

ERNW Newsletter 36 / October 2011

Certificate Based Device Authentication with iOS Devices

Version: 1.0

Date: 5 Oct 2011

Author: Rene Graf (rgraf@ernw.de)



Table of contents

1	INTRODUCTION	3
2	BACKGROUND INFORMATION	4
2.1	Cryptography	4
2.1.1	Authentication	4
2.1.2	Encryption Technologies	4
2.1.3	Certificates	5
2.1.4	Certificate Authority	5
2.1.5	Public Key Infrastructure	6
2.1.6	Automatic Certificate Enrollment using SCEP	6
2.2	Wireless LAN	7
2.2.1	Extensible Authentication Protocol (EAP)	7
2.2.2	EAP-TLS	7
2.2.3	PEAP	7
2.3	Virtual Private Network	8
2.3.1	IPSec	8
2.3.2	SSL VPN	8
3	CERTIFICATE SUPPORT ON IOS DEVICES.....	9
3.1	Configuration profiles	9
3.2	Certificate usage on iOS devices	10
3.2.1	Encrypted connections	10
3.2.2	Certificates for device authentication	10
3.2.3	Certificates for user authentication	10
3.2.4	Encrypted profiles	11
3.3	Import of private root certificates	12
3.4	Import of device certificates	13
3.5	Certificate auto enrollment using SCEP	14
3.5.1	Known limitations	15
3.5.2	Supported Certificate Authorities	15
3.6	Over the air provisioning	16
3.7	Import of user certificates for Exchange Active Sync.....	17
3.8	Supported EAP methods for WLAN Authentication.....	18
3.9	VPN	18
4	EXAMPLE CONFIGURATION USING A MANAGED PKI.....	19
4.1	Creation of an authentication challenge	19
4.2	Configuration profile.....	20
4.2.1	General settings	20
4.2.2	Credentials	21
4.2.3	SCEP settings	22
4.2.4	WLAN settings	23

1 INTRODUCTION

Mobile device like iPhones and iPads are used more and more often in corporate environments. Companies not supporting such devices find their users unhappy and doing all kinds of silly stuff like forwarding confidential data to freemail accounts to be able to access them on their private iOS devices. The pressure on the IT departments to support these devices in an official way grows from day to day.

To integrate those devices into the corporate environment a lot of use cases require access to internal resources. Those devices basically have two ways to connect to a corporate network: WLAN and VPN connections. WLAN is a wireless technology and a VPN typically can be accessed over the InternetInternetInternet – also a very insecure network.

In order to protect the internal corporate network, strong authentication mechanisms are required. Thus all devices must support these authentication methods.

Using cryptographic authentication methods, such as client certificates, can fulfill the need for a strong authentication mechanism.

This newsletter shows how certificates can be used for device authentication on iOS devices – Enjoy!



2 BACKGROUND INFORMATION

This chapter gives a brief overview of the technologies used in this document.

2.1 Cryptography

2.1.1 Authentication

Authentication is the process of identifying a person, system or other digital entity. Based on a performed authentication, privileges for resources of some kind are granted (e.g. access to a certain asset). When thinking of network security, the authentication process must be reliable and strong (by means of "very hard to circumvent"), as it is used by an entity to prove its identity.

Generally, there are different ways to authenticate an user by:

- something you know (e.g. a password)
- something you have (e.g. a key, a smartcard or some token)
- something you are (e.g. biometric stuff like a fingerprint)

No matter which of the above categories are used to authenticate, the specific implementation must be able to resist attacks such as password brute force, cryptographic analysis or other ways of spoofing a certain identity.

Modern authentication methods are based on cryptographic technologies to provide the required level of security.

If at least two of the above authentication categories are combined to further improve the security, this is called two-factor authentication. If one of the two authentication factors gets "lost", the protected asset is still secure, because an attacker needs both factors to authenticate. Everybody already uses two-factor authentication in his daily life for withdrawing cash at the ATM. First factor – something you have – is your debit card, second factor – something you know – is the assigned PIN code.

2.1.2 Encryption Technologies

The cryptographic world basically knows two kinds of cryptographic algorithms. Symmetric and Asymmetric. Symmetric algorithms use the same key for encrypting and decrypting data, while asymmetric algorithms use different keys – a public key and a private key. Both kinds of algorithms have their advantages and disadvantages.

While symmetric algorithms exhibit quite good performance and can easily be implemented in hardware, they scale very badly (each pair of communication partners needs their own key) and need a way to exchange the key confidentially.

