EAP−FAST Authentication with Wireless LAN Controllers and External RADIUS Server Configuration Example

Document ID: 99791

Introduction

This document explains how to configure the wireless LAN controller (WLC) for Extensible Authentication Protocol (EAP) − Flexible Authentication via Secure Tunneling (FAST) authentication with the use of an external RADIUS server. This configuration example uses the Cisco Secure Access Control Server (ACS) as the external RADIUS server to authenticate the wireless client.

This document focuses on how to configure the ACS for Anonymous and Authenticated In−Band (Automatic) Protected Access Credentials (PAC) provisioning to the wireless clients. In order to configure Manual/Out−of−Band PAC provisioning, refer to Manual PAC Provisioning and PAC File Generation for Manual Provisioning for more information.

Prerequisites

Requirements

Ensure that you meet these requirements before you attempt this configuration:

- Basic knowledge of the configuration of lightweight access points (LAPs) and Cisco WLCs
- Basic knowledge of Lightweight Access Point Protocol (LWAPP)
- Knowledge of how to configure an external RADIUS server, such as the Cisco Secure ACS
- Functional knowledge on general EAP framework
- Basic knowledge on security protocols, such as MS−CHAPv2 and EAP−GTC, and knowledge on digital certificates

Components Used

The information in this document is based on these software and hardware versions:

- Cisco 2000 Series WLC that runs firmware 4.0.217.0
- Cisco Aironet 1000 Series LAP
- Cisco Secure ACS that runs version 4.1
- Cisco Aironet 802.11 a/b/g Client Adapter
- Cisco Aironet Desktop Utility (ADU) that runs firmware version 3.6

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. If your network is live, make sure that you understand the potential impact of any command.

Conventions

Refer to the Cisco Technical Tips Conventions for more information on document conventions.

Background Information

The EAP−FAST protocol is a new, publicly accessible IEEE 802.1X EAP type that Cisco developed to support customers that cannot enforce a strong password policy and want to deploy an 802.1X EAP type that does not require digital certificates.

The EAP−FAST protocol is a client−server security architecture that encrypts EAP transactions with a Transport Level Security (TLS) tunnel. EAP−FAST tunnel establishment is based on strong secrets that are unique to users. These strong secrets are called PACs, which the ACS generates by using a master key known only to the ACS.

EAP−FAST occurs in three phases:

- **Phase zero (Automatic PAC provisioning phase)** EAP−FAST phase zero, an optional phase is a tunnel−secured means of providing an EAP−FAST end−user client with a PAC for the user requesting network access. **Providing a PAC to the end−user client is the sole purpose of phase zero**.

  Note: Phase zero is optional because PACs can also be manually provisioned to clients instead of using phase zero.

See the PAC Provisioning Modes section of this document for details.

- **Phase one** In phase one, the ACS and the end−user client establish a TLS tunnel based on the user's PAC credential. This phase requires that the end−user client has been provided a PAC for the user who is attempting to gain network access, and that the PAC is based on a master key that has not expired. No network service is enabled by phase one of EAP−FAST.
- **Phase two** In phase two, user authentication credentials are passed securely using an inner EAP method supported by EAP−FAST within the TLS tunnel to the RADIUS created using the PAC between the client and RADIUS server. EAP−GTC, TLS and MS−CHAP are supported as inner EAP methods. No other EAP types are supported for EAP−FAST.

Refer to How EAP−FAST works for more information.

PAC

PACs are strong shared secrets that enable the ACS and an EAP−FAST end−user client to authenticate each other and establish a TLS tunnel for use in EAP−FAST phase two. The ACS generates PACs by using the active master key and a username.

PAC comprises:

- **PAC−Key** Shared secret bound to a client (and client device) and server identity.
- **PAC Opaque** Opaque field that the client caches and passes to the server. The server recovers the PAC−Key and the client identity to mutually authenticate with the client.
- **PAC−Info** At a minimum, includes the server's identity to enable the client to cache different PACs. Optionally, it includes other information such as the PAC's expiration time.

PAC Provisoning Modes

As mentioned earlier, phase zero is an optional phase.

EAP−FAST offers two options to provision a client with a PAC:

- **Automatic PAC provisioning (EAP−FAST Phase 0, or In−band PAC provisioning)**
- **Manual (Out−of−band) PAC provisioning**

In−band/Automatic PAC provisioning sends a new PAC to an end−user client over a secured network connection. Automatic PAC provisioning requires no intervention of the network user or an ACS administrator, provided that you configure the ACS and the end−user client to support automatic provisioning.

The latest EAP−FAST version supports two different in−band PAC provisioning configuration options:

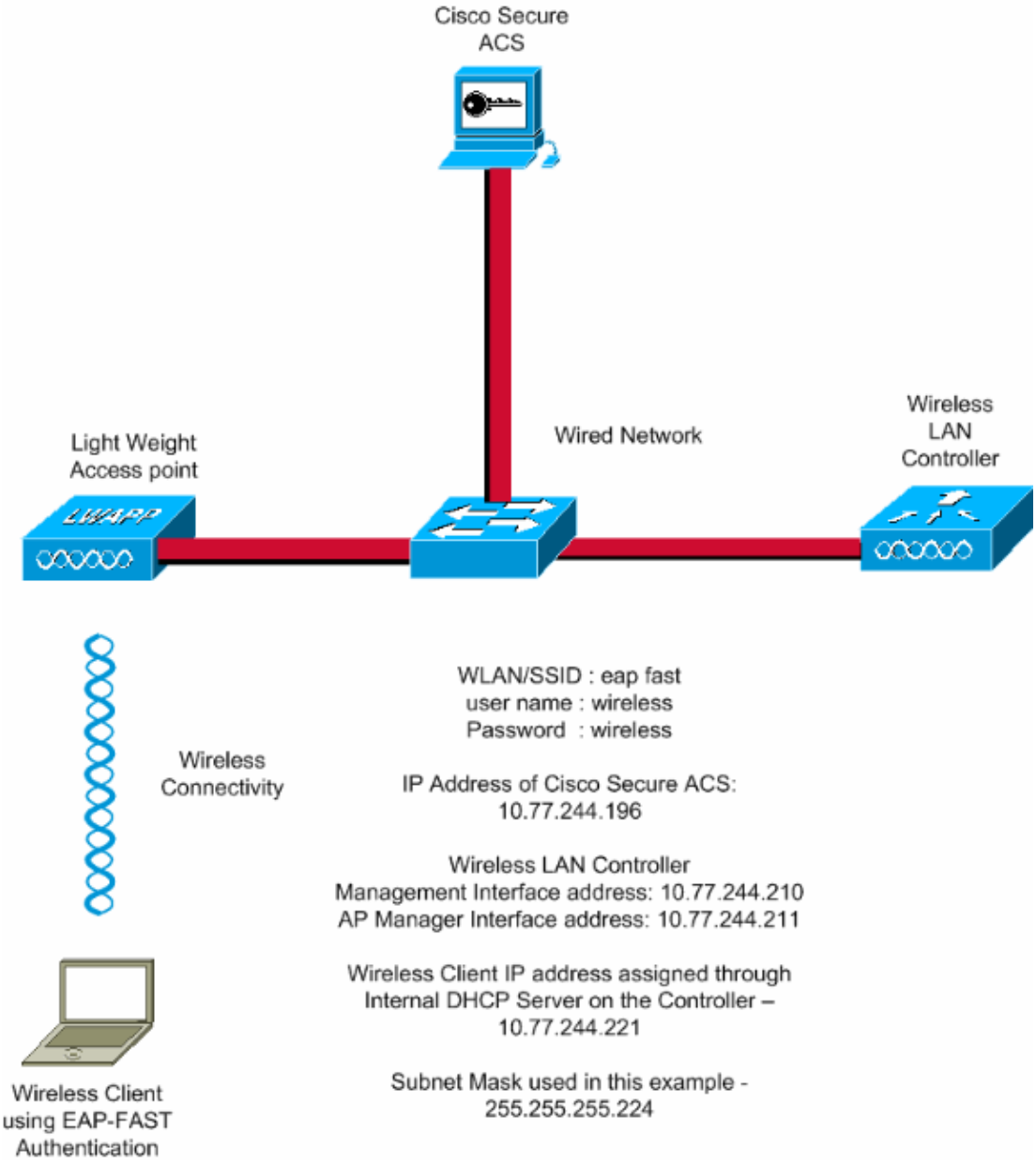
- **Anonymous In−band PAC provisioning**
- **Authenticated In−band PAC provisioning**

Note: This document discusses these in−band PAC provisioning methods and how to configure them.

Out−of−band/Manual PAC provisioning requires an ACS administrator to generate PAC files, which must then be distributed to the applicable network users. Users must configure end−user clients with their PAC files.

Network Diagram and Configuration Setup

In this setup, a Cisco 2006 WLC and a LAP are connected through a hub. An external RADIUS server (Cisco Secure ACS) is also connected to the same hub. All the devices are in the same subnet. The access point (AP) is initially registered to the controller. You must configure the WLC for EAP−FAST authentication. The clients that connect to the AP use EAP−FAST authentication in order to associate with the AP. Cisco Secure ACS is used in order to perform RADIUS authentication.



Configure the WLC for EAP−FAST Authentication

Perform these steps in order to configure the WLC for EAP−FAST authentication:

1. Configure the WLC for RADIUS Authentication through an External RADIUS Server
2. Configure the WLAN for EAP−FAST Authentication

Configure the WLC for RADIUS Authentication through an External RADIUS Server

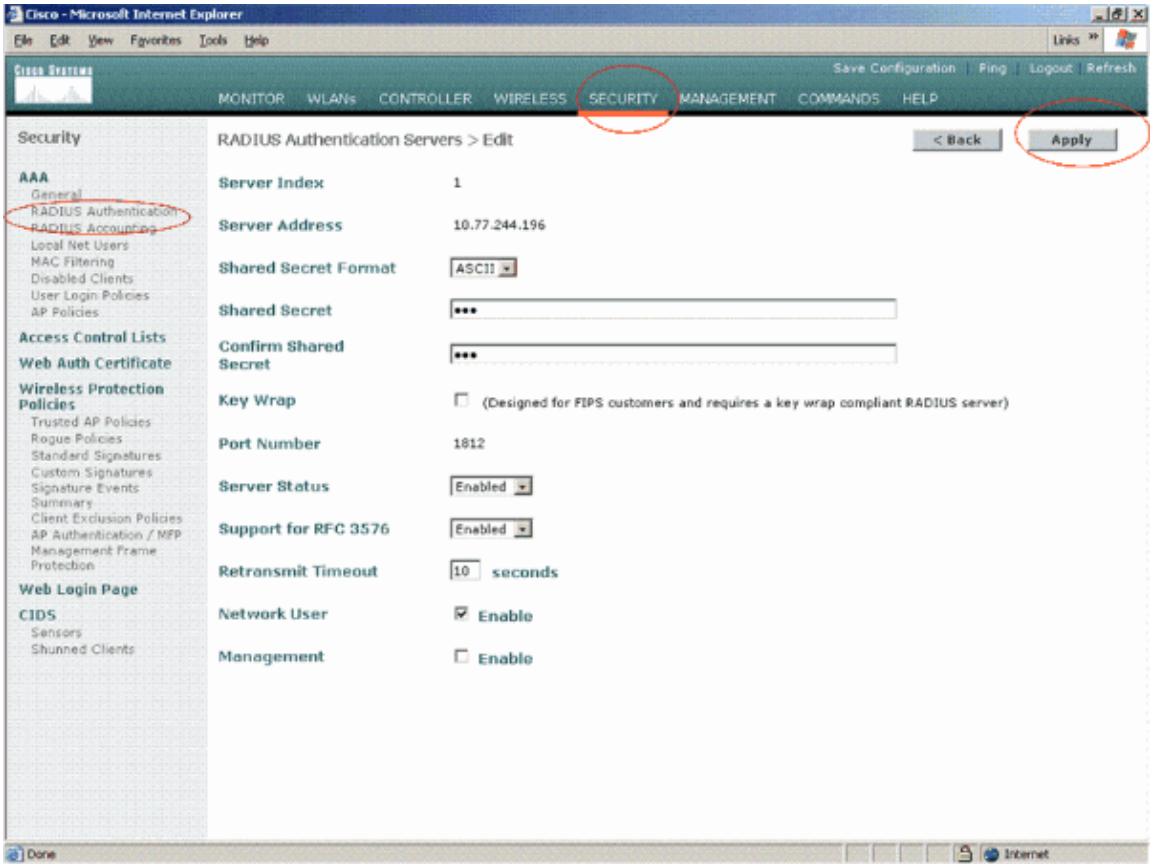
The WLC needs to be configured in order to forward the user credentials to an external RADIUS server. The external RADIUS server then validates the user credentials using EAP−FAST and provides access to the wireless clients.

Complete these steps in order to configure the WLC for an external RADIUS server:

1. Choose **Security** and **RADIUS Authentication** from the controller GUI to display the RADIUS Authentication Servers page. Then, click **New** in order to define a RADIUS server.
2. Define the RADIUS server parameters on the **RADIUS Authentication Servers > New** page. These parameters include:

   - RADIUS Server IP Address
   - Shared Secret
   - Port Number
   - Server Status

   This document uses the ACS server with an IP address of 10.77.244.196.



3. Click **Apply**.

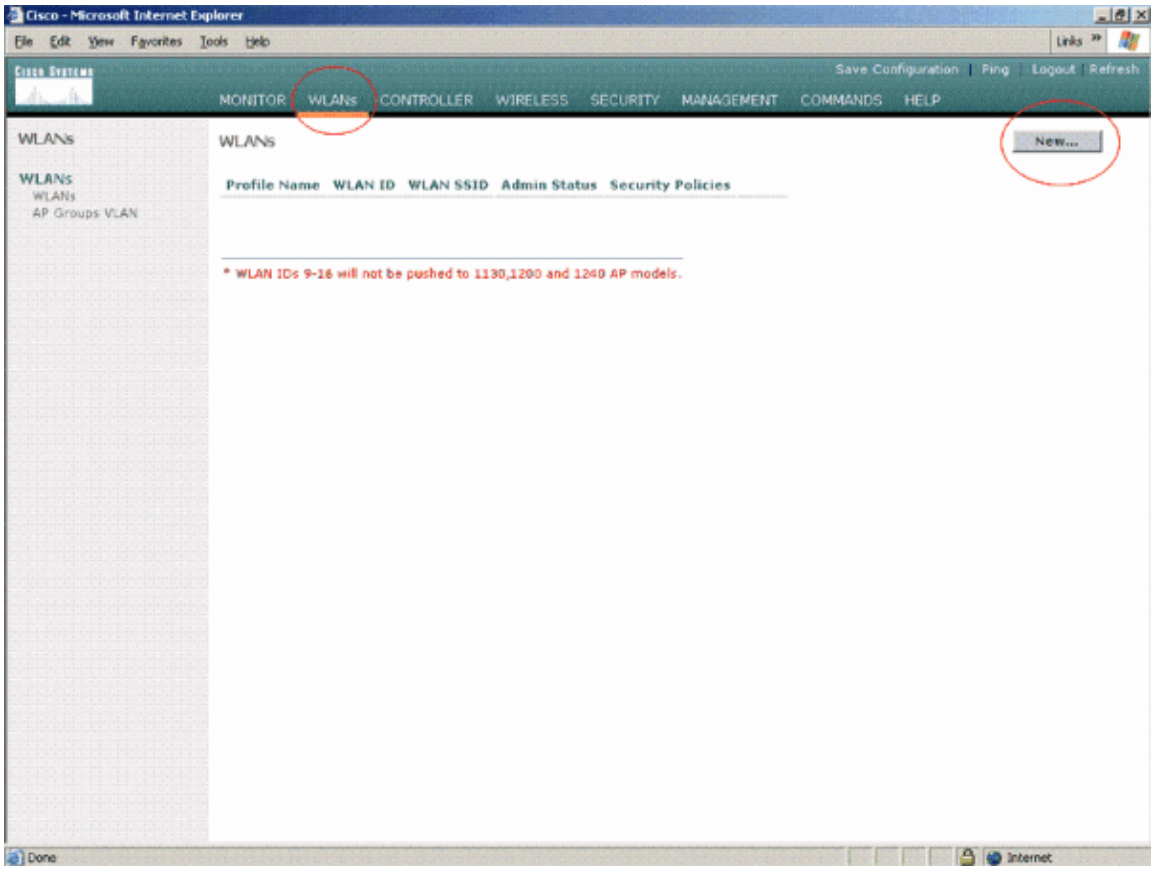
Configure the WLAN for EAP−FAST Authentication

Next, configure the WLAN which the clients use to connect to the wireless network for EAP−FAST authentication and assign to a dynamic interface. The WLAN name configured in this example is **eap fast**. This example assigns this WLAN to the management interface.

