMsConnTask_0: Jun 21 21:46:06.516: 00:40:96:b7:ab:5c
  Processing RSN IE type 48, length 20 for mobile
  00:40:96:b7:ab:5c

!--- The WLC/AP finds an Information Element that claims
     PMKID Caching support on the Association request that
     is sent from the client.



*apfMsConnTask_0: Jun 21 21:46:06.516: 00:40:96:b7:ab:5c
  Received RSN IE with 0 PMKIDs from mobile
  00:40:96:b7:ab:5c

!--- Since this is an initial association, the Association
     Request comes without any PMKID.



*apfMsConnTask_0: Jun 21 21:46:06.516: 00:40:96:b7:ab:5c
  Setting active key cache index 0 ---> 8

*apfMsConnTask_0: Jun 21 21:46:06.516: 00:40:96:b7:ab:5c
  Sending Assoc Response to station on BSSID
  84:78:ac:f0:68:d2 (status 0) ApVapId 3 Slot

!--- The Association Response is sent to the client.



*dot1xMsgTask: Jun 21 21:46:06.522: 00:40:96:b7:ab:5
  Sending EAP-Request/Identity to mobile 00:40:96:b7:ab:5c
  (EAP Id 1)

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.614: 00:40:96:b7:ab:5c
  Received EAPOL START from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.614: 00:40:96:b7:ab:5c
  Sending EAP-Request/Identity to mobile 00:40:96:b7:ab:5c
  (EAP Id 2)

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.623: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.623: 00:40:96:b7:ab:5c
  Received Identity Response (count=2) from mobile
  00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.630: 00:40:96:b7:ab:5c
  Processing Access-Challenge for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.630: 00:40:96:b7:ab:5c
  Sending EAP Request from AAA to mobile 00:40:96:b7:ab:5c
  (EAP Id 3)
```

```
*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.673: 00:40:96:b7:ab:5c
  Received EAPOL EAPPKT from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.673: 00:40:96:b7:ab:5c
  Received EAP Response from mobile 00:40:96:b7:ab:5c
  (EAP Id 3, EAP Type 25)

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.843: 00:40:96:b7:ab:5c
  Processing Access-Accept for mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.844: 00:40:96:b7:ab:5c
  Creating a PKC PMKID Cache entry for station
  00:40:96:b7:ab:5c (RSN 2)
*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.844: 00:40:96:b7:ab:5c
  Setting active key cache index 8 ---> 8
*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.844: 00:40:96:b7:ab:5c
  Setting active key cache index 8 ---> 0
*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.844: 00:40:96:b7:ab:5c
  Adding BSSID 84:78:ac:f0:68:d2 to PMKID cache at index 0
  for station 00:40:96:b7:ab:5
*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.844: New PMKID: (16)
*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.844:
  [0000] 4e a1 7f 5a 75 48 9c f9 96 e3 a8 71 25 6f 11 d0
```

!--- WLC creates a PMK cache entry for this client, which is used for OKC in this case, so the PMKID is computed with the AP MAC address (BSSID 84:78:ac:f0:68:d2).

```
*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.844: 00:40:96:b7:ab:5c
  PMK sent to mobility group
```

!--- The PMK cache entry for this client is shared with the WLCs on the mobility group.

```
*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.844: 00:40:96:b7:ab:5c
  Sending EAP-Success to mobile 00:40:96:b7:ab:5c (EAP Id 13)

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.844: 00:40:96:b7:ab:5c
  Found an cache entry for BSSID 84:78:ac:f0:68:d2 in PMKID
  cache at index 0 of station 00:40:96:b7:ab:5

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.844: Including PMKID
  in M1  (16)
```

!--- The hashed PMKID is included on the Message-1 of the WPA/WPA2 4-Way handshake.

```
*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.844:
  [0000] 4e a1 7f 5a 75 48 9c f9 96 e3 a8 71 25 6f 11 d0
```

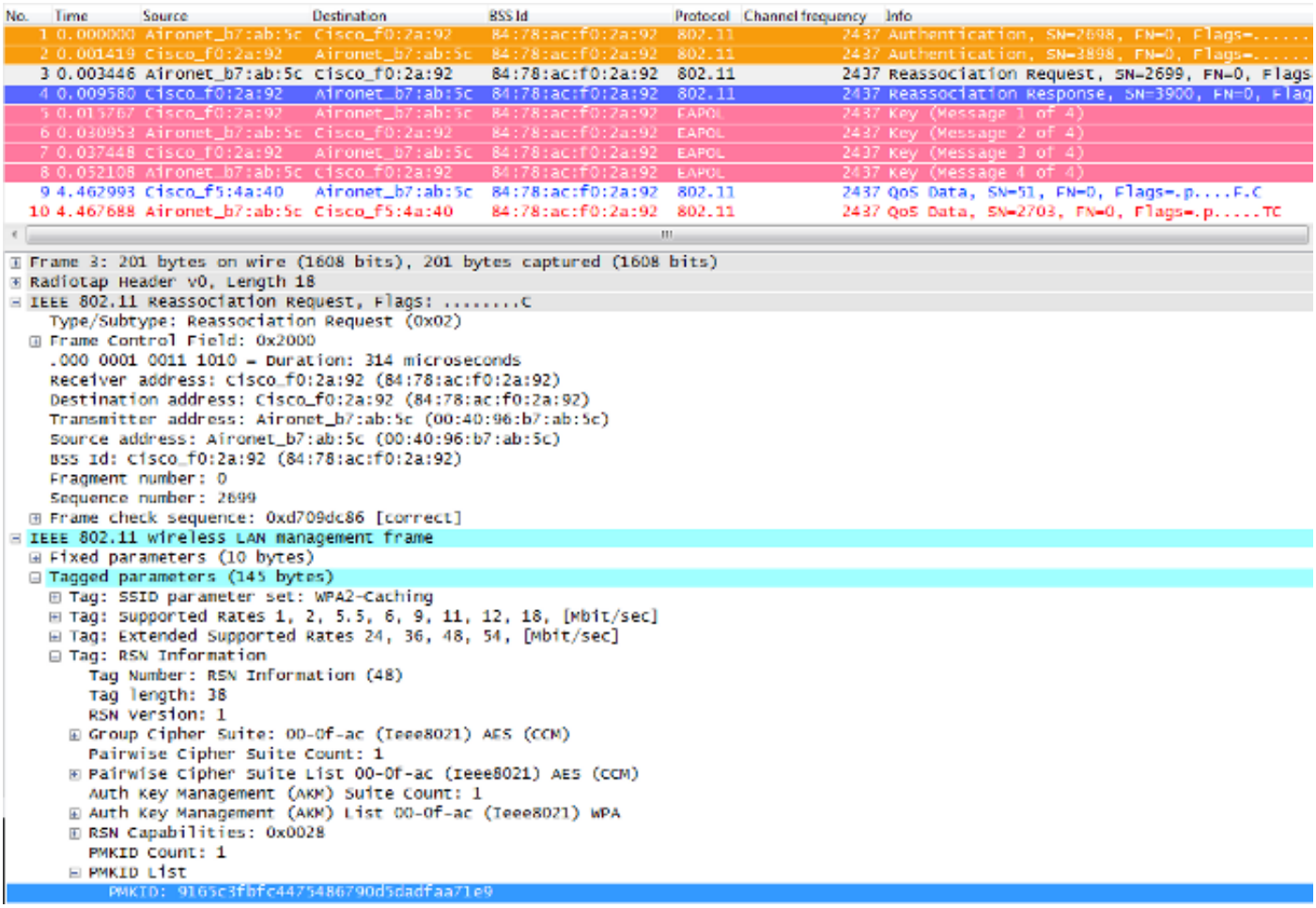
!--- This is the hashed PMKID. The next messages are the same WPA/WPA2 4-Way handshake messages described thus far that are used in order to finish the encryption keys generation/installation.

```
*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.844: 00:40:96:b7:ab:5c
```

```
  Sending EAPOL-Key Message to mobile 00:40:96:b7:ab:5c state
  INITPMK (message 1), replay counter 00.00.00.00.00.00.00.00

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.865: 00:40:96:b7:ab:5c
  Received EAPOL-Key from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.865: 00:40:96:b7:ab:5c
  Received EAPOL-key in PTK_START state (message 2)
  from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.865: 00:40:96:b7:ab:5c
  PMK: Sending cache add

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.865: 00:40:96:b7:ab:5c
  Sending EAPOL-Key Message to mobile 00:40:96:b7:ab:5c state
  PTKINITNEGOTIATING (message 3), replay counter
  00.00.00.00.00.00.00.01

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.889: 00:40:96:b7:ab:5c
  Received EAPOL-Key from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 21:46:06.890: 00:40:96:b7:ab:5c
  Received EAPOL-key in PTKINITNEGOTIATING state (message 4)
  from mobile 00:40:96:b7:ab:5c
```

Avec cette méthode, le client sans fil et le WLC (pour tous les AP gérés) mettent en cache le PMK d'origine de l'association sécurisée initialement établie. En fait, chaque fois que le client sans fil se connecte à un point d'accès spécifique, un PMKID est haché en fonction de : l'adresse MAC du client, l'adresse MAC du point d'accès (BSSID du WLAN), et le PMK dérivé avec ce point d'accès. Par conséquent, étant donné qu'OKC met en cache la même PMK d'origine pour tous les AP et le client spécifique, quand ce client (re)associe à un autre AP, la seule valeur qui change afin de hacher le nouveau PMKID est la nouvelle adresse MAC d'AP.

Lorsque le client lance l'itinérance vers un nouveau point d'accès et envoie la trame de demande de réassociation, il ajoute le PMKID sur l'élément d'information RSN WPA2 s'il veut informer le point d'accès qu'un PMK mis en cache est utilisé pour l'itinérance sécurisée rapide. Il connaît déjà l'adresse MAC du BSSID (AP) pour lequel il se déplace, puis le client hache simplement le nouveau PMKID qui est utilisé sur cette demande de réassociation. Lorsque le point d'accès reçoit cette demande du client, il hache également le PMKID avec les valeurs qu'il a déjà (le PMK mis en cache, l'adresse MAC du client et sa propre adresse MAC de point d'accès), et répond avec la réponse de réassociation réussie qui confirme que les PMKID correspondent. La clé PMK mise en cache peut être utilisée comme valeur initiale pour lancer une connexion en quatre étapes WPA2 afin de dériver les nouvelles clés de cryptage (et ignorer EAP) :



Dans cette image, le cadre Demande de réassociation du client est sélectionné et développé afin que vous puissiez voir plus de détails sur le cadre. Les informations d'adresse MAC ainsi que l'élément d'information Robuste Security Network (RSN, conformément à la norme 802.11i - WPA2), où les informations sur les paramètres WPA2 utilisés pour cette association sont affichées (en surbrillance est le PMKID obtenu à partir de la formule hachée).

Voici un résumé des débogages WLC pour cette méthode d'itinérance rapide et sécurisée avec OKC :

<#root>

```
*apfMsConnTask_2: Jun 21 21:48:50.562: 00:40:96:b7:ab:5c
  Reassociation received from mobile on BSSID
  84:78:ac:f0:2a:92
```

!--- This is the Reassociation Request from the client.