Asymmetric algorithms (like RSA) do not have the scaling problem (as each communication partner simply has its own key pair). They also do not have the problem of confidential key exchange. The two problems left with asymmetric encryption are the performance (they are much slower than their symmetric pendants) and the integrity of the exchanged public key.

When two communication partners (person 2 person, browser 2 webserver) want to establish a secure, encrypted communication, they first need to exchange their public keys. Data which is encrypted using a certain public key can only be decrypted with the corresponding private key. An attacker could replace the transferred public key with his own key to be able to decrypt the transmitted data. This is called a "man in the middle" attack.

To prevent such attacks, the communication partners need a reliable way to check if the received public keys are authentic. This is done by not only transferring public keys over the network, but including

some additional information. The bundle of this additional information and the public key is called certificate. The public key and the additional information (like name, validity) are then signed by an trusted third party – some entity both communication partners trust – a so called certificate authority. The signature ensures that nobody can manipulate a transmitted certificate unnoticed. It is very important that the communicating entities are able to validate a certain certificate, cause this is the only protection against man in the middle attacks.

2.1.3 Certificates

A digital certificate is an electronic document bonded to a real identity and contains its public key. With given information like the name of the person or an organization, their address, etc. the public key can be matched to an individual. The RFC 5280 standard describes the structure of a digital certificate. This includes the certificate signature algorithm, the certificate signature and a certificate with the following information:

- Version
- Serial Number
- Algorithm ID
- Issuer
- Validity (from, until)
- Subject
- Subject Public Key Info (Public Key Algorithm, Subject Public Key)
- Issuer Unique Identifier (optional)
- Subject Unique Identifier (optional)
- Extensions (optional)

Extensions are used to indicate the usage purpose of the certificate. They can be marked with the so called critical flag. A system must reject certificates having unrecognizable extensions where the critical flag is set. Common used extensions are Key Usage describing the usage of the certificate and Basic Constraints determining whether the certificate belongs to a Certificate authority (CA).

2.1.4 Certificate Authority

As mentioned above, a certificate authority is a kind of “trusted 3rd party” which issues certificates and therefore proves the authenticity of a certain identity by signing the parameters in the certificate. Thus making them tamper proof (at least if the communicating parties perform proper checks like validating the signature, checking the revocation list, and so on).

There are several public certificate authorities, which are trusted by operating systems and other software. Server administrators can buy a certificate from a public certificate authority and use it to identify their HTTPS webserver. People visiting the website are then able to verify the identity and make sure nobody “sits in between”. Using public certificate authorities is necessary for public resources like websites and the like.

In closed environments (like the VPN of a company) where all participating entities (clients, servers, ...) are under the control of the company itself, an own certificate authority could be operated. This on one hand saves costs for buying certificates, but on the other hand also introduces operational costs –

someone would have to operate this certificate authority. On top of that, a certificate authority alone is not enough. It is only part of a so called public key infrastructure (PKI). By operating an own PKI a company also solves a trust problem. No one would allow lets say "all Verisign issued certificates" to authenticate against the corporate VPN gateway and get access to the corporate network.

2.1.5 Public Key Infrastructure

A PKI is a generic term for creating, distributing, revoking and managing digital certificates.

A certificate authority itself does not qualify as a PKI. A PKI also includes the whole certificate management process (rollout, revocation, renewal) technically and on an organizational level (operational processes).

2.1.5.1 Managed PKI

A managed PKI solution is an outsourced private PKI – one could also call it Cloud PKI ;-).

2.1.6 Automatic Certificate Enrollment using SCEP

There are several protocols available for automating certificate rollout for a large amount of devices. One of those protocols is called simple certificate enrollment protocol (SCEP). It is already used by other devices like VPN clients and the like.

SCEP currently has the state of an IETF draft. It is not standardized as a RFC yet. The latest revision is revision 23 and can be viewed on the IETF website.

<http://tools.ietf.org/html/draft-nourse-scep-23>

The following abstract is cited from the IETF draft:

"This document specifies the Simple Certificate Enrollment Protocol (SCEP), a Public Key Infrastructure (PKI) communication protocol which leverages existing technology by using PKCS#7 and PKCS#10 over HTTP. SCEP is the evolution of the enrollment protocol developed by VeriSign, Inc. for Cisco Systems, Inc. It now enjoys wide support in both client and a Certification Authority implementations."

SCEP is already supported by most commercial and free PKI solutions and can (as defined in the standard) perform the following actions:

- Automatic certificate enrollment based on authorization codes
- Automatic certificate enrollment with manual authorization
- Certificate renewal with authorization based on the old, still valid certificate
- Online Certificate revocation check
- Using a special proxy role to proxy requires through the DMZ tot he internal network

A subset of the SCEP features is supported by iOS Devices to rollout certificates during the so called over the air enrollment.