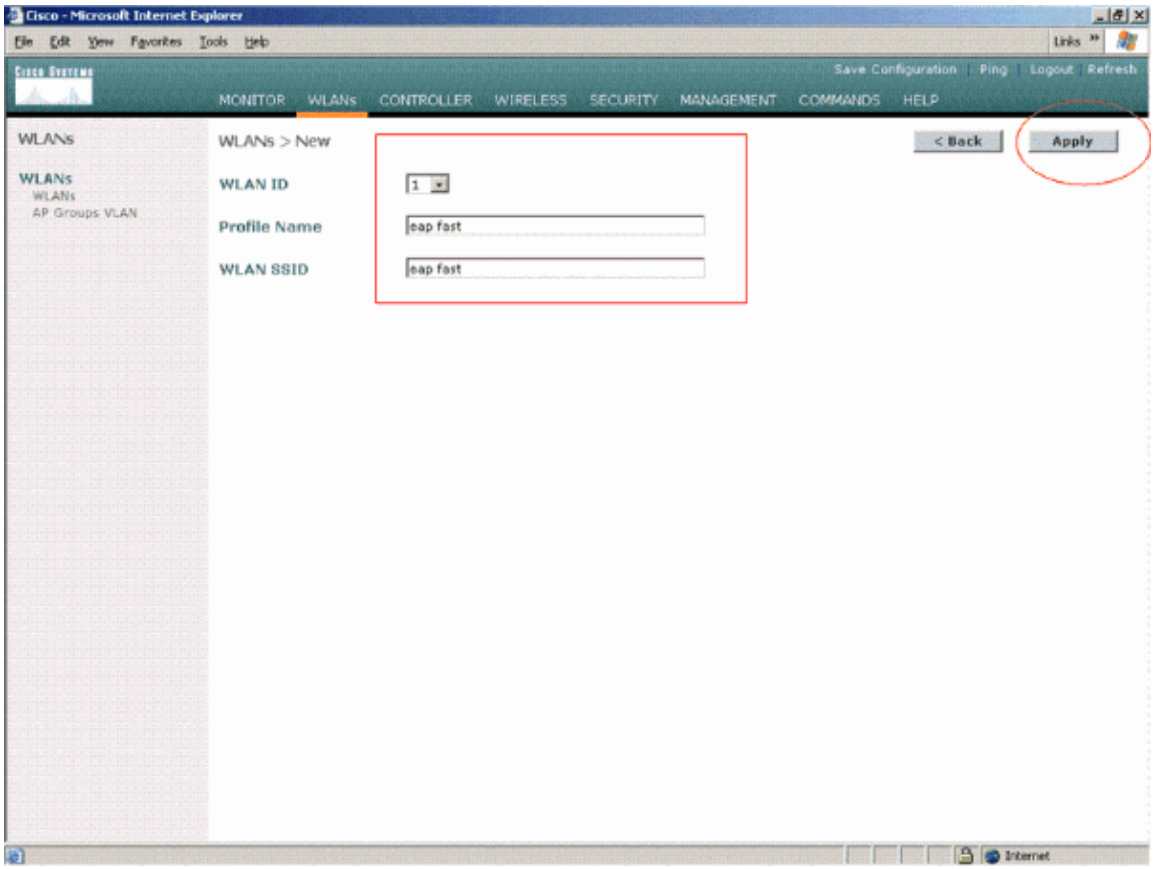
Complete these steps in order to configure the **eap fast** WLAN and its related parameters:

1. Click **WLANs** from the GUI of the controller in order to display the WLANs page.

   This page lists the WLANs that exist on the controller.
2. Click **New** in order to create a new WLAN.

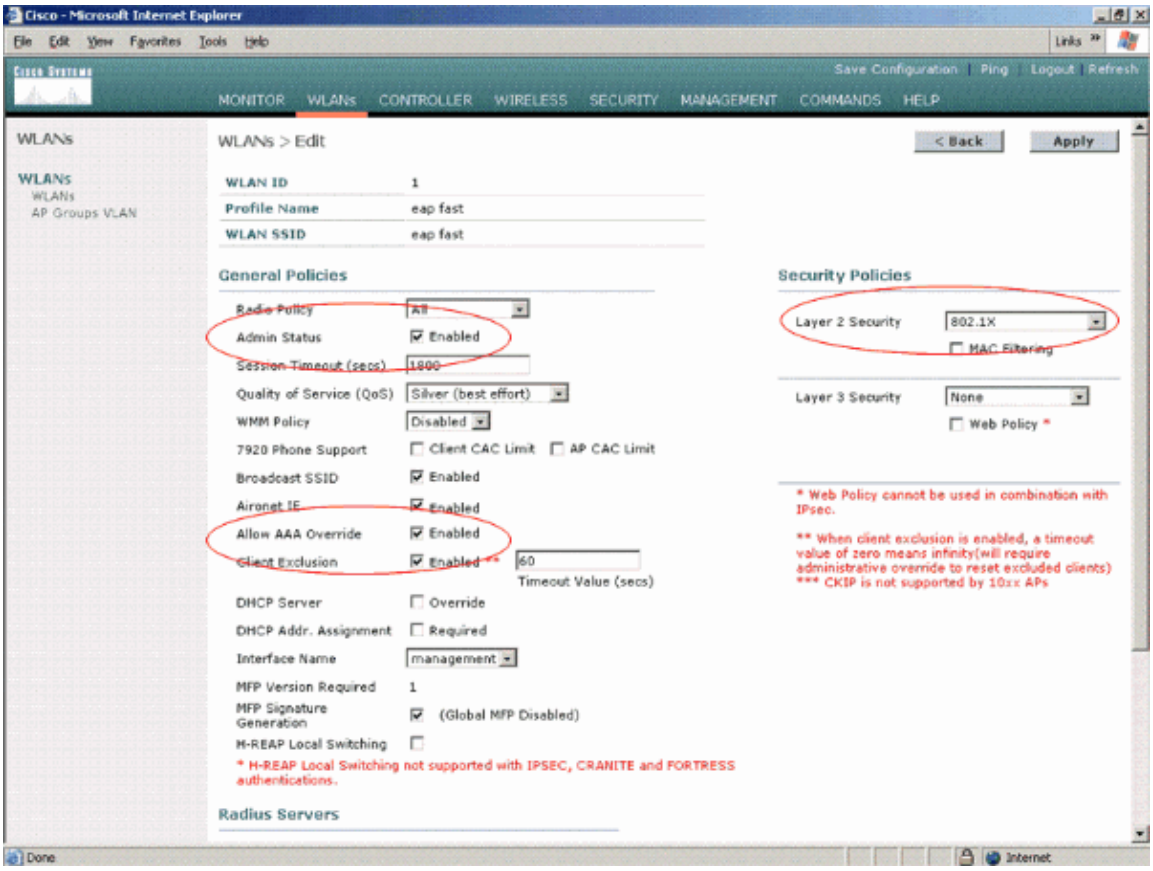


3. Configure the **eap fast** WLAN SSID name, profile name and WLAN ID on the WLANs > New page. Then, click **Apply**.



4. Once you create a new WLAN, the **WLAN > Edit** page for the new WLAN appears. On this page, you can define various parameters specific to this WLAN. This includes General Policies, RADIUS Servers, Security Policies, and 802.1x Parameters.
5. Check the **Admin Status** check box under General Policies in order to enable the WLAN.
6. If you want the AP to broadcast the SSID in its beacon frames, check the **Broadcast SSID** check box. Check the **Allow AAA Override** check box if you want to override the WLC configurations by the RADIUS server.
7. Choose the appropriate RADIUS server from the pull−down menu under RADIUS Servers. Under Security Policies, choose **802.1x** from the Layer 2 Security drop−down menu. You can also use any other authentication method (WPA/WPA2 with 802.1x) that involves RADIUS server for authentication.

   This example uses 802.1x with dynamic WEP encryption as Layer 2 security for this WLAN. The other parameters can be modified based on the requirement of the WLAN network.



8. Click **Apply**.

   Note: This is the only EAP setting that needs to be configured on the controller for EAP authentication. All other configurations specific to EAP−FAST need to be done on the RADIUS server and the clients that need to be authenticated.

Configure the RADIUS Server for EAP−FAST Authentication

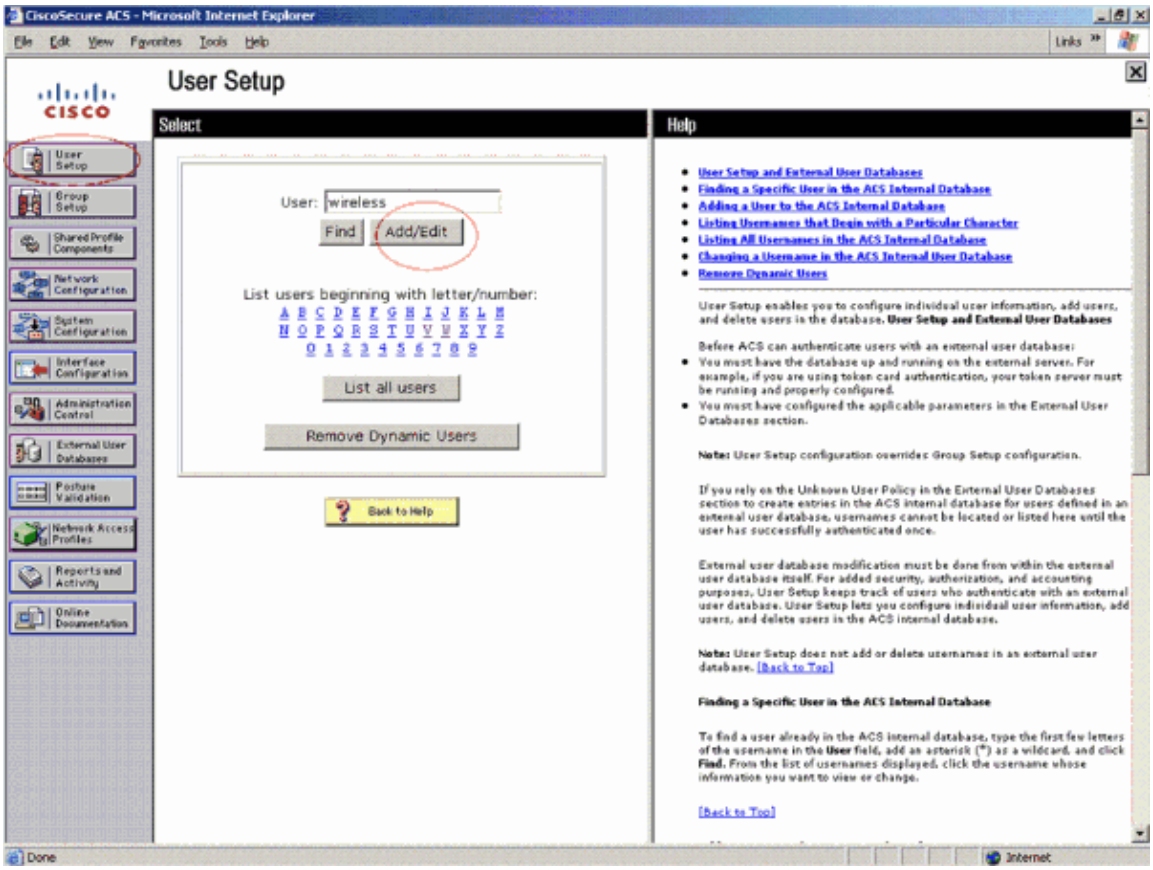
Perform these steps in order to configure the RADIUS server for EAP−FAST authentication:

1. Create a User Database to Authenticate EAP−FAST Clients
2. Add the WLC as AAA Client to the RADIUS Server
3. Configure EAP−FAST Authentication on the RADIUS Server with Anonymous In−band PAC Provisioning
4. Configure EAP−FAST Authentication on the RADIUS Server with Authenticated In−band PAC Provisioning

Create a User Database to Authenticate EAP−FAST Clients

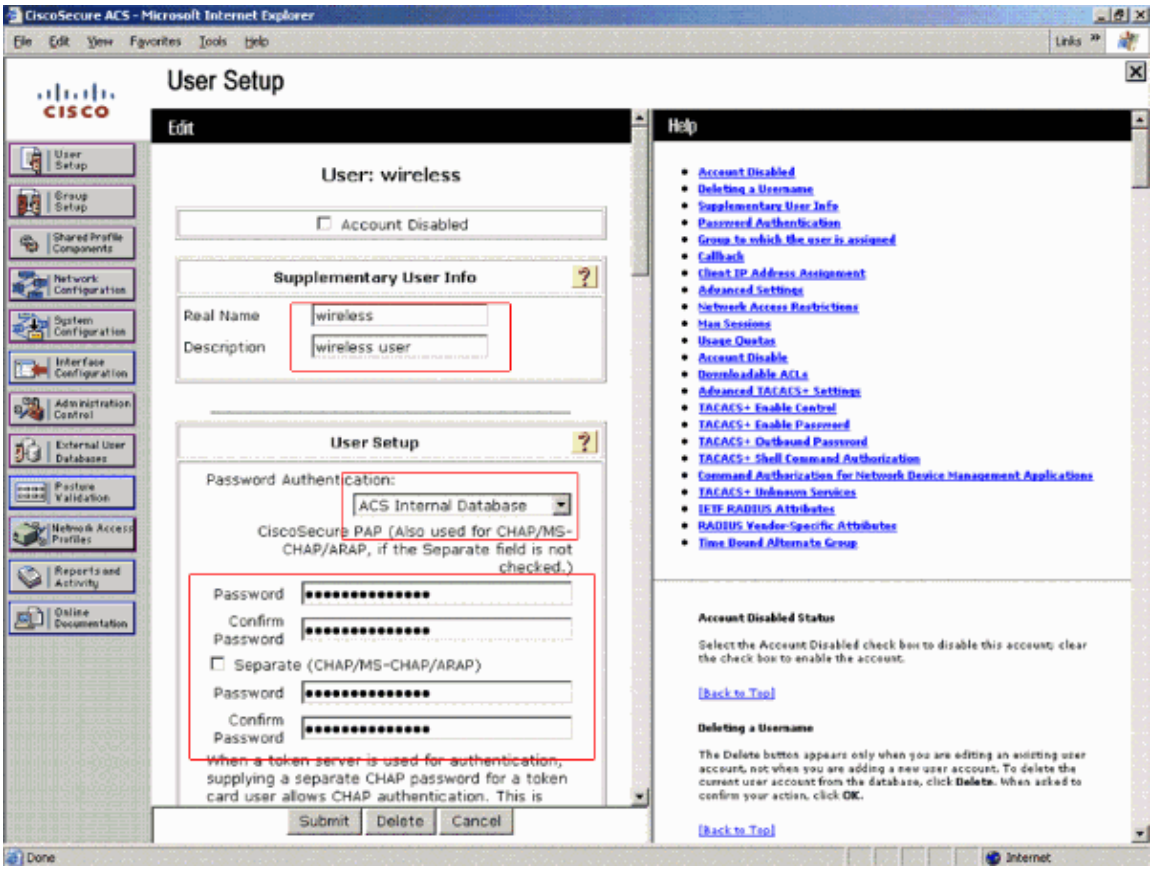
Complete these steps in order to create a user database for EAP−FAST clients on the ACS. This example configures username and password of the EAP−FAST client as **wireless** and **wireless**, respectively.

1. From the ACS GUI in the navigation bar, select **User Setup**. Create a new user **wireless**, then click **Add/Edit** in order to go to the Edit page of this user.



2. From the User Setup Edit page, configure Real Name and Description as well as the Password settings as shown in this example.

   This document uses **ACS Internal Database** for Password Authentication.
3. Choose **ACS Internal Database** from the **Password Authentication** drop−down box.