```
*apfMsConnTask_2: Jun 21 21:48:50.563: 00:40:96:b7:ab:5c
  Processing RSN IE type 48, length 38 for mobile
  00:40:96:b7:ab:5c
```

**!--- The WLC/AP finds and Information Element that claims
PMKID Caching support on the Association request that
is sent from the client.**

```
*apfMsConnTask_2: Jun 21 21:48:50.563: 00:40:96:b7:ab:5c
  Received RSN IE with 1 PMKIDs from mobile
  00:40:96:b7:ab:5c
```

!--- The Reassociation Request from the client comes with one PMKID.

```
*apfMsConnTask_2: Jun 21 21:48:50.563:
  Received PMKID:  (16)

*apfMsConnTask_2: Jun 21 21:48:50.563:
  [0000] 91 65 c3 fb fc 44 75 48 67 90 d5 da df aa 71 e9

*apfMsConnTask_2: Jun 21 21:48:50.563: 00:40:96:b7:ab:5c
  Searching for PMKID in MSCB PMKID cache for mobile
  00:40:96:b7:ab:5c
*apfMsConnTask_2: Jun 21 21:48:50.563: 00:40:96:b7:ab:5c
  No valid PMKID found in the MSCB PMKID cache for mobile
  00:40:96:b7:ab:5
```

!--- As the client has never authenticated with this new AP, the WLC cannot find a valid PMKID to match the one provided by the client. However, since the client performs OKC and not SKC (as per the following messages), the WLC computes a new PMKID based on the information gathered (the cached PMK, the client MAC address, and the new AP MAC address).

```
*apfMsConnTask_2: Jun 21 21:48:50.563: 00:40:96:b7:ab:5c
  Trying to compute a PMKID from MSCB PMK cache for mobile
  00:40:96:b7:ab:5c

*apfMsConnTask_2: Jun 21 21:48:50.563:
  CCKM: Find PMK in cache: BSSID =  (6)
*apfMsConnTask_2: Jun 21 21:48:50.563:
  [0000] 84 78 ac f0 2a 90
*apfMsConnTask_2: Jun 21 21:48:50.563:
  CCKM: Find PMK in cache: realAA =  (6)
*apfMsConnTask_2: Jun 21 21:48:50.563:
  [0000] 84 78 ac f0 2a 92
*apfMsConnTask_2: Jun 21 21:48:50.563:
  CCKM: Find PMK in cache: PMKID =  (16)
*apfMsConnTask_2: Jun 21 21:48:50.563:
  [0000] 91 65 c3 fb fc 44 75 48 67 90 d5 da df aa 71 e9
*apfMsConnTask_2: Jun 21 21:48:50.563:
  CCKM: AA (6)
*apfMsConnTask_2: Jun 21 21:48:50.563:
  [0000] 84 78 ac f0 2a 92
*apfMsConnTask_2: Jun 21 21:48:50.563:
  CCKM: SPA (6)
*apfMsConnTask_2: Jun 21 21:48:50.563:
  [0000] 00 40 96 b7 ab 5c
*apfMsConnTask_2: Jun 21 21:48:50.563: 00:40:96:b7:ab:5c
  Adding BSSID 84:78:ac:f0:2a:92 to PMKID cache at
  index 0 for station 00:40:96:b7:ab:5c
*apfMsConnTask_2: Jun 21 21:48:50.563:
  New PMKID: (16)
*apfMsConnTask_2: Jun 21 21:48:50.563:
  [0000] 91 65 c3 fb fc 44 75 48 67 90 d5 da df aa 71 e9
```

```
*apfMsConnTask_2: Jun 21 21:48:50.563: 00:40:96:b7:ab:5c
  Computed a valid PMKID from MSCB PMK cache for mobile
  00:40:96:b7:ab:5c
```

!--- The new PMKID is computed and validated to match the one provided by the client, which is also computed with the same information. Hence, the fast-secure roam is possible.

```
*apfMsConnTask_2: Jun 21 21:48:50.563: 00:40:96:b7:ab:5c
  Setting active key cache index 0 ---> 0

*apfMsConnTask_2: Jun 21 21:48:50.564: 00:40:96:b7:ab:5c
  Sending Assoc Response to station on BSSID 84:78:ac:f0:2a:92
  (status 0) ApVapId 3 Slot
```

!--- The Reassociation response is sent to the client, which validates the fast-roam with OKC.

```
*dot1xMsgTask: Jun 21 21:48:50.570: 00:40:96:b7:ab:5c
  Initiating RSN with existing PMK to mobile
  00:40:96:b7:ab:5c
```

!--- WLC initiates a Robust Secure Network association with this client-and AP pair with the cached PMK found. Hence, EAP is avoided, as per the the next message.

```
*dot1xMsgTask: Jun 21 21:48:50.570: 00:40:96:b7:ab:5c
  Skipping EAP-Success to mobile 00:40:96:b7:ab:5c

*dot1xMsgTask: Jun 21 21:48:50.570: 00:40:96:b7:ab:5c
  Found an cache entry for BSSID 84:78:ac:f0:2a:92 in
  PMKID cache at index 0 of station 00:40:96:b7:ab:5c

*dot1xMsgTask: Jun 21 21:48:50.570:
  Including PMKID in M1  (16)
```

!--- The hashed PMKID is included on the Message-1 of the WPA/WPA2 4-Way handshake.

```
*dot1xMsgTask: Jun 21 21:48:50.570:
  [0000] 91 65 c3 fb fc 44 75 48 67 90 d5 da df aa 71 e9
```

!--- The PMKID is hashed. The next messages are the same WPA/WPA2 4-Way handshake messages described thus far, which are used in order to finish the encryption keys generation/installation.

```
*dot1xMsgTask: Jun 21 21:48:50.570: 00:40:96:b7:ab:5c
  Sending EAPOL-Key Message to mobile 00:40:96:b7:ab:5c state
  INITPMK (message 1), replay counter 00.00.00.00.00.00.00.00

*Dot1x_NW_MsgTask_4: Jun 21 21:48:50.589: 00:40:96:b7:ab:5
  Received EAPOL-Key from mobile 00:40:96:b7:ab:5c
```

```
*Dot1x_NW_MsgTask_4: Jun 21 21:48:50.589: 00:40:96:b7:ab:5c
  Received EAPOL-key in PTK_START state (message 2) from mobile
  00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 21:48:50.589: 00:40:96:b7:ab:5c
  PMK: Sending cache add

*Dot1x_NW_MsgTask_4: Jun 21 21:48:50.590: 00:40:96:b7:ab:5c
  Sending EAPOL-Key Message to mobile 00:40:96:b7:ab:5c state
  PTKINITNEGOTIATING (message 3), replay counter
  00.00.00.00.00.00.00.01

*Dot1x_NW_MsgTask_4: Jun 21 21:48:50.610: 00:40:96:b7:ab:5c
  Received EAPOL-Key from mobile 00:40:96:b7:ab:5c