2.2 Wireless LAN

Clients can authenticate against a WLAN infrastructure in several ways. There are the very old and insecure methods using MAC Address based authentication, SSID based authentication or WEP authentication.

The newer and more secure authentication methods are based on the WPA/WPA2 standard. WPA is basically a subset of WPA2 using the algorithms already used in WEP for encryption but without the known weaknesses. This enables users to use their old hardware.

For the SOHO sector WPA with pre-shared key authentication is used. For the enterprise sector there is the EAP protocol available which defines an overall authentication framework. On top of that an authentication protocol has to be used – this is called EAP method. EAP based authentication enables personalized user authentication by either username and password or client certificates – the specific implementation depends on the used EAP method. The two most known and widely used EAP methods in WLAN environments are PEAP and EAP-TLS. The EAP framework is defined in RFC 5247.

2.2.1 Extensible Authentication Protocol (EAP)

The RFC defines it as follows:

„The Extensible Authentication Protocol (EAP), defined in RFC 3748, enables extensible network access authentication. This document specifies the EAP key hierarchy and provides a framework for the transport and usage of keying material and parameters generated by EAP authentication algorithms, known as "methods". It also provides a detailed system-level security analysis, describing the conditions under which the key management guidelines described in RFC 4962 can be satisfied.“

2.2.2 EAP-TLS

EAP-TLS is a standardized protocol defined in [RFC 5216](#) which commonly supported by all hardware vendors of wireless LAN devices. It supports certificate-based mutual authentication and key derivation. In combination with an authentication server and PKI it offers a strong security requiring client-side certificates and user credentials. The RFC abstract defines it as follows:

"The Extensible Authentication Protocol (EAP), defined in RFC 3748, provides support for multiple authentication methods. Transport Layer Security (TLS) provides for mutual authentication, integrity-protected ciphersuite negotiation, and key exchange between two endpoints. This document defines EAP-TLS, which includes support for certificate-based mutual authentication and key derivation.“

2.2.3 PEAP

Protected EAP was initially invented by Cisco, Microsoft and RSA and defines a chaining of different EAP methods. PEAP itself does not define an actual authentication method. Most widely supported are MSCHAPv2 and EAP-GTC for authentication based on username and password combination.

2.3 Virtual Private Network

Virtual private networks (VPN) are typically used to interconnect networks across insecure networks. There are basically three different deployment scenarios:

- Site to site
- End to site
- End to end

The only relevant deployment scenario for connecting mobile devices is end to site. It is used to establish a "secure" connection to the corporate network in order to allow the mobile devices to access internal resources.

Most people think of VPN by means of a technology encrypting traffic over insecure networks, but not all of the available VPN technologies use encryption. PPTP for example is a widely supported VPN protocol but does not encrypt data at all. Also most of the provider VPN technologies are based on traffic separation instead of encryption (e.g. MPLS, Famerelay). The most common VPN technologies used for connecting mobile devices are the IPSec standard and proprietary SSL VPN solutions.

2.3.1 IPSec

The IPSec standard basically is not a single protocol but rather a complete protocol family. The ISAKMP protocol is used to authenticate the peers, perform the key exchange and negotiate the required connection parameters (so called "security associations") between the peers. For the actual data transport additional protocols are used. There is the AH protocol (authentication header) which only authenticates the packets to ensure data integrity. The protocol used most widely for data transportation is called ESP – Encapsulated security payload – which authenticates and encrypts the traffic. IPSec is a really complex technology which supports a lot of different configuration variants which this document does not cover.

2.3.2 SSL VPN

In the meantime SSL based VPN solutions are becoming more and more popular. Nearly every "big player" in the VPN business has its own solution. SSL VPNs are not standardized at all. While SSL (Secure Socket Layer) is used to authenticate the peers and to encrypt the transferred packets, the actual way the packet encapsulation (IP in IP) is done is proprietary – thus making the different solutions incompatible.

In general there are also different levels on which a SSL VPN operates:

- Clientless HTTP access only – kind of a reverse proxy solution.
- Clientless browser based solution with some Java client embedded (like Citrix).
- Client based with full IP connection – comparable to an IPSec VPN.

