Apple Technical White Paper

802.1X Authentication

OS X 10.7.3 and iOS 5.1

May 25, 2012

Contents

About 802.1X

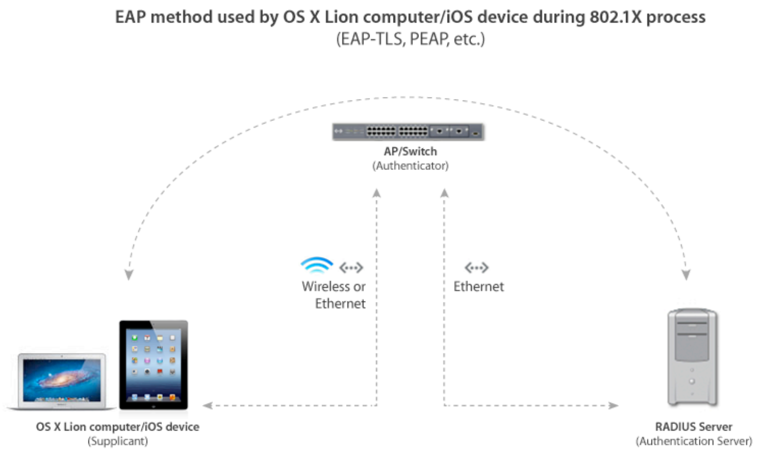
For many organizations, the growing mobility trend has created a greater need to securely manage access for both users and mobile devices between public (unprotected) and private (protected) networks. These organizations need a strong, yet flexible, system to identify and validate users and control device access to protected resources.

802.1X authentication enhances the security of local area networks (LANs) by preventing unauthorized devices from gaining port-level access to protected networks through wired or wireless LAN connections. It leverages an extensible architecture that supports a variety of authentication methods, including passwords, RSA keys, token cards, and certificates.

Key Components

802.1X relies on an integrated system of components to manage the authentication process:

- **Supplicant.** Software or service running on a device that seeks access to a protected network.
- **Authenticator.** Software or service running on a wireless access point or switch that manages the authentication process between the supplicant and authentication server.
- **Authentication server.** Software or service that provides authentication services to the authenticator. Using the credentials provided by the supplicant, the authentication server controls whether the supplicant is authorized to access the services provided on the authentication server's protected network.
- **Port.** A service access point, typically on a router or switch, whose state is either unauthorized or authorized. An unauthenticated supplicant device initially connects to an unauthorized port on the public network. After successful authentication, the device is connected to an authorized port and able to access resources on the protected network.
- **Extensible Authentication Protocol (EAP).** A generic authentication framework that supports many different types of authentication methods with different options, including Kerberos, public-key encryption, and one-time passwords. EAP communication between supplicants and authenticators is typically encapsulated using the EAP over LAN (EAPoL) protocol. EAP communication between authenticators and the authentication server is typically encapsulated using Remote Authentication Dial In User Service (RADIUS).



EAP Methods

EAP methods define the type of authentication used between the supplicant and the authenticator to validate a supplicant's identity. These authentication methods are commonly divided into two major groups:

- **Password-based methods.** Password-based authentication methods require each device to present a user name and password to authenticate to the network. Password-based methods use either an encrypted tunnel to send the password or a challenge/response mechanism to encrypt a piece of data to prove to the server that the client has access to the password. It's common to use the user's network password or a device trust account password as the password for this exchange.
- **Certificate-based methods.** Certificate-based authentication methods use X.509 identities instead of passwords. An X.509 identity consists of a private key, a certificate that provides information about the identity, and a public key. The certificate and the public key can be publicly shared; the public key is commonly included in the certificate. The private key is never shared. Both the authentication server and the supplicant have unique X.509 identities installed.

Supported EAP types
OS X computers and iOS devices support the most common EAP methods, including:

- PEAP
- TLS
- TTLS
- LEAP
- EAP-FAST
- EAP-SIM

Common EAP methods

The extensible nature of EAP has resulted in about 40 method definitions. Each method defines how EAP messages are handled and which authentication options and settings it provides. The most common EAP methods, both password based and certificate based, require a server-side identity.

Networks that use WPA Enterprise or WPA2 Enterprise encryption commonly use the following EAP methods for 802.1X authentication:

- **Protected Extensible Authentication Protocol (PEAP).** PEAP uses password-based authentication for device authentication and certificate-based authentication to verify server identities. It can use an outer identity, which is different from the user name and is passed in the clear as part of the preauthentication process, to prevent discovery of which user is authenticating. PEAP is the most common form of password-based authentication in 802.1X environments.
- **Transport Layer Security (TLS).** TLS uses client-side certificate-based authentication. It requires that an identity be provisioned ahead of time to the device that's connecting to the network. This requirement makes TLS a very secure option but also makes it a challenge to deploy since it requires provisioning prior to first connection to the network. TLS is the most common form of certificate-based authentication.
- **Tunneled Transport Layer Security (TTLS).** TTLS is an extension of TLS that relies on a secure tunnel between the authentication server and the supplicant as opposed to a preinstalled certificate on the supplicant. TTLS is similar to PEAP authentication in that it uses password-based authentication on the client side and certificates to verify the server's identity.
- **Lightweight Extensible Authentication Protocol (LEAP).** LEAP is a password-based authentication developed by Cisco for use mainly with Cisco hardware. It's considered a legacy protocol.
- **Flexible Authentication via Secure Tunneling (EAP-FAST).** EAP-FAST is a password-based authentication that can use directory authentication, Protected Access Credential (PAC), and an outer identity.
- **Subscriber Identity Module (EAP-SIM).** EAP-SIM uses a subscriber identity module (SIM) card for authentication. Because the SIM is intimately tied to a particular wireless carrier, this method is, in practice, applicable only to carrier-provided Wi-Fi hotspots.

802.1X Authentication

The 802.1X authentication process is divided into two basic stages:

Preauthentication

The 802.1X preauthentication process begins with a client device that contains a **supplicant** service or software that's used for negotiation and authentication. When the device connects to a wireless or wired network on an unauthorized port, the **authenticator** software in the switch or access point blocks the device's connection to the network. Using one of the EAP methods, the authenticator establishes a security negotiation with the supplicant and creates an 802.1X session.

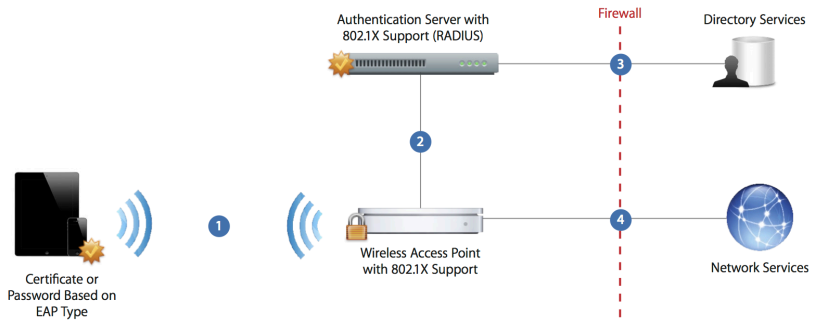
The supplicant provides its authentication information to the authenticator, which then proxies the information to an **authentication server**.

Authentication

After the authentication server authenticates the supplicant, the authentication server initiates the authentication stage of the process. During this phase the authenticator facilitates an exchange of keys between the supplicant and the authentication server. After these keys are established, the authenticator grants the supplicant access to the protected network on an authorized port.