*Dot1x_NW_MsgTask_4: Jun 21 21:48:50.610: 00:40:96:b7:ab:5c
  Received EAPOL-key in PTKINITNEGOTIATING state (message 4)
  from mobile 00:40:96:b7:ab:5c
```

Comme indiqué au début des débogages, le PMKID doit être calculé après la réception de la demande de réassociation du client. Ceci est nécessaire afin de valider le PMKID et de confirmer que le PMK mis en cache est utilisé avec la connexion en 4 étapes WPA2 pour dériver les clés de cryptage et terminer l'itinérance sécurisée rapide. Ne confondez pas les entrées CCKM sur les débogages ; ceci n'est pas utilisé afin d'effectuer CCKM, mais OKC, comme expliqué précédemment. CCKM est simplement un nom utilisé par le WLC pour ces sorties, comme le nom d'une fonction qui gère les valeurs afin de calculer le PMKID.

FlexConnect avec mise en cache des clés opportuniste

- L'authentification centrale est prise en charge. Cela inclut la commutation de données locale et centrale. Si le point d'accès fait partie du même groupe FlexConnect, l'itinérance de sécurité rapide est contrôlée par le point d'accès, sinon l'itinérance de sécurité rapide est contrôlée par le contrôleur.

  Remarque : cette configuration peut fonctionner si les points d'accès ne sont pas sur le même groupe FlexConnect, mais ce n'est pas une configuration recommandée ou prise en charge.

- L'authentification locale flexible est prise en charge. En mode connecté, le cache peut être distribué du point d'accès au contrôleur, puis au reste des points d'accès du groupe FlexConnect.
- Le mode autonome est pris en charge. Si le cache est déjà présent sur le point d'accès (en raison de la distribution précédente), l'itinérance rapide sécurisée doit fonctionner. La nouvelle authentification en mode autonome ne prend pas en charge l'itinérance rapide et sécurisée.

Avantages de la mise en cache opportuniste

- Le client sans fil et l'infrastructure WLAN n'ont pas besoin de mémoriser plusieurs PMKID, mais simplement de mettre en cache le PMK d'origine de l'authentification initiale vers le

WLAN. Ensuite, vous devez ressaisir le PMKID approprié (utilisé sur la demande de réassociation) requis avec chaque association sécurisée AP afin de valider l'itinérance sécurisée rapide.
- Ici, le client sans fil effectue une itinérance rapide et sécurisée vers un nouveau point d'accès sur le même WLAN/SSID, même s'il n'a jamais été associé à ce point d'accès (ce qui n'est pas le cas dans SKC). Tant que le client effectue l'authentification 802.1X/EAP initiale avec un point d'accès géré par le déploiement centralisé qui gère le cache PMK pour tous les points d'accès pour lesquels le client se déplace, aucune authentification complète n'est requise pour le reste de la durée de vie du client sur ce WLAN.

Inconvénients avec mise en cache opportuniste

- Cette méthode n'est déployée que sur un environnement centralisé où tous les AP sont sous une sorte de contrôle administratif (tel qu'un contrôleur WLAN) qui est responsable de la mise en cache et du partage de la PMK d'origine de la session client. Par conséquent, il s'agit d'une limitation sur les environnements AP autonomes.
- Les techniques qui sont appliquées dans cette méthode ne sont pas suggérées ou décrites sur la norme 802.11, de sorte que la prise en charge varie largement d'un périphérique à l'autre. Néanmoins, c'est toujours la méthode qui a été la plus adoptée en attendant 802.11r.

Remarque sur le terme « Mise en cache proactive des clés »

La mise en cache proactive des clés (ou PKC) est connue sous le nom d'OKC (Opportunistic Key Caching) et les deux termes sont utilisés de manière interchangeable lorsqu'ils décrivent la même méthode expliquée ici. Cependant, ce n'était qu'un terme utilisé par Airspace en 2001 pour une ancienne méthode de mise en cache des clés, qui a ensuite été utilisée par la norme 802.11i comme base pour la « pré-authentification » (une autre méthode d'itinérance sécurisée rapide brièvement expliquée ci-dessous). PKC n'est pas une préauthentification ou OKC (Opportunistic Key Caching), mais lorsque vous entendez parler de PKC, la référence est essentiellement à OKC, et non à la préauthentification.

Itinérance rapide et sécurisée avec préauthentification

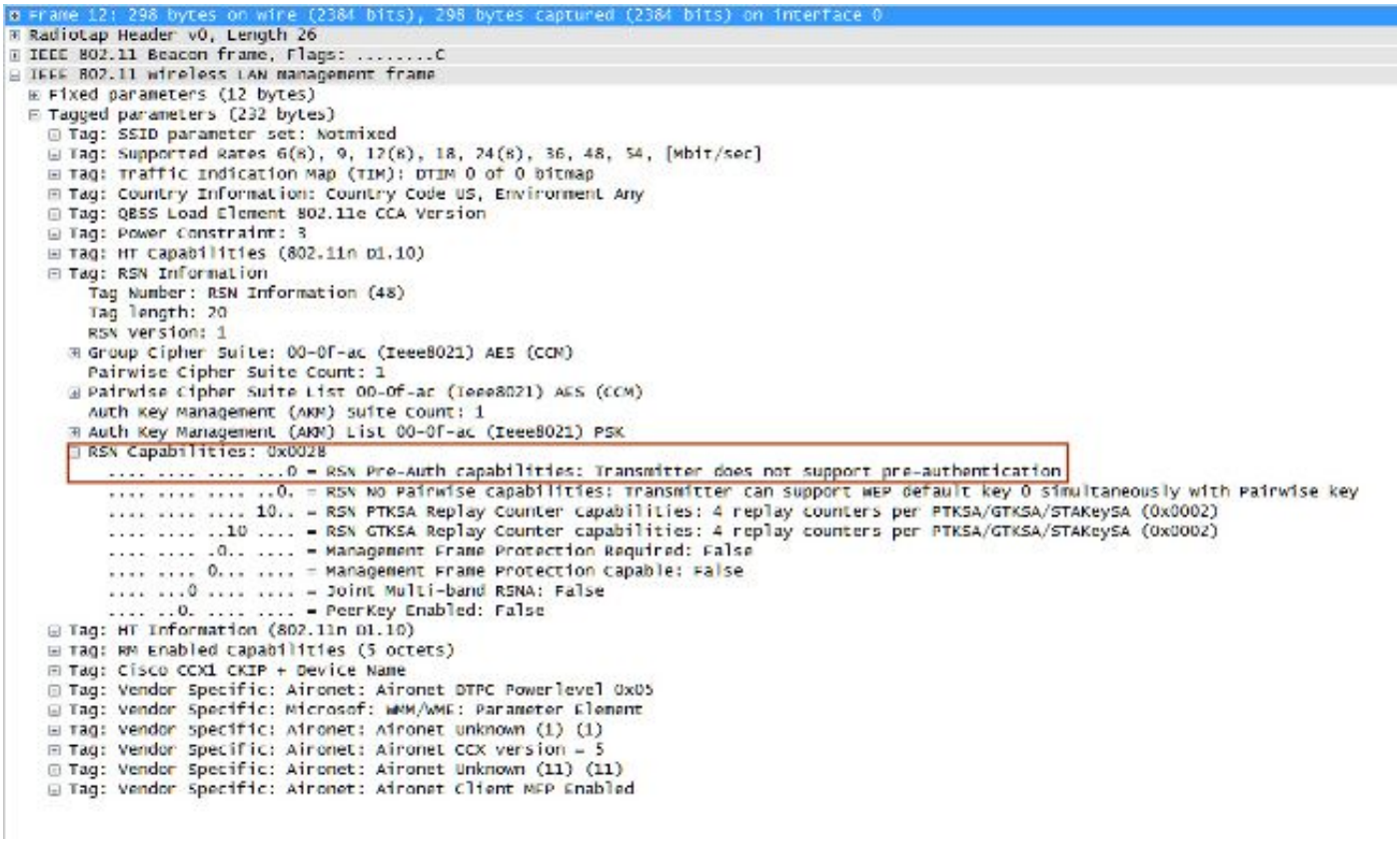
Cette méthode est également suggérée par la norme IEEE 802.11 dans la modification de sécurité 802.11i. Elle fonctionne donc également avec WPA2, mais c'est la seule méthode d'itinérance rapide et sécurisée qui n'est pas prise en charge par l'infrastructure WLAN Cisco. Pour cette raison, il n'est expliqué que brièvement ici et sans résultats.

Avec la préauthentification, les clients sans fil peuvent s'authentifier avec plusieurs points d'accès à la fois tout en étant associés au point d'accès actuel. Lorsque cela se produit, le client envoie les trames d'authentification EAP au point d'accès actuel où il est connecté, mais il est destiné aux autres points d'accès où le client veut effectuer la pré-authentification (points d'accès voisins qui sont des candidats possibles pour l'itinérance). Le point d'accès actuel envoie ces trames au(x) point(s) d'accès cible sur le système de distribution. Le nouveau point d'accès effectue une authentification complète sur le serveur RADIUS pour ce client, de sorte qu'une nouvelle connexion d'authentification EAP complète est effectuée et que ce nouveau point d'accès agit

comme authentificateur.

L'idée est d'effectuer l'authentification et de dériver PMK avec les AP voisins avant que le client ne les contacte réellement, donc quand il est temps d'errer, le client est déjà authentifié et avec un PMK déjà mis en cache pour cette nouvelle association sécurisée AP-client, de sorte qu'ils n'ont besoin d'effectuer la connexion en 4 étapes et de connaître une itinérance rapide après que le client ait envoyé sa demande de réassociation initiale.

Voici une image d'une balise AP qui montre le champ RSN IE qui annonce la prise en charge de la préauthentification (celle-ci provient d'un point d'accès Cisco, où il est confirmé que la préauthentification n'est pas prise en charge) :



Avantages de la préauthentification

Il y a un PMK pour chaque association sécurisée AP-client, qui pourrait être considéré comme un avantage de sécurité dans le cas où un AP est compromis et les clés sont volées (ne peut pas être utilisé avec d'autres AP). Cependant, cet avantage de sécurité est géré par l'infrastructure WLAN de différentes manières par rapport à d'autres méthodes.

Inconvénients avec préauthentification

- Comme il y a un PMK par AP, les clients ont une limite sur la quantité d'AP qui peuvent être pré-authentifiés.
- Chaque fois qu'un client effectue une pré-authentification avec un nouveau point d'accès, il y a un échange d'authentification EAP complet, ce qui signifie plus de charge sur le réseau et sur le serveur d'authentification.

- La plupart des clients sans fil ne prennent pas en charge cette méthode, car elle n'a jamais été très utilisée (OKC a été plus utilisé).

Itinérance rapide et sécurisée avec 802.11r

La technique d'itinérance rapide et sécurisée basée sur l'amendement 802.11r (officiellement appelée Fast BSS Transition par la norme 802.11, et connue sous le nom de FT) est la première méthode officiellement ratifiée (en 2008) par l'IEEE pour la norme 802.11 comme solution pour effectuer des transitions rapides entre les AP (Basic Service Sets ou BSS), qui définit clairement la hiérarchie de clés utilisée lorsque vous gérez et mettez en cache des clés sur un WLAN. Cependant, son adoption a été lente, principalement en raison des autres solutions déjà disponibles lorsque des transitions rapides étaient réellement nécessaires, comme avec les implémentations VoWLAN lorsqu'elles sont utilisées avec l'une des méthodes précédemment expliquées dans ce document. Seuls quelques périphériques prennent actuellement en charge certaines des options FT (d'ici 2013).

Cette technique est plus complexe à expliquer que les autres méthodes, car elle introduit de nouveaux concepts et plusieurs couches de PMK qui sont mises en cache sur différents périphériques (chaque périphérique ayant un rôle différent), et fournit encore plus d'options pour l'itinérance rapide et sécurisée. Par conséquent, un bref résumé est fourni sur cette méthode et la façon dont elle est mise en oeuvre avec chaque option disponible.

La norme 802.11r est différente des normes SKC et OKC, principalement pour les raisons suivantes :

- La messagerie d'échange (PMKID, ANonce et SNonce, par exemple) se produit dans les trames d'authentification 802.11 ou dans les trames d'action au lieu des trames de réassociation. Contrairement aux méthodes de mise en cache PMKID, la phase d'échange en quatre étapes, qui est effectuée après l'échange de messages de (ré)association, est évitée. La connexion de clé avec le nouveau point d'accès commence avant que le client ne se déplace entièrement/ne se réassocie avec ce nouveau point d'accès.
- Il fournit deux méthodes pour la connexion en itinérance rapide : via AIR et via le système de distribution (DS).
- La norme 802.11r comporte davantage de couches hiérarchiques de clés.
- Comme ce protocole évite la connexion en 4 étapes pour la gestion des clés lorsqu'un client se déplace (génère de nouvelles clés de cryptage - PTK et GTK - sans avoir besoin de cette connexion), il peut également être appliqué pour les configurations WPA2 avec un PSK, et pas seulement lorsque 802.1X/EAP est utilisé pour l'authentification. Cela accélère encore plus l'itinérance pour ces configurations, où aucun échange EAP ou 4-Way handshake n'a lieu.

Grâce à cette méthode, le client sans fil effectue une seule authentification initiale sur l'infrastructure WLAN lorsqu'une connexion est établie avec le premier point d'accès, et effectue une itinérance rapide et sécurisée entre les points d'accès du même domaine de mobilité FT.

C'est l'un des nouveaux concepts, qui se réfère essentiellement aux AP qui utilisent le même SSID (connu sous le nom de Extended Service Set ou ESS) et gèrent les mêmes clés FT. Ceci

est similaire aux autres méthodes expliquées jusqu'à présent. La façon dont les AP gèrent les clés de domaine de mobilité FT est normalement basée sur une configuration centralisée, telle que le WLC ou les groupes de mobilité ; cependant, cette méthode peut également être implémentée sur des environnements d'AP autonomes.

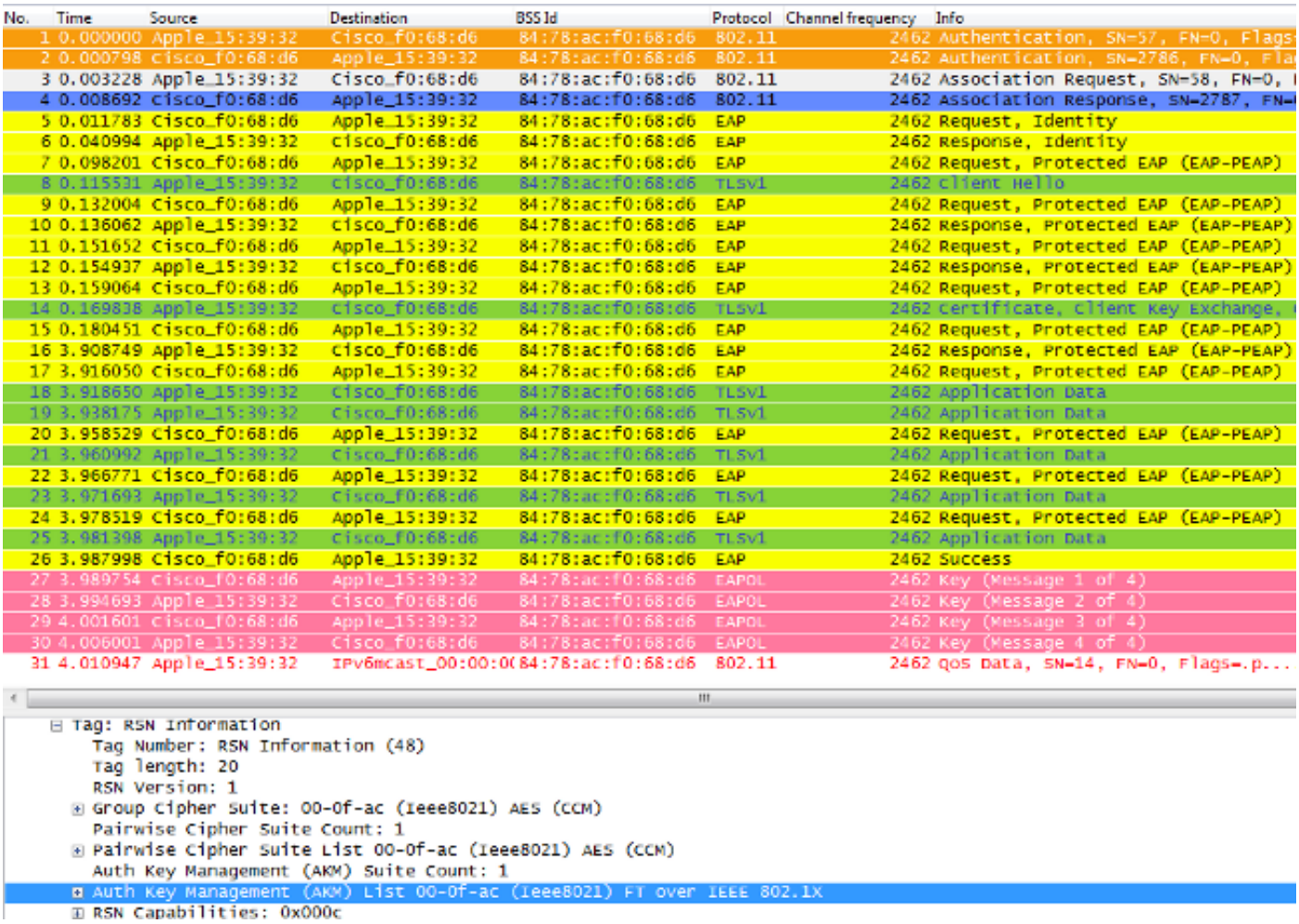
Voici un résumé de la hiérarchie des clés :

- Un MSK est toujours dérivé sur le demandeur client et le serveur d'authentification à partir de la phase d'authentification 802.1X/EAP initiale (transféré du serveur d'authentification à l'authentificateur (WLC) une fois l'authentification réussie). Ce MSK, comme dans les autres méthodes, est utilisé comme valeur de départ pour la hiérarchie de clés FT. Lorsque vous utilisez WPA2-PSK au lieu d'une méthode d'authentification EAP, le PSK est essentiellement ce MSK.
- Une clé principale par paire R0 (PMK-R0) est dérivée de la clé MSK, qui est la clé de premier niveau de la hiérarchie de clés FT. Les détenteurs de clé pour ce PMK-R0 sont le WLC et le client.
- Une clé de second niveau, appelée clé maître par paire R1 (PMK-R1), est dérivée de la clé PMK-R0, et les détenteurs de clé sont le client et les points d'accès gérés par le WLC qui détient la clé PMK-R0.
- La clé de troisième et dernier niveau de la hiérarchie de clés FT est la clé PTK, qui est la dernière clé utilisée pour chiffrer les trames de données de monodiffusion 802.11 (similaire aux autres méthodes qui utilisent WPA/TKIP ou WPA2/AES). Ce PTK est dérivé sur FT du PMK-R1, et les détenteurs de clé sont le client et les AP gérés par le WLC.

Remarque : selon le fournisseur du WLAN et les configurations de mise en oeuvre (telles que les points d'accès autonomes, FlexConnect ou Mesh), l'infrastructure WLAN peut transférer et gérer les clés d'une manière différente. Il peut même modifier les rôles des détenteurs de clés, mais comme cela sort du cadre de ce document, les exemples basés sur le résumé de hiérarchie de clés donné précédemment constituent le prochain point à traiter. Les différences ne sont pas vraiment pertinentes pour comprendre le processus, à moins que vous ayez réellement besoin d'analyser en profondeur les périphériques d'infrastructure (et leur code) afin de découvrir un problème logiciel.

Transition BSS rapide par liaison radio

Avec cette méthode, la première association à un point d'accès est une première authentification régulière au WLAN, où l'authentification 802.1X/EAP complète par rapport au serveur d'authentification et la connexion en 4 étapes pour la génération de clé doivent avoir lieu avant l'envoi des trames de données, comme illustré dans cette image d'écran :



Les principales différences sont les suivantes :

- La négociation de gestion de clé d'authentification est légèrement différente de la négociation WPA/WPA2 standard. Certaines informations supplémentaires sont donc utilisées afin d'effectuer cette négociation lorsque l'association à une infrastructure WLAN qui prend en charge FT se produit. Comme l'illustre l'image, la trame de demande d'association du client est sélectionnée et le champ AKM de l'élément d'information RNS est mis en surbrillance afin de montrer que ce client veut effectuer un FT sur 802.1X/EAP.
- L'élément d'information du domaine de mobilité (partie de FT) est également représenté, où le champ Capacité et politique FT indique si la transition BSS rapide est terminée en mode Over-the-Air ou Over-the-DS en mode d'itinérance rapide (ceci indique Over-the-Air dans cette image).
- Un autre élément d'information est également ajouté (Fast BSS Transition ou FT IE, qui est décrit plus loin dans ce document) avec des informations qui sont nécessaires pour effectuer la séquence d'authentification FT lors de l'itinérance FT.
- La génération de clé est différente en raison de la hiérarchie des clés. Ainsi, même si la connexion FT en 4 étapes ressemble à la connexion WPA/WPA2 en 4 étapes, son contenu est légèrement différent.

Les débogages montrent essentiellement le même échange de trames d'authentification EAP que les autres méthodes lors de l'authentification initiale au WLAN (comme remarqué sur les images), mais certaines sorties qui concernent les techniques de mise en cache des clés utilisées par le

WLC sont ajoutées ; ainsi, cette sortie de débogage est coupée afin de montrer uniquement les informations pertinentes :

<#root>

