Configuration Profile Reference

Developer

Contents

Tables

Configuration Profile Key Reference

Note: This document was previously titled *iPhone Configuration Profile Reference*. It now supports both iOS and OS X.

A configuration profile is an XML file that allows you to distribute configuration information. If you need to configure a large number of devices or to provide lots of custom email settings, network settings, or certificates to a large number of devices, configuration profiles are an easy way to do it.

A configuration profile contains a number of settings that you can specify, including:

- Restrictions on device features
- Wi-Fi settings
- VPN settings
- Email server settings
- Exchange settings
- LDAP directory service settings
- CalDAV calendar service settings
- Web clips
- Credentials and keys

Note: OSX versions 10.10 and later honor a `true` value of the `PayloadRemovalDisallowed` key to prevent manual removal of profiles installed through an MDM server. Such profiles cannot be removed using the Profiles preference pane, nor the profiles command line tool even when run as root. Only the MDM server can remove such profiles. Profiles installed manually, with `PayloadRemovalDisallowed` set to `true`, can be removed manually, but only by using administrative authority.

Configuration profiles are written in property list format, with `Data` values stored in Base64 encoding. The `.plist` format can be read and written by any XML library.

There are five ways to deploy configuration profiles:

- Using Apple Configurator 2, available in the App Store
- In an email message
- On a webpage
- Using over-the air configuration as described in *Over-the-Air Profile Delivery and Configuration*
- Over the air using a Mobile Device Management Server

> **Note:** Profile installation fails when the device is locked with a passcode.

Both iOS and OS X support using encryption to protect the contents of profiles. Profiles can also be signed to guarantee data integrity. To learn about encrypted profile delivery, read *Over-the-Air Profile Delivery and Configuration* .

Devices can be supervised when preparing them for deployment with Apple Configurator 2 (iOS 5 or later) or by using the Device Enrollment Program (iOS 7 or later). .

For general information about the Device Enrollment Program, visit Apple's Corporate-owned deployments made simple or IT in Education. For details, go to Apple Deployment Programs Help.

When a device is supervised, you can use configuration profiles to control many of its settings. This document describes the available keys in a profile and provides examples of the resulting XML payloads.

> **Note:** Before you get started working with configuration profiles, you should create a skeleton profile. This provides a useful starting point that you can then modify as desired.

Configuration Profile Keys

At the top level, a profile property list contains the following keys:

Key	Type	Content
`PayloadContent`	Array	Optional. Array of payload dictionaries. Not present if IsEncrypted is `true`.
`PayloadDescription`	String	Optional. A description of the profile, shown on the Detail screen for the profile. This should be descriptive enough to help the user decide whether to install the profile.

Key	Type	Content
`PayloadDisplayName`	String	Optional. A human-readable name for the profile. This value is displayed on the Detail screen. It does not have to be unique.
`PayloadExpiration-Date`	Date	Optional. A date on which a profile is considered to have expired and can be updated over the air. This key is only used if the profile is delivered via Over The Air profile delivery.
`PayloadIdentifier`	String	A reverse-DNS style identifier (com.example.myprofile, for example) that identifies the profile. This string is used to determine whether a new profile should replace an existing one or should be added.
`PayloadOrganization`	String	Optional. A human-readable string containing the name of the organization that provided the profile.
`PayloadUUID`	String	A globally unique identifier for the profile. The actual content is unimportant, but it must be globally unique. In OS X, you can use `uuidgen` to generate reasonable UUIDs.
`PayloadRemoval-Disallowed`	Boolean	Optional. Supervised only. If present and set to `true`, the user cannot delete the profile (unless the profile has a removal password and the user provides it).
`PayloadType`	String	The only supported value is `Configuration`.
`PayloadVersion`	Number	The version number of the profile format. This describes the version of the configuration profile as a whole, not of the individual profiles within it. Currently, this value should be `1`.
`PayloadScope`	String	Optional. Determines if the profile should be installed for the system or the user. In many cases, it determines the location of the certificate items, such as keychains. Though it is not possible to declare different payload scopes, payloads, like VPN, may automatically install their items in both scopes if needed. Legal values are `System` and `User`, with `User` as the default value. **Availability:** Available in OS X 10.7 and later.

Key	Type	Content
`RemovalDate`	date	Optional. The date on which the profile will be automatically removed.
`DurationUntilRemoval`	float	Optional. Number of seconds until the profile is automatically removed. If the `RemovalDate` keys is present, whichever field yields the earliest date will be used.
`ConsentText`	Dictionary	Optional. This dictionary's keys must be locale strings that contain a canonicalized IETF BCP 47 language identifier. Additionally, the key `default` may be present to provide the default localization. The system chooses a localized version in the order of preference specified by the user (OS X) or based on the user's current language setting (iOS). If no exact match is found, the default localization is used. If there is no default localization, the `"en"` localization is used. If there is no `"en"` localization, then the first available localization is used. You should provide a default localization. No warning will be displayed if the user's locale does not match any of the localizations in the consentText dictionary.