3 CERTIFICATE SUPPORT ON IOS DEVICES

Devices running the iPhone Operating System (iOS) have support for encrypted SSL connections and client certificate based authentication. This section describes what kind of certificates can be used for which purposes on iOS devices. The Apple devices running iOS are:

- iPhone
- iPad
- iPod Touch (Could be used in business as a kind of "small iPad")
- Apple TV (Already seen in corporate environments attached to projectors as Airplay receiver)

3.1 Configuration profiles

Apple has a well-defined interface to configure devices. The so-called configuration profiles are used to apply device configurations to the devices. A configuration profile is basically a simple XML-File which includes different payloads (e.g. a WLAN profile, a root certificate, and the like). There are different types of these payloads:

- Mandatory payloads which can only appear once (The general settings).
- Optional payloads which can only appear once (Restrictions, MDM, APN).
- Optional payloads which can appear multiple times (All others, e.g. WLAN profiles, certificates, ...).

Apple provides an "iPhone configuration utility (IPCU)" which can be used to create the configuration profiles. The tool is available for Windows and MAC OS X free of charge from the Apple website.

Configuration profiles can also be signed and encrypted for a specific device. Configuration profiles can be deployed to the device in different ways:

- iPhone Configuration utility via USB port.
- Email or download after export from the IPCU.
- Over the air provisioning – typically used by MDM solutions to provision devices.

For encrypting the profiles the installed device certificate is used. Every Apple device has a device certificate preinstalled – which can be exchanged by a custom certificate.

3.2 Certificate usage on iOS devices

iOS devices can use certificates for different use cases. First there are the general SSL encrypted connections which validate the server certificate if it is valid and trusted. Therefore – if certificates from private PKI installations are used – the corresponding root certificates have to be imported. Second, iOS devices can authenticate against some type of backend using client device certificates. Third, iOS devices can authenticate against some backend using client user certificates. And last, iOS devices can be provisioned using encrypted configuration profiles (encrypted for a specific device using its device certificate).

3.2.1 Encrypted connections

iOS devices can use encrypted SSL/TLS connections to access different backend systems. Nearly all configurable backends can secure their communications this way. The following backends support this:

- Email Backend systems, including POP3, IMAP and SMTP
- Exchange Active Sync
- LDAP directory access
- CalDAV (Calendar access)
- CardDAV (Adressbook)
- Subscribed Calendars (Read Only)
- The browser, of course, (Safari)

In order to have a secure connection, the root certificate must be known and trusted by the device (see importing root certificates).

3.2.2 Certificates for device authentication

Some backend systems support device authentication using client certificates. Each device has a device certificate from Apple preinstalled. This can be replaced by a custom certificate. The backends supporting device authentication are basically the technologies to access corporate networks. These include authenticating the device for WLAN connections, IPSec VPN connections and SSL VPN connections. Certificates enrolled using SCEP can be used for device authentication.

3.2.3 Certificates for user authentication

To separate certificates for device and user authentication basically originates from the traditional multi-user workstations. The device certificate was used to authenticate the device in order to get a VPN, WLAN or wired LAN network connection. Every user on the device (e.g. PC) could access/use those certificates. On top of that, user certificates were used to authenticate the user to a specific backend system (e.g. web server, mail server, and the like) – Only the owner (user) can access those certificates.

The modern smartphone and tablet devices don't have a multi-user concept – and they are typically assigned to only one user. If multiple users are sharing one device (e.g. an iPad) there is no way to have a device certificate for all users and separate user certificates for each user installed on the device. Apple however differentiates the two types of certificates. This means, that the device certificates enrolled using SCEP cannot be used to authenticate the user.

User certificate authentication is currently only supported for the Exchange Active Sync (EAS) backend and a separate certificate has to be imported manually – no auto enrollment for this. All other SSL capable backends do not support certificate-based authentication.

3.2.4 Encrypted profiles


Apple configuration profiles can be encrypted so they can only be used by the device the profile was created for. Encrypted profiles can be created by the iPhone configuration utility export function. This is recommended if the profile is to be sent by mail or placed on a website for download – at least if the profile includes confidential data like certificates. The over the air enrollment process also uses encrypted profiles to provision the iOS device – as commonly used by MDM solutions.