```
*apfMsConnTask_0: Jun 27 19:25:23.426: ec:85:2f:15:39:32
  Association received from mobile on BSSID
  84:78:ac:f0:68:d6

!--- This is the Association request from the client.



*apfMsConnTask_0: Jun 27 19:25:23.427: ec:85:2f:15:39:32
  Marking this mobile as TGr capable.

!--- WLC recognizes that the client is 802.11r-capable.



*apfMsConnTask_0: Jun 27 19:25:23.427: ec:85:2f:15:39:32
  Processing RSN IE type 48, length 20 for mobile
  ec:85:2f:15:39:32

!--- The WLC/AP finds an Information Element that claims FT
     support on the Association request that is sent from the client.



*apfMsConnTask_0: Jun 27 19:25:23.427:
  Sending assoc-resp station:ec:85:2f:15:39:32
  AP:84:78:ac:f0:68:d0-00 thread:144be808
*apfMsConnTask_0: Jun 27 19:25:23.427:
  Adding MDIE, ID is:0xaaf0
*apfMsConnTask_0: Jun 27 19:25:23.427: ec:85:2f:15:39:32
  Including FT Mobility Domain IE (length 5) in Initial
  assoc Resp to mobile
*apfMsConnTask_0: Jun 27 19:25:23.427: ec:85:2f:15:39:32
  Sending R0KH-ID as:-84.30.6.-3
*apfMsConnTask_0: Jun 27 19:25:23.427: ec:85:2f:15:39:32
  Sending R1KH-ID as 3c:ce:73:d8:02:00
*apfMsConnTask_0: Jun 27 19:25:23.427: ec:85:2f:15:39:32
  Including FT IE (length 98) in Initial Assoc Resp to mobile
*apfMsConnTask_0: Jun 27 19:25:23.427: ec:85:2f:15:39:32
  Sending Assoc Response to station on BSSID 84:78:ac:f0:68:d6
  (status 0) ApVapId 7 Slot 0

!--- The Association Response is sent to the client once the
     FT information is computed (as per the previous messages),
     so this is included in the response.



*dot1xMsgTask: Jun 27 19:25:23.432: ec:85:2f:15:39:32
  Sending EAP-Request/Identity to mobile ec:85:2f:15:39:32
  (EAP Id 1)

!--- EAP begins, and follows the same exchange explained so far.



*apfMsConnTask_0: Jun 27 19:25:23.436: ec:85:2f:15:39:32
```

```
  Got action frame from this client.

*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.449: ec:85:2f:15:39:32
  Received EAPOL EAPPKT from mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.449: ec:85:2f:15:39:32
  Received Identity Response (count=1) from mobile
  ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.456: ec:85:2f:15:39:32
  Processing Access-Challenge for mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.456: ec:85:2f:15:39:32
  Sending EAP Request from AAA to mobile ec:85:2f:15:39:32
  (EAP Id 2)

*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.479: ec:85:2f:15:39:32
  Received EAPOL EAPPKT from mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.479: ec:85:2f:15:39:32
  Received EAP Response from mobile ec:85:2f:15:39:32
  (EAP Id 2, EAP Type 25)

*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.627: ec:85:2f:15:39:32
  Processing Access-Accept for mobile ec:85:2f:15:39:32
```

!--- The client is validated/authenticated by the RADIUS Server.

```
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.627: ec:85:2f:15:39:32
  Creating a PKC PMKID Cache entry for station
  ec:85:2f:15:39:32 (RSN 2)
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.627: ec:85:2f:15:39:32
  Resetting MSCB PMK Cache Entry 0 for station
  ec:85:2f:15:39:32
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.627: ec:85:2f:15:39:32
  Setting active key cache index 8 ---> 8
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.628: ec:85:2f:15:39:32
  Setting active key cache index 8 ---> 0
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.628: ec:85:2f:15:39:32
  Adding BSSID 84:78:ac:f0:68:d6 to PMKID cache at index 0
  for station ec:85:2f:15:39:32
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.628: New PMKID: (16)
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.628:
  [0000] 52 b8 8f cf 50 a7 90 98 2b ba d6 20 79 e4 cd f9
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.629: ec:85:2f:15:39:32
  Created PMK Cache Entry for TGr AKM:802.1x ec:85:2f:15:39:32
```

!--- WLC creates a PMK cache entry for this client, which is used for FT with 802.1X in this case, so the PMKID is computed with the AP MAC address (BSSID 84:78:ac:f0:68:d6).

```
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.629:
  ec:85:2f:15:39:32   R0KH-ID:172.30.6.253
  R1KH-ID:3c:ce:73:d8:02:00  MSK Len:48 pmkValidTime:1807
```

!--- The R0KH-ID and R1KH-ID are defined, as well as the PMK cache validity period.

```
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.630: ec:85:2f:15:39:32
  PMK sent to mobility group
```

!--- The FT PMK cache entry for this client is shared with the WLCs on the mobility group.