Note: Profile payload dictionary keys that are prefixed with "Payload" are reserved key names and must never be treated as managed preferences. Any other key in the payload dictionary may be considered a managed preference for that preference domain.

Keys in the payload dictionary are described in detail in the next section.

Payload Dictionary Keys Common to All Payloads

If a `PayloadContent` value is provided in a payload, each entry in the array is a dictionary representing a configuration payload. The following keys are common to all payloads:

Key	Type	Content
`PayloadType`	String	The payload type. The payload types are described in Payload-Specific Property Keys (page 9).

Key	Type	Content
`PayloadVersion`	Number	The version number of the individual payload. A profile can consist of payloads with different version numbers. For example, changes to the VPN software in iOS might introduce a new payload version to support additional features, but Mail payload versions would not necessarily change in the same release.
`PayloadIdentifier`	String	A reverse-DNS-style identifier for the specific payload. It is usually the same identifier as the root-level `PayloadIdentifier` value with an additional component appended.
`PayloadUUID`	String	A globally unique identifier for the payload. The actual content is unimportant, but it must be globally unique. In OS X, you can use `uuidgen` to generate reasonable UUIDs.
`PayloadDisplayName`	String	A human-readable name for the profile payload. This name is displayed on the Detail screen. It does not have to be unique.
`PayloadDescription`	String	Optional. A human-readable description of this payload. This description is shown on the Detail screen.
`PayloadOrganization`	String	Optional. A human-readable string containing the name of the organization that provided the profile. The payload organization for a payload need not match the payload organization in the enclosing profile.

Payload-Specific Property Keys

In addition to the standard payload keys (described in Payload Dictionary Keys Common to All Payloads (page 8)), each payload type contains keys that are specific to that payload type. The sections that follow describe those payload-specific keys.

Active Directory Certificate Profile Payload

The Active Directory Certificate Profile payload is designated by specifying `com.apple.ADCertificate.managed` as the `PayloadType` value.

You can request a certificate from a Microsoft Certificate Authority (CA) using DCE/RPC and the Active Directory Certificate profile payload instructions detailed at support.apple.com/kb/HT5357.

This payload includes the following unique keys:

Key	Type	Value
`AllowAllAppsAccess`	Boolean	If `true`, apps have access to the private key.
`CertServer`	String	Fully qualified host name of the Active Directory issuing CA.
`CertTemplate`	String	Template Name as it appears in the General tab of the template's object in the Certificate Templates' Microsoft Management Console snap-in component.
`Certificate-AcquisitionMechanism`	String	Most commonly `RPC`. If using 'Web enrollment,' `HTTP`.
`CertificateAuthority`	String	Name of the CA. This value is determined from the Common Name (CN) of the Active Directory entry: CN=<your CA name>, CN='Certification Authorities', CN='Public Key Services', CN='Services', or CN='Configuration', <your base Domain Name>.
`CertificateRenewal-TimeInterval`	Integer	Number of days in advance of certificate expiration that the notification center will notify the user.
`Description`	String	User-friendly description of the certification identity.
`KeyIsExtractable`	Boolean	If `true`, the private key can be exported.
`PromptForCredentials`	Boolean	This key applies only to user certificates where Manual Download is the chosen method of profile delivery. If `true`, the user will be prompted for credentials when the profile is installed. Omit this key for computer certificates.
`Keysize`	Integer	Optional; defaults to 2048. The RSA key size for the Certificate Signing Request (CSR). **Availability:** Available in OS X 10.11 and later.

AirPlay Payload

The AirPlay payload is designated by specifying `com.apple.airplay` as the `PayloadType` value.

This payload is supported on iOS 7.0 and later and on OS X 10.10 and later.

Key	Type	Value
`Whitelist`	Array of dictionaries	Optional. Supervised only (ignored otherwise). If present, only AirPlay destinations present in this list are available to the device. The dictionary format is described below.
`Passwords`	Array of dictionaries	Optional. If present, sets passwords for known AirPlay destinations. The dictionary format is described below.

Each entry in the `Whitelist` array is a dictionary that can contain the following fields:

Key	Type	Value
`DeviceID`	String	The Device ID of the AirPlay destination, in the format `xx:xx:xx:xx:xx:xx`. This field is not case sensitive.

Each entry in the `Passwords` array is a dictionary that contains the following fields:

Key	Type	Value
`DeviceName`	String	The name of the AirPlay destination (used on iOS).
`DeviceID`	String	The `DeviceID` of the AirPlay destination (used on OS X).
`Password`	String	The password for the AirPlay destination.

AirPrint Payload

The AirPrint payload adds AirPrint printers to the user's AirPrint printer list. This makes it easier to support environments where the printers and the devices are on different subnets. An AirPrint payload is designated by specifying `com.apple.airprint` as the `PayloadType` value.