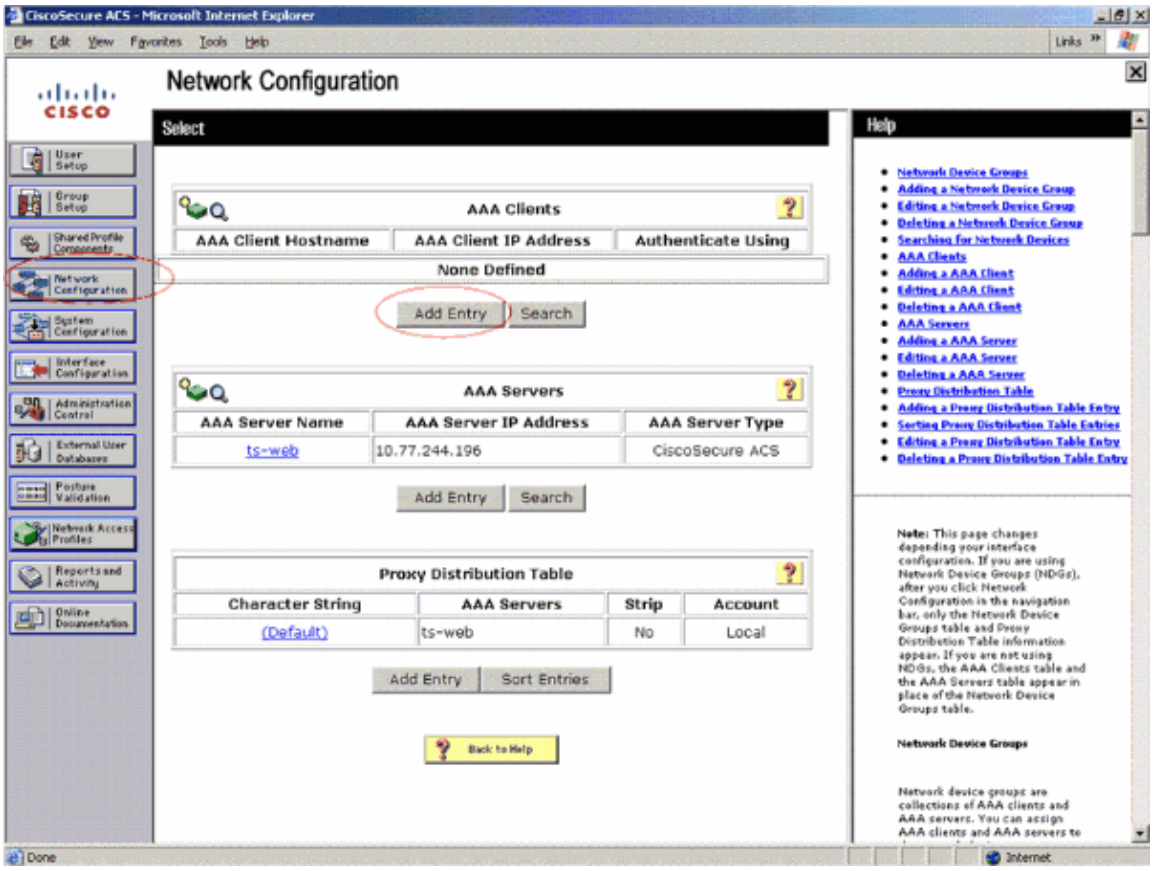


4. Configure all the other required parameters and click **Submit**.

Add the WLC as AAA Client to the RADIUS Server

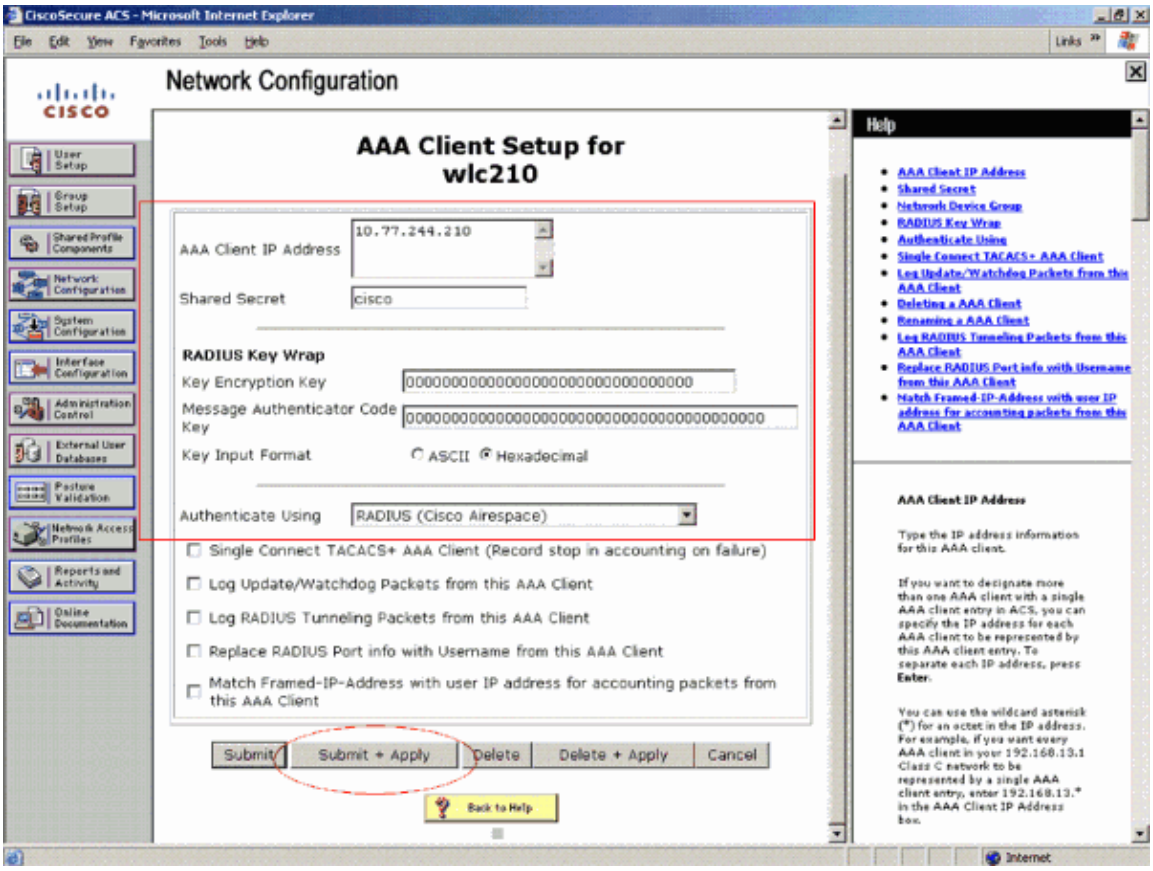
Complete these steps in order to define the controller as an AAA client on the ACS server:

1. Click **Network Configuration** from the ACS GUI. Under the **Add AAA client** section of the Network Configuration page, click **Add Entry** in order to add the WLC as the AAA client to the RADIUS server.



2. From the AAA Client page, define the name of the WLC, IP address, shared secret and authentication method (RADIUS/Cisco Airespace). Refer to the documentation from the manufacturer for other non−ACS authentication servers.

   Note: The shared secret key that you configure on the WLC and the ACS server must match. The shared secret is case sensitive.
3. Click **Submit+Apply**.



Configure EAP−FAST Authentication on the RADIUS Server with Anonymous In−band PAC Provisioning

Anonymous In−band Provisioning

This is one of the two in−band provisioning methods in which the ACS establishes a secured connection with the end−user client for the purpose of providing the client with a new PAC. This option allows an anonymous TLS handshake between the end−user client and ACS.

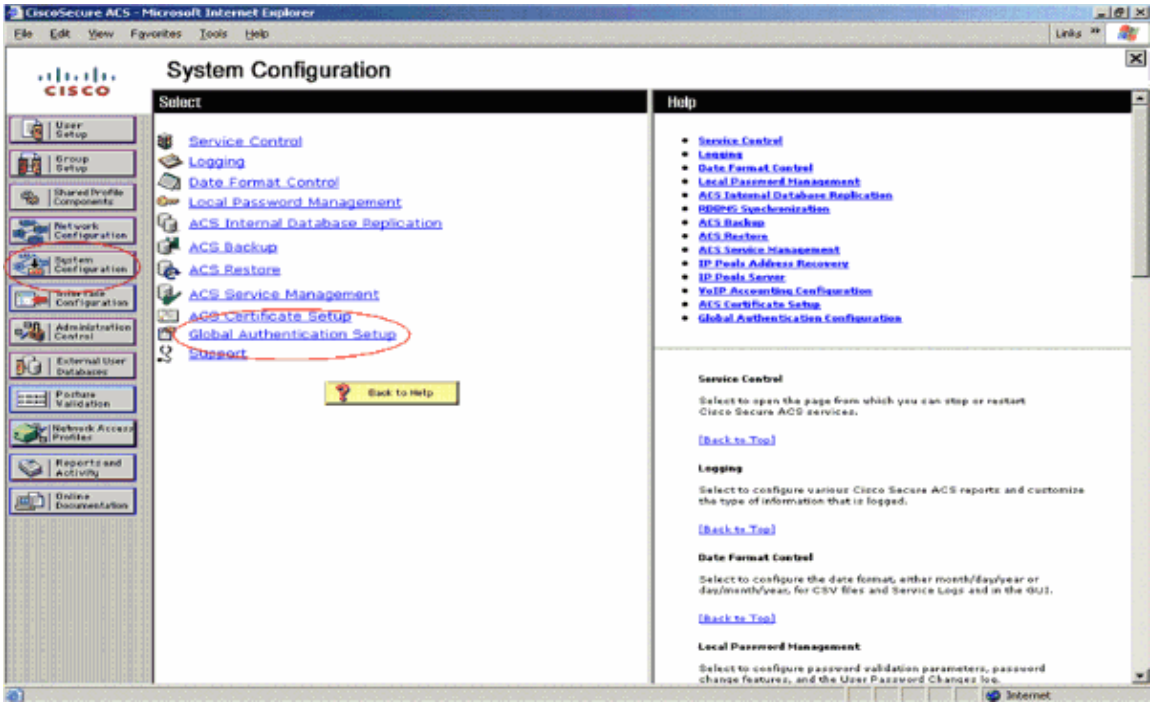
This method operates inside an Authenticated Diffie−HellmanKey Agreement Protocol (ADHP) tunnel before the peer authenticates the ACS server.

Then, the ACS requires EAP−MS−CHAPv2 authentication of the user. At successful user authentication, the ACS establishes a Diffie−Hellman tunnel with the end−user client. The ACS generates a PAC for the user and sends it to the end−user client in this tunnel, along with information about this ACS. This method of provisioning uses EAP−MSCHAPv2 as the authentication method in phase zero and EAP−GTC in phase two.

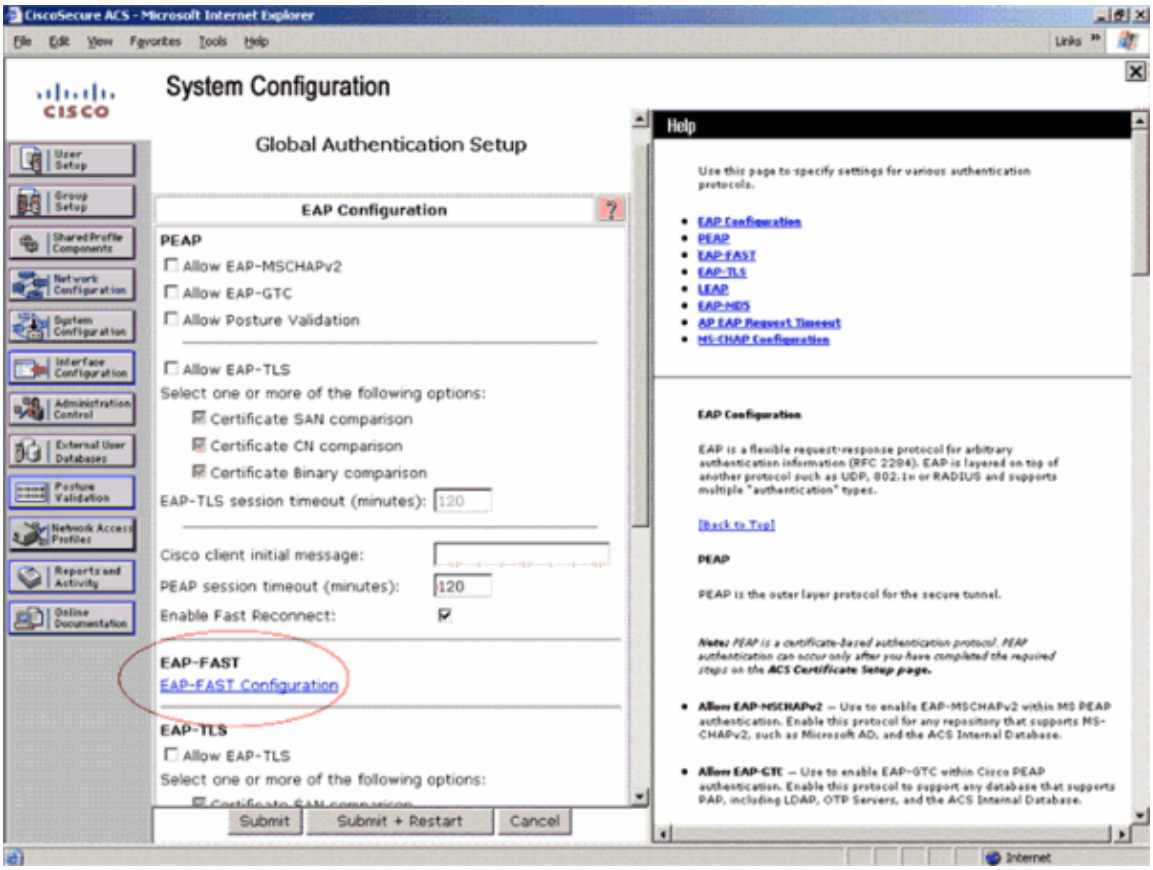
Because an unauthenticated server is provisioned, it is not possible to use a plain text password. Therefore, only MS−CHAP credentials can be used inside the tunnel. MS−CHAPv2 is used to prove the peer's identity and receive a PAC for further authentication sessions (EAP−MS−CHAP will be used as inner method only).

Complete these steps in order to configure EAP−FAST authentication in the RADIUS server for anonymous in−band provisioning:

1. Click **System Configuration** from the RADIUS server GUI. From the System Configuration page, choose **Global Authentication Setup**.



2. From the Global Authentication setup page, click **EAP−FAST Configuration** in order to go to the EAP−FAST settings page.



3. From the EAP−FAST Settings page, check the **Allow EAP−FAST** check box to enable EAP−FAST in the RADIUS server.
4. Configure the **Active/Retired master key TTL** (Time−to−Live) values as desired, or set it to the default value as shown in this example.

   Refer to Master Keys for information about Active and Retired master keys. Also, refer to Master Keys and PAC TTLs for more information.

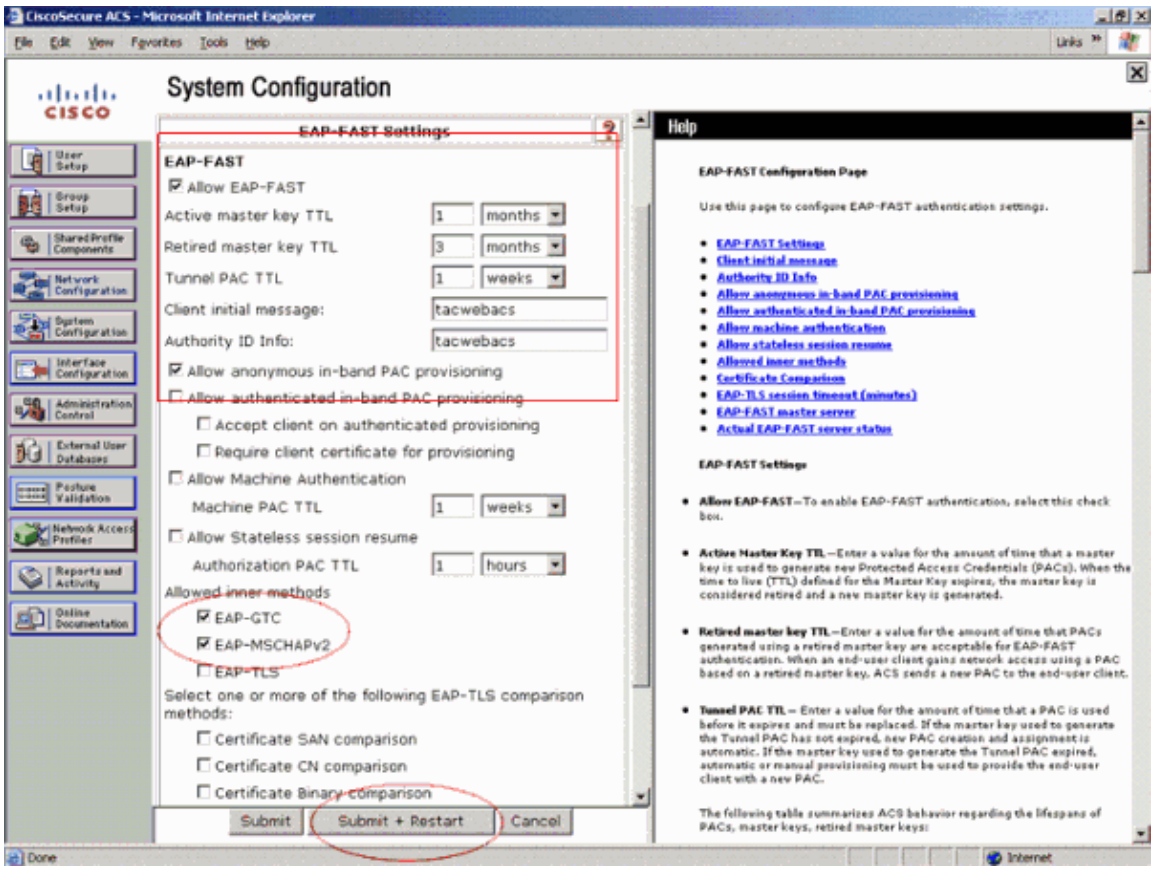
The **Authority ID Info** field represents the textual identity of this ACS server, which an end user can use to determine which ACS server to be authenticated against. Filling in this field is mandatory.

The **Client initial display message** field specifies a message to be sent to users who authenticate with an EAP−FAST client. Maximum length is 40 characters. A user will see the initial message only if the end−user client supports the display.

5. If you want the ACS to perform anonymous in−band PAC provisioning, check the **Allow anonymous in−band PAC provisioning** check box .

   Allowed inner methods This option determines which inner EAP methods can run inside the EAP−FAST TLS tunnel. For anonymous in−band provisioning, you must enable EAP−GTC and EAP−MS−CHAP for backward compatibility. If you select **Allow anonymous in−band PAC provisioning**, you must select EAP−MS−CHAP (phase zero) and EAP−GTC (phase two).
6. Click **Submit+Restart**.

This screenshot shows the steps in this section:



Configure EAP−FAST Authentication on the RADIUS Server with Authenticated In−band PAC Provisioning

Authenticated In−band Provisioning

EAP−FAST has been enhanced to support an authenticated tunnel (using the server certificate), which is where PAC provisioning occurs. This mode provisions an end−user client with a PAC by using EAP−FAST phase zero with TLS server−side authentication. This option requires that you install a server certificate and a trusted root CA on the ACS.

The new cipher suites, that are enhancements to EAP−FAST and specifically the server certificate, are used. Because the server is authenticated as part of setting up the tunnel, weaker EAP methods such as EAP−GTC can be used inside the tunnel to provide supplicant authentication.

By default, the ACS supports TLS server−side authentication. However, if the client sends the user certificate to the ACS, mutual TLS authentication is performed and inner methods are bypassed.