```
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.630: ec:85:2f:15:39:32
  Sending EAP-Success to mobile ec:85:2f:15:39:32 (EAP Id 12)

*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.630: ec:85:2f:15:39:32
  Found an cache entry for BSSID 84:78:ac:f0:68:d6 in PMKID
  cache at index 0 of station ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.630: Including PMKID in
  M1  (16)
```

!--- The hashed PMKID is included on the Message-1 of the initial FT 4-Way handshake.

```
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.630:
  [0000] 52 b8 8f cf 50 a7 90 98 2b ba d6 20 79 e4 cd f9

*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.630: ec:85:2f:15:39:32
  Sending EAPOL-Key Message to mobile ec:85:2f:15:39:32 state
  INITPMK (message 1), replay counter 00.00.00.00.00.00.00.0
```

!--- Message-1 of the FT 4-Way handshake is sent from the WLC/AP to the client.

```
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.639: ec:85:2f:15:39:32
  Received EAPOL-Key from mobile ec:85:2f:15:39:32
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.639: ec:85:2f:15:39:32
  Received EAPOL-key in PTK_START state (message 2) from
  mobile ec:85:2f:15:39:32
```

!--- Message-2 of the FT 4-Way handshake is received successfully from the client.

```
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.639: ec:85:2f:15:39:32
  Calculating PMKR0Name
```

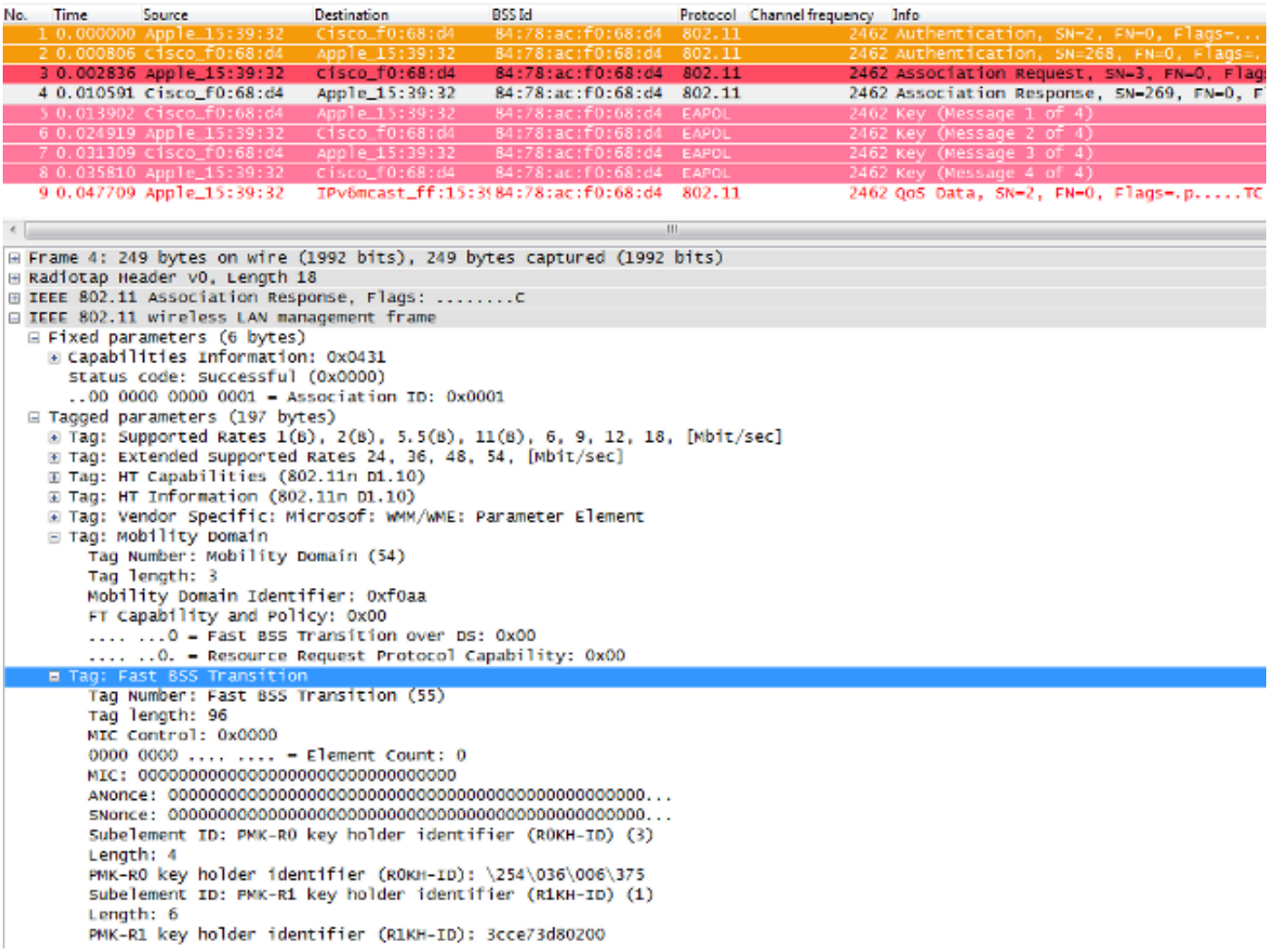
!--- The PMKR0Name is calculated.

```
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.639: ec:85:2f:15:39:32
  DOT11R: Sending cache add

*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.639: Adding MDIE,
  ID is:0xaaf0
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.639: ec:85:2f:15:39:32
  Adding TIE for reassociation deadtime:20000 milliseconds
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.639: ec:85:2f:15:39:32
  Adding TIE for R0Key-Data valid time :1807
*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.640: ec:85:2f:15:39:32
  Sending EAPOL-Key Message to mobile ec:85:2f:15:39:32 state
  PTKINITNEGOTIATING (message 3), replay counter
```

```
  00.00.00.00.00.00.00.01

!--- After the MDIE, TIE for reassociation deadtime, and TIE
     for R0Key-Data valid time are calculated, the Message-3
     of this FT 4-Way handshake is sent from the WLC/AP to the
     client with this information.



*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.651: ec:85:2f:15:39:32
  Received EAPOL-Key from mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 27 19:25:23.651: ec:85:2f:15:39:32
  Received EAPOL-key in PTKINITNEGOTIATING state (message 4)
  from mobile ec:85:2f:15:39:32

!--- Message-4 (final message) of this initial FT 4-Way handshake
     is received successfully from the client, which confirms the
     installation of the derived keys. They can now be used in order
     to encrypt data frames with the current AP.
```

Remarque : afin de déboguer cette méthode et d'atteindre les sorties 802.11r/FT supplémentaires montrées ici, un débogage supplémentaire est activé avec le client debug, qui est le debug ft events enable.

Voici les images et les débogages d'une association initiale au WLAN lorsque vous effectuez un FT avec WPA2-PSK (au lieu d'une méthode 802.1X/EAP), où la trame de réponse d'association du point d'accès est sélectionnée afin d'afficher l'élément d'information de transition BSS rapide (mis en surbrillance). Certaines des informations clés nécessaires à l'exécution de la connexion FT en 4 étapes sont également présentées :



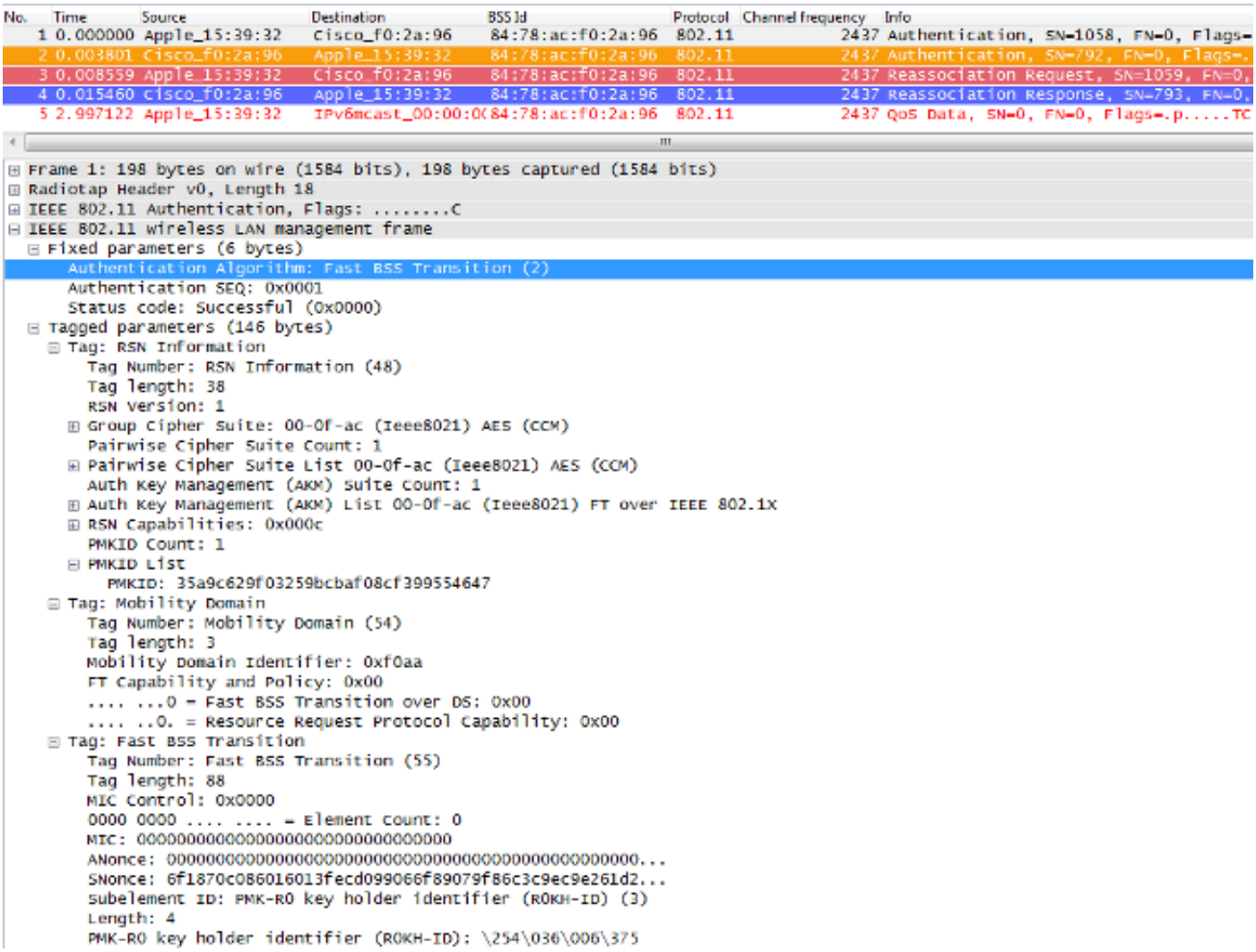

```
*apfMsConnTask_0: Jun 27 19:29:09.136: ec:85:2f:15:39:32
  Association received from mobile on BSSID
  84:78:ac:f0:68:d4

*apfMsConnTask_0: Jun 27 19:29:09.137: ec:85:2f:15:39:32
  Marking this mobile as TGr capable.