This payload is supported on iOS 7.0 and later and on OS X 10.10 and later.

Key	Type	Value
`AirPrint`	Array of dictionaries	An array of AirPrint printers that should always be shown.

Each dictionary in the `AirPrint` array must contain the following keys and values:

Key	Type	Value
IPAddress	String	The IP Address of the AirPrint destination.
ResourcePath	String	The Resource Path associated with the printer. This corresponds to the rp parameter of the _ipps.tcp Bonjour record. For example: printers/Canon_MG5300_series printers/Xerox_Phaser_7600 ipp/print Epson_IPP_Printer

APN Payload

The APN (Access Point Name) payload is designated by specifying com.apple.apn.managed as the PayloadType value.

In iOS 7 and later, the APN payload is deprecated in favor of the Cellular payload.

The APN Payload is not supported in OS X.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
DefaultsData	Dictionary	This dictionary contains two key/value pairs.
DefaultsDomainName	String	The only allowed value is com.apple.managedCarrier.
apns	Array	This array contains an arbitrary number of dictionaries, each describing an APN configuration, with the key/value pairs below.
apn	String	This string specifies the Access Point Name.
username	String	This string specifies the user name for this APN. If it is missing, the device prompts for it during profile installation.
password	Data	Optional. This data represents the password for the user for this APN. For obfuscation purposes, the password is encoded. If it is missing from the payload, the device prompts for the password during profile installation.
Proxy	String	Optional. The IP address or URL of the APN proxy.

Key	Type	Value
ProxyPort	Number	Optional. The port number of the APN proxy.

Per-App VPN Payload

The Per-App VPN payload is used for configuring add-on VPN software, and it works only on VPN services of type 'VPN'. It should not be confused with the standard VPN payload, described in VPN Payload (page 55).

This payload is supported only in iOS 7.0 and later and OS X v10.9 and later.

The VPN payload is designated by specifying `com.apple.vpn.managed.applayer` as the `PayloadType` value. The Per-App VPN payload supports all of the keys described in VPN Payload (page 55) plus the following additional keys:

Key	Type	Value
VPNUUID	String	A globally-unique identifier for this VPN configuration. This identifier is used to configure apps so that they use the Per-App VPN service for all of their network communication. See App-to-Per-App VPN Mapping (page 14).
SafariDomains	Array	This optional key is a special case of App-to-Per App VPN Mapping. It sets up the app mapping for Safari (Webkit) with a specific identifier and a designated requirement. The array contains strings, each of which is a domain that should trigger this VPN connection in Safari. The rule matching behavior is as follows: • Before being matched against a host, all leading and trailing dots are stripped from the domain string. For example, if the domain string is `".com"` the domain string used to match is `"com"`. • Each label in the domain string must match an entire label in the host string. For example, a domain of `"example.com"` matches `"www.example.com"`, but not `"foo.badexample.com"`. • Domain strings with only one label must match the entire host string. For example, a domain of `"com"` matches `"com"`, not `"www.example.com"`.

Key	Type	Value
OnDemandMatch-AppEnabled	Boolean	This key is placed in the VPN payload sub-dictionary. If true, the Per-App VPN connection starts automatically when apps linked to this Per-App VPN service initiate network communication. If false, the Per-App VPN connection must be started manually by the user before apps linked to this Per-App VPN service can initiate network communication. If this key is not present, the value of the OnDemandEnabled key is used to determine the status of Per-App VPN On Demand.

VPN Dictionary Keys

In addition to the VPN Dictionary keys defined in the com.apple.vpn.managed payload, the VPN Dictionary within the com.apple.vpn.managed.applayer payload can also contain the following keys:

Key	Type	Value
ProviderType	String	Optional. Either packet-tunnel or app-proxy. The default is app-proxy. If the value of this key is app-proxy, then the VPN service will tunnel traffic at the application layer. If the value of this key is packet-tunnel, then the VPN service will tunnel traffic at the IP layer.

App-to-Per-App VPN Mapping

The App-to-Per-App mapping payload is designated by specifying com.apple.vpn.managed.appmapping as the PayloadType value.

This payload is supported only in OS X v10.9 and later. It is not supported in iOS.

Key	Type	Value
AppLayerVPNMapping	Array of dictionaries	An array of mapping dictionaries.

Each dictionary in the array can contain the following keys:

Key	Type	Value
Identifier	String	The app's bundle ID.
VPNUUID	String	The VPNUUID of the Per-App VPN defined in a Per-App VPN payload.

App Lock Payload

The App Lock payload is designated by specifying `com.apple.app.lock` as the `PayloadType` value. Only one of this payload type can be installed at any time. This payload can be installed only on a Supervised device.

By installing an app lock payload, the device is locked to a single application until the payload is removed. The home button is disabled, and the device returns to the specified application automatically upon wake or reboot.

This payload is supported only in iOS 6.0 and later.