3.3 Import of private root certificates

Root certificates which identify an private/corporate certificate authority must be imported to the device configuration in order to be able to validate server certificates. This can be done using the "credential payload" in a configuration profile. The certificate can be imported from a file (MAC OS X only) and from the Windows Certificate store (Windows only). To perform this using the apple configuration utility a new credential payload has to be added – then a popup will show up to select the certificate. The following screenshot shows an imported root certificate:

Credential Name
Name or description of the credential

Certificate or Identity Data
PKCS1 (.cer, etc) or PKCS12 (.p12) files for inclusion on device



CA01
Root certificate authority
Expired: Thursday, December 16, 2010
4:56:05 PM Germany Time
+ This certificate is marked as trusted for this account

▶ **Details**

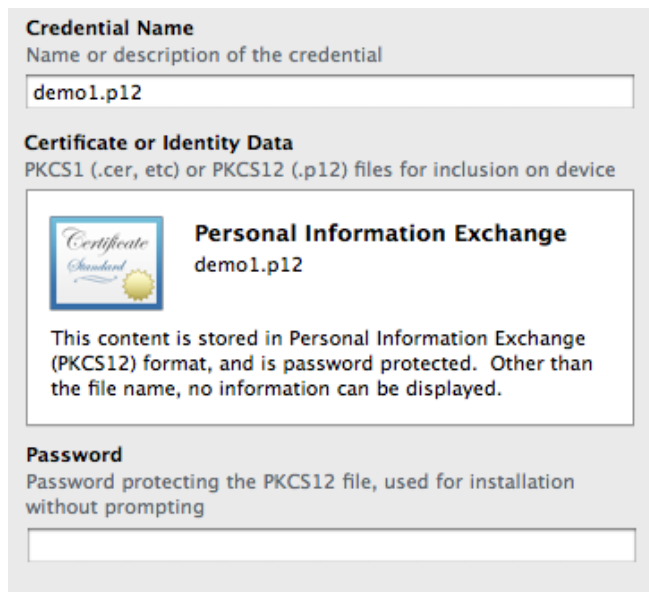
Password
Password protecting the PKCS12 file, used for installation without prompting

Own root certificates will show up as "not trusted" cause they cannot be validated. The password protection field is only required when importing client certificates which include a private key.

3.4 Import of device certificates


If no auto enrollment with SCEP is to be used, client device certificates can also be imported using the credentials payload. The certificates can then be selected for WLAN or VPN authentication in the payloads. The process using the iPhone configuration utility is the same as for the root certificate import which the exception that a container password has to be provided (which protects the private key) in order for the device to be able to read the private key.

The following screenshot shows an imported device certificate including a private key:



Credential Name
Name or description of the credential

Certificate or Identity Data
PKCS1 (.cer, etc) or PKCS12 (.p12) files for inclusion on device



Personal Information Exchange
demo1.p12

This content is stored in Personal Information Exchange (PKCS12) format, and is password protected. Other than the file name, no information can be displayed.

Password
Password protecting the PKCS12 file, used for installation without prompting

3.5 Certificate auto enrollment using SCEP

The only auto enrollment method supported by iOS devices is the simple certificate enrollment protocol. As described in the background section, this still is an IETF draft – not a standard. This auto enrollment method is part of the over the air enrollment flow defined by apple (described later). But it can also be performed manually without a complete device provisioning.

For this to work, a “SCEP payload” has to be included in the configuration profile. The enrollment then takes place during the profile installation.

The following options can be configured in this payload:

- URL

„This is the address of the SCEP server.“

- Name [optional]

“This can be any string that is understood by the certificate authority. It can be used to distinguish between instances, for example.”

- Subject [optional]:

„The representation of a X.500 name represented as an array of OID and value. For example, /C=US/O=Apple Inc./CN=foo/1.2.5.3=bar, which translates to: [[[“C”, “US”]], [[“O”, “Apple Inc.”]], ..., [[“1.2.5.3”, “bar”]]]”

- Subject alternative name type

“Specify 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.”

- Subject Alternative Name

„A pre-shared secret the SCEP server can use to identify the request or user.“

- NT Principal name [optional]

“NT principal to be used in the request”

- Challenge

„A pre-shared secret the SCEP server can use to identify the request or user.“

- Key size

“1024 | 2048”

- Key usage

"digital signature and key encipherment"

■ Fingerprint

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

3.5.1 Known limitations

Although iOS devices support SCEP, they do not support all of its features. One of the most important limitations is that certificates must be automatically issued. Manual request authorization by an administrator is not possible. If the certificate is not issued automatically, the profile installation will fail (SCEP enrollment takes place during profile installation).

3.5.2 Supported Certificate Authorities

Currently Apple supports the certificate authorities from Microsoft and Cisco. Other CAs also work if they have a compatible SCEP implementation. There can occur compatibility issues because SCEP is not yet standardized. The IETF draft is updated every six months. If a different CA should be used in a certain environment, a proof of concept should be performed.

See section "Example certificate enrollment using a managed PKI" for an example configuration using a unsupported CA.