*apfMsConnTask_0: Jun 27 19:29:09.137: ec:85:2f:15:39:32
  Processing RSN IE type 48, length 20 for mobile
  ec:85:2f:15:39:32

*apfMsConnTask_0: Jun 27 19:29:09.137: Sending assoc-resp
  station:ec:85:2f:15:39:32 AP:84:78:ac:f0:68:d0-00
  thread:144be808

*apfMsConnTask_0: Jun 27 19:29:09.137: Adding MDIE,
  ID is:0xaaf0

*apfMsConnTask_0: Jun 27 19:29:09.137: ec:85:2f:15:39:32
  Including FT Mobility Domain IE (length 5) in Initial
  assoc Resp to mobile

*apfMsConnTask_0: Jun 27 19:29:09.137: ec:85:2f:15:39:32
  Sending R0KH-ID as:-84.30.6.-3

*apfMsConnTask_0: Jun 27 19:29:09.137: ec:85:2f:15:39:32
```

```
  Sending R1KH-ID as 3c:ce:73:d8:02:00

*apfMsConnTask_0: Jun 27 19:29:09.137: ec:85:2f:15:39:32
  Including FT IE (length 98) in Initial Assoc Resp to mobile

*apfMsConnTask_0: Jun 27 19:29:09.138: ec:85:2f:15:39:32
  Sending Assoc Response to station on BSSID 84:78:ac:f0:68:d4
  (status 0) ApVapId 5 Slot 0

*dot1xMsgTask: Jun 27 19:29:09.141: ec:85:2f:15:39:32
  Creating a PKC PMKID Cache entry for station
  ec:85:2f:15:39:32 (RSN 2)

*dot1xMsgTask: Jun 27 19:29:09.141: ec:85:2f:15:39:32
  Resetting MSCB PMK Cache Entry 0 for station
  ec:85:2f:15:39:32

*dot1xMsgTask: Jun 27 19:29:09.141: ec:85:2f:15:39:32
  Setting active key cache index 8 ---> 8

*dot1xMsgTask: Jun 27 19:29:09.141: ec:85:2f:15:39:32
  Setting active key cache index 8 ---> 0

*dot1xMsgTask: Jun 27 19:29:09.141: ec:85:2f:15:39:32
  Adding BSSID 84:78:ac:f0:68:d4 to PMKID cache at
  index 0 for station ec:85:2f:15:39:32

*dot1xMsgTask: Jun 27 19:29:09.142: New PMKID: (16)

*dot1xMsgTask: Jun 27 19:29:09.142:
  [0000] 17 4b 17 5c ed 5f c7 1d 66 39 e9 5d 3a 63 69 e7

*dot1xMsgTask: Jun 27 19:29:09.142: ec:85:2f:15:39:32
  Creating global PMK cache for this TGr client

*dot1xMsgTask: Jun 27 19:29:09.142: ec:85:2f:15:39:32
  Created PMK Cache Entry for TGr AKM:PSK
  ec:85:2f:15:39:32

*dot1xMsgTask: Jun 27 19:29:09.142: ec:85:2f:15:39:32
  R0KH-ID:172.30.6.253   R1KH-ID:3c:ce:73:d8:02:00
  MSK Len:48 pmkValidTime:1813

*dot1xMsgTask: Jun 27 19:29:09.142: ec:85:2f:15:39:32
  Initiating RSN PSK to mobile ec:85:2f:15:39:32

*dot1xMsgTask: Jun 27 19:29:09.142: ec:85:2f:15:39:32
  Found an cache entry for BSSID 84:78:ac:f0:68:d4 in
  PMKID cache at index 0 of station ec:85:2f:15:39:32

*dot1xMsgTask: Jun 27 19:29:09.142: Including PMKID
  in M1  (16)

*dot1xMsgTask: Jun 27 19:29:09.142:
  [0000] 17 4b 17 5c ed 5f c7 1d 66 39 e9 5d 3a 63 69 e7

*dot1xMsgTask: Jun 27 19:29:09.143: ec:85:2f:15:39:32
  Sending EAPOL-Key Message to mobile ec:85:2f:15:39:32
  state INITPMK (message 1), replay counter
  00.00.00.00.00.00.00.00

*apfMsConnTask_0: Jun 27 19:29:09.144: ec:85:2f:15:39:32
```

```
  Got action frame from this client.

*Dot1x_NW_MsgTask_2: Jun 27 19:29:09.152: ec:85:2f:15:39:32
  Received EAPOL-Key from mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 27 19:29:09.153: ec:85:2f:15:39:32
  Received EAPOL-key in PTK_START state (message 2) from
  mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 27 19:29:09.153: ec:85:2f:15:39:32
  Calculating PMKR0Name

*Dot1x_NW_MsgTask_2: Jun 27 19:29:09.153: Adding MDIE,
  ID is:0xaaf0

*Dot1x_NW_MsgTask_2: Jun 27 19:29:09.153: ec:85:2f:15:39:32
  Adding TIE for reassociation deadtime:20000 milliseconds

*Dot1x_NW_MsgTask_2: Jun 27 19:29:09.153: ec:85:2f:15:39:32
  Adding TIE for R0Key-Data valid time :1813

*Dot1x_NW_MsgTask_2: Jun 27 19:29:09.154: ec:85:2f:15:39:32
  Sending EAPOL-Key Message to mobile ec:85:2f:15:39:32 state
  PTKINITNEGOTIATING (message 3), replay counter
  00.00.00.00.00.00.00.01

*Dot1x_NW_MsgTask_2: Jun 27 19:29:09.163: ec:85:2f:15:39:32
  Received EAPOL-Key from mobile ec:85:2f:15:39:32

*Dot1x_NW_MsgTask_2: Jun 27 19:29:09.163: ec:85:2f:15:39:32
  Received EAPOL-key in PTKINITNEGOTIATING state (message 4)
  from mobile ec:85:2f:15:39:32
```

Avec la norme 802.11r, l’association initiale au réseau local sans fil est la base utilisée pour dériver les clés de base utilisées par cette technique, tout comme dans les autres méthodes d’itinérance rapide et sécurisée. Les principales différences viennent lorsque le client commence à se déplacer ; FT évite non seulement 802.1X/EAP lorsqu'il est utilisé, mais il exécute en fait une méthode d'itinérance plus efficace qui combine les trames d'authentification et de réassociation de système ouvert 802.11 (qui sont toujours utilisées et requises lors de l'itinérance entre AP) afin d'échanger des informations FT et de dériver de nouvelles clés de chiffrement dynamiques à la place de la connexion en 4 étapes.

L'image suivante montre les trames échangées lors d'une transition BSS Fast Over-the-Air avec sécurité 802.1X/EAP. La trame Open System Authentication du client au point d'accès est sélectionnée afin de voir les éléments d'information du protocole FT qui sont requis pour commencer la négociation de clé FT. Ceci est utilisé afin de dériver le nouveau PTK avec le nouveau AP (basé sur le PMK-R1). Le champ qui montre l'algorithme d'authentification est mis en surbrillance afin de montrer que ce client n'effectue pas une simple authentification Open System, mais une transition BSS rapide :



Voici les sorties de débogage du WLC lorsque cet événement d'itinérance FT se produit avec 802.1X/EAP :

<#root>

```
*apfMsConnTask_2: Jun 27 19:25:48.751: ec:85:2f:15:39:32
  Doing preauth for this client over the Air

!--- WLC begins FT fast-secure roaming over-the-Air with
      this client and performs a type of preauthentication,
      because the client asks for this with FT on the Authentication
      frame that is sent to the new AP over-the-Air
      (before the Reassociation Request).



*apfMsConnTask_2: Jun 27 19:25:48.751: ec:85:2f:15:39:32
  Doing local roaming for destination address
  84:78:ac:f0:2a:96

!--- WLC performs the local roaming event with the new AP to
      which the client roams.



*apfMsConnTask_2: Jun 27 19:25:48.751: ec:85:2f:15:39:32
  Got 1 AKMs in RSNIE
```

```
*apfMsConnTask_2: Jun 27 19:25:48.751: ec:85:2f:15:39:32
  RSNIE AKM matches with PMK cache entry :0x3
```

!--- WLC receives one PMK from this client (known as AKM here), which matches the PMK cache entry hold for this client.

```
*apfMsConnTask_2: Jun 27 19:25:48.751: ec:85:2f:15:39:32
  Created a new preauth entry for AP:84:78:ac:f0:2a:96
*apfMsConnTask_2: Jun 27 19:25:48.751: Adding MDIE,
  ID is:0xaaf0
```

!--- WLC creates a new preauth entry for this AP-and-Client pair, and adds the MDIE information.

```
*apfMsConnTask_2: Jun 27 19:25:48.763: Processing assoc-req
  station:ec:85:2f:15:39:32 AP:84:78:ac:f0:2a:90-00
  thread:144bef38
*apfMsConnTask_2: Jun 27 19:25:48.763: ec:85:2f:15:39:32
  Reassociation received from mobile on BSSID
  84:78:ac:f0:2a:96
```

!--- Once the client receives the Authentication frame reply from the WLC/AP, the Reassociation request is sent, which is received at the new AP to which the client roams.

```
*apfMsConnTask_2: Jun 27 19:25:48.764: ec:85:2f:15:39:32
  Marking this mobile as TGr capable.

*apfMsConnTask_2: Jun 27 19:25:48.764: ec:85:2f:15:39:32
  Processing RSN IE type 48, length 38 for mobile
  ec:85:2f:15:39:32