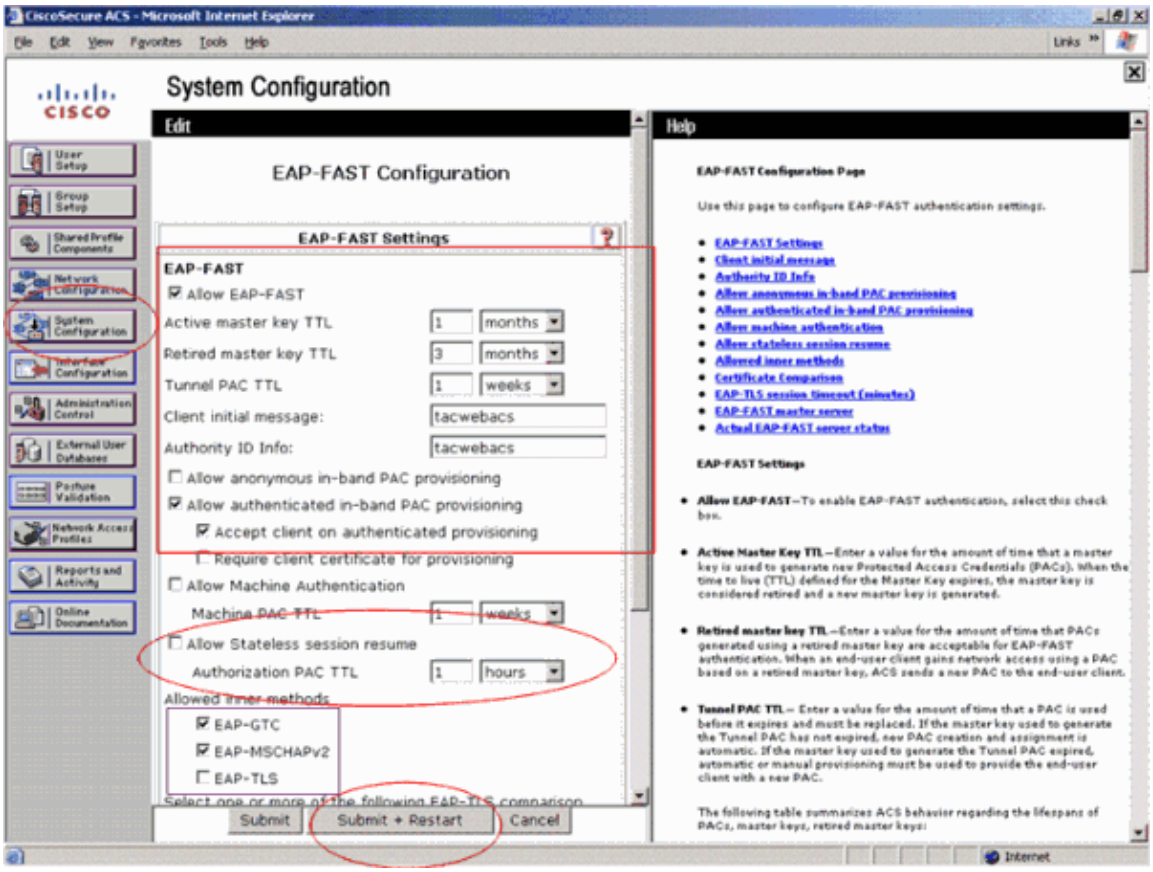
1. Repeat steps1 to 4 of the Anonymous In−band Provisioning configuration in order to configure the other EAP−FAST settings on the RADIUS server.
2. If you want the ACS to perform anonymous in−band PAC provisioning, check the **Allow authenticated in−band PAC provisioning** check box.

    a. **Accept client on authenticated provisioning** This option is only available when the allow authenticated in−band PAC provisioning option is selected. The server always sends an Access−Reject at the end of the provisioning phase, which forces the client to reauthenticate using the tunnel PAC. This option enables the ACS to send an Access−Accept to the client at the end of the provisioning phase.
    b. Uncheck the **Allow Stateless session resume option** check box if you do not want the ACS to provision authorization PACs for EAP−FAST clients and to always perform phase two of EAP−FAST.
    c. **Allowed inner methods** If you select **Allow authenticated in−band PAC provisioning**, the inner method in the authentication phase is negotiable (**EAP−GTC is used by default in phase zero**). Select one or more of these inner methods:

        ◊ EAP−GTC In order to enable EAP−GTC in EAP FAST authentication, check this box.
        ◊ EAP−MS−CHAPv2 In order to enable EAP−MS−CHAPv2 in EAP FAST authentication, check this box.
        ◊ EAP−TLS In order to enable EAP−TLS in EAP FAST authentication, check this box.

    Note: The ACS always runs the first enabled EAP method. For example, if you select EAP−GTC and EAP−MS−CHAPv2, then the first enabled EAP method is EAP−GTC.

    All these configurations specific to authenticated in−band PAC provisioning are mentioned in this example:



3. Click **Submit+Restart**.

Set up the Server Certificate on ACS for Authenticated In−band Provisioning

EAP−FAST has been enhanced to support an authenticated tunnel (using the server certificate) inside which PAC provisioning occurs.

Note: This option requires setting up a server certificate and a trusted root CA on the ACS.

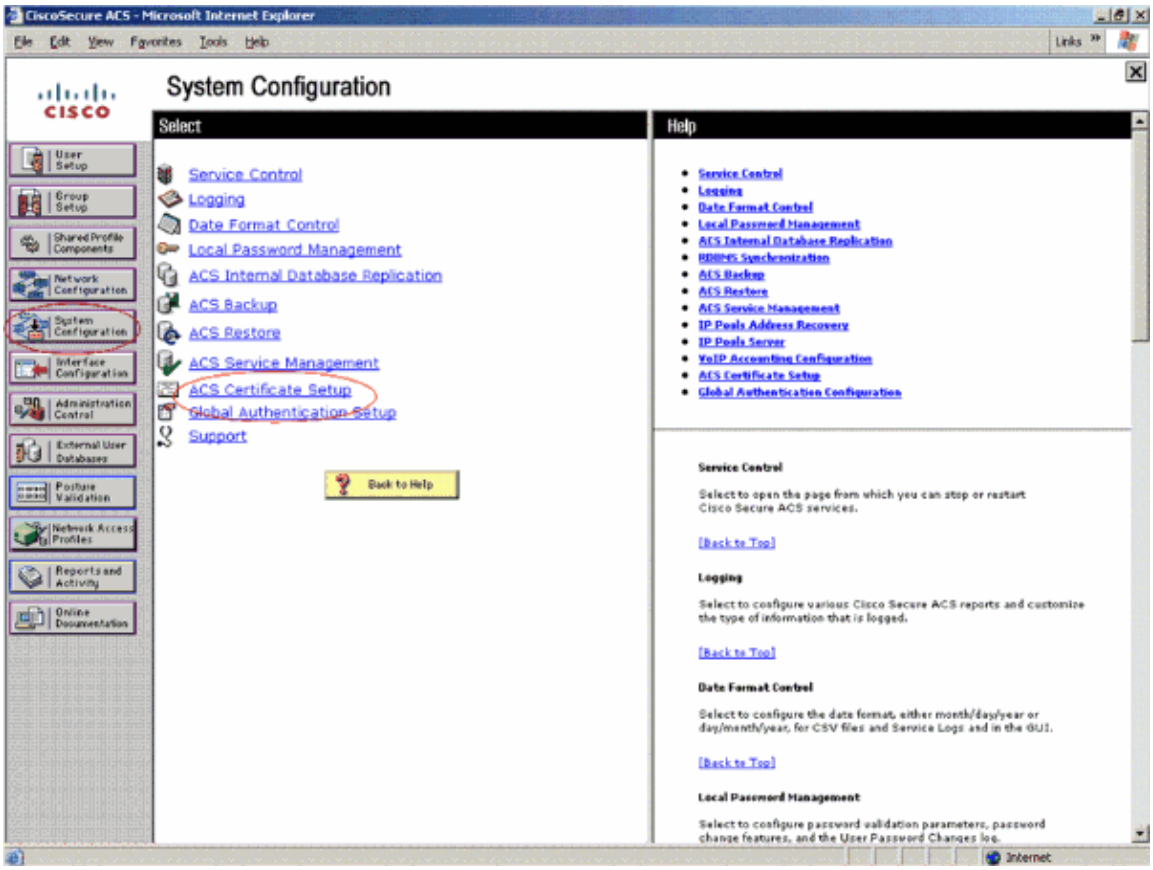
There are several methods available to set up the server certificate on the ACS. This document explains how to generate a self−signed certificate on the ACS and import it to the client's trusted Certificate Authority list.

Generate Self−Signed Certificate in ACS

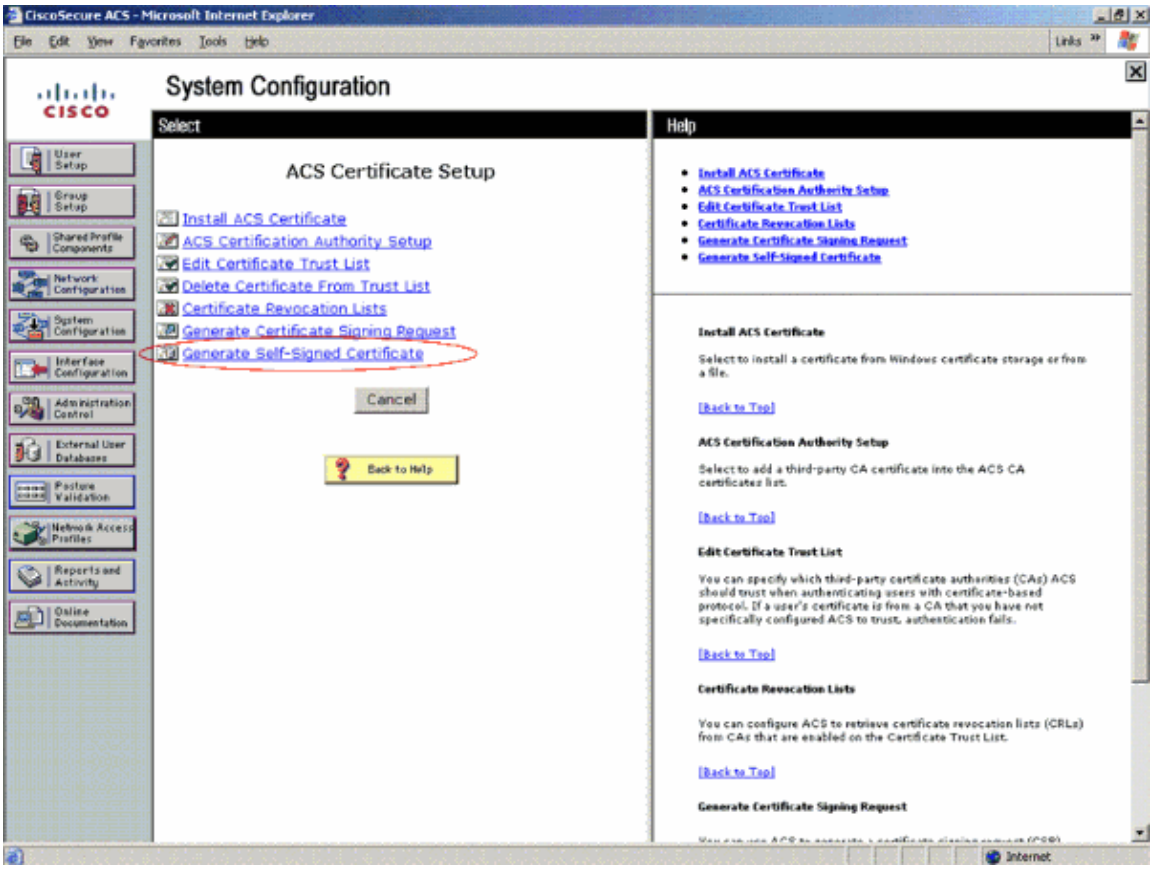
The ACS supports TLS/SSL−related protocols, which includes PEAP, EAP−FAST, and HTTPS, that require the use of digital certificates. The employment of self−signed certificates is a way for administrators to meet this requirement without having to interact with a CA to obtain and install the certificate for the ACS.

Complete these steps in order to generate the self−signed certificate in the ACS:

1. Click **System Configuration** from the RADIUS server GUI. From the System Configuration page, choose **ACS Certificate Setup**.



2. The ACS Certificate Setup lists various options to set up the certificate on the ACS. For this example, choose **Generate Self−Signed Certificate**.



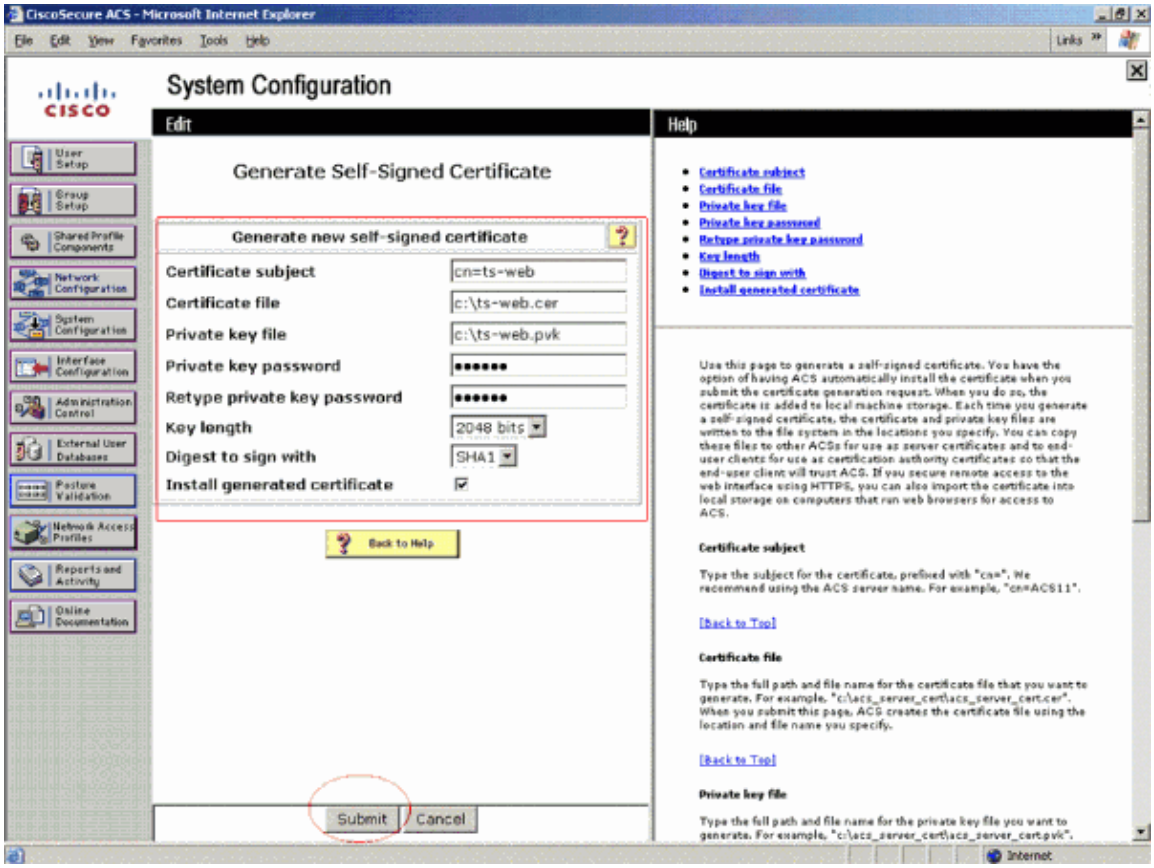
The Generate Self−Signed Certificate Edit page appears.

3. From this page, complete these steps:

    a. In the **Certificate subject** box, enter the certificate subject in the form **cn=XXXX**.
    b. In the **Certificate file** box, enter the full path and file name for the certificate file. In the Key length box, select the key length.
    c. In the **Private key file** box, enter the full path and file name for the private key file.
    d. In the **Private key password** box, enter the private key password. In the **Retype private key password** box, retype the private key password.
    e. In the **Digest to sign with** box, select the hash digest (**SHA1** in this example) to be used to encrypt the key.
    f. In order to install the self−signed certificate when you submit the page, select the **Install generated certificate** option.

    Note: If you select the Install generated certificate option, you must restart ACS services after you submit this form for the new settings to take effect. If you do not select the Install generated certificate option, the certificate file and private key file are generated and saved when you click Submit in the next step. However, these are not installed in local machine storage.

Here is an example of an Editing a Self−Signed Certificate page:



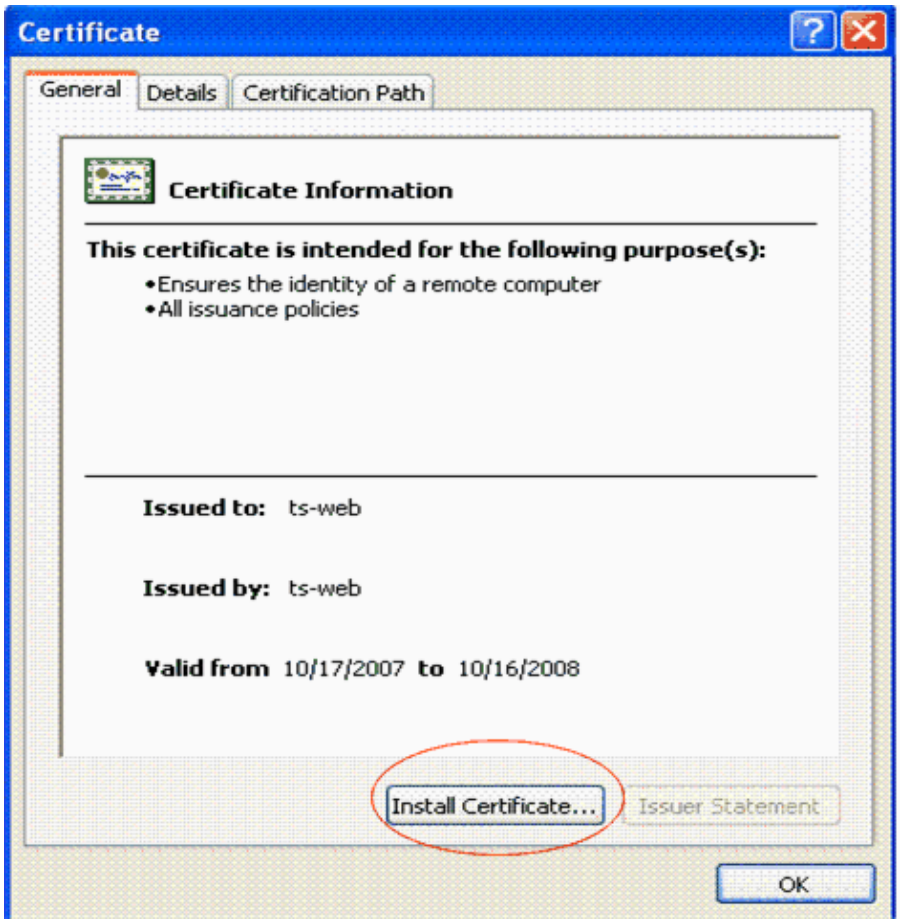
Note: The field values entered (ts−web) here are the example values. You can use any field values or names to generate a self−signed certificate on the ACS. All the fields must be filled in.