The payload contains the following key:

Key	Type	Value
`App`	Dictionary	A dictionary containing information about the app.

The `App` dictionary, in turn, contains the following key:

Key	Type	Value
`Identifier`	String	The bundle identifier of the application.
`Options`	Dictionary	Optional. Described below. **Availability:** Available only in iOS 7.0 and later.
`UserEnabledOptions`	Dictionary	Optional. Described below. **Availability:** Available only in iOS 7.0 and later.

The `Options` dictionary, if present, can contain the following keys (in iOS 7.0 and later):

Key	Type	Value
`DisableTouch`	Boolean	Optional. If `true`, the touch screen is disabled. Default is `false`.
`DisableDevice-Rotation`	Boolean	Optional. If `true`, device rotation sensing is disabled. Default is `false`.
`DisableVolumeButtons`	Boolean	Optional. If `true`, the volume buttons are disabled. Default to `false`.
`DisableRingerSwitch`	Boolean	Optional. If `true`, the ringer switch is disabled. Default is `false`. When disabled, the ringer behavior depends on what position the switch was in when it was first disabled.

Key	Type	Value
`DisableSleep-WakeButton`	Boolean	Optional. If `true`, the sleep/wake button is disabled. Default is `false`.
`DisableAutoLock`	Boolean	Optional. If `true`, the device will not automatically go to sleep after an idle period.
`EnableVoiceOver`	Boolean	Optional. If `true`, VoiceOver is turned on. Default is `false`.
`EnableZoom`	Boolean	Optional. If `true`, Zoom is turned on. Default is `false`.
`EnableInvertColors`	Boolean	Optional. If `true`, Invert Colors is turned on. Default is `false`.
`EnableAssistiveTouch`	Boolean	Optional. If `true`, AssistiveTouch is turned on. Default is `false`.
`EnableSpeakSelection`	Boolean	Optional. If `true`, Speak Selection is turned on. Default is `false`.
`EnableMonoAudio`	Boolean	Optional. If `true`, Mono Audio is turned on. Default is `false`.

The `UserEnabledOptions` dictionary, if present, can contain the following keys (in iOS 7.0 and later):

Key	Type	Value
`VoiceOver`	Boolean	Optional. If `true`, allow VoiceOver adjustment. Default is `false`.
`Zoom`	Boolean	Optional. If `true`, allow Zoom adjustment. Default is `false`.
`InvertColors`	Boolean	Optional. If `true`, allow Invert Colors adjustment. Default is `false`.
`AssistiveTouch`	Boolean	Optional. If `true`, allow AssistiveTouch adjustment. Default is `false`.

CalDAV Payload

This payload configures a CalDAV account.

The payload is designated by specifying `com.apple.caldav.account` as the `PayloadType`

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`CalDAVAccount-Description`	String	Optional. The description of the account.
`CalDAVHostName`	String	The server address. In OS X, this key is required.
`CalDAVUsername`	String	The user's login name. In OS X, this key is required.
`CalDAVPassword`	String	Optional. The user's password
`CalDAVUseSSL`	Boolean	Whether or not to use SSL. In OS X, this key is optional.
`CalDAVPort`	Number	Optional. The port on which to connect to the server.
`CalDAVPrincipalURL`	String	Optional. The base URL to the user's calendar. In OS X this URL is required if the user doesn't provide a password, because auto-discovery of the service will fail and the account won't be created.

Calendar Subscription Payload

The calendar subscription payload is designated by specifying `com.apple.subscribedcalendar.account` as the `PayloadType` value.

A calendar subscription payload adds a subscribed calendar to the user's calendars list.

The calendar subscription payload is not supported in OS X.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`SubCalAccountDescription`	String	Optional. Description of the account.
`SubCalAccountHostName`	String	The server address.
`SubCalAccountUsername`	String	The user's login name
`SubCalAccountPassword`	String	The user's password.
`SubCalAccountUseSSL`	Boolean	Whether or not to use SSL.

CardDAV Payload

The CardDAV payload is designated by specifying `com.apple.carddav.account` as the `PayloadType` value.

As of OS X v10.8 and later, this payload type supports obtaining `CardDAVUsername` and `CardDAVPassword` from an Identification Payload, if present.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`CardDAVAccountDescription`	String	Optional. The description of the account.
`CardDAVHostName`	String	The server address.
`CardDAVUsername`	String	The user's login name.
`CardDAVPassword`	String	Optional. The user's password
`CardDAVUseSSL`	Boolean	Optional. Whether or not to use SSL.
`CardDAVPort`	Number	Optional. The port on which to connect to the server.
`CardDAVPrincipalURL`	String	Optional. Not supported on OS X. The base URL to the user's address book.

Cellular Payload

A cellular payload configures cellular network settings on the device. It is not supported on OS X. On iOS 7 and later, a cellular payload is designated by specifying `com.apple.cellular` as the `PayloadType` value. Cellular payloads have two important installation requirements:

- No more than one cellular payload can be installed at any time.
- A cellular payload cannot be installed if an APN payload is already installed.