The following diagram summarizes an implementation of the 802.1X authentication process using a RADIUS server as the authentication server to a wireless network access point:

RADIUS
Remote Authentication Dial In User Service (RADIUS) servers are commonly used as authentication servers. You can integrate them with the accounts in a directory service, such as Open Directory or Active Directory, to provide centralized authentication, authorization, and accounting services.


Authentication Server with 802.1X Support (RADIUS)
Firewall
Directory Services
1
2
3
4
Wireless Access Point with 802.1X Support
Network Services
Certificate or Password Based on EAP Type

1 The device requests access to the network when a user selects an available wireless network or the device detects a previously configured network.

2 When the access point receives the request, it passes the request to the RADIUS authentication server for authentication.

3 The RADIUS server uses directory services to validate the user account.

4 After the user is authenticated, the access point provides network access with policies and permissions as instructed by the RADIUS server.

Apple Product Compatibility with 802.1X

802.1X is the most widely accepted form of port-based network access control in use and is available in both OS X and iOS.

Apple provides 802.1X support in the following product families:

Wireless security protocols
OS X computers and iOS devices support the most common wireless security protocols, including:

- WEP
- WPA PSK (WPA Personal)
- WPA2 PSK (WPA2 Personal)
- 802.1X/WEP (WEP Enterprise)
- WPA Enterprise
- WPA2 Enterprise

OS X

OS X computers, such as MacBook Air, MacBook Pro, iMac, Mac mini, and Mac Pro, contain built-in supplicant software support for 802.1X authentication over both wireless networks and wired networks.

OS X computers detect WPA/WPA2 Enterprise LEAP, EAP-FAST, TTLS (MCHAPv2), and PEAP v0 and v1 on Ethernet and wireless networks. For EAP-TLS authentication without a Network payload, install the necessary identity certificates and tell users to select EAP-TLS mode in the 802.1X credentials dialog that appears when they connect to the network. Other EAP types require a Network payload that must specify the correct settings for the network.

Users on OS X computers can join dynamic 802.1X/WEP (WEP Enterprise) networks manually by choosing Join Other Network from the Wi-Fi menu. To allow users to join these types of networks automatically, install a Network payload that specifies 802.1X/WEP as the security type.

The OS X supplicant operates in three modes: User Mode, System Mode, and Login Window Mode. User Mode, the most basic mode, is used when the user joins the network and authenticates when prompted. System Mode is used primarily for computer authentication, which will occur even when a user isn't logged in to the computer. Finally, Login Window Mode is used when the computer is bound to an external directory such as Active Directory. When Login Window Mode is configured and a user types in an user name and password at the login window, two things will happen. First, the login window will authenticate the computer via 802.1X to the network using the user name and password the user entered. After the 802.1X authentication is successful, login window will authenticate the same user name and password to the external directory. It's possible to use System Mode and Login Window Mode together. For detailed information about the settings related to these modes, see Appendix A.

For computers with OS X Lion or later, you'll use configuration profiles to make 802.1X settings. Device and device group settings in OS X profiles are applied at the system level. To configure 802.1X System or Login Window modes, you would edit a device or device group setting. User and user group settings are applied at the user level. For more information about configuration profiles, see About Configuration Profiles and Appendix A in this document.

OS X Server

OS X Server includes the following services and technologies that you can integrate into an 802.1X authentication environment.

RADIUS

Use RADIUS as an authentication server to authorize Open Directory users and groups so they can access an AirPort Base Station on a network. By configuring RADIUS and Open Directory, you can control who has access to your wireless network.

Open Directory (authentication/authorization)

Use the Open Directory and Password Server to provide centralized authentication and authorization services for users and groups. Open Directory works with RADIUS to grant authorized users access to the network through an AirPort Base Station.

Profile Manager (profile creation/distribution)

Use Profile Manager, a component of OS X Server, to simplify the creation and distribution of configuration profiles with settings for specific users, groups of users, or devices. Profile Manager features web-based administration so you can manage your server from any modern web browser. You can even use Profile Manager to give users access to a self-service web portal where they can download and install new configuration profiles, clear passcodes, and remotely lock or wipe devices that are lost or stolen.

iOS

iOS devices, such as iPhone, iPad, and iPod touch, contain built-in supplicant software support for 802.1X authentication over wireless networks using industry-standard network protocols, including WPA2 Enterprise.

You can integrate iOS devices into a broad range of RADIUS authentication environments. 802.1X wireless authentication methods supported on iPhone, iPad, and iPod touch include EAP-TLS, EAP-TTLS, EAP-FAST, EAP-SIM, PEAPv0, PEAPv1, and LEAP.

For iOS devices, you'll use configuration profiles to make 802.1X settings. All 802.1X settings for iOS profiles are applied at the device level.

AirPort

You can configure Apple AirPort products, including AirPort Extreme, AirPort Express, and Time Capsule, to provide authenticator services between wireless supplicants (OS X computers and iOS devices) and a RADIUS server.

When a user attempts to access an AirPort Base Station, AirPort uses EAP to authenticate and authorize the user. Users are given access to the network if their user credentials are valid and they're authorized to use the AirPort Extreme Base Station. Users who aren't authorized can't access the network through the AirPort Base Station.

Configuring 802.1X Settings

In order to configure Apple products to integrate successfully into an 802.1X authentication environment, you must create and deploy to your devices the configuration profiles containing the settings and certificates for accessing your protected network.

The Trust Chain

Some EAP methods, such as EAP-TLS, use certificate-based authentication to uniquely identify a client and provide the ability for the client to authenticate to the network without user intervention. Both the client and server have an identity, and these identities, along with other certificates, are used to establish a **trust chain** between them.

In 802.1X authentication environments, it's important to understand the role certificates play in the trust chain. Client devices should be able to verify server-side certificates, and those certificates must be trusted for EAP. This trust is established by the user. The first time the user joins a device to an 802.1X-protected network, the device will prompt the user to trust the server's certificate.

The certificate presented by the client device to the server must also be valid and, in certain configurations, information in the certificate is validated against the authentication server (usually a directory server) to grant or deny that user's access to the protected network. If a configuration profile is used to configure the client's 802.1X settings, then the profile should include the necessary server-side certificate(s) and enable the trust setting for those certificate(s).

802.1X Integration Summary

The successful deployment of 802.1X authentication using Apple products relies on the interaction of key components. Together, these components form an integrated system that, when configured properly, provides the foundation for secure and reliable authentication using 802.1X.

When planning the 802.1X integration of Apple products into your environment, you may find it useful to keep the following general process and important considerations in mind.

1. **Verify network infrastructure**
    - Verify network appliances for compatibility and determine the supported authentication types (EAP types).
    - Configure your certificate infrastructure to support the corresponding key distribution process.
    - Verify certificate format and authentication server compatibility.

2. **Configure authentication servers**
    - Gather information about network settings, including host names, MAC addresses, and IP addresses.
    - Generate and install any necessary certificates and identities.
    - Configure services including RADIUS, directory services, web hosting, email, and VPN.
    - Assign network access permissions to users and groups.
    - Create configuration profiles and configure settings.
    - Distribute profiles to devices.
3. **Configure authenticators**
    - Gather information about network settings.
    - Configure access points for 802.1X (RADIUS) authentication.
4. **Configure supplicants**
    - Enroll devices for management.
    - Install profiles on devices.
    - Validate the trust chain for installed certificates and identities.

About Configuration Profiles

Configuration profiles provide a way to standardize settings and configure devices to interact with servers on networks in a company, school, or other organization. After you create a configuration profile for one or more devices, you must install it onto the devices in order to apply the settings within the profile.