*apfMsConnTask_2: Jun 27 19:25:48.765: ec:85:2f:15:39:32
  Roaming succeed for this client.
```

!--- WLC confirms that the FT fast-secure roaming is successful for this client.

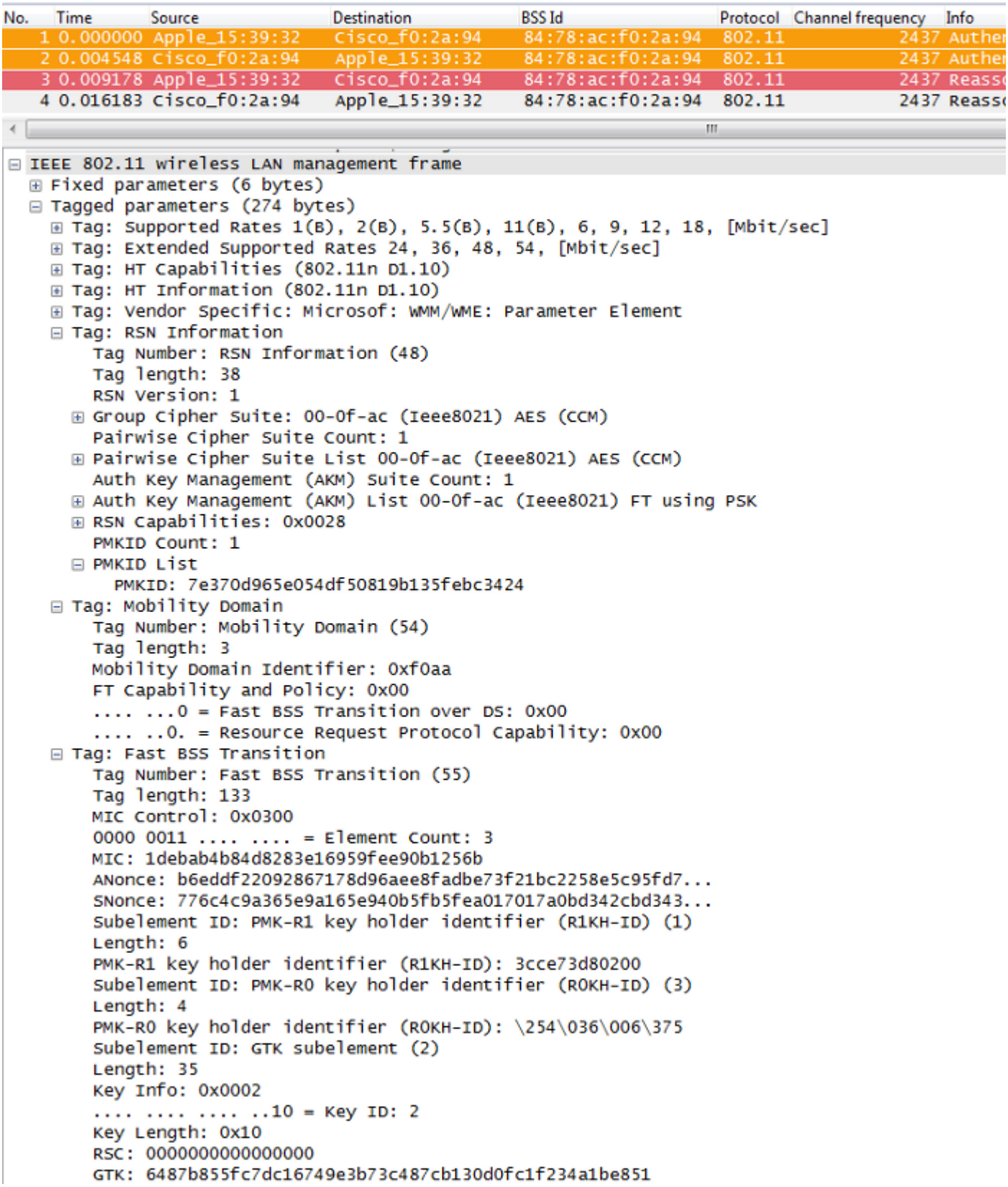
```
*apfMsConnTask_2: Jun 27 19:25:48.765: Sending assoc-resp
  station:ec:85:2f:15:39:32 AP:84:78:ac:f0:2a:90-00
  thread:144bef38
*apfMsConnTask_2: Jun 27 19:25:48.766: Adding MDIE,
  ID is:0xaaf0
*apfMsConnTask_2: Jun 27 19:25:48.766: ec:85:2f:15:39:32
  Including FT Mobility Domain IE (length 5) in
  reassociation assoc Resp to mobile
*apfMsConnTask_2: Jun 27 19:25:48.766: ec:85:2f:15:39:32
  Sending Assoc Response to station on BSSID 84:78:ac:f0:2a:96
  (status 0) ApVapId 7 Slot 0
```

!--- The Reassociation response is sent to the client, which includes the FT Mobility Domain IE.

```
*dot1xMsgTask: Jun 27 19:25:48.769: ec:85:2f:15:39:32
  Finishing FT roaming for mobile ec:85:2f:15:39:32
```

```
!--- FT roaming finishes and EAP is skipped (as well as any
      other key management handshake), so the client is ready
      to pass encrypted data frames with the current AP.


*dot1xMsgTask: Jun 27 19:25:48.769: ec:85:2f:15:39:32
  Skipping EAP-Success to mobile ec:85:2f:15:39:32
```

Voici une image qui montre une transition BSS rapide en direct avec la sécurité WPA2-PSK, où la trame de réponse de réassociation finale du point d'accès au client est sélectionnée afin d'afficher plus de détails sur cet échange FT :



Voici les sorties de débogage lorsque cet événement d'itinérance FT se produit avec PSK, qui sont similaires à celles lorsque 802.1X/EAP est utilisé :

```
*apfMsConnTask_2: Jun 27 19:29:29.854: ec:85:2f:15:39:32
  Doing preauth for this client over the Air
```

```
*apfMsConnTask_2: Jun 27 19:29:29.854: ec:85:2f:15:39:32
  Doing local roaming for destination address
  84:78:ac:f0:2a:94

*apfMsConnTask_2: Jun 27 19:29:29.854: ec:85:2f:15:39:32
  Got 1 AKMs in RSNIE

*apfMsConnTask_2: Jun 27 19:29:29.854: ec:85:2f:15:39:32
  RSNIE AKM matches with PMK cache entry :0x4

*apfMsConnTask_2: Jun 27 19:29:29.854: ec:85:2f:15:39:32
  Created a new preauth entry for AP:84:78:ac:f0:2a:94

*apfMsConnTask_2: Jun 27 19:29:29.854: Adding MDIE,
  ID is:0xaaf0

*apfMsConnTask_2: Jun 27 19:29:29.867: Processing assoc-req
  station:ec:85:2f:15:39:32 AP:84:78:ac:f0:2a:90-00
  thread:144bef38

*apfMsConnTask_2: Jun 27 19:29:29.867: ec:85:2f:15:39:32
  Reassociation received from mobile on BSSID
  84:78:ac:f0:2a:94

*apfMsConnTask_2: Jun 27 19:29:29.868: ec:85:2f:15:39:32
  Marking this mobile as TGr capable.

*apfMsConnTask_2: Jun 27 19:29:29.868: ec:85:2f:15:39:32
  Processing RSN IE type 48, length 38 for mobile
  ec:85:2f:15:39:32

*apfMsConnTask_2: Jun 27 19:29:29.869: ec:85:2f:15:39:32
  Roaming succeed for this client.

*apfMsConnTask_2: Jun 27 19:29:29.869: Sending assoc-resp
  station:ec:85:2f:15:39:32 AP:84:78:ac:f0:2a:90-00
  thread:144bef38

*apfMsConnTask_2: Jun 27 19:29:29.869: Adding MDIE,
  ID is:0xaaf0

*apfMsConnTask_2: Jun 27 19:29:29.869: ec:85:2f:15:39:32
  Including FT Mobility Domain IE (length 5) in
  reassociation assoc Resp to mobile

*apfMsConnTask_2: Jun 27 19:29:29.870: ec:85:2f:15:39:32
  Sending Assoc Response to station on BSSID
  84:78:ac:f0:2a:94 (status 0) ApVapId 5 Slot 0