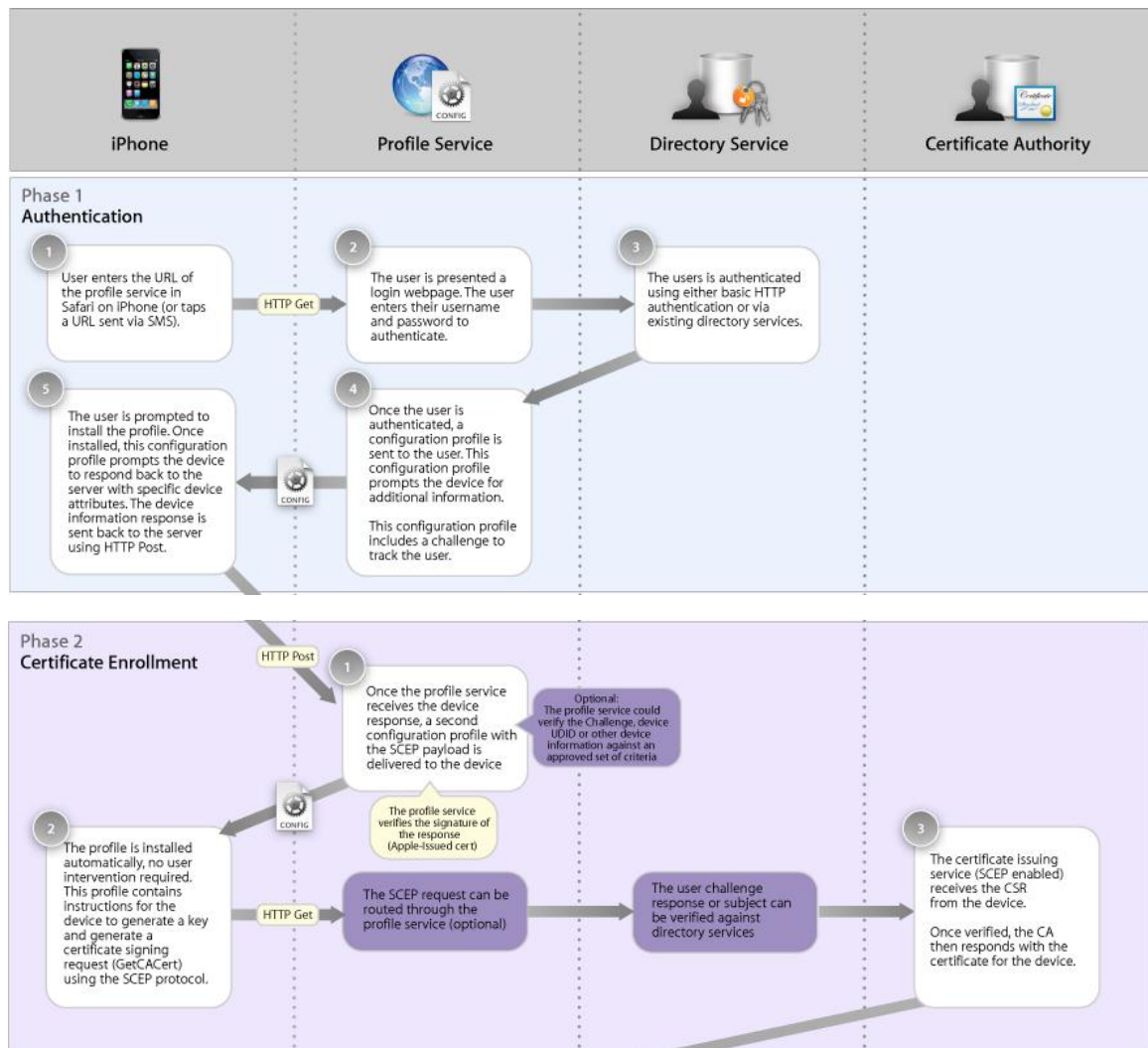
3.6 Over the air provisioning

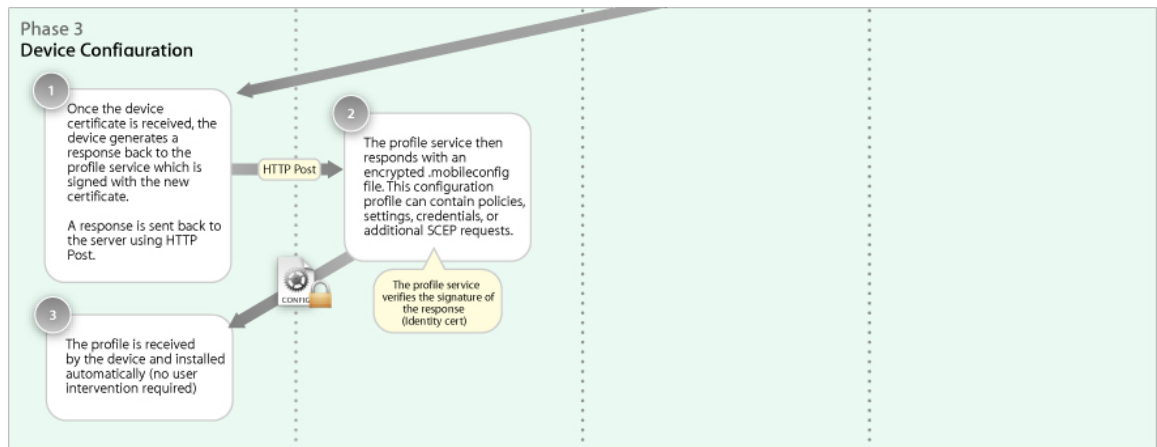
The over the air provisioning is the only really wireless way to provision iOS devices. The process itself is not covered by this document, but only the certificate enrollment part. The over the air provisioning process is divided in three phases:

- Device authentication/Initial configuration
- Certificate Enrollment using SCEP
- Final device configuration (with encrypted profile – using the just issued certificate)

All available mobile device management solutions use this process to provision iOS devices as this is the way defined by Apple.

The following graphics are from the apple over the air provisioning description:

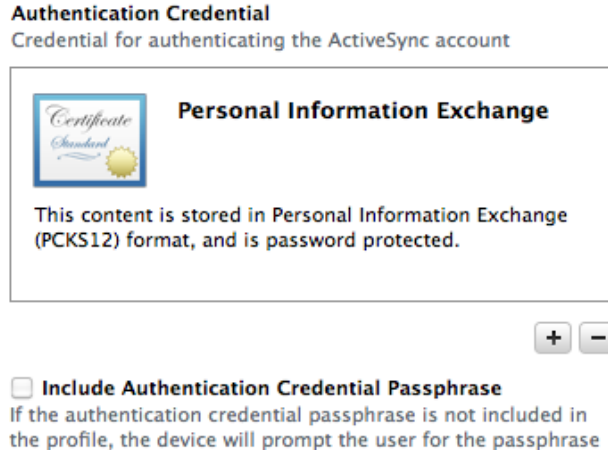




3.7 Import of user certificates for Exchange Active Sync

The only backend which supports user client certificates is the Exchange Active Sync (EAS) connection. Unfortunately, EAS cannot use the certificates issued by SCEP nor can it use the certificates imported as credential payload. The configuration profile payload for EAS has its own certificate selection dialog. To automatically enroll EAS user certificates, the mobile device management solution must enroll these by including them in the profile installed on the device.

The following screenshot shows the certificate section in the EAS payload:



EAS configuration with and without certificates is not covered in this document more detailed.

3.8 Supported EAP methods for WLAN Authentication