Import the Self−Signed Certificate to the Client

You need to import the self−signed certificate generated on the ACS to the client's root Certification Authoritiy list in order for the client to authenticate the server as using a valid certificate.

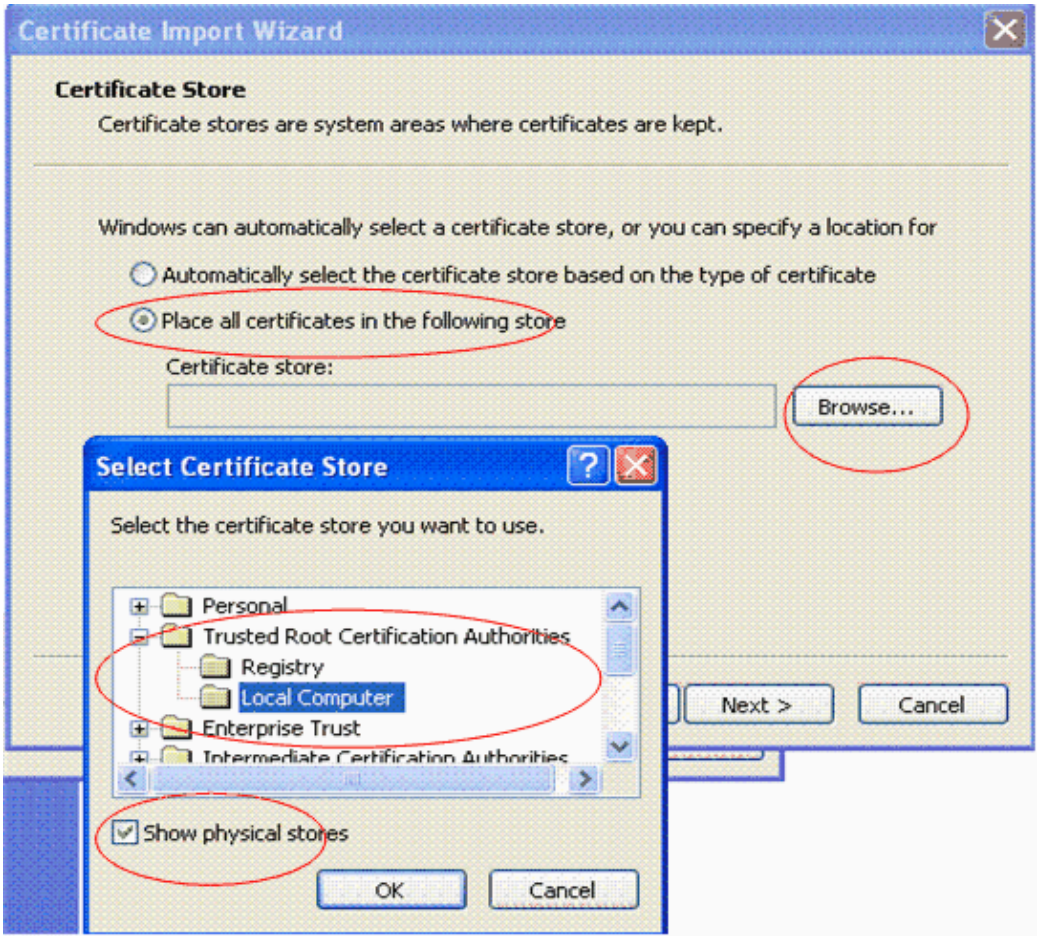
Complete these steps in the client:

1. Copy the certificate from its location on the ACS to the client.
2. Right−click the **.cer** file and click **install certificate**.



3. Click **Next**.
4. Choose **Place all certificates in the following store** and click **Browse**.

   The Select Certificate store window will pop−up.
5. From the Select Certificate Store window, check the **Show physical stores** check box.
6. Expand **Trusted Root Certification Authorities** from the certificate tree, select **Local Computer**, and click **OK**.



7. Click **Next**, click **Finish**, and click **OK**.

   A Certificate Import Wizard appears that shows the import was successful.

Configure the Client for EAP−FAST Authentication

Complete these steps in order to configure the client for EAP−FAST authentication:

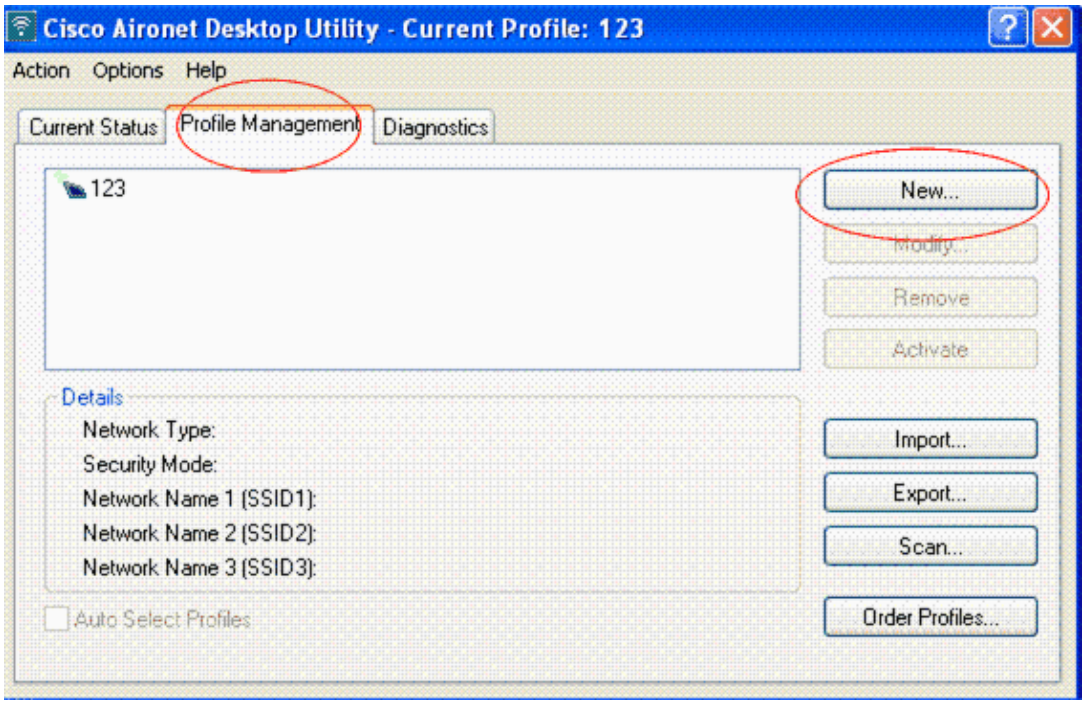
1. Configure Client for Anonymous In−band Provisioning
2. Configure Client for Authenticated In−band Provisioning

Configure Client for Anonymous In−band Provisioning

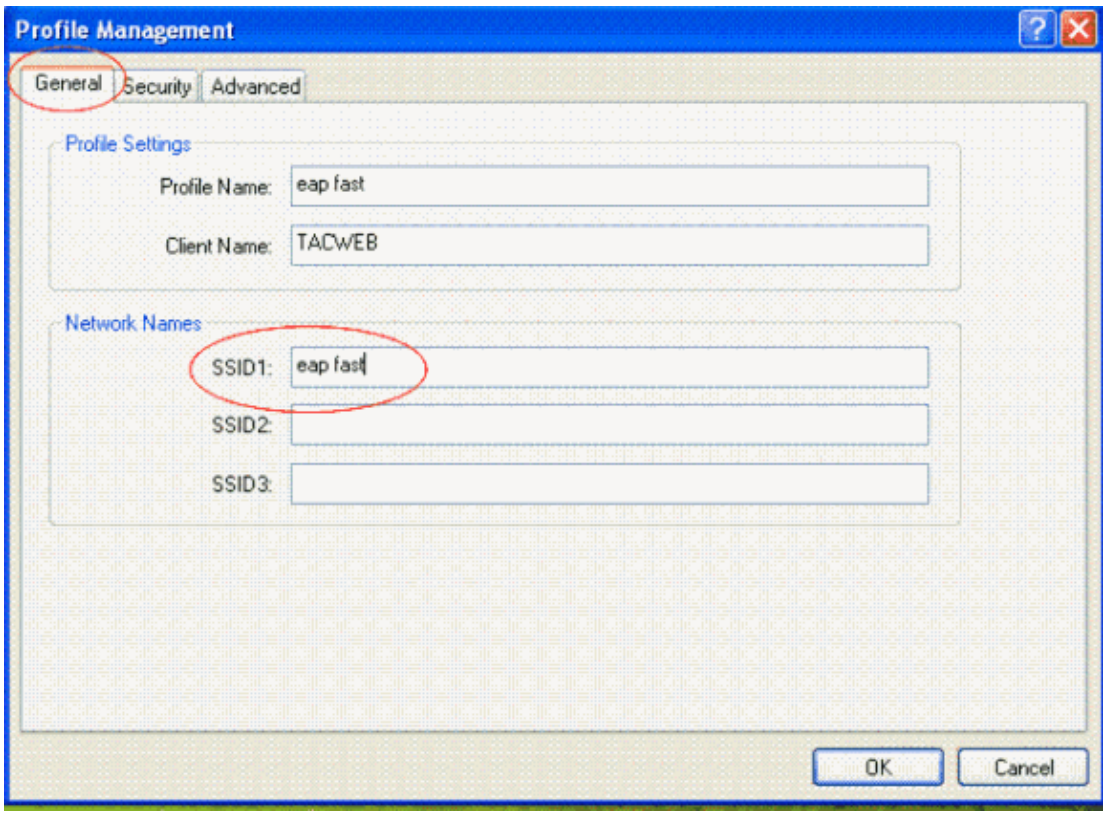
Complete these steps in order to configure the wireless client for anonymous in−band provisioning:

1. From the Aironet Desktop Utility window, click **Profile Management** > **New** in order to create a profile for EAP−FAST user.

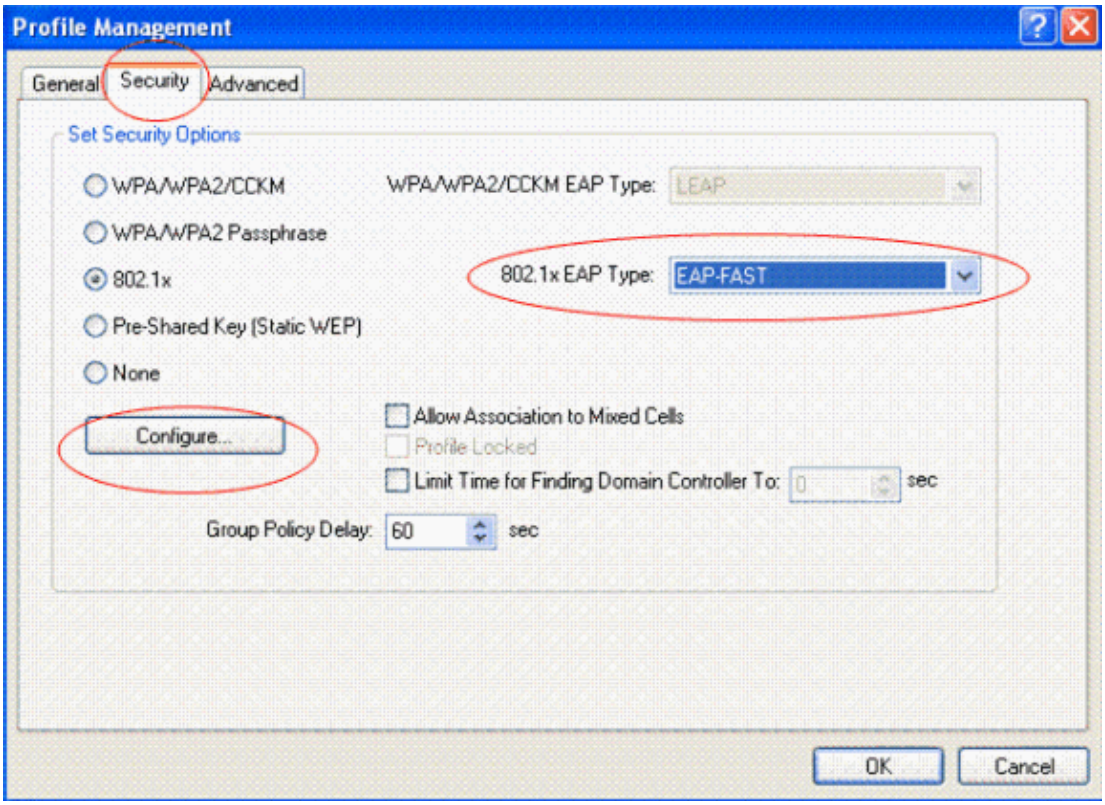
   As mentioned earlier, this document uses the WLAN/SSID name as **eap fast** for wireless client.



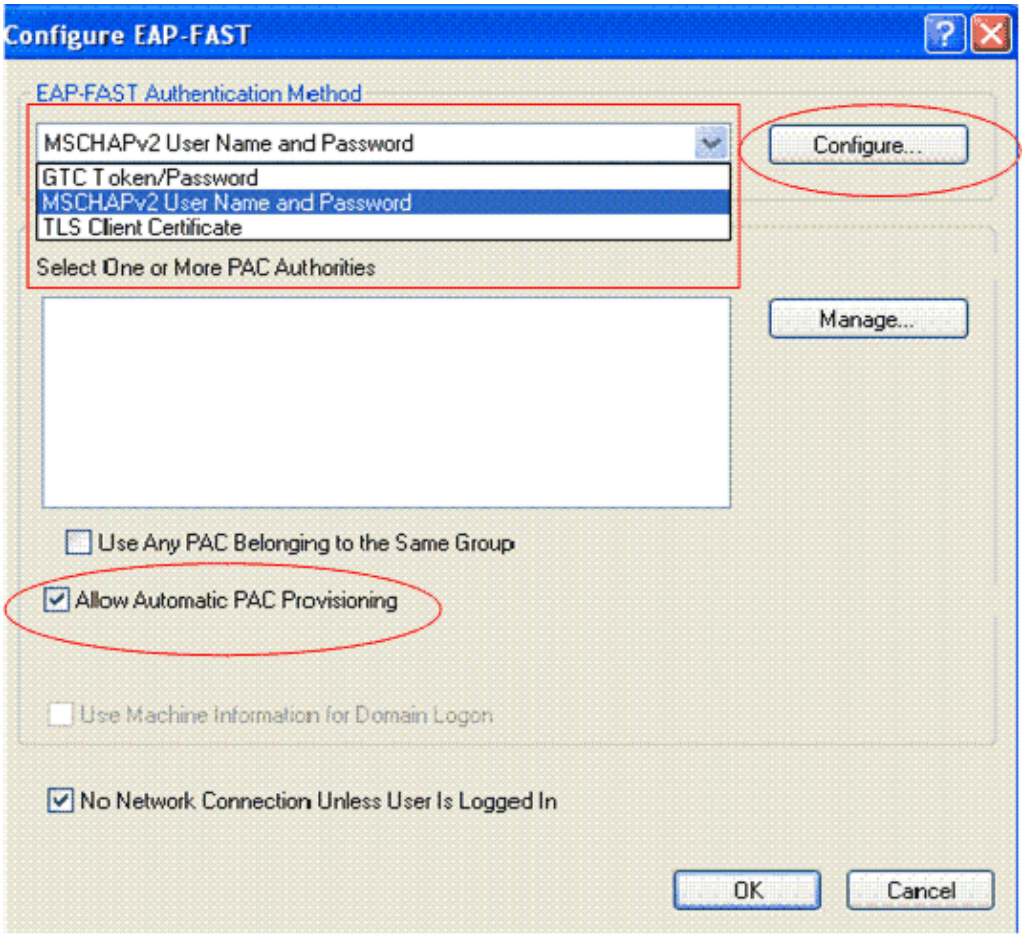
2. From the Profile Management window, click the **General** tab and configure the Profile Name, Client Name and SSID name as shown in this example. Then, click **OK**.



3. Click the **Security** tab and choose **802.1x** as the Set Security Option with the 802.1x EAP Type as **EAP−FAST**. Click **Configure** in order to configure the EAP−FAST setting.



4. From the Configure EAP−FAST window, check the **Allow Automatic PAC Provisioning** check box. If you want to configure anonymous PAC provisioning, EAP−MS−CHAP will be used as the only inner method in phase zero.
5. Choose **MSCHAPv2 User Name and Password** as the authentication method from the EAP−FAST Authentication Method drop−down box. Click **Configure**.