You can install a variety of configuration profiles on a single device. For example, you can create one configuration profile with settings for configuring a user's Mail account and another with network settings for Wi-Fi and VPN. You can distribute configuration profiles in a signed and encrypted format to protect their contents.

If you want to configure OS X computers and iOS devices to authenticate to an 802.1X network, you must use configuration profiles to create and install the settings. With a Mobile Device Management (MDM) solution such as Profile Manager, you can create, distribute, and even remove profiles automatically.

Configuration profiles
You manage 802.1X settings for OS X computers and iOS devices using configuration profiles. These XML files store key-value pairs in a property list (.plist) format and have a .mobileconfig suffix in their filenames.

To learn more about configuration profiles, review the Managing OS X with Configuration Profiles white paper on Apple's website.

Creating Configuration Profiles

For OS X computers and iOS devices, Apple uses an XML document structure to define the content and format for configuration profiles.

Device profiles and user profiles
In OS X, a profile created for a device or device group is applied at the system level. A profile created for a user or user group is applied at the user level.

In iOS, all profiles are applied at the same level, and the profiles apply to the device.

Configuration profiles are stored in the property list (.plist) format. You can use a variety of tools to create them, including:

- Profile Manager, included as part of OS X Server
- MDM solutions from third-party providers
- iPhone Configuration Utility, available as a free download from Apple
- Apple Configurator, available on the Mac App Store
- A text editor or scripting solution for manual creation

For more information and documentation about configuration profiles and creation tools, see Resources in this document.

Configuring Payload Settings

Each configuration profile contains one or more **payloads**, which are collections of a certain type of settings, such as settings for Wi-Fi or VPN.

When you create a new configuration profile, you specify settings in one or more of the profile's payloads. Each payload setting field provides a brief description of its purpose and how it's used.

You can configure these settings in one general configuration profile or divide them into smaller, specialized configuration profiles.

Payloads

The payloads typically used to configure 802.1X settings on OS X computers and iOS devices include:

Multiple payloads
A configuration profile can contain multiple payloads to configure multiple services and settings. For example, you can create separate Network payloads for both Wi-Fi and Ethernet.

- **General.** The General payload uniquely identifies the profile and defines its removal policy.
- **Network.** The Network payload contains the primary settings for 802.1X configuration and defines the EAP method and security options for the specified network.
- **Certificates.** The Certificates payload contains the necessary credentials and identities to establish the certificate trust chain between the device and protected resources.
- **SCEP.** The SCEP payload defines the settings necessary to obtain an identity from a Simple Certificate Enrollment Protocol (SCEP) server.

The only required payloads in a configuration profile are General and Network. It's common, however, to configure additional payloads. For 802.1X authentication, you must also have a Certificates payload for any necessary certificates and, optionally, a SCEP payload if a client is acquiring an identity via SCEP.

The following paragraphs describe the general contents of each payload type listed above and highlight those settings relevant to 802.1X. For more detailed descriptions of these payload settings, see Appendix A: Payload Settings for 802.1X.

General settings

General settings
- Organization
- Description
- Security

The General payload, which is the only mandatory payload in a configuration profile, sets the name and identifier of a configuration profile. You can also use it to specify whether end users may remove the profile after it's installed.

To keep configuration profiles organized, name them with consistent conventions and clear descriptions with version numbers and dates. It's important that you specify a unique identifier for each configuration profile because any subsequent profile created with an identical identifier will replace the original.

Network settings

Network settings
- Network Interface
- SSID
- Hidden Network
- Auto Join
- Proxy Setup
- Security Type
- Network Security Settings

The majority of the settings in the Network payload apply to both OS X and iOS. Some of the settings apply only to OS X, and OS X has settings that are applied at either the user level or the device level. For example, the ability to use the computer credentials when authenticating via PEAP is available only at the device level.

Both password- and certificate-based protocols require a Network payload. This payload determines what network interface (Wi-Fi or Ethernet) is used, provides security settings for the Wi-Fi protocol, and contains references to certificates and identities associated with the 802.1X configuration.

For 802.1X, the primary settings are established in the Security Type and Network Security Settings sections of the payload. The Security Type section defines the wireless security protocol settings. The Network Security Settings section defines the EAP protocol and trust settings.

The trust section of the Network payload is used to establish explicit trust for the authentication server's certificate and any additional certificates in the server's trust chain. You can also optionally specify wildcard names for certificates associated with your authentication server(s).

Certificate settings

Certificate Settings
- Certificate Name
- Certificate or Identity Data
- Passphrase

The Certificate payload allows you to specify trusted root certificates, intermediate certificates, server certificates, and X.509 identities. Use this payload to specify the X.509 certificates you want to install on devices. Add the certificates necessary to establish the trust chain that will be used when authenticating the device's access to your protected network.

The following table outlines the file formats, extensions, and support:

Type	Usage	OS X	iOS	Extension
PKCS#12	Identity	Yes	Yes	.p12
PKCS#7	Identity	Yes	No	.pfx
X.509	Root, intermediate, leaf	Yes	Yes	.cer, .pem, .der

SCEP Settings
- Name
- Subject
- Subject Alternative Name Type
- Subject Alternative Name Value
- NT Principal Name
- Challenge
- Key Size
- Use as digital signature
- Use for key encipherment
- Fingerprint

SCEP settings

Both OS X and iOS allow you to securely request an identity via Simple Certificate Enrollment Protocol (SCEP) over HTTP. The settings configured by this payload are used when generating a certificate signing request (CSR) for use with SCEP.

Issuing an identity to OS X computers and iOS devices

In order to authenticate using EAP-TLS, the authenticating device must have an identity installed prior to authentication. How it requests the identity is based on your environment. Common ways to request the identity include:

- **Manually.** You create a CSR, submit it to the certificate authority (CA), request the template found in the prior step, and generate a PKCS#12 (.p12) file, which is then password protected. You then place this file in the payload along with the 802.1X configuration.
- **SCEP.** You configure a payload to use SCEP services to request an identity, then specify this payload in the Network configuration payload. The 802.1X configuration then uses the resulting identity from the SCEP request. You must ensure that the SCEP service is turned on and configured in the SCEP environment and that a one-time passphrase has been generated and included in the payload or that the SCEP service is set up to allow reuse of the passphrase.
- **ADCertificate profile (OS X).** OS X can request a certificate based on information in a user's or computer's Active Directory account. This option requires that Microsoft Web enrollment be turned on and configured in a Microsoft environment. For more information, see How to request a certificate from a Microsoft Certificate Authority using the ADCertificatePayloadPlugin on Apple's support website.
- **Web browser (OS X only).** Safari on OS X can request an identity directly from the Microsoft Web enrollment website using the keygen tag. The private key is stored in the login keychain when the request is generated. After the certificate is issued, it can be imported into the keychain and the resulting identity exported as a PKCS#12 (.p12) file that can be used in a payload. This payload can then be used by the Network payload for configuration.

Deploying 802.1X Configuration Profiles

Deploying 802.1X configuration profiles to OS X and iOS devices involves two steps:

- Distributing the configuration profile to the device containing the required root certificates and any necessary intermediate certificates.
- Installing the configuration profile onto the device containing the root and intermediate certificates and user identity.

Apple Push Notification service
MDM servers such as Profile Manager can use the Apple Push Notification service to wirelessly distribute configuration profiles to devices that have been enrolled for management.

You can distribute configuration profiles to devices by several means. You can post them to a website for users to download or email them to users directly. You can also use an MDM solution, such as Profile Manager, to distribute them automatically to devices that have been enrolled for management.