This payload replaces the `com.apple.managedCarrier` payload, which is supported, but deprecated.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`AttachAPN`	Dictionary	Optional. An `AttachAPN` configuration dictionary, described below.

Key	Type	Value
APNs	Array	Optional. An array of APN dictionaries, described below. Only the first entry is currently used.

The AttachAPN dictionary contains the following keys:

Key	Type	Value
Name	String	Required. The Access Point Name.
AuthenticationType	String	Optional. Must contain either CHAP or PAP. Defaults to PAP.
Username	String	Optional. A user name used for authentication.
Password	String	Optional. A password used for authentication.

Each APN dictionary contains the following keys:

Key	Type	Value
Name	String	Required. The Access Point Name.
AuthenticationType	String	Optional. Must contain either CHAP or PAP. Defaults to PAP.
Username	String	Optional. A user name used for authentication.
Password	String	Optional. A password used for authentication.
ProxyServer	String	Optional. The proxy server's network address.
ProxyPort	Number	Optional. The proxy server's port.

Certificate Payload

The PayloadType of a certificate payload must be one of the following:

Payload type	Container format	Certificate type
com.apple.security.root	PKCS#1(.cer)	Alias for com.apple.security.pkcs1.
com.apple.security.pkcs1	PKCS#1(.cer)	DER-encoded certificate without private key. May contain root certificates.

Payload type	Container format	Certificate type
`com.apple.security.pem`	PKCS#1(.cer)	PEM-encoded certificate without private key. May contain root certificates.
`com.apple.security.pkcs12`	PKCS#12(.p12)	Password-protected identity certificate. Only one certificate may be included.

In addition to the settings common to all payloads, all Certificate payloads define the following keys:

Key	Type	Value
`PayloadCertificate-FileName`	String	Optional. The file name of the enclosed certificate.
`PayloadContent`	Data	Mandatory. The binary representation of the payload.
`Password`	String	Optional. For PKCS#12 certificates, contains the password to the identity.

> **Caution:** Because the password string is stored in the clear in the profile, it is recommended that the profile be encrypted for the device.

Education Configuration Payload

The Education Configuration Payload is designated by specifying `com.apple.education` as the `PayloadType` value. It can contain only one payload, which must be supervised. It is not supported on the User Channel.

The Education Configuration Payload defines the users, groups, and departments within an educational organization. It is supported on iOS 9.3 and later.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`OrganizationUUID`	String	Required. The organization's UUID identifier.
`OrganizationName`	String	Required. The organization's display name.
`PayloadCertificate-UUID`	String	Required. The UUID of an identity certificate payload that will be used to perform client authentication with other devices.

Key	Type	Value
`LeaderPayload-CertificateAnchor-UUID`	Array	Optional. An array of UUIDs referring to certificate payloads that will be used to authorize leader peer certificate identities.
`MemberPayload-CertificateAnchor-UUID`	Array	Optional. An array of UUIDs referring to certificate payloads that will be used to authorize group member peer certificate identities.
`UserIdentifier`	String	Optional. A unique string that identifies the user of this device within the organization.
`Departments`	Array	Optional. Shared: An array of dictionaries that define departments that are shown in the iOS login window. Leader: An array of dictionaries that define departments that are shown in the Classroom app.
`Groups`	Array	Required. Shared: An array of dictionaries that define groups that the user can select in the login window. Leader: An array of dictionaries that define the groups that the user can control. Member: An array of dictionaries that define the groups of which the user is a member.
`Users`	Array	Required. Shared: An array of dictionaries that define the users that are shown in the iOS login window. Leader: An array of dictionaries that define users that are members of the leader's groups. Member: An array of a dictionaries that must contain the definition of the user specified in the `UserIdentifier` key.
`DeviceGroups`	Array	Optional. Leader: An array of dictionaries that define the device groups that are members of device groups to which the leader can assign devices.

The `Departments` key must contain an array of dictionaries with the following key-value pairs:

Key	Type	Content
`Name`	String	Required: the display name of the department.
`GroupBeaconIDs`	Array	Required: group beacon identifiers that are members of this department.

The `Groups` key must contain an array of dictionaries with the following key-value pairs:

Key	Type	Content
`BeaconID`	Number	Required: unsigned 16 bit integer specifying this group's unique beacon ID.
`Name`	String	Required: the display name of the group.
`Description`	String	Optional: description of the group.
`ImageURL`	String	Optional: URL of an image for the group.
`ConfigurationSource`	String	Optional: the source that provided this group; e.g. iTunesU, SIS, or MDM.
`LeaderIdentifiers`	Array	Optional: user identifiers that are leaders of this group.
`MemberIdentifiers`	Array	Required: strings that refer to entries in the `Users` array that are members of the group.
`DeviceGroup-Identifiers`	Array	Required: identifier strings that refer to entries in the `DeviceGroups` array that are device groups to which the instructor can assign users from this class.