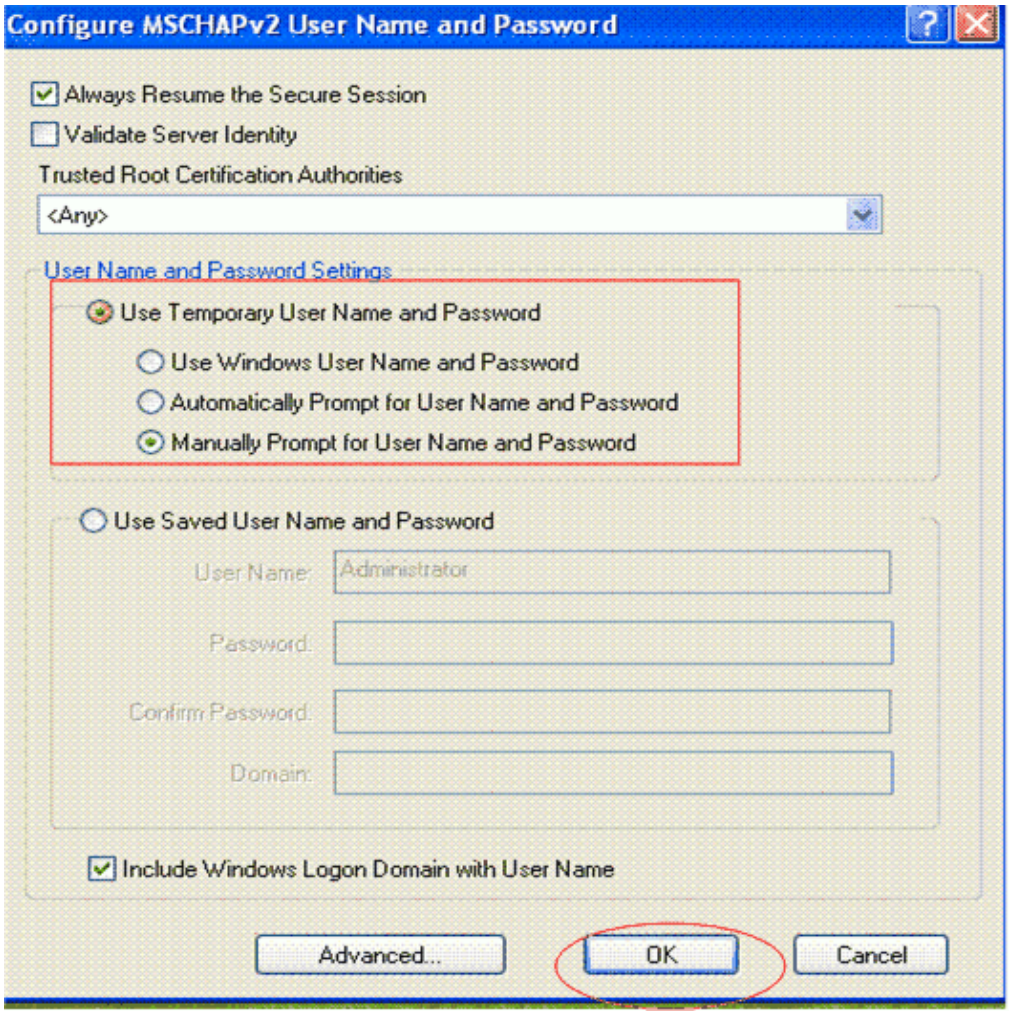


6. From the Configure MSCHAPv2 User Name and Password window, choose the appropriate username and password settings.

   This example chooses **Manually Prompt for User Name and Password**.

   The same username and password should be registered at the ACS. As mentioned earlier, this example uses **wireless** and **wireless** respectively as the username and password.

   Also, note that this is an anonymous in−band provisioning. Therefore, the client cannot validate the server certificate. You need to make sure that the **Validate Server Identity** check box is **unchecked**.
7. Click **OK**.



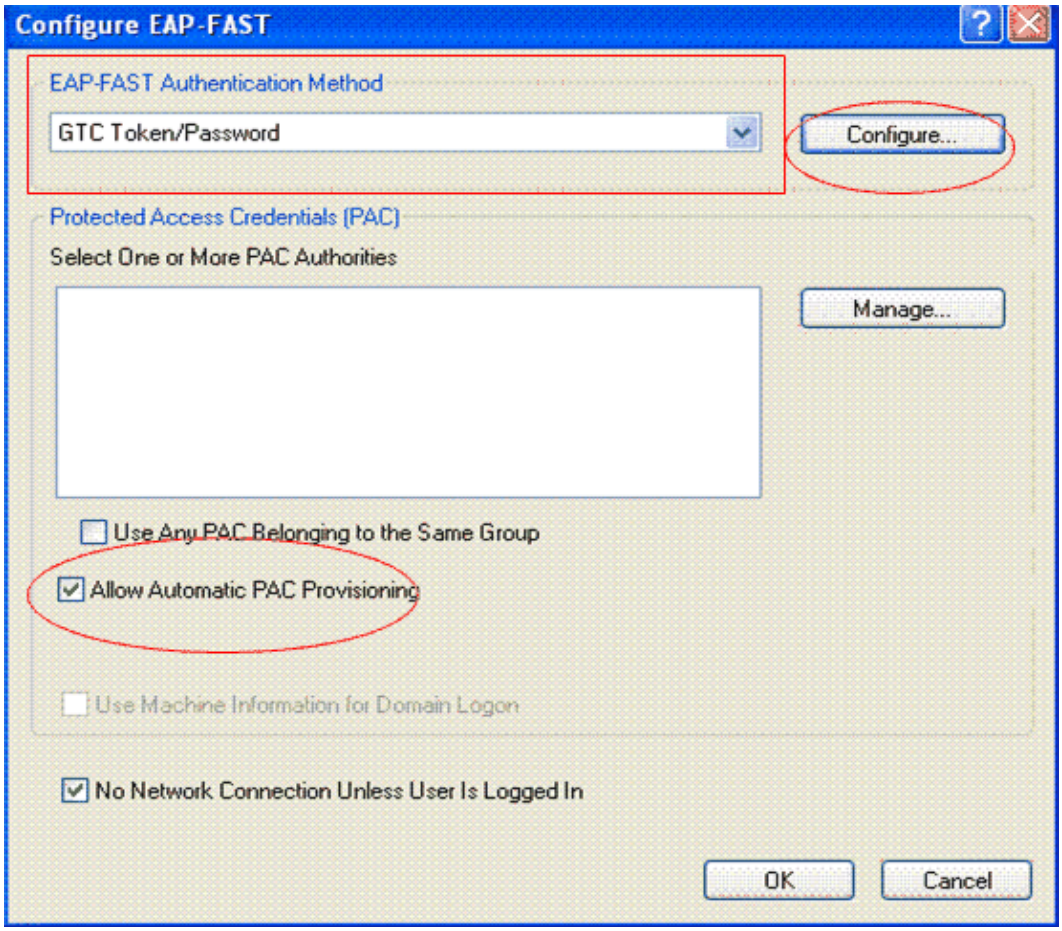
Configure Client for Authenticated In−band Provisioning

Complete these steps in order to configure wireless client for authenticated in−band provisioning:

1. Repeat steps 1 to 3 of the Configure Client for Anonymous In−band Provisioning section of this document.
2. From the Configure EAP−FAST window, check the **Allow Automatic PAC Provisioning** check box.

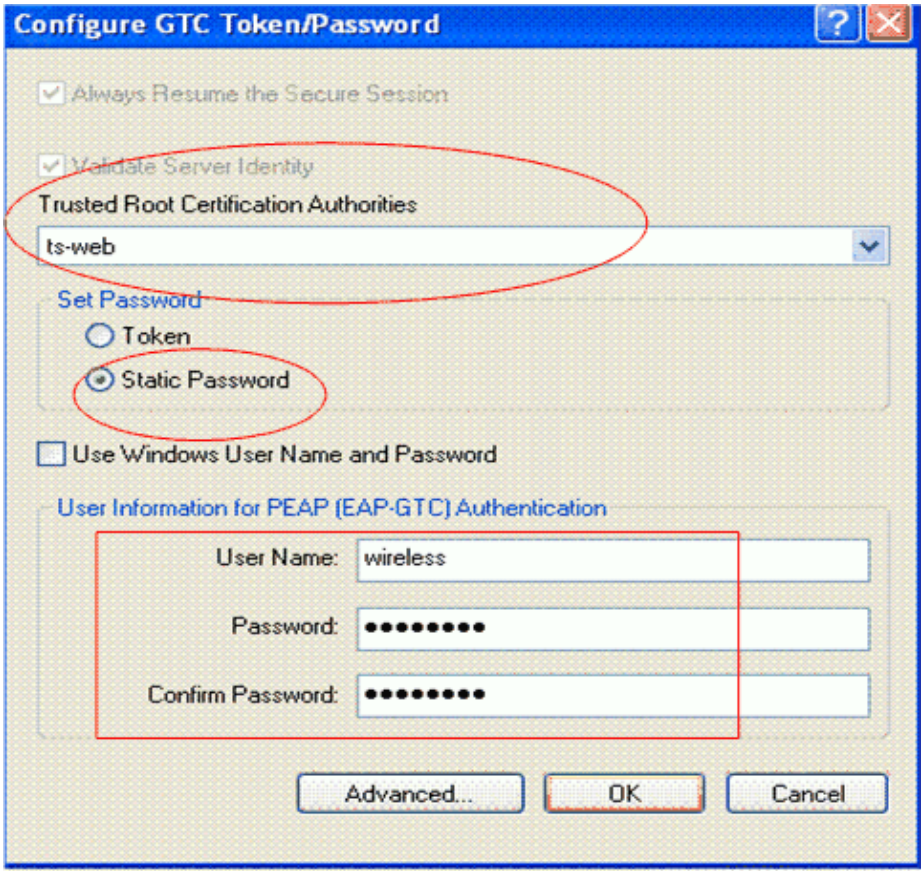
   With authenticated in−band PAC provisioning, any of the inner methods (EAP−GTC, EAP−MSCHAPv2, TLS Client Certificate) allowed at the ACS side can be selected on the client as the authentication method from the EAP−FAST Authentication Method drop−down box.

   This example chooses **GTC Token/Password** as the EAP−FAST Authentication Method.
3. Click **Configure**.



4. From the GTC configuration window, you can either use a static password or a token to be prompted at the time of authentication.

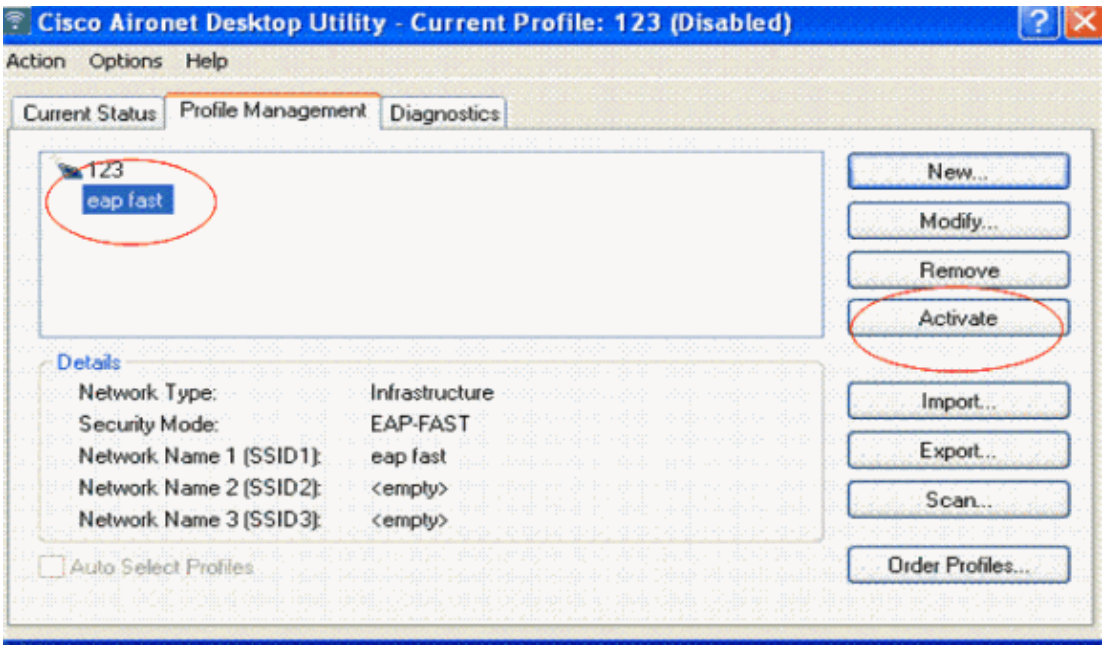
   This example uses a static username and password. The same username and password should be registered at the ACS. As mentioned earlier, this example uses **wireless** and **wireless** respectively as the username and password.
5. Also, note that this is an authenticated in−band provisioning configuration. Therefore, check the **Validate Server Identity** check box. Also, from the **Trusted Root Certification Authorities** drop−down box, scroll down to look for the Imported Self−Signed Certificate from the ACS (ts−web in this example). Choose the certificate as the Trusted Root Certification Authority. Click **OK**.



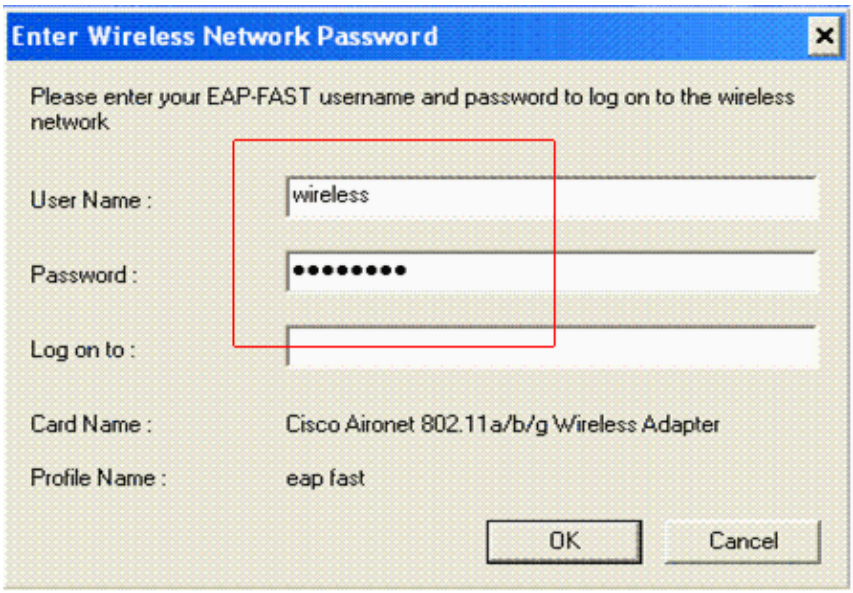
Verify In−band Provisioning

Complete these steps in order to verify whether your EAP−FAST configuration works properly:

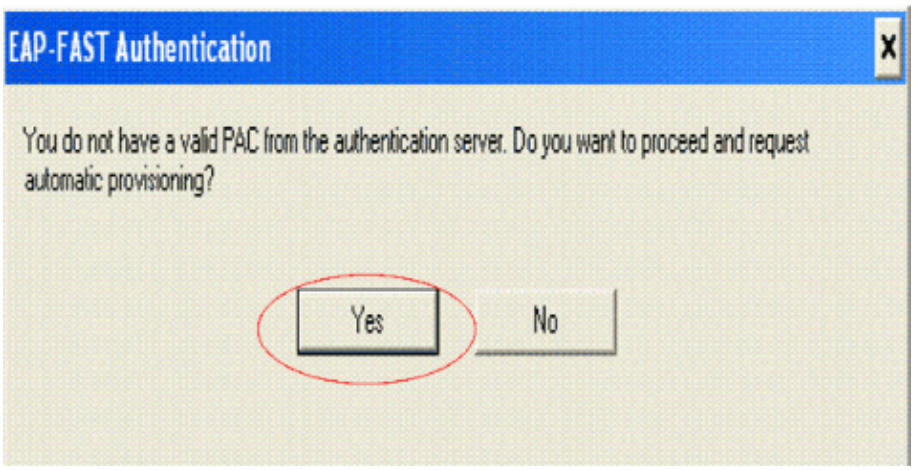
1. Select the profile **eap fast** and click **Activate** in order to activate the wireless client profile.



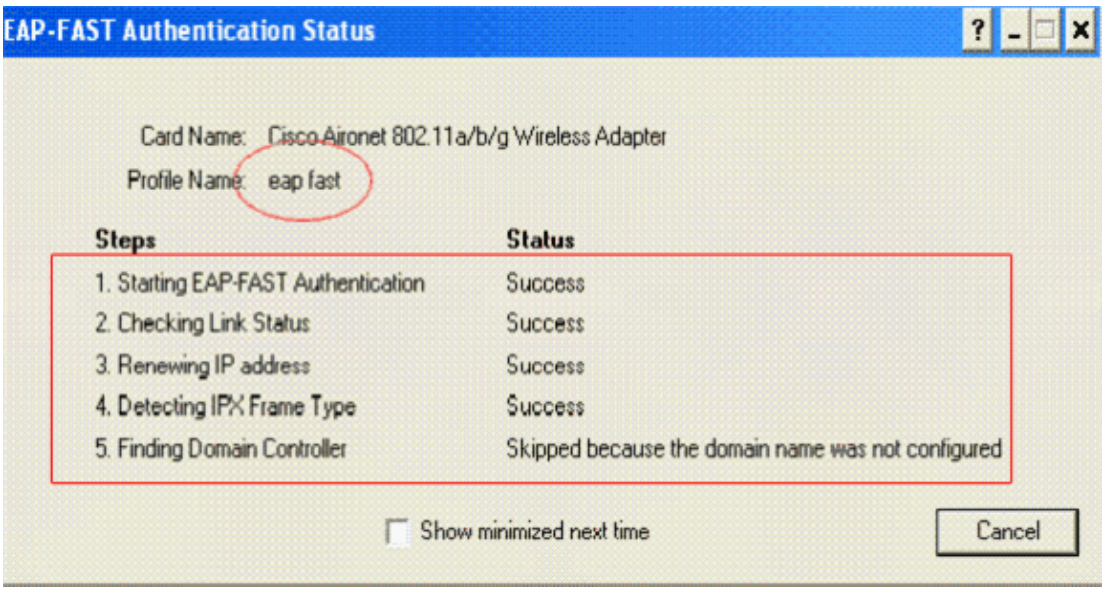
2. If you have enabled MS−CHAP ver2 as your authentication method (with Anonymous, this is the only successful method as explained earlier), then the client will prompt for the username and password.