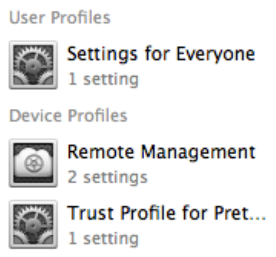
Devices must be able to establish a trusted, secure connection to the MDM server. If your server's SSL certificate isn't issued by a trusted CA known to OS X or iOS, then users must install the necessary root certificates on their devices to verify the certificate. For Profile Manager, they do so by downloading the Trust Profile from the user portal.

Profile Manager can sign configuration profiles so devices can verify that they haven't been modified. This requires a code-signing certificate, which Profile Manager can generate for you.

Alternatively, you can use a signing certificate with an established chain of trust. In the Profile Manager pane of the Server app, enable profile signing and select your installed code-signing certificate from the system keychain. Tell your users to download the Trust Profile from the user portal to install the intermediate certificates to verify signed profiles.

Finder (OS X)

System Preferences (Profiles)



When users receive configuration profiles, they're prompted to review the profile and enter administrator credentials to install the profile.

You can also make profiles available for manual installation by simply copying the profile to a location that the user can access on the computer, such as the desktop, and then instructing the user to double-click the profile to install it.

After the profile is installed, its name will appear in the Profiles preference pane of System Preferences. The Profiles preference pane lists all user and device profiles installed on the system, including Remote Management and Trust profiles. Users can review the details associated with each profile and selectively add and remove profiles.

Terminal (OS X)

You can manage configuration profiles from the command line, or via a script, by using Terminal's `profiles` command.

Command Line
To learn more about the `profiles` command, launch Terminal and type `man profiles` at the prompt.

To install a configuration profile called `testfile.mobileconfig` into the current user:

```
profiles -I -F /testfile.mobileconfig
```

To remove the profile `/profiles/testfile2.mobileconfig` from the current user:

```
profiles -R -F /profiles/testfile2.mobileconfig
```

To display information about the installed configuration profiles for the current user:

```
profiles -L
```

iOS

When users receive configuration profiles, they're prompted to review the profile before tapping Install to install the profile. Users can review and remove installed profiles by navigating to Settings > General > Profiles.

Resources

802.1X specification

http://standards.ieee.org/getieee802/download/802.1X-2010.pdf

Extensible Authentication Protocol (EAP) specification

http://tools.ietf.org/html/rfc5247

Wi-Fi Protected Access (WPA)

www.wi-fi.org

Configuration profiles reference

http://help.apple.com/profilemanager/mac/10.7/#apd88330954-6FA0-4568-A88E-7F6828E763A7X.509

https://developer.apple.com/library/ios/featuredarticles/iPhoneConfigurationProfileRef/

Certificates

http://www.apple.com/iphone/business/docs/iOS_Certificates.pdf

OS X Server

Profile Manager Help: http://help.apple.com/profilemanager/mac/10.7/

RADIUS: https://help.apple.com/advancedserveradmin/mac/10.7/#apd48AEB083-F53C-498D-B245-FA7993D92F57

Accessing 802.1X networks in OS X

Mobile Device Management (MDM)

http://images.apple.com/ipad/business/docs/iOS_MDM.pdf

iPCU

iPhone Configuration Utility 3.5 for Mac OS X

iPhone Configuration Utility 3.5 for Windows

AirPort specifications

http://www.apple.com/wifi/

Requesting certificates from a Microsoft CA

http://support.apple.com/kb/HT4784

Appendix A: Payload Settings for 802.1X

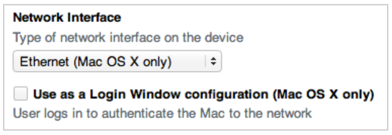
This appendix contains a summary of the settings typically required for configuring 802.1X in OS X and iOS configuration profiles.

Settings for the Network, Certificate, and SCEP payloads are described as viewed in OS X Server Profile Manager.

The individual settings and number of payloads you'll specify depend on your network architecture and organizational policies.

Network Payload Settings

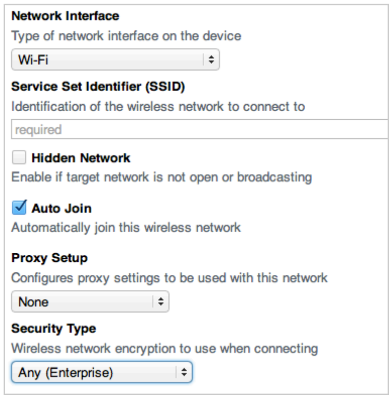
Network Settings (OS X Device Profile)



Network Interface
Type of network interface on the device
Ethernet (Mac OS X only)

☐ **Use as a Login Window configuration (Mac OS X only)**
User logs in to authenticate the Mac to the network

Network Settings



Network Interface
Type of network interface on the device
Wi-Fi

Service Set Identifier (SSID)
Identification of the wireless network to connect to
required

☐ **Hidden Network**
Enable if target network is not open or broadcasting

✓ **Auto Join**
Automatically join this wireless network

Proxy Setup
Configures proxy settings to be used with this network
None

Security Type
Wireless network encryption to use when connecting
Any (Enterprise)

Network interface

Use the Network Interface setting to specify whether the settings apply to the Ethernet (wired) or wireless network interface. (Ethernet applies to OS X computer profiles only.)

Use as a Login Window configuration (device profiles, OS X only)

Enable this setting to use the same credentials entered at the login window to authenticate the user via 802.1X and the configured directory server. If you select the "Use as a Login Window configuration" checkbox, which is only available on device profiles, you're specifically defining a Login Window Mode configuration.

SSID (wireless only)

Enter the identification of the wireless network to connect to. The 802.1X configuration is associated with an SSID (Service Set Identifier), or network name. When this network is joined, the client will authenticate with the 802.1X settings specified.

Hidden Network (wireless only)

Enable this setting if the target network is not open or broadcasting its SSID. If the network is hidden, then selecting this option will request it without requiring the SSID to be broadcast.

Auto Join (wireless only)

Select this option to have the device automatically attempt to join the wireless network if it's available.

Security Type (wireless only)

Select the wireless network encryption to use when connecting to the network, then specify the Protocols and Trust settings for the selected security type. For 802.1X, select either WEP Enterprise or WPA/WPA2 Enterprise. The most common is WPA/WPA2 Enterprise because WEP is considered a legacy security type.

Network Security settings

Use the Network Security settings to configure the primary settings related to 802.1X network authentication. Network Security has two tabs: Protocols, for choosing the EAP protocols, and Trust, for defining trust options related to certificates and exceptions.

Note: When configuring the network settings described below for device profiles, if you enter authentication credentials you'll be creating a System Mode configuration. Authentication credentials can be established using any of the following Network security settings:

- Enter a user name and password
- Select an identity that was previously defined in the Certificates payload
- Select Use Directory Authentication
- Select a previously defined SCEP payload

Protocols (accepted EAP types). Use the Protocols tab to specify which EAP methods to use for authentication. Select the protocol that applies to your network. You can select multiple EAP methods at one time. The user name, password, and outer identity are used for all methods.

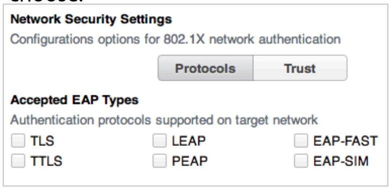
For TTLS, LEAP, PEAP, and EAP-FAST, if you choose directory authentication, the credentials for the OS X directory login are used to authenticate.

The following options are supported by some EAP methods, as noted below. Some options are applicable only to device profiles or user profiles.