The `Users` key must contain an array of dictionaries with the following key-value pairs:

Key	Type	Content
`Identifier`	String	Required: uniquely identifies a user in the organization.
`Name`	String	Required: will be displayed as the name of the user.
`GivenName`	String	Optional: will be displayed as the given name of the user.
`FamilyName`	String	Optional: will be displayed as the family name of the user.
`ImageURL`	String	Optional: URL pointing to an image of the user. The `ResourcePayloadCertificateUUID` identity certificate or the MDM client identity will be used to perform authentication when fetching the specified resource.
`FullScreenImageURL`	String	Optional: URL pointing to an image of the user. The `ResourcePayloadCertificateUUID` identity certificate or the MDM client identity will be used to perform authentication when fetching the specified resource.

Key	Type	Content
AppleID	String	Optional: the Managed Apple ID for this user.
PasscodeType	String	Optional: the passcode UI to show when the user is at the login window; possible values are complex, four, or six.

The DeviceGroups key must contain an array of dictionaries with the following key-value pairs:

Key	Type	Content
Identifier	String	Required: uniquely identifies the device group in the organization.
Name	String	Required: will be displayed as the name of the device group.
SerialNumbers	Array	Required: strings containing the serial numbers of the devices in the group.

Email Payload

The email payload is designated by specifying com.apple.mail.managed as the PayloadType value.

An email payload creates an email account on the device.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
EmailAccount-Description	String	Optional. A user-visible description of the email account, shown in the Mail and Settings applications.
EmailAccountName	String	Optional. The full user name for the account. This is the user name in sent messages, etc.
EmailAccountType	String	Allowed values are EmailTypePOP and EmailTypeIMAP. Defines the protocol to be used for that account.
EmailAddress	String	Designates the full email address for the account. If not present in the payload, the device prompts for this string during profile installation.
IncomingMailServer-Authentication	String	Designates the authentication scheme for incoming mail. Allowed values are EmailAuthPassword and EmailAuthNone.

Key	Type	Value
`IncomingMailServer-HostName`	String	Designates the incoming mail server host name (or IP address).
`IncomingMailServer-PortNumber`	Number	Optional. Designates the incoming mail server port number. If no port number is specified, the default port for a given protocol is used.
`IncomingMailServer-UseSSL`	Boolean	Optional. Default `false`. Designates whether the incoming mail server uses SSL for authentication.
`IncomingMailServer-Username`	String	Designates the user name for the email account, usually the same as the email address up to the @ character. If not present in the payload, and the account is set up to require authentication for incoming email, the device will prompt for this string during profile installation.
`IncomingPassword`	String	Optional. Password for the Incoming Mail Server. Use only with encrypted profiles.
`OutgoingPassword`	String	Optional. Password for the Outgoing Mail Server. Use only with encrypted profiles.
`OutgoingPasswordSame-AsIncomingPassword`	Boolean	Optional. If set, the user will be prompted for the password only once and it will be used for both outgoing and incoming mail.
`OutgoingMailServer-Authentication`	String	Designates the authentication scheme for outgoing mail. Allowed values are EmailAuthPassword and EmailAuthNone.
`OutgoingMailServer-HostName`	String	Designates the outgoing mail server host name (or IP address).
`OutgoingMailServer-PortNumber`	Number	Optional. Designates the outgoing mail server port number. If no port number is specified, ports 25, 587 and 465 are used, in this order.
`OutgoingMailServer-UseSSL`	Boolean	Optional. Default `false`. Designates whether the outgoing mail server uses SSL for authentication.
`OutgoingMailServer-Username`	String	Designates the user name for the email account, usually the same as the email address up to the @ character. If not present in the payload, and the account is set up to require authentication for outgoing email, the device prompts for this string during profile installation.

Key	Type	Value
PreventMove	Boolean	Optional. Default false. If true, messages may not be moved out of this email account into another account. Also prevents forwarding or replying from a different account than the message was originated from. **Availability:** Available only in iOS 5.0 and later.
PreventAppSheet	Boolean	Optional. Default false. If true, this account is not available for sending mail in any app other than the Apple Mail app. **Availability:** Available only in iOS 5.0 and later.
SMIMEEnabled	Boolean	Optional. Default false. If true, this account supports S/MIME. **Availability:** Available only in iOS 5.0 and later.
SMIMESigning-CertificateUUID	String	Optional. The PayloadUUID of the identity certificate used to sign messages sent from this account. **Availability:** Available only in iOS 5.0 and later.
SMIMEEncryption-CertificateUUID	String	Optional. The PayloadUUID of the identity certificate used to decrypt messages sent to this account. **Availability:** Available only in iOS 5.0 and later.
SMIMEEnablePer-MessageSwitch	Boolean	Optional. If set to true, enable the per-message signing and encryption switch. Defaults to true. **Availability:** Available only in iOS 8.0 and later.
disableMailRecents-Syncing	Boolean	If true, this account is excluded from address Recents syncing. This defaults to false. **Availability:** Available only in iOS 6.0 and later.