iOS devices support the following EAP methods for authentication in WLAN networks:

- EAP-TLS
- EAP-TTLS
- EAP-FAST
- EAP-SIM
- PEAP v0, PEAP v1
- LEAP

See the WLAN configuration example section for an example configuration using EAP-TLS.

In addition to the EAP method, the WLAN connection trust settings can be configured. This means it can be configured which server certificates the iOS device should trust and if the user is allowed to accept certificate validation errors.

NOTE:

iOS devices can have a per connection proxy configuration. As of iOS 4.x this cannot be configured through a configuration profile but has to be set manually by the user. VPN connections do not have this limitation. However, as the beta version of the upcoming iOS 5 shows, this option reappeared in the configuration utility. This means that iOS 5 fixes this issue.

3.9 VPN

iOS devices support the following VPN backends:

- L2TP
- PPTP
- Cisco IPsec
- SSL
 - Cisco Anyconnect
 - Juniper SSL
 - F5 SSL
 - Custom SSL

Cisco IPsec and SSL based VPNs support client certificate based authentication. Because SSL VPNs are proprietary, every SSL VPN requires an additional application to be installed on the device.

4 EXAMPLE CONFIGURATION USING A MANAGED PKI

This section describes an example SCEP configuration using a hosted/managed PKI solution. Please note that if a different CA/Hosted PKI is used, the settings possible must be adjusted. This example also includes a WLAN payload to use the issued certificate to authenticate the device in an EAP-TLS WLAN environment.