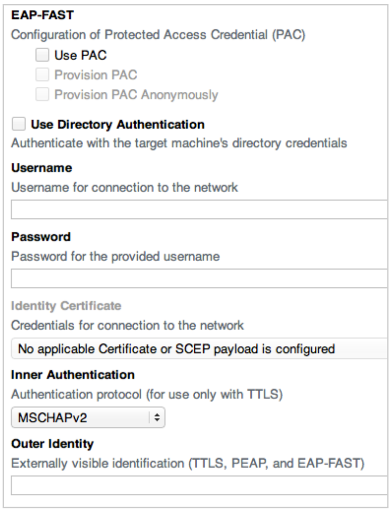
- **Use Directory Authentication.** In OS X device profiles, select this checkbox to authenticate with the target machine's Active Directory credentials. When the computer is bound to Active Directory and you use a password-based EAP type, the computer credentials used to bind to Active Directory are used to authenticate via 802.1X. This is sometimes called "computer authentication."
- **Protected Access Credential (PAC).** This setting is used with EAP-FAST to configure an encrypted shared secret used during the preauthentication phase. Enable this option to have the PAC establish a preshared key that will be used to authenticate the device to the RADIUS server over a secure tunnel. The PAC can be automatically provisioned when PAC provisioning is enabled.
- **User name.** If using a password-based protocol, enter the user name that will be used to authenticate the device. If using TLS, the value entered for Username will be sent in response to an identity request, instead of a value from the identity certificate itself. If the value from the certificate doesn't work properly for authentication, you may enter something like `host/computername.domain.com` or `computername$`.

  On OS X computers, you can use variables in the 802.1X user name fields. These variables are resolved on the device during installation. You can combine these variables with static text to create a compound user

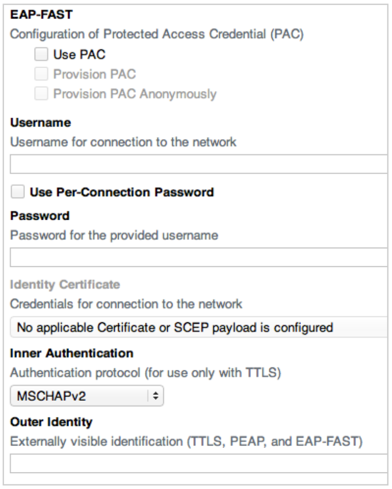
Network Security Settings (Protocols)
When you enable a protocol in this section, additional options will appear below the protocol. The options presented will depend on the protocol(s) you choose.


Network Security Settings
Configurations options for 802.1X network authentication
Protocols | Trust
Accepted EAP Types
Authentication protocols supported on target network
☐ TLS ☐ LEAP ☐ EAP-FAST
☐ TTLS ☐ PEAP ☐ EAP-SIM

EAP Options (OS X Device Profile)


EAP-FAST
Configuration of Protected Access Credential (PAC)
☐ Use PAC
☐ Provision PAC
☐ Provision PAC Anonymously
☐ **Use Directory Authentication**
Authenticate with the target machine's directory credentials
Username
Username for connection to the network
Password
Password for the provided username
Identity Certificate
Credentials for connection to the network
No applicable Certificate or SCEP payload is configured
Inner Authentication
Authentication protocol (for use only with TTLS)
MSCHAPv2
Outer Identity
Externally visible identification (TTLS, PEAP, and EAP-FAST)

EAP Options (User Profile)

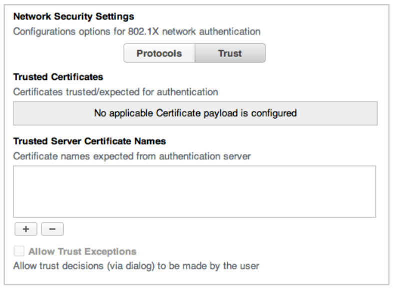

EAP-FAST
Configuration of Protected Access Credential (PAC)
☐ Use PAC
☐ Provision PAC
☐ Provision PAC Anonymously
Username
Username for connection to the network
☐ **Use Per-Connection Password**
Password
Password for the provided username
Identity Certificate
Credentials for connection to the network
No applicable Certificate or SCEP payload is configured
Inner Authentication
Authentication protocol (for use only with TTLS)
MSCHAPv2
Outer Identity
Externally visible identification (TTLS, PEAP, and EAP-FAST)

name. To review the complete list of variables, see Profile Manager Help: Network settings.

- **Use Per-Connection Password.** In OS X user profiles, enable this option to require users to type in a different password each time they connect. This option is used for RSA tokens and other one-time passwords.
- **Password.** Enter the password for the provided user name to embed the user password in the profile. If you don't specify a password, the user will be prompted during the first connection and the password may be saved in the user's keychain.
- **Identity Certificate.** Choose the identity certificate to specify the identity to use for certificate-based authentication. You can select a SCEP payload and the identity that's created with that payload will be used when authenticating. You can also select a PKCS#12 identity file that's provided in the Credentials section. Note that you MUST have both the certificate and private key in the Credentials payload. (See below.) The certificate and private key are both contained in a single PKCS#12 (.p12) file and are usually password protected.
- **Inner Authentication.** Choose an authentication protocol for use with TTLS. This option uses an encrypted tunnel and allows choice of MS-CHAPv2 (Microsoft Challenge Handshake Authentication Protocol v2), MSCHAP (Microsoft Challenge Handshake Authentication Protocol), CHAP (Challenge Handshake Authentication Protocol), or PAP (Password Authentication Protocol).
- **Outer Identity.** Enter the name to specify as the externally visible identification. This is a different outer identity than the "real" identity used when authenticating. The outer identity is sent during the initial part of the negotiation, and is available in clear text on the network.

Setting	TLS	TTLS	LEAP	PEAP	EAP-FAST
Use Directory Authentication		•	•	•	•
Protected Access Credentials (PAC)					•
Username		•	•	•	•
Use Per-Connection Password		•	•	•	•
Password		•	•	•	•
Identity Certificate	•				
Inner Authentication		•			
Outer Identity		•		•	•

Network Security Settings (Trust)


Network Security Settings
Configurations options for 802.1X network authentication
Protocols | Trust
Trusted Certificates
Certificates trusted/expected for authentication
No applicable Certificate payload is configured
Trusted Server Certificate Names
Certificate names expected from authentication server
+ −
☐ Allow Trust Exceptions
Allow trust decisions (via dialog) to be made by the user

Trust. Use the Trust tab to specify which certificates should be trusted to validate the authentication server for the network connection. Because the trusted root certificate store may contain roots that you don't want to trust for 802.1X authentication, this setting provides the ability to trust only specific certificates or specific certificate authorities.

Trusted Certificates. This list contains certificates from the Certificates payload and allows you to explicitly trust the intermediate, or leaf, certificates.

- **Trusted Server Certificate Names.** Add or remove the certificate names that will be expected from the authentication server. If you don't want to trust every 802.1X certificate issued by your trusted certificate authority, you can specify the common name of the server certificate (the common name on the certificate sent from the server to the client during initial negotiation). You may also specify wildcard characters so that you don't have to enter every server name. For example: `*.mycompany.radius.com`.

  For device configuration profiles, you must provide the trusted certificates necessary to authenticate the connection.

- **Allow Trusted Exceptions.** Enable this option to allow users to trust a server when the chain of trust can't be established. To avoid these prompts and permit connections only to trusted services, turn off this option and embed all necessary certificates in a profile.

  When you're creating a profile for a user, the settings are for 802.1X User Mode. When you're creating a profile for a device, the settings are for System Mode or Login Window Mode.