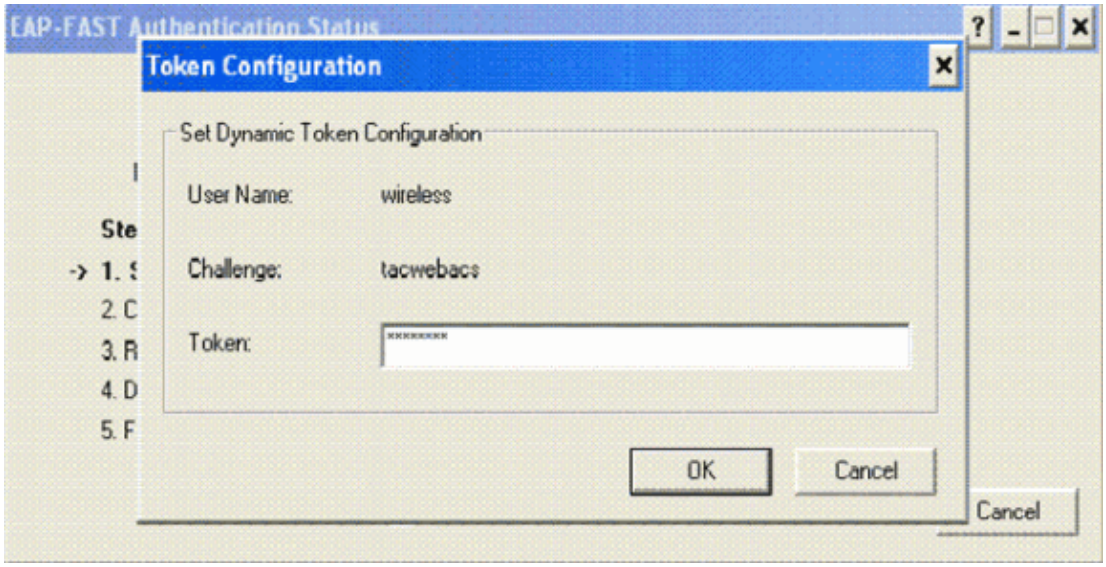
3. During EAP−FAST processing of the user, you will be prompted by the client to request PAC from the RADIUS server. When you click **YES**, PAC provisioning starts.



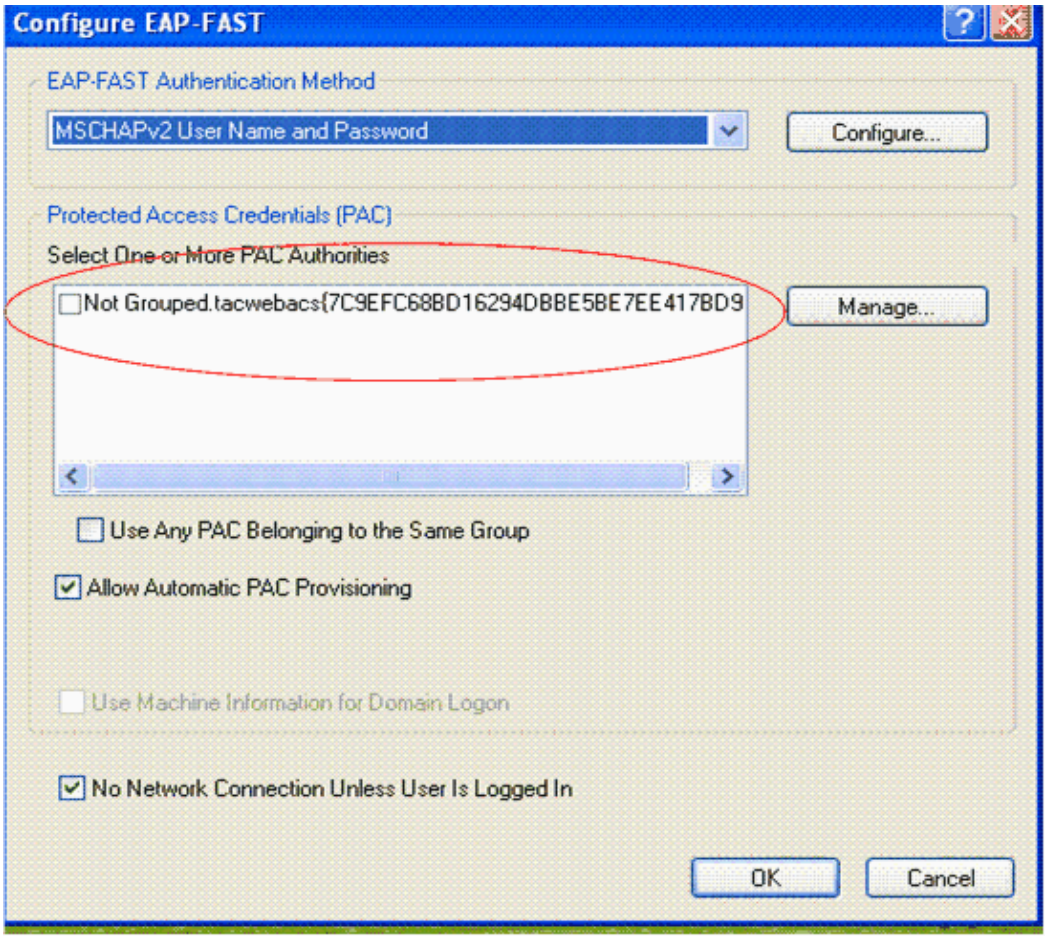
After successful PAC provisioning in phase zero, phase one and two follow and a successful authentication procedure takes place.



4. With authenticated in−band provisioning, if you have enabled GTC/Token as your EAP−FAST Authentication Method, you will be prompted to enter the token. As mentioned earlier, this example uses **wireless** as the user credential. Click **OK**.

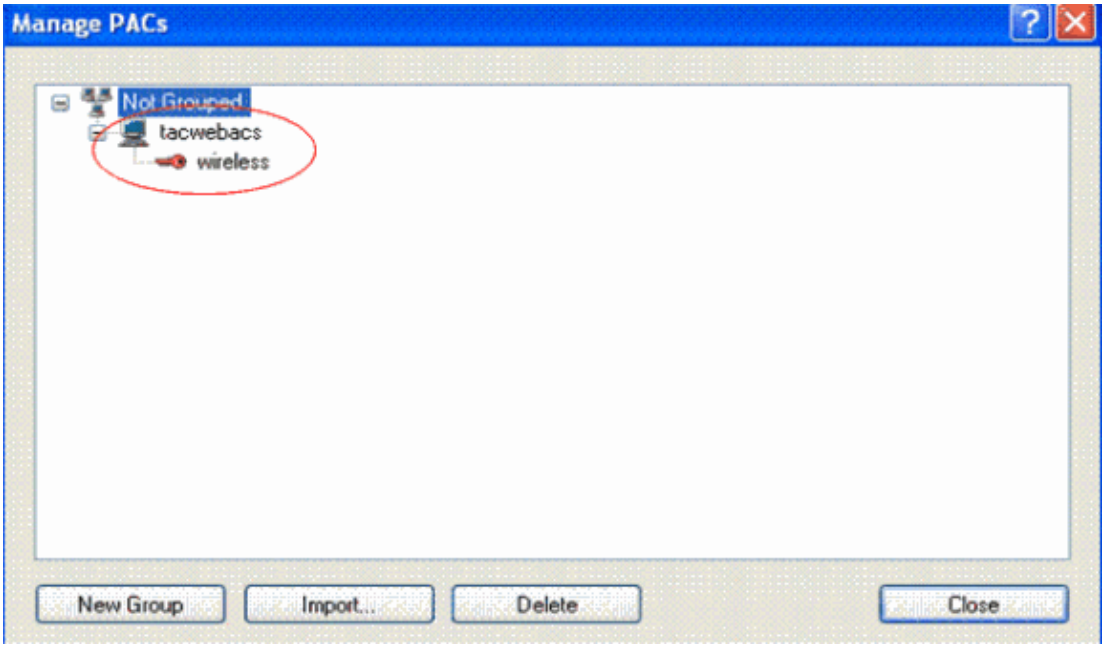


You can see the PAC file received by the client in the EAP−FAST configuration page.

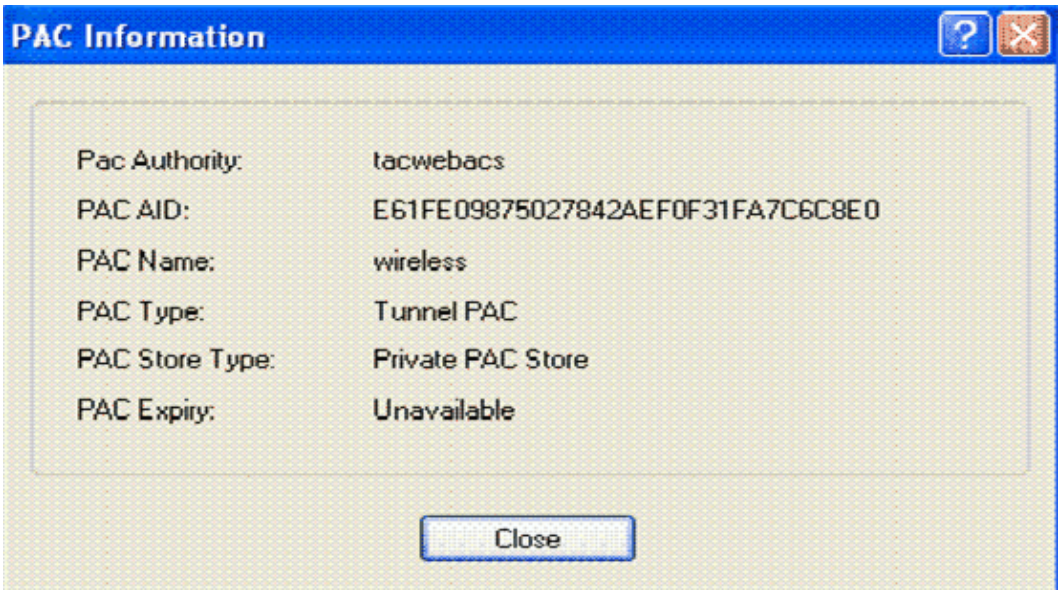


This PAC can be used for further authentication sessions of this user as per the settings of the PAC/Master Key TTL values.

5. Click **Manage** to see the contents of the PAC file by browsing through the PAC management tree as shown here. Double−click on the **wireless** PAC file to display the contents of the PAC file.



This is the content of a PAC file:



Verify

You can verify whether the RADIUS server receives and validates the authentication request from the wireless client. Check the **Passed Authentications and Failed Attempts** reports on the ACS server in order to accomplish this. These reports are available under **Reports and Activities** on the ACS server.

Here is an example when the RADIUS server authentication is actually successful. However, because phase zero of EAP−FAST does not enable a network service, even a successful EAP−FAST phase zero transaction is recorded in the ACS Failed Attempts log. If you see the "EAP−FAST user was provisioned with new PAC" message, this indicates PAC is provisioned successfully to the client.

These **debug** commands might be useful for some troubleshooting purposes:

Note: Refer to Important Information on Debug Commands before you use **debug** commands.

- **debug dot1x events enable** Enables the debug of all dot1x events. Here is an example debug output based on successful authentication:

```
(Cisco Controller)>debug dot1x events enable

Mon Oct 22 20:34:57 2007: 00:40:96:af:3e:93 Sending EAP
-Request/Identity to mobile 00:40:96:af:3e:93 (EAP Id 1)
Mon Oct 22 20:34:57 2007: 00:40:96:af:3e:93 Received EAPOL START from mobile 00:
40:96:af:3e:93
Mon Oct 22 20:34:57 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 2)
Mon Oct 22 20:34:57 2007: 00:40:96:af:3e:93 Received EAP Response packet with mi
smatching id (currentid=2, eapid=1) from mobile 00:40:96:af:3e:93
Mon Oct 22 20:34:57 2007: 00:40:96:af:3e:93 Received Identity Response (count=2)
 from mobile 00:40:96:af:3e:93
Mon Oct 22 20:34:57 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
.....................................................................
.....................................................................
.....................................................................
.....................................................................

Mon Oct 22 20:35:00 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 19, EAP Type 43)
Mon Oct 22 20:35:00 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:00 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 20)
Mon Oct 22 20:35:01 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 20, EAP Type 43)
Mon Oct 22 20:35:29 2007: Creating dot1x interface with key 00:0b:85:91:c3:c0 -
0
Mon Oct 22 20:35:29 2007: Resetting the group key timer for 3689 seconds on AP 0
0:0b:85:91:c3:c0
Mon Oct 22 20:35:29 2007: Creating dot1x interface with key 00:0b:85:91:c3:c0 -
1
Mon Oct 22 20:35:29 2007: Resetting the group key timer for 3696 seconds on AP 0
0:0b:85:91:c3:c0
Mon Oct 22 20:35:30 2007: 00:40:96:af:3e:93 Received EAPOL START from mobile 00:
40:96:af:3e:93
Mon Oct 22 20:35:30 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 22)
Mon Oct 22 20:35:30 2007: 00:40:96:af:3e:93 Received Identity Response (count=3)
 from mobile 00:40:96:af:3e:93
Mon Oct 22 20:35:30 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:30 2007: 00:40:96:af:3e:93 WARNING: updated EAP-Identifer 22 ==
=> 19 for STA 00:40:96:af:3e:93
Mon Oct 22 20:35:30 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 19)
Mon Oct 22 20:35:30 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 19, EAP Type 3)
Mon Oct 22 20:35:30 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
```

```
Mon Oct 22 20:35:30 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 20)
Mon Oct 22 20:35:30 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 20, EAP Type 43)
Mon Oct 22 20:35:30 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:30 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 21)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 21, EAP Type 43)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 22)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 22, EAP Type 43)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 23)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 23, EAP Type 43)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 24)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 24, EAP Type 43)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 25)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 25, EAP Type 43)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 26)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 26, EAP Type 43)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 27)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 27, EAP Type 43)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Processing Access-Reject for mobile
00:40:96:af:3e:93
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Sending EAP-Failure to mobile 00:40:
96:af:3e:93 (EAP Id 27)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Setting quiet timer for 5 seconds fo
r mobile 00:40:96:af:3e:93
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 1)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 1)
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Received EAPOL START from mobile 00:
40:96:af:3e:93
Mon Oct 22 20:35:31 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 2)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Received Identity Response (count=2)
 from mobile 00:40:96:af:3e:93
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 WARNING: updated EAP-Identifer 2 ===
> 20 for STA 00:40:96:af:3e:93
```

```
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 20)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 20, EAP Type 3)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 21)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 21, EAP Type 43)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 22)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 22, EAP Type 43)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 WARNING: updated EAP-Identifer 22 ==
=> 24 for STA 00:40:96:af:3e:93
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 24)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 24, EAP Type 43)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 25)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 25, EAP Type 43)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Processing Access-Accept for mobile
00:40:96:af:3e:93
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Creating a new PMK Cache Entry for s
tation 00:40:96:af:3e:93 (RSN 0)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Sending EAP-Success to mobile 00:40:
96:af:3e:93 (EAP Id 25)
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Sending default RC4 key to mobile 00
:40:96:af:3e:93
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Sending Key-Mapping RC4 key to mobil
e 00:40:96:af:3e:93
Mon Oct 22 20:35:32 2007: 00:40:96:af:3e:93 Received Auth Success while in Authe
nticating state for mobile 00:40:96:af:3e:93
```

Here is an example of **failed authentication** from the output of the **debug dot1x events enable** command:

```
(Cisco Controller) >debug dot1x events enable

Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Sending EAP-Requ
est/Identity to mobile 00:40:96:af:3e:93 (EAP Id 1)
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Received EAPOL START from mobile 00:
40:96:af:3e:93
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 2)
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Received Identity Response (count=2)
 from mobile 00:40:96:af:3e:93
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 WARNING: updated EAP-Identifer 2 ===
> 11 for STA 00:40:96:af:3e:93
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 11)
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 11, EAP Type 3)
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
```

```
le 00:40:96:af:3e:93
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 12)
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 12, EAP Type 43)
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Processing Access-Challenge for mobi
le 00:40:96:af:3e:93
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Sending EAP Request from AAA to mobi
le 00:40:96:af:3e:93 (EAP Id 13)
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Received EAP Response from mobile 00
:40:96:af:3e:93 (EAP Id 13, EAP Type 43)
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Processing Access-Reject for mobile
00:40:96:af:3e:93
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Sending EAP-Failure to mobile 00:40:
96:af:3e:93 (EAP Id 13)
Mon Oct 22 20:26:11 2007: 00:40:96:af:3e:93 Setting quiet timer for 5 seconds fo
r mobile 00:40:96:af:3e:93
Mon Oct 22 20:26:16 2007: 00:40:96:af:3e:93 802.1x 'quiteWhile' Timer expired fo
r station 00:40:96:af:3e:93
Mon Oct 22 20:26:16 2007: 00:40:96:af:3e:93 quiet timer completed for mobile 00:
40:96:af:3e:93
Mon Oct 22 20:26:16 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 15)
Mon Oct 22 20:26:17 2007: 00:40:96:af:3e:93 802.1x 'txWhen' Timer expired for st
ation 00:40:96:af:3e:93
Mon Oct 22 20:26:17 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 16)
Mon Oct 22 20:26:18 2007: 00:40:96:af:3e:93 802.1x 'txWhen' Timer expired for st
ation 00:40:96:af:3e:93
Mon Oct 22 20:26:18 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 17)
Mon Oct 22 20:26:19 2007: 00:40:96:af:3e:93 802.1x 'txWhen' Timer expired for st
ation 00:40:96:af:3e:93
Mon Oct 22 20:26:19 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 18)
Mon Oct 22 20:26:20 2007: 00:40:96:af:3e:93 802.1x 'txWhen' Timer expired for st
ation 00:40:96:af:3e:93
Mon Oct 22 20:26:20 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 19)
Mon Oct 22 20:26:21 2007: 00:40:96:af:3e:93 802.1x 'txWhen' Timer expired for st
ation 00:40:96:af:3e:93
Mon Oct 22 20:26:21 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 20)
Mon Oct 22 20:26:22 2007: 00:40:96:af:3e:93 802.1x 'txWhen' Timer expired for st
ation 00:40:96:af:3e:93
Mon Oct 22 20:26:22 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 21)
Mon Oct 22 20:26:23 2007: 00:40:96:af:3e:93 802.1x 'txWhen' Timer expired for st
ation 00:40:96:af:3e:93
Mon Oct 22 20:26:23 2007: 00:40:96:af:3e:93 Sending EAP-Request/Identity to mobi
le 00:40:96:af:3e:93 (EAP Id 22)
```