4.1 Creation of an authentication challenge

In order to use automatic enrollment of certificates, the SCEP client has to authenticate its request. This is done by creating a so called "passcode" on the PKI admin interface.

A passcode is always associated to a name. The passcode and the name have to be included in the SCEP client configuration.

In the admin interface of the PKI, a passcode has to be created.

Managed PKI Control Center

Create Passcode

This page enables you to create and upload a single new passcode match value set.

Note: All values except Passcode are matched in a case-sensitive manner. Passcodes are case-insensitive.

Fill in all fields and click **Submit**.

If your account supports CEP Enrollment for devices, then enter this exact text: **DO-NOT-MATCH** for fields that are irrelevant values to DO-NOT-MATCH.

Passcode Information

Passcode	<input type="text" value="0123456789"/>
First Name	<input type="text" value="rgraf-ipad.ernw-pki.de"/>

BACK

HELP

SUBMIT

If the certificate rollout is done by a mobile device management solution, this step would be automatically performed by the MDM solution. However, the integration of a CA with a MDM solution is not covered by this document.

4.2 Configuration profile

An iPhone configuration profile was created using the Apple iPhone Configuration Utility.

The following payloads are required for a working configuration profile to enroll certificates:

- General settings
- Credentials (Root Certificate)
- SCEP

4.2.1 General settings

General settings identify the configuration profile. It has to include a unique identifier.

Name

Display name of the profile (shown on the device)

Identifier

Unique identifier for the profile (e.g. com.company.profile)

Organization

Name of the organization for the profile

Description

Brief explanation of the contents or purpose of the profile

Security

Controls when the profile can be removed

4.2.2 Credentials

The credentials payload includes the required Root certificates (CA).

The following screenshot shows the imported root certificate:


Credential Name

Name or description of the credential

CA01

Certificate or Identity Data

PKCS1 (.cer, etc) or PKCS12 (.p12) files for inclusion on device



CA01
Root certificate authority
Expired: Thursday, December 16, 2010
4:56:05 PM Germany Time
⚙ This certificate is marked as trusted for
this account

▶ Details

Password

Password protecting the PKCS12 file, used for installation
without prompting

4.2.3 SCEP settings

The following screenshots show the required SCEP payload settings:

URL
The base URL for the SCEP server

Name
The name of the instance: CA-IDENT

Subject
Representation of a X.500 name

Subject Alternative Name Type
The type of a subject alternative name

Subject Alternative Name Value
The value of a subject alternative name

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

Challenge
Used as the pre-shared secret for automatic enrollment

Key Size
Key size in bits

Use as digital signature

Use for key encipherment

Fingerprint
Enter hex string to be used as a fingerprint or use button to create fingerprint from Certificate

The passcode name must be included in the "Subject Alternative Name Value" Field with type "DNS Name" and the passcode must be included as "Challenge". Also the name attribute must end with "ernw-pki.com".

4.2.4 WLAN settings

The following screenshots show the required WLAN payload settings:

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

ERNW

Hidden Network
Enable if target network is not open or broadcasting

Security Type
Wireless network encryption to use when connecting

WPA / WPA2 Enterprise

Enterprise Settings
Configuration of protocols, authentication, and trust

Protocols Authentication Trust

Accepted EAP Types
Authentication protocols supported on target network

TLS LEAP EAP-FAST
 TTLS PEAP EAP-SIM

EAP-FAST
Configuration of Protected Access Credential (PAC)

Use PAC
 Provision PAC
 Provision PAC Anonymously

The SSID and security type must be specified. Additionally, the accepted EAP methods must be specified (EAP-TLS in this case).

Protocols Authentication Trust

Username
Username for connection to wireless network

rgraf-ipad

Use Per-Connection Password
Request during connection and send with authentication

Password
Password for the provided username

Identity Certificate
Credentials for connection to wireless network

SCEP: http://ca01.ernw-pki.de - CA01

Outer Identity
Externally visible identification (for TTLS, PEAP, and EAP-FAST)

The username must only be specified if the used WLAN environment matches the authenticating user with some backend for authorization. The identity certificate selection simply specifies the created SCEP profile instead of an imported certificate.

Protocols Authentication **Trust**

Trusted Certificates
Certificates trusted/expected for authentication

<input checked="" type="checkbox"/> hdb-ca-001

Trusted Server Certificate Names
Certificate names expected from authentication server

+ -

Allow Trust Exceptions
Allow trust decisions (via dialog) to be made by the user

Finally, the trust settings tell the device which root certificates to trust. Optionally the server names of the authentication backend can be specified. The "Allow trust exceptions" checkbox specifies, if the user has the choice to accept validation errors.