  If the Allow Trusted Exceptions checkbox is selected, the device will prompt the user to trust the untrusted certificate. If the checkbox is not selected, the device will fail authentication.

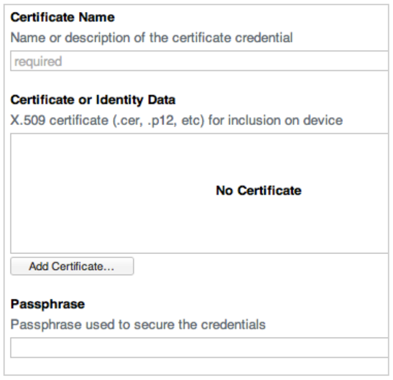
  Don't select this checkbox if you establish trust via Trusted Certificates or Trusted Server Certificate Names. Doing so places the decision in the hands of users who may not have the understanding to make trust decisions.

Certificate Settings

The Certificate Settings payload is used for both the authentication server certificate and the User or Computer Identity (if using EAP-TLS). You can use the Certificate settings payload to add certificates and identities to the device. OS X computers and iOS devices can use X.509 certificates with RSA keys. The file extensions .cer, .crt, and .der are recognized.

Certificates in PKCS1 and PKCS12 format are supported. Use P12 (PKCS#12 standard) files that contain exactly one identity. The file extensions .p12 and .pfx are recognized.

Certificate Settings


Certificate Name
Name or description of the certificate credential
required

Certificate or Identity Data
X.509 certificate (.cer, .p12, etc) for inclusion on device
No Certificate
Add Certificate…

Passphrase
Passphrase used to secure the credentials

When you install credentials, you should also install the intermediate certificates to establish a chain to a trusted certificate on the device. The only certificates required for server authentication are the server certificates themselves. Optionally, you could use intermediate or root certificates and use wildcards in the names field. If you're installing a user identity, then you need to include only the identity and not any intermediate server certificates.

To view a list of preinstalled roots for iOS devices, see the Apple Support article iOS 5: List of available trusted root certificates. On OS X, use Keychain Access to view the System Roots keychain.

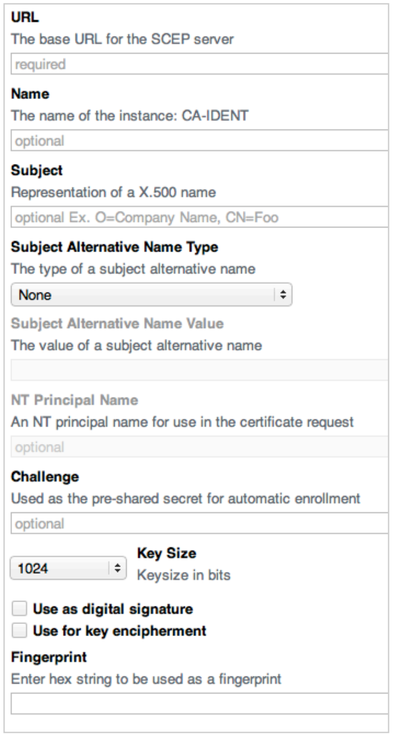
To add an identity for use with Microsoft Exchange, use the Exchange payload.

If you omit the certificate's passphrase, the user is asked to enter it when the profile is installed. Depending on how you create and distribute profiles, the contents of the payload may not be encrypted, so if you include the passphrase be sure it's available only to authorized users.

When using 802.1X User Mode and user identities, instead of using a configuration profile to install certificates, you can let users use Safari to download the certificates to their devices from a web page, or you can email certificates to users. You can also use SCEP settings to specify how the device obtains certificates over the air when the profile is installed.

SCEP Settings

SCEP Settings


URL
The base URL for the SCEP server
required

Name
The name of the instance: CA-IDENT
optional

Subject
Representation of a X.500 name
optional Ex. O=Company Name, CN=Foo

Subject Alternative Name Type
The type of a subject alternative name
None

Subject Alternative Name Value
The value of a subject alternative name

NT Principal Name
An NT principal name for use in the certificate request
optional

Challenge
Used as the pre-shared secret for automatic enrollment
optional

1024
Key Size
Keysize in bits

Use as digital signature
Use for key encipherment

Fingerprint
Enter hex string to be used as a fingerprint

You can use the SCEP payload to specify settings that allow the device to obtain certificates from a certificate authority (CA) using the Simple Certificate Enrollment Protocol.

URL

Enter the address of the SCEP server to define where SCEP requests will be sent, over HTTP or HTTPS. Because the private key isn't sent with the CSR, it may be safe to send the request unencrypted. However, if the one-time password is allowed to be reused, you should use HTTPS to protect the password.

Name

Use this option to specify which certificate authority is to be used when signing the certificate. (In Microsoft SCEP environments, this value is ignored.) The name can be any string understood by the certificate authority. It can be used to distinguish between instances, for example.

Subject, Subject Alt Name, NT Principal Name

Use these settings to specify values associated with alternate names. The Subject is the representation of a X.500 name. You can use variables to have this field defined dynamically by the OS X computer being configured. The subject name is usually specified in distinguished name (DN) format (dc=com,dc=example,cn=ipod$) to a computer account on the system, or simply the relative distinguished name (RDN) format (cn=ipad$). DNs must be specified by an object identifier (OID) (0.9.2342.19200300.100.1.25=com. 0.9.2342.19200300.100.1.25=example,CN=ipod$).

The SubjectAltName and NT Principal names are usually DNS names.

The NT Principal Name is for NT networks and OS X computers only.

The Subject Alternative Name specifies the type and value of an alternative name for the SCEP server. Valid values are an email address (RFC-822), the DNS name of the server, or the server's fully qualified URL.

The Subject, Subject Alt Name, and NT Principal Name are all critical pieces of information that MUST be correct, or in some cases omitted, for 802.1X authentication to succeed. There are no set values for these attributes, which require that you either have the requirements identified or follow the format on an existing certificate that is known to work. Achieving this can be tricky, because authentication can be based on specific paths in the common name or host names found in the NT Principal Name or Subject Alt Name.

For example, a RADIUS server may be configured to specify that only computers in an Organizational Unit called "remote access" are allowed to use 802.1X. The RADIUS server would then check the Subject name to verify that the computer account is in that OU and accept the authentication only if that is true. A client with a certificate that had a DN of dc=example,dc=com,OU=remote access, cn=ipad would succeed, while a client with a certificate of dc=example,dc=com,OU=computers, cn=ipad would fail. The RADIUS server logs would clearly show this kind of failure. Getting a certificate issued by your certificate authority via SCEP does not guarantee that it can be used for 802.1X authentication since the RADIUS server must actually accept it.

Challenge

If the SCEP server uses a one-time password, enter it here. This is a preshared secret the SCEP server can use to identify the request or user.

Key Size

Choose the number of bits for the key size, 1024 or 2048. When generating the key, the keysize setting determines how long the key size will be. Longer key sizes are generally more secure but are more computationally expensive. Since the public key is sent with the CSR, you need to specify a supported key size in order for the CSR to be accepted and a certificate issued.

"Use as digital signature" and "Use as key encipherment"

Use these options to specify how the certificate can be used. If someone is using the certificate to verify a signature, such as verifying whether a certificate was issued by a certificate authority, the SCEP server would verify that the certificate can be used in this manner prior to using the public key to decrypt the hash. If a server is using the public key in a certificate provided by a client to verify that a piece of data was encrypted using the private key, the server would first check to see if the certificate can be used for key encipherment. If not, it would fail the operation.

Leaf certificates (such as those on a client and on a server) require key encipherment, and intermediate/root certificates require a digital signature. In SCEP requests, it's common to see both of these options selected.