- **debug dot1x packet enable** Enables the debug of 802.1x packet messages. Here is an example debug output based on successful authentication:

```
(Cisco Controller) debug dot1x packet enable

 00:40:96:af:3e:93 Sending 802
.11 EAPOL message  to mobile 00:40:96:af:3e:93
Mon Oct 22 20:41:13 2007: 00000000: 01 00 00 2c 01 01 00 2c  01 00 6e 65 74 77 6
f 72  ...,...,..networ
                          00000010: 6b 69 64 3d 65 61 70 20  66 61 73 74 2c 6e 61 73
  kid=eap.fast,nas
                        00000020: 69 64 3d 57 4c 43 31 2c  70 6f 72 74 69 64 3d 32  id
=WLC1,portid=2
Mon Oct 22 20:41:13 2007: 00:40:96:af:3e:93 Received 802.11 EAPOL message (len 4
) from mobile 00:40:96:af:3e:93
```

```
Mon Oct 22 20:41:13 2007: 00000000: 01 01 00 00
      ....
Mon Oct 22 20:41:13 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:13 2007: 00000000: 01 00 00 2c 01 02 00 2c  01 00 6e 65 74 77 6
f 72  ...,...,..networ
                          00000010: 6b 69 64 3d 65 61 70 20  66 61 73 74 2c 6e 61 73
  kid=eap.fast,nas
                 00000020: 69 64 3d 57 4c 43 31 2c  70 6f 72 74 69 64 3d 32  id
=WLC1,portid=2
Mon Oct 22 20:41:13 2007: 00:40:96:af:3e:93 Received 802.11 EAPOL message (len 3
1) from mobile 00:40:96:af:3e:93
Mon Oct 22 20:41:13 2007: 00000000: 01 00 00 1b 02 02 00 1b  01 50 45 41 50 2d 3
0 30  .........PEAP-00
                          00000010: 2d 34 30 2d 39 36 2d 41  46 2d 33 45 2d 39 33
  -40-96-AF-3E-93
Mon Oct 22 20:41:13 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:13 2007: 00000000: 01 00 00 26 01 19 00 26  11 01 00 08 52 97 b
2 df  ...&...&....R...
                          00000010: 38 d2 ce 74 50 45 41 50  2d 30 30 2d 34 30 2d 39
  8..tPEAP-00-40-9
                 00000020: 36 2d 41 46 2d 33 45 2d  39 33                    6-
AF-3E-93
Mon Oct 22 20:41:13 2007: 00:40:96:af:3e:93 Received 802.11 EAPOL message (len 1
0) from mobile 00:40:96:af:3e:93
Mon Oct 22 20:41:13 2007: 00000000: 01 00 00 06 02 19 00 06  03 2b
      .........+
Mon Oct 22 20:41:13 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:13 2007: 00000000: 01 00 00 1a 01 1a 00 1a  2b 21 00 04 00 10 7
c 9e  ........+!....|.
                          00000010: fc 68 bd 16 29 4d bb e5  be 7e e4 17 bd 9f
  .h..)M...~....
Mon Oct 22 20:41:14 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:14 2007: 00000000: 01 00 00 1a 01 1a 00 1a  2b 21 00 04 00 10 7
c 9e  ........+!....|.
                          00000010: fc 68 bd 16 29 4d bb e5  be 7e e4 17 bd 9f
  .h..)M...~....
Mon Oct 22 20:41:15 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:15 2007: 00000000: 01 00 00 1a 01 1a 00 1a  2b 21 00 04 00 10 7
c 9e  ........+!....|.
                          00000010: fc 68 bd 16 29 4d bb e5  be 7e e4 17 bd 9f
  .h..)M...~....
Mon Oct 22 20:41:16 2007: 00:40:96:af:3e:93 Received 802.11 EAPOL message (len 6
0) from mobile 00:40:96:af:3e:93
Mon Oct 22 20:41:16 2007: 00000000: 01 00 00 38 02 1a 00 38  2b 01 16 03 01 00 2
d 01  ...8...8+.....-.
                          00000010: 00 00 29 03 01 47 1c bd  d7 0f 4c 38 41 29 51 34
  ..)..G....L8A)Q4
                 00000020: dc 42 4d bb 3d 3f ca 79  a6 34 5c 52 d1 ab ba ec  .B
M.=?.y.4\R....
              00000030: f5 d7 44 0d
Mon Oct 22 20:41:16 2007: 00:40:96:af:3e:93 Received 802.11 EAPOL message (len 6
0) from mobile 00:40:96:af:3e:93
Mon Oct 22 20:41:16 2007: 00000000: 01 00 00 38 02 1a 00 38  2b 01 16 03 01 00 2
d 01  ...8...8+.....-.
                          00000010: 00 00 29 03 01 47 1c bd  d7 0f 4c 38 41 29 51 34
  ..)..G....L8A)Q4
                 00000020: dc 42 4d bb 3d 3f ca 79  a6 34 5c 52 d1 ab ba ec  .B
M.=?.y.4\R....
              00000030: f5 d7 44 0d
Mon Oct 22 20:41:16 2007: 00:40:96:af:3e:93 Received 802.11 EAPOL message (len 6
0) from mobile 00:40:96:af:3e:93
```

```
Mon Oct 22 20:41:16 2007: 00000000: 01 00 00 38 02 1a 00 38  2b 01 16 03 01 00 2
d 01  ...8...8+.....-.
                          00000010: 00 00 29 03 01 47 1c bd  d7 0f 4c 38 41 29 51 34
  ..)..G....L8A)Q4
                        00000020: dc 42 4d bb 3d 3f ca 79  a6 34 5c 52 d1 ab ba ec  .B
M.=?.y.4\R....
                    00000030: f5 d7 44 0d
Mon Oct 22 20:41:16 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:16 2007: 00000000: 01 00 02 72 01 1b 02 72  2b 81 00 00 02 68 1
6 03  ...r...r+....h..
                          00000010: 01 00 4a 02 00 00 46 03  01 47 1c bd b1 3a a8 8e
  ..J...F..G...:..
                        00000020: de fc 62 51 e0 fe 93 c2  1d 21 08 51 59 ba c6 90  ..
bQ.....!.QY...
                    00000030: ad 14 1b bc
Mon Oct 22 20:41:17 2007: 00:40:96:af:3e:93 Received 802.11 EAPOL message (len 3
36) from mobile 00:40:96:af:3e:93
Mon Oct 22 20:41:17 2007: 00000000: 01 00 01 4c 02 1b 01 4c  2b 01 16 03 01 01 0
6 10  ...L...L+.......
                          00000010: 00 01 02 01 00 f4 92 8c  54 b1 34 ae 1e 13 a3 63
  ........T.4....c
                        00000020: 03 35 e9 33 e6 45 01 6a  87 18 ba 3d 03 0f 5f a5  .5
.3.E.j...=.._.
                    00000030: 1d 3a ae fa
Mon Oct 22 20:41:46 2007: 00:40:96:af:3e:93 Received 802.11 EAPOL message (len 4
) from mobile 00:40:96:af:3e:93
Mon Oct 22 20:41:46 2007: 00000000: 01 01 00 00
      ....
Mon Oct 22 20:41:46 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
Mon Oct 22 20:41:47 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:47 2007: 00000000: 01 00 00 45 01 1d 00 45  2b 81 00 00 00 3b 1
4 03  ...E...E+....;..
                          00000010: 01 00 01 01 16 03 01 00  30 cc 1f c4 54 ac a2 88
  ........0...T...
                        00000020: 14 11 92 c4 5a 78 2c 57  06 57 77 d7 6b 4e b1 db  ..
..Zx,W.Ww.kN..
eived 802.11 EAPOL message (len 1
11) from mobile 00:40:96:af:3e:93
Mon Oct 22 20:41:47 2007: 00000000: 01 00 00 6b 02 1f 00 6b  2b 01 17 03 01 00 6
0 8b  ...k...k+.....`.
                          00000010: 25 2f c6 1a 61 de af 51  a4 1d ae 9e 8c e8 45 80
  %/..a..Q......E.
                        00000020: e0 14 69 01 d8 ab eb 08  af 21 44 44 70 9c 25 d2  ..
i......!DDp.%.
                    00000030: 39 6f 75 ce
Mon Oct 22 20:41:47 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:47 2007: 00000000: 01 00 00 5b 01 20 00 5b  2b 01 17 03 01 00 5
0 0a  ...[...[+.....P.
                          00000010: 7e da 91 4e d7 ae 6b 87  7c 31 7f 7a 3c af 67 71
  ~..N..k.|1.z<.gq
                        00000020: 8a a1 0b aa 0a ac 6f b5  e8 65 83 3b d1 39 bd 1b  ..
....o..e.;.9..
                    00000030: f4 dd f9 68
Mon Oct 22 20:41:47 2007: 00:40:96:af:3e:93 Received 802.11 EAPOL message (len 4
7) from mobile 00:40:96:af:3e:93
Mon Oct 22 20:41:47 2007: 00000000: 01 00 00 2b 02 20 00 2b  2b 01 17 03 01 00 2
0 73  ...+...++......s
                          00000010: 02 d6 0e d3 55 57 7c 0c  58 f0 7c 1f 5e 61 09 40
  ....UW|.X.|.^a.@
                        00000020: 00 85 0b 8e 11 b5 23 1b  fc 27 77 71 ad b8 ed     ..
....#..'wq...
Mon Oct 22 20:41:47 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
```

```
Mon Oct 22 20:41:47 2007: 00000000: 01 00 00 6b 01 21 00 6b  2b 01 17 03 01 00 6
0 c8  ...k.!.k+.....`.
                      00000010: c4 69 43 5d ad 34 88 05  24 d7 7e 90 e3 65 87 37
  .iC].4..$.~..e.7
                      00000020: 9c 94 2f 99 5d be ca 82  1a 11 89 68 44 08 4c ef  ..
/.].......hD.L.
                00000030: 56 89 18 7a
Mon Oct 22 20:41:47 2007: 00:40:96:af:3e:93 Received 802.11 EAPOL message (len 1
11) from mobile 00:40:96:af:3e:93
Mon Oct 22 20:41:47 2007: 00000000: 01 00 00 6b 02 21 00 6b  2b 01 17 03 01 00 6
0 16  ...k.!.k+.....`.
                      00000010: 04 f2 92 41 da 4c 9c 4a  64 d8 88 9e 4b ac b9 76
  ...A.L.Jd...K..v
                      00000020: a0 94 2a 7c 5f 7b 9b be  8b 07 e1 42 79 f1 4f 06  ..
*|_{.....By.O.
                00000030: 8e 50 c8 25
Mon Oct 22 20:41:47 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:47 2007: 00000000: 01 00 01 1b 01 22 01 1b  2b 01 17 03 01 01 1
0 0f  ....."..+.......
                      00000010: 9b 84 da 57 60 8b a7 f6  e2 09 33 38 0c d8 fc 1f
  ...W`.....38....
                      00000020: 5f 2f 6a 59 72 4a a7 db  3a 5d 32 5c ac ed 1d d0  _/
jYrJ..:]2\....
                00000030: 46 6a 1c 93
Mon Oct 22 20:41:47 2007: 00:40:96:af:3e:93 Received 802.11 EAPOL message (len 7
9) from mobile 00:40:96:af:3e:93
Mon Oct 22 20:41:47 2007: 00000000: 01 00 00 4b 02 22 00 4b  2b 01 17 03 01 00 4
0 3f  ...K.".K+.....@?
                      00000010: 93 1f aa c6 2f 61 f8 45  84 f5 60 bd 35 87 8e 3a
  ..../a.E..`.5..:
                      00000020: 6b 96 7c 9a f6 79 91 73  c1 e9 68 78 82 88 29 07  k.
|..y.s..hx..).
                00000030: 8f 71 ef 09
Mon Oct 22 20:41:47 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:47 2007: 00000000: 01 00 00 04 04 22 00 04
      ....."..
Mon Oct 22 20:41:47 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:47 2007: 00000000: 01 00 00 2c 01 01 00 2c  01 00 6e 65 74 77 6
f 72  ...,...,..networ
                      00000010: 6b 69 64 3d 65 61 70 20  66 61 73 74 2c 6e 61 73
  kid=eap.fast,nas
                      00000020: 69 64 3d 57 4c 43 31 2c  70 6f 72 74 69 64 3d 32  id
=WLC1,portid=2
Mon Oct 22 20:41:48 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct
                00000030: 2c 14 ca 45
Mon Oct 22 20:41:49 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:49 2007: 00000000: 01 00 00 04 03 20 00 04
      ........
Mon Oct 22 20:41:49 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:49 2007: 00000000: 01 03 00 39 01 00 0d ca  b3 f9 35 00 0a 60 b
a 20  ...9......5..`..
                      00000010: 82 55 76 30 27 6f 92 2e  ee ce 49 c8 c2 5c 24 01
  .Uv0'o....I..\$.
                      00000020: 33 8e 39 fb a6 f4 fa 72  09 8b ae 1a d5 ab 84 73  3.
9....r.......s
                00000030: e9 b9 28 85
Mon Oct 22 20:41:49 2007: 00:40:96:af:3e:93 Sending 802.11 EAPOL message  to mob
ile 00:40:96:af:3e:93
Mon Oct 22 20:41:49 2007: 00000000: 01 03 00 2c 01 00 0d ca  b3 f9 35 00 0a 60 b
```

```
b 2b  ...,......5..`.+
                        00000010: fa 79 fc 30 50 de dd a0  95 95 cb 50 25 19 0a 80
  .y.0P......P%...
                      00000020: 1e e8 3e bf 8b 7a e3 a1  37 2a 07 8d ef 24 72 47  ..
>..z..7*...$rG
```

- **debug aaa events enable** Enables the debug output of all aaa events. Here is a sample debug output of a failed authentication process:

```
(Cisco Controller) >debug aaa events enable

Mon Oct 22 20:14:35 2007: 00:40:96:af:3e:93 Successful trans
mission of Authentication Packet (id 145) to 10.77.244.196:1812, proxy state 00:
40:96:af:3e:93-96:af
Mon Oct 22 20:14:35 2007: ****Enter processIncomingMessages: response code=11
Mon Oct 22 20:14:35 2007: ****Enter processRadiusResponse: response code=11
Mon Oct 22 20:14:35 2007: 00:40:96:af:3e:93 Access-Challenge received from RADIU
S server 10.77.244.196 for mobile 00:40:96:af:3e:93 receiveId = 9
Mon Oct 22 20:14:35 2007: 00:40:96:af:3e:93 Successful transmission of Authentic
ation Packet (id 146) to 10.77.244.196:1812, proxy state 00:40:96:af:3e:93-96:af
.....................................................................
.....................................................................
.....................................................................
.....................................................................
Mon Oct 22 20:14:35 2007: ****Enter processIncomingMessages: response code
Mon Oct 22 20:14:35 2007: ****Enter processRadiusResponse: response code=3
Mon Oct 22 20:14:35 2007: 00:40:96:af:3e:93 Access-Reject received from RADIUS s
erver 10.77.244.196 for mobile 00:40:96:af:3e:93 receiveId = 9
Mon Oct 22 20:14:42 2007: 00:40:96:af:3e:93 Successful transmission of Authentic
ation Packet (id 149) to 10.77.244.196:1812, proxy state 00:40:96:af:3e:93-96:af
```

Troubleshoot

Troubleshooting Tips

- Ensure that the **Validate Server Identity** check box is disabled for the client profile for anonymous in−band provisioning.
- Ensure that **EAP−MSCHAPver2** is selected as the authenticated method on the client profile for anonymous in−band provisioning. This is the only applicable EAP inner method in phase zero for anonymous in−band provisioning.
- Make sure the user credentials entered at the client side at the time of authentication are already configured in the ACS.
- Check if the RADIUS server is selected from the drop−down menu of the WLAN (SSID).
- If you use Wi−Fi Protected Access (WPA), then you have to install the latest Microsft WPA hotfix for Windows XP SP2. Also, you should upgrade the driver for your client supplicant to the latest version.
- The `[SECURITY] 1x_ptsm.c 391: MAX EAPOL-Key M3 retransmissions reached` error message means that the card did not respond to a request for its identity. You can extend the EAP timers on the controllers to wait for client 802.1x info if you use these commands on the WLC CLI:

    - **config advanced eap identity−request−timeout 120 c**
    - **config advanced eap identity−request−retries 20**
    - **config advanced eap request−timeout 120**
    - **config advanced eap request−retries 20 save config**

Extracting the Package File from ACS RADIUS Server for Troubleshooting

If you use ACS as the external RADIUS server, this section can be used to troubleshoot. The package.cab is a zip file that contains all the necessary files needed in order to troubleshoot the ACS efficiently. You can use the CSSupport.exe utility to create the package.cab, or you can collect the files manually.

Refer to the Creating a package.cab File section of Obtaining Version and AAA Debug Information for Cisco Secure ACS for Windows for more information on how to create and extract the package file from the wireless control system (WCS).

Related Information

- **Cisco Secure Services Client with EAP−FAST Authentication**
- **EAP Authentication with RADIUS Server**
- **EAP−FAST Version 1.02 Configuration Guide**
- **Cisco EAP−FAST Q & A**
- **EAP Authentication with WLAN Controllers (WLC) Configuration Example**
- **EAP−FAST Authentication**
- **Technical Support & Documentation − Cisco Systems**

Updated: Oct 26, 2007 Document ID: 99791