*dot1xMsgTask: Jun 27 19:29:29.874: ec:85:2f:15:39:32
  Finishing FT roaming for mobile ec:85:2f:15:39:32
```

Comme l'illustre l'image, une fois que la transition BSS rapide est négociée lors de l'association initiale au WLAN, les quatre trames utilisées et requises pour l'itinérance (authentification système ouverte du client, authentification système ouverte du point d'accès, demande de réassociation et réponse de réassociation) sont essentiellement utilisées comme une connexion en quatre étapes FT afin de dériver les nouvelles clés PTK (clé de cryptage monodiffusion) et GTK (clé de cryptage

multidiffusion/diffusion).

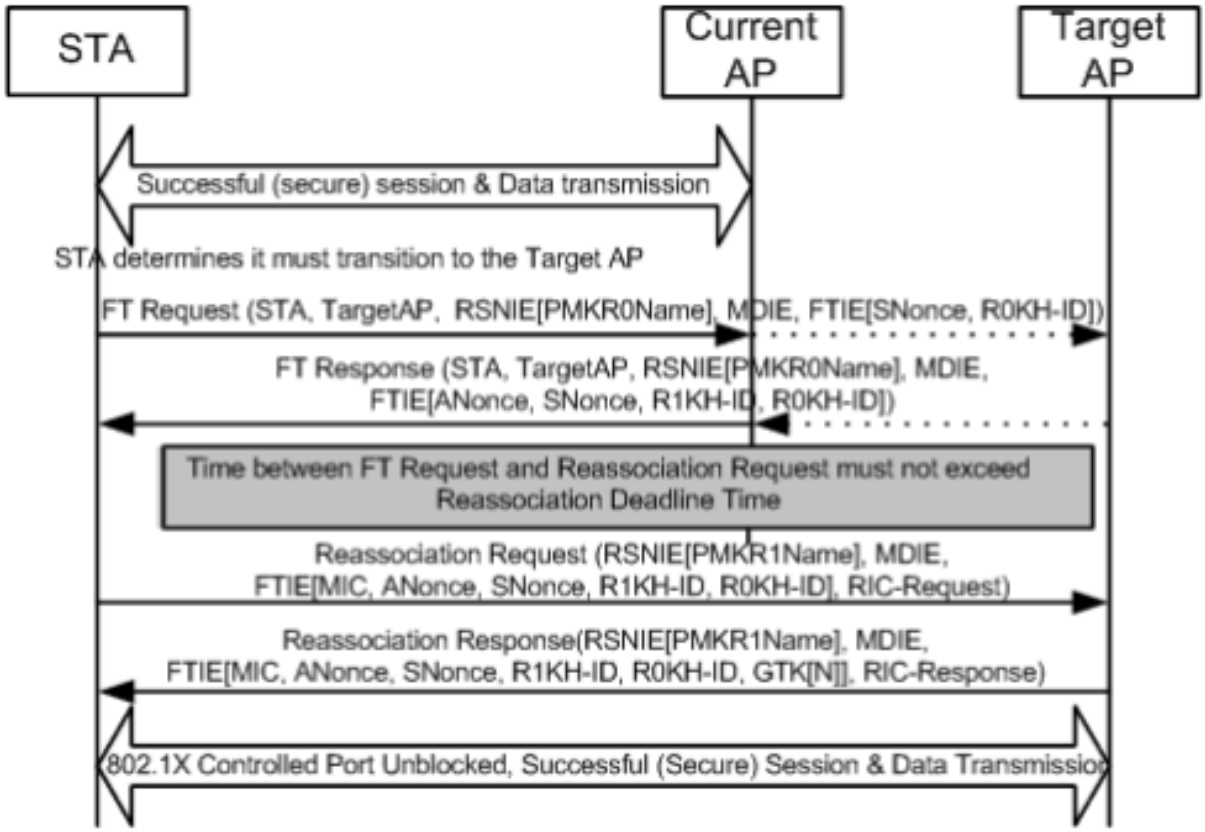
Cela remplace l'échange en quatre étapes qui se produit normalement après l'échange de ces trames, et le contenu FT et la négociation de clé sur ces trames sont fondamentalement les mêmes, que vous utilisiez 802.1X/EAP ou PSK comme méthode de sécurité. Comme le montre l'image, le champ AKM est la principale différence, ce qui confirme si le client exécute FT avec PSK ou 802.1X. Par conséquent, il est important de noter que ces quatre trames ne disposent normalement pas de ce type d'informations de sécurité pour la négociation de clé, mais uniquement lorsque le client FT est en itinérance si la norme 802.11r est implémentée et négociée entre le client et l'infrastructure WLAN lors de l'association initiale.

Transition BSS rapide sur la liaison descendante

802.11r permet une autre implémentation de la transition BSS rapide, où l'itinérance FT est initiée par le client avec le nouveau point d'accès pour lequel le client se déplace via le DS (système de distribution), et non via l'émission. Dans ce cas, les trames Action FT sont utilisées afin d'initier la négociation de clé au lieu des trames Authentification Système Ouvert.

Fondamentalement, une fois que le client décide qu'il peut se déplacer vers un meilleur AP, le client envoie une trame de demande d'action FT à l'AP d'origine où il est actuellement connecté avant de se déplacer. Le client indique le BSSID (adresse MAC) du point d'accès cible où il veut se déplacer FT. Le point d'accès d'origine transfère cette trame de demande d'action FT au point d'accès cible sur le système de distribution (normalement l'infrastructure filaire), et le point d'accès cible répond au client avec une trame de réponse d'action FT (également sur le DS, afin qu'il puisse enfin l'envoyer par liaison radio au client). Une fois cet échange de trame d'action FT réussi, le client termine l'itinérance FT ; le client envoie la requête de réassociation au point d'accès cible (cette fois par liaison radio) et reçoit une réponse de réassociation du nouveau point d'accès afin de confirmer l'itinérance et la dérivation finale des clés.

En résumé, il y a quatre trames pour négocier la transition BSS rapide et dériver de nouvelles clés de chiffrement, mais ici les trames d'authentification de système ouvert sont remplacées par les trames de requête/réponse d'action FT, qui sont échangées avec le point d'accès cible sur le système de distribution avec le point d'accès actuel. Cette méthode est également valide pour les méthodes de sécurité 802.1X/EAP et PSK, toutes prises en charge par les contrôleurs LAN sans fil Cisco. Cependant, étant donné que cette transition Over-the-DS n'est pas prise en charge et mise en oeuvre par la plupart des clients sans fil du secteur Wi-Fi (et étant donné que les sorties d'échange de trames et de débogage sont fondamentalement les mêmes), des exemples ne sont pas fournis dans ce document. À la place, cette image est utilisée afin de visualiser la transition BSS rapide sur le DS :



FlexConnect avec 802.11r

- L'authentification centrale est prise en charge. Cela inclut la commutation de données locale et centrale. Les points d'accès doivent faire partie du même groupe FlexConnect.
- Authentification locale non prise en charge.
- Le mode autonome n'est pas pris en charge.

Avantages de la norme 802.11r

- Cette méthode est la première qui utilise une hiérarchie de clés clairement définie par l'IEEE sur la norme 802.11 en tant qu'amendement (802.11r), de sorte que la mise en oeuvre de ces techniques FT sont plus compatibles entre les fournisseurs et sans interprétations différentes.
- La norme 802.11r permet d'utiliser plusieurs techniques utiles, en fonction de vos besoins (Over-the-Air et Over-the-DS, pour la sécurité 802.1x/EAP et pour la sécurité PSK).
- Le client sans fil effectue une itinérance rapide et sécurisée vers un nouveau point d'accès sur le même WLAN/SSID, même s'il n'est jamais associé à ce point d'accès, et sans avoir besoin d'enregistrer plusieurs PMKID.
- Il s'agit de la première méthode d'itinérance sécurisée rapide qui permet une itinérance plus rapide, même avec la sécurité PSK, et évite la connexion en 4 étapes qui est requise lors de l'itinérance entre les AP avec WPA/WPA2 PSK. L'objectif principal des méthodes d'itinérance rapide et sécurisée est d'éviter la connexion 802.1X/EAP lorsque cette méthode de sécurité est implémentée ; cependant, pour la sécurité PSK, l'événement d'itinérance est encore plus accéléré avec la norme 802.11r lorsque la connexion en 4 étapes est évitée.

Inconvénients de la norme 802.11r

- Quelques périphériques clients sans fil prennent en charge les transitions Fast BSS et, dans la plupart des cas, ils ne prennent pas en charge toutes les techniques disponibles sur la norme 802.11r.
- Étant donné que ces implémentations sont très jeunes, il n'y a pas suffisamment de résultats de test provenant d'environnements de production réels ou suffisamment de résultats de débogage pour répondre aux éventuelles mises en garde qui peuvent apparaître.
- Lorsque vous configurez un WLAN/SSID afin d'utiliser l'une des méthodes FT, seuls les clients sans fil qui prennent en charge 802.11r peuvent se connecter à ce WLAN/SSID. Les paramètres FT ne sont pas facultatifs pour les clients, de sorte que les clients sans fil qui ne prennent pas en charge 802.11r doivent se connecter avec un WLAN/SSID distinct où FT n'est pas configuré du tout.

802.11r adaptatif

- Certains clients hérités ne peuvent pas s'associer à un WLAN/SSID dont la norme 802.11r est activée, même pour le « mode mixte » (que vous espérez pouvoir avoir sur les mêmes clients SSID qui prennent en charge et qui ne prennent pas en charge la norme 802.11r). C'est lorsque le pilote du demandeur client qui est responsable de l'analyse de l'élément d'information du réseau de sécurité robuste (IE RSN) est ancien et ne connaît pas les suites AKM supplémentaires dans l'IE. En raison de cette limitation, les clients ne peuvent pas envoyer de demandes d'association aux WLAN qui annoncent la prise en charge de la norme 802.11r, et par conséquent, vous devez configurer un WLAN/SSID pour les clients 802.11r et un WLAN/SSID distinct pour les clients qui ne prennent pas en charge la norme 802.11r.

- Pour remédier à ce problème, l'infrastructure LAN sans fil Cisco a introduit la fonctionnalité Adaptive 802.11r. Lorsque le mode FT est défini sur Adaptive au niveau du WLAN, le WLAN annonce l'ID de domaine de mobilité 802.11r sur un WLAN compatible 802.11i. Certains périphériques clients Apple iOS10 identifient la présence de MDIE sur un WLAN 80211i/WPA2 et effectuent une connexion propriétaire afin d'établir une association 802.11r. Une fois que le client a réussi l'association 802.11r, il peut effectuer l'itinérance FT comme dans un WLAN normal compatible 802.11r. L'Adaptatif FT s'applique uniquement à certains appareils Apple iOS10 (et versions ultérieures). Tous les autres clients peuvent continuer à avoir une association 802.11i/WPA2 sur le WLAN, et exécuter la méthode FSR applicable comme prise en charge.

- Pour plus d'informations sur cette nouvelle fonctionnalité introduite pour les périphériques iOS10 afin d'exécuter la norme 802.11r sur un WLAN/SSID où la norme 802.11r n'est pas réellement activée (afin que d'autres clients non-802.11r puissent se connecter), consultez Meilleures pratiques d'entreprise pour les périphériques Cisco IOS sur le LAN sans fil Cisco.

Conclusions

- Gardez à l'esprit que le client est toujours celui qui décide de se déplacer vers un AP

spécifique, et le WLC/AP ne peut pas décider cela pour le client. L'événement d'itinérance est initié par le client sans fil lorsqu'il considère qu'il doit se déplacer.

- Le WLC prend en charge une combinaison de la plupart ou de la totalité des méthodes FSR (Fast-Secure Roaming) sur le même WLAN/SSID. Cependant, sachez que cela ne fonctionne normalement pas, car cela dépend fortement du comportement du client (très différent entre les différents périphériques mobiles) afin de prendre en charge ou même comprendre ce que le WLC tente d'annoncer comme pris en charge. Au lieu de réaliser l'interopérabilité dans un seul SSID, il y a normalement plus de problèmes que ceux qui sont censés être résolus, donc ce n'est pas recommandé. Des tests approfondis avec tous les clients possibles à utiliser sur ce WLAN doivent être effectués si cela est vraiment nécessaire.
- Il est très important de comprendre que des méthodes d'itinérance rapide et sécurisée sont développées afin d'accélérer le processus d'itinérance WLAN lorsque vous passez d'un point d'accès à un autre si la sécurité du WLAN/SSID est activée. Quand aucune sécurité n'est en place, il n'y a rien à accélérer, car le client-AP échange simplement les trames de gestion sans fil qui sont toujours nécessaires lors de l'itinérance entre les AP avant que les trames de données soient envoyées (Open System Authentication from the client, Open System Authentication from the AP, Reassociation Request, and Reassociation Response). Par conséquent, cela ne peut pas aller plus vite. Si vous rencontrez des problèmes d'itinérance sans sécurité, alors il n'y a pas de méthodes d'itinérance rapide pour améliorer l'itinérance, seulement des méthodes pour confirmer si la configuration et la conception du WLAN/SSID sont appropriées pour que les stations clientes sans fil se déplacent en conséquence entre les cellules de couverture AP.
- La norme 802.11r/FT est mise en oeuvre avec WPA2-PSK afin d'accélérer les événements d'itinérance avec cette sécurité et d'éviter la connexion en 4 étapes, comme expliqué dans la section 802.11r.
- Toutes les méthodes présentent des avantages et des inconvénients, mais en fin de compte, vous devez toujours vérifier si les stations clientes sans fil prennent en charge la méthode spécifique que vous souhaitez mettre en oeuvre et si l'infrastructure WLAN Cisco prend en charge toutes les méthodes disponibles. Par conséquent, vous devez sélectionner la meilleure méthode réellement prise en charge par les clients sans fil qui se connectent au WLAN/SSID spécifique. Par exemple, dans certains déploiements, vous pouvez créer un WLAN/SSID avec CCKM pour les téléphones IP sans fil Cisco (qui prennent en charge WPA2/AES avec CCKM, mais pas 802.11r), puis un autre WLAN/SSID avec WPA2/AES via 802.11r/FT pour les clients sans fil qui prennent en charge cette méthode d'itinérance sécurisée rapide (ou utiliser OKC, si c'est ce qui est pris en charge).
- Si les clients sans fil ne prennent pas en charge l'une des méthodes d'itinérance rapide et sécurisée disponibles, alors vous devez accepter le fait que ces clients peuvent toujours expérimenter les retards expliqués dans ce document lors de l'itinérance entre les AP sur un WLAN/SSID avec la sécurité 802.1X/EAP (ce qui peut causer des interruptions sur les applications/services clients).
- Toutes les méthodes, à l'exception de SKC (WPA2 PMKID Caching), sont prises en charge pour l'itinérance rapide et sécurisée entre les AP gérés par différents WLC (intercontroller roaming), tant qu'ils sont sur le même groupe de mobilité.
- CUWN prend entièrement en charge toutes les méthodes d'itinérance Fast-Secure décrites dans cet article lorsque l'authentification 802.1X/EAP est utilisée pour WPA/WPA2. CUWN

ne prend pas en charge l'itinérance Fast-Secure sur les méthodes qui fonctionnent avec WPA2-RSN (CCKM, PMKID Caching/SKC, OKC/PKC) lorsque PSK (WPA2-Personal) est utilisé, où les méthodes Fast-Roaming ne sont généralement pas nécessaires. Cependant, CUWN prend en charge l'itinérance Fast-Secure dans le cas de WPA2-FT (802.11r) avec PSK, comme expliqué dans cet article.

Informations connexes

- Guide de déploiement de la transition rapide BSS 802.11r
- Assistance technique de Cisco et téléchargements

À propos de cette traduction
Cisco a traduit ce document en traduction automatisée vérifiée par une personne dans le cadre d'un service mondial permettant à nos utilisateurs d'obtenir le contenu d'assistance dans leur propre langue.

Il convient cependant de noter que même la meilleure traduction automatisée ne sera pas aussi précise que celle fournie par un traducteur professionnel.