Depending on the SCEP server, these options may affect different settings as well. For example, in a Microsoft Network Device Enrollment Service (NDES), different templates are selected based on this option.

Fingerprint

If your CA uses HTTP, use this field to provide the fingerprint of the CA's certificate, which the device uses to confirm authenticity of the CA's response during enrollment. You can enter a SHA1 or MD5 fingerprint, or select a certificate to import its signature.

The provided fingerprint (a hash of the root certificate) is compared against the issuing root certificate that is provided during the SCEP negotiation. If the fingerprint does not match, the SCEP process will fail. If the fingerprint is not provided, the check is not done.

Appendix B: Sample Configuration Profiles

System EAP-TLS Configuration Profile with PKCS#12 Identity

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST
  1.0//EN" "http://www.apple.com/DTDs/
  PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
  <key>PayloadContent</key>
  <array>
     <dict>
          <key>AutoJoin</key>
          <true/>
          <key>EAPClientConfiguration</key>
          <dict>
               <key>AcceptEAPTypes</key>
               <array>
                    <integer>13</integer>
               </array>
          </dict>
          <key>EncryptionType</key>
          <string>WPA</string>
          <key>HIDDEN_NETWORK</key>
          <false/>
          <key>Interface</key>
          <string>BuiltInWireless</string>
          <key>PayloadCertificateUUID</key>
          <string>85E6B54F-008C-4A38-92E9-DEEC79811959</
  string>
          <key>PayloadDisplayName</key>
          <string>WiFi (TestSSID)</string>
          <key>PayloadEnabled</key>
          <true/>
          <key>PayloadIdentifier</key>
          <string>com.apple.mdm.earbuds.apple.com.
  543751e0-
  e897-012e-17ca-0017f20564ec.alacarte.interfaces.
  864faa40-e897-012e-17cc-0017f20564ec</string>
```

```
        <key>PayloadType</key>
        <string>com.apple.wifi.managed</string>
        <key>PayloadUUID</key>
        <string>864faa40-e897-012e-17cc-0017f20564ec</
string>
        <key>PayloadVersion</key>
        <integer>1</integer>
        <key>ProxyType</key>
        <string>None</string>
        <key>SSID_STR</key>
        <string>TestSSID</string>
        <key>SetupModes</key>
        <array>
            <string>System</string>
        </array>
    </dict>
    <dict>
        <key>PayloadContent</key>
        <data>
            <<bas64DATA>>
        </data>
        <key>PayloadDisplayName</key>
        <string>Example Certificate Authority</string>
        <key>PayloadEnabled</key>
        <true/>
        <key>PayloadIdentifier</key>
        <string>com.apple.mdm.earbuds.apple.com.
543751e0-
e897-012e-17ca-0017f20564ec.alacarte.certificate.CD4D7
BFA-CF18-4289-85EA-636B9B2D28FD</string>
        <key>PayloadType</key>
        <string>com.apple.security.root</string>
        <key>PayloadUUID</key>
        <string>CD4D7BFA-CF18-4289-85EA-636B9B2D28FD</
string>
        <key>PayloadVersion</key>
        <integer>1</integer>
    </dict>
    <dict>
```

```
        <key>Password</key>
        <string>abc</string>
        <key>PayloadContent</key>
        <data>
            <<bas64DATA>>
        </data>
        <key>PayloadDisplayName</key>
        <string>id.p12</string>
        <key>PayloadEnabled</key>
        <true/>
        <key>PayloadIdentifier</key>
        <string>com.apple.mdm.earbuds.apple.com.
543751e0-
e897-012e-17ca-0017f20564ec.alacarte.certificate.
85E6B54F-008C-4A38-92E9-DEEC79811959</string>
        <key>PayloadType</key>
        <string>com.apple.security.pkcs12</string>
        <key>PayloadUUID</key>
        <string>85E6B54F-008C-4A38-92E9-DEEC79811959</
string>
        <key>PayloadVersion</key>
        <integer>1</integer>
    </dict>
</array>
<key>PayloadDisplayName</key>
<string>Settings for test</string>
<key>PayloadIdentifier</key>
<string>com.apple.mdm.earbuds.apple.com.543751e0-
e897-012e-17ca-0017f20564ec.alacarte</string>
<key>PayloadOrganization</key>
<string>AppleEnterprise</string>
<key>PayloadRemovalDisallowed</key>
<false/>
<key>PayloadScope</key>
<string>System</string>
<key>PayloadType</key>
<string>Configuration</string>
<key>PayloadUUID</key>
<string>543751e0-e897-012e-17ca-0017f20564ec</string>
```

```
  <key>PayloadVersion</key>
  <integer>1</integer>
</dict>
</plist>
```

System EAP-TLS Configuration Profile with CA Certificate Payload

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
  "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
  <key>PayloadContent</key>
  <array>
    <dict>
        <key>PayloadCertificateFileName</key>
        <string>Internal Issuing CA 1</string>
        <key>PayloadContent</key>
        <data>ASNFZ4mrze8=</data>
        <key>PayloadDescription</key>
        <string>Trust Radius Server Certificate</string>
        <key>PayloadDisplayName</key>
        <string>Trust Radius Server Certificate</string>
        <key>PayloadIdentifier</key>
        <string>my.payload.identifier</string>
        <key>PayloadType</key>
        <string>com.apple.security.root</string>
        <key>PayloadUUID</key>
        <string>9768C058-9437-4F51-B7E6-AEAEF9717531</
  string>
        <key>PayloadVersion</key>
        <integer>1</integer>
    </dict>
    <dict>
        <key>PayloadCertificateFileName</key>
        <string>Internal Root CA</string>
        <key>PayloadContent</key>
        <data>/ty6mHZUMhA=</data>
        <key>PayloadDescription</key>
        <string>Trust Radius Server Certificate</string>
        <key>PayloadDisplayName</key>
        <string>Trust Radius Server Certificate</string>
        <key>PayloadIdentifier</key>
        <string>my.payload.identifier</string>
        <key>PayloadType</key>
        <string>com.apple.security.root</string>
        <key>PayloadUUID</key>
        <string>65295CEA-70C9-431A-86D1-F5581F2FED4F</
string>
        <key>PayloadVersion</key>
        <integer>1</integer>
   </dict>
   <dict>
        <key>EAPClientConfiguration</key>
        <dict>
            <key>AcceptEAPTypes</key>
            <array>
                <integer>13</integer>
            </array>
            <key>PayloadCertificateAnchorUUID</key>
            <array>
                <string>9768C058-9437-4F51-B7E6-
AEAEF9717531</string>
                <string>65295CEA-70C9-431A-86D1-
F5581F2FED4F</string>
            </array>
        </dict>
        <key>EncryptionType</key>
        <string>WPA</string>
        <key>HIDDEN_NETWORK</key>
        <true/>
        <key>Interface</key>
        <string>BuiltInWireless</string>
        <key>PayloadCertificateUUID</key>
        <string>0EF3981E-0DD8-4A62-A792-A859E734DCB6</
string>
        <key>PayloadDescription</key>
        <string>EAP-TLS 802.1x configuration</string>
        <key>PayloadDisplayName</key>
        <string>Configuration Profile</string>
        <key>PayloadIdentifier</key>
        <string>com.example.eaptls.8021x.wifi</string>
        <key>PayloadOrganization</key>
        <string>Apple Inc.</string>
        <key>PayloadType</key>
        <string>com.apple.wifi.managed</string>
        <key>PayloadUUID</key>
        <string>9574A054-8A51-46B3-8766-D8542DB0D843</
string>
        <key>PayloadVersion</key>
        <integer>1</integer>
        <key>SSID_STR</key>
        <string>hidden_ssid</string>
        <key>SetupModes</key>
        <array>
            <string>System</string>
        </array>
    </dict>
    <dict>
        <key>CertServer</key>
        <string>https://pki.apple.com/certsrv</string>
        <key>CertTemplate</key>
        <string>Workstation</string>
        <key>PayloadDescription</key>
        <string>EAP-TLS 802.1x configuration</string>
        <key>PayloadDisplayName</key>
        <string>Configuration Profile</string>
        <key>PayloadIdentifier</key>
        <string>my.payload.identifier</string>
        <key>PayloadOrganization</key>
        <string>Apple Inc.</string>
        <key>PayloadType</key>
        <string>com.apple.ADCertificate.managed</string>
        <key>PayloadUUID</key>
        <string>0EF3981E-0DD8-4A62-A792-A859E734DCB6</
string>
        <key>PayloadVersion</key>
        <integer>1</integer>
        <key>deleted</key>
        <false/>
        <key>PromptForCredentials</key>
        <true/>
    </dict>
  </array>
  <key>PayloadDescription</key>
  <string>EAP-TLS 802.1x configuration</string>
  <key>PayloadDisplayName</key>
  <string>Configuration Profile</string>
  <key>PayloadIdentifier</key>
  <string>my.payload.identifier</string>
  <key>PayloadOrganization</key>
  <string>Apple Inc.</string>
  <key>PayloadRemovalDisallowed</key>
  <false/>
  <key>PayloadType</key>
  <string>SystemConfiguration</string>
  <key>PayloadUUID</key>
  <string>78BB1EE4-EC9E-463A-86D7-00DA73F26733</string>
  <key>PayloadVersion</key>
  <integer>1</integer>
</dict>
</plist>
```