802.1x Ethernet Payload

The 802.1x Ethernet payload is designated by specifying one of the following as the PayloadType value:

com.apple.firstactiveethernet.managed [default]

com.apple.firstethernet.managed

com.apple.secondactiveethernet.managed

```
com.apple.secondethernet.managed
com.apple.thirdactiveethernet.managed
com.apple.thirdethernet.managed
```

Payloads with "active" in their name apply to Ethernet interfaces that are working at the time of profile installation. If there is no active Ethernet interface working, the `com.apple.firstactiveethernet.managed` payload will configure the interface with the highest service order priority.

Payloads without "active" in the name apply to Ethernet interfaces according to service order regardless of whether the interface is working or not.

There is currently no support for a BSD level specifier.

To specify an enterprise profile for a given 802.1x network, include the `EAPClientConfiguration` key in the payload, as described in EAPClientConfiguration Dictionary (page 76).

Exchange Payload

In iOS, the Exchange payload is designated by specifying `com.apple.eas.account` as the `PayloadType` value. This payload configures an Exchange Active Sync account on the device.

In OS X, the Exchange payload is designated by specifying `com.apple.ews.account` as the `PayloadType` value. This payload will configure an Exchange Web Services account for Contacts, Mail, Notes, Reminders, and Calendar.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
Available in both iOS and OS X		
`EmailAddress`	String	Specifies the full email address for the account. If not present in the payload, the device prompts for this string during profile installation. In OS X, this key is required.
`Host`	String	Specifies the Exchange server host name (or IP address). In OS X 10.11 and later, this key is optional.
`SSL`	Boolean	Optional. Default YES. Specifies whether the Exchange server uses SSL for authentication.

Key	Type	Value
`UserName`	String	This string specifies the user name for this Exchange account. If missing, the devices prompts for it during profile installation. In OS X, this key is required.
`Password`	String	Optional. The password of the account. Use only with encrypted profiles.
Available in iOS only		
`Certificate`	NSData blob	Optional. For accounts that allow authentication via certificate, a .p12 identity certificate in NSData blob format.
`CertificateName`	String	Optional. Specifies the name or description of the certificate.
`CertificatePassword`	data	Optional. The password necessary for the p12 identity certificate. Used with mandatory encryption of profiles.
`PreventMove`	Boolean	Optional. Default `false`. If set to `true`, messages may not be moved out of this email account into another account. Also prevents forwarding or replying from a different account than the message was originated from. **Availability:** Available in iOS 5.0 and later.
`PreventAppSheet`	Boolean	Optional. Default `false`. If set to `true`, this account will not be available for sending mail in any app other than the Apple Mail app. **Availability:** Available in iOS 5.0 and later.
`PayloadCertificate-UUID`	String	UUID of the certificate payload to use for the identity credential. If this field is present, the `Certificate` field is not used. **Availability:** Available in iOS 5.0 and later.
`SMIMEEnabled`	Boolean	Optional. Default `false`. If set to `true`, this account supports S/MIME. **Availability:** Available in iOS 5.0 and later.

Key	Type	Value
SMIMESigning-CertificateUUID	String	Optional. The PayloadUUID of the identity certificate used to sign messages sent from this account. **Availability:** Available in iOS 5.0 and later.
SMIMEEncryption-CertificateUUID	String	Optional. The PayloadUUID of the identity certificate used to decrypt messages sent to this account. **Availability:** Available in iOS 5.0 and later.
SMIMEEnablePer-MessageSwitch	Boolean	Optional. If set to true, enable the per-message signing and encryption switch. Defaults to true.
disableMailRecents-Syncing	Boolean	If true, this account is excluded from address Recents syncing. This defaults to false. **Availability:** Available only in iOS 6.0 and later.
MailNumberOfPast-DaysToSync	Integer	The number of days since synchronization.
Available in OS X Only		
Path	String	Optional.
Port	Number	Optional.
ExternalHost	String	Optional.
ExternalSSL	Boolean	Optional.
ExternalPath	String	Optional.
ExternalPort	Number	Optional.

Note: Note: As with VPN and Wi-Fi configurations, it is possible to associate an SCEP credential with an Exchange configuration via the PayloadCertificateUUID key.

FileVault 2

In OS X 10.9, you can use FileVault 2 to perform full XTS-AES 128 encryption on the contents of a volume. FileVault 2 payloads are designated by specifying com.apple.MCX.FileVault2 as the PayloadType value. Removal of the FileVault payload does not disable FileVault.

Key	Type	Value
`Enable`	String	Set to 'On' to enable FileVault. Set to 'Off' to disable FileVault. This value is required.
`Defer`	Boolean	Set to true to defer enabling FileVault until the designated user logs out. For details, see *fdesetup(8)* . The person enabling FileVault must be either a local user or a mobile account user.
`UserEntersMissingInfo`	Boolean	Set to true for manual profile installs to prompt for missing user name or password fields.
`UseRecoveryKey`	Boolean	Set to true to create a personal recovery key. Defaults to true.
`ShowRecoveryKey`	Boolean	Set to false to not display the personal recovery key to the user after FileVault is enabled. Defaults to true.
`OutputPath`	String	Path to the location where the recovery key and computer information plist will be stored.
`Certificate`	Data	DER-encoded certificate data if an institutional recovery key will be added.
`PayloadCertificateUUID`	String	UUID of the payload containing the asymmetric recovery key certificate payload.
`Username`	String	User name of the Open Directory user that will be added to FileVault.
`Password`	String	User password of the Open Directory user that will be added to FileVault. Use the `UserEntersMissingInfo` key if you want to prompt for this information.
`UseKeychain`	Boolean	If set to true and no certificate information is provided in this payload, the keychain already created at /Library/Keychains/FileVaultMaster.keychain will be used when the institutional recovery key is added.

Key	Type	Value
`DeferForceAtUserLoginMaxBypassAttempts`	Integer	When using the `Defer` option you can optionally set this key to the maximum number of times the user can bypass enabling FileVault before it will require that it be enabled before the user can log in. If set to 0, it will always prompt to enable FileVault until it is enabled, though it will allow you to bypass enabling it. Setting this key to –1 will disable this feature. **Availability:** Available in OS X 10.10 and later.
`DeferDontAskAtUserLogout`	Boolean	When using the `Defer` option, set this key to true to not request enabling FileVault at user logout time. **Availability:** Available in OS X 10.10 and later.

A personal recovery user will normally be created unless the `UseRecoveryKey` key value is false. An institutional recovery key will be created only if either there is certificate data available in the `Certificate` key value, a specific certificate payload is referenced, or the `UseKeychain` key value is set to true and a valid `FileVaultMaster.keychain` file was created. In all cases, the certificate information must be set up properly for FileVault or it will be ignored and no institutional recovery key will be set up.

FileVault Recovery Key Redirection Payload

FileVault full-volume encryption (FDE) recovery keys are, by default, sent to Apple if the user requests it. With this key, you can redirect those recovery keys to a corporate server. FileVault Recovery Key Redirection payloads are designated by specifying `com.apple.security.FDERecoveryRedirect` as the `PayloadType` value. Only one payload of this type is allowed per system.

A site providing support for archiving the recovery key must implement its own HTTPS server. The client issues a POST request to the server with XML data in the request body containing the recovery key and serial number of the client computer. The server must respond with XML data echoing the device's serial number and provide a RecordNumber, which can be any data that locates the recovery key.

The SSL certificate chain of the server is evaluated by the client, which must trust it. If needed, the configuration profile can include an additional certificate to set up a chain of trust.

Key	Type	Value
`RedirectURL`	String	The URL to which FDE recovery keys should be sent instead of Apple. Must begin with `https://`.

Key	Type	Value
EncryptCertPayload–UUID	String	The UUID of a payload within the same profile that contains a certificate whose public key is used to encrypt the recovery key when it is sent to the redirected URL. The referenced payload must be of type com.apple.security.pkcs1.

Once installed, this payload causes any FileVault recovery keys to be redirected to the specified URL instead of being sent to Apple. This will require sites to implement their own HTTPS server that will receive the recovery keys via a POST request.

This payload is valid only in system-scoped profiles (where PayloadScope is System). Installing more than one payload of this type per machine causes an error. The SSL certificate chain of the server is evaluated by the client, which must trust it. If needed, the configuration profile may contain another payload with the server's root certificate to be marked as trusted when the profile is installed.

FileVault Client Request

The client issues a HTTPS POST request to the server with XML data containing the following:

Key	Type	Value
VersionNumber	String	Currently set to '1.0'.
SerialNumber	String	The serial number of the client computer. The server must include this value in its response back to the client (see below).
RecoveryKeyCMS64	String	The recovery key encrypted using the encryption certificate provided in the configuration profile (referenced by the EncryptCertPayloadUUID key). The encrypted payload contains only the recovery key string without any XML wrapper. The encrypted data is wrapped in a CMS envelope and is then Base-64 encoded.

These tags are enclosed within a parent FDECaptureRequest tag. An example of an XML message body is:

```
<FDECaptureRequest>
<VersionNumber>1.0</VersionNumber>
<SerialNumber>A02FE08UCC8X</SerialNumber>
<RecoveryKeyCMS64>MIAGCSqGSIb3DQEHA ... AAAAAAAAA==</RecoveryKeyCMS64>
</FDECaptureRequest>
```

FileVault Server Response

Upon receiving the client