Appendix C: Using Active Directory Certificates

In Microsoft environments, it's common to see PFX and PKCS#12 formats. The PFX format isn't supported by iOS and should be converted to P12 for use on iOS.

Note: When exporting identities from a Windows workstation, it's common to have the identity marked to not allow the private key to be exported. If you can't export the private key, you won't be able to use the identity for authentication. Exporting just the certificate isn't sufficient for certificate-based authentication.

Active Directory Certificates Payload

In OS X, autoenrollment for a certificate can be required via a Windows CA Web enrollment service. The request is authenticated with Kerberos, and the resulting certificate is associated with the requesting account. The administrator doesn't generate a one-time password, and a user or computer password is used to authenticate the connection. The CSR is sent in a similar fashion as SCEP. There's no UI for providing this payload type.

For more information on creating this payload, see "Requesting certificates from a Microsoft CA" in the Resources section.

Creating an EAP-TLS Profile in a Microsoft Certificate Authority Environment

It's a common deployment environment to have clients authenticating via EAP-TLS with certificates issued by a Microsoft Certificate Authority. In both OS X and iOS, you can leverage the information known about the certificates and the environment to quickly and easily configure and deploy configuration profiles.

In order for the authentication to be successful, an OS X computer or iOS device must have a valid X.509 identity installed and this identity must be accepted for authentication to the RADIUS server. Because certificates can contain a wide variety of information and can present the same information in a variety of formats, this requirement can lead to failed authentication. Some best practices when deploying iOS and OS X in Microsoft CA environments include:

- **Root certificates.** If your organization's root certificates aren't contained in the root certificate store on OS X or iOS, you need to include the root certificate in the Certificates payload. You can export the root certificates either from a PC's computer certificate store or via the Microsoft Web Enrollment CA.
- **Templates.** An X.509 certificate can contain many different pieces of information, depending on the purpose of the certificate. For example, a computer may have a DNS name associated with its certificate, while a user may have an email address associated with its certificate.

In a Microsoft CA attributes are grouped together into templates, which define what information will be contained in the issued template. The specific information can be obtained from either the CSR or from the object's Active Directory information. It's common for a template to require that information for a certificate be populated with information from Active Directory specified by the template. Because the information required by the RADIUS server to authenticate a user, computer, or device is based on the information contained in the certificate, it's common to base RADIUS server requirements on information in the template.

If possible, use the same template for requesting a certificate from the Microsoft CA that's currently being used in a known, good configuration. While it may be tempting to create a new template for OS X computers or iOS devices on your network, using an existing template helps ensure that any future updates to the template won't adversely affect new requests.

However, it's possible that the issued certificate won't have a user or computer name in a form that is acceptable during the initial negotiation. (It's common to have a required format of "host/client.domain.com" or "DOMAIN\computername.") To resolve this issue, you can either specify a user name to be used in the Network payload that is inserted prior to the profile installed, or issue a new template that populates a usable user name.

- **Discovering the template name from a Microsoft Certificate Authority.** In order to use an existing template, you must know the name of the template. Every certificate that is issued by a Microsoft Certificate Authority has an Extended Key Usage that specifies the template name used when issuing the certificate. You can read this attribute by opening the local certificate store on a PC and viewing the certificate.

  To view the Certificate Manager, open the Microsoft Management Console and select the Certificates snap-in. If it's a computer-based certificate, select Computer Account when asked what the snap-in will manage. Under Personal, you should then see the certificate issued to the machine. The name of the template appears under Certificate Template Information.

- **User name.** During the initial EAPOL request and response stage of the preauthentication process, the name of the user or device is sent by the supplicant to the authenticator. If the name isn't in a format that the RADIUS server can resolve to a user or device account in Active Directory, the negotiation will fail. Because EAP-TLS uses certificates for authentication, the user name will be derived from the certificate by the authenticator. You can, however, specify it in the profile as well. The following is the order in which the user name will be determined:
  - User name in the profile payload
  - NTPrincipalName in the identity certificate
  - CommonName in the identity certificate
  - RFC822Name in the identity certificate

  If authentication fails due to a bad user name, or if the user is being prompted to select an identity and provide an optional user name, select the correct identity and try these forms for the user name:

  - `host/fqdn.of.host` (where *fqdn.of.host* is the fully qualified domain name of the computer. Note that the word "host" is literal, and should not be substituted)
  - `computername$` (where *computername* is the computer name of the computer in Active Directory)
  - `DOMAIN\computername` (where *DOMAIN* is the name of the domain the computer is in, and *computername* is the name of the computer account in Active Directory)

When you have determined the correct name, either make sure that the name is included in the NTPrincipalName, RFC822Name, or CommonName in the certificate, or specify the user name by including the EAPClientConfiguration payload dictionary in the configuration payload.

See Appendix B for an example of a hardcoded user name.

SCEP

The client generates a private and public key pair, and then generates a CSR. The CSR contains the requested information from the payload and a copy of the public key. When the certificate authority processes the CSR, it may or may not use the requested information. Note also that the information in the certificate may be used to determine whether the certificate is accepted for authentication. For example, it's common to require that the Subject Alt Name match a valid computer account in Active Directory during 802.1X authentication.

SCEP doesn't provide user-based authentication but rather depends on a one-time password. When an administrator is creating the profile, a one-time password is generated on the SCEP server (usually via a web form) and this one-time password is included in the payload. After this password is used, it can't be reused. The SCEP server can be configured to allow a one-time password to be used multiple times, or not required at all.

Renewing Certificates

When a certificate is nearing expiration, you can either renew it or request a new identity. Since 802.1X profiles are associated with specific certificates, either of these options requires reconfiguration of the 802.1X service. There's no mechanism for automatically requesting a new certificate when the old one is nearing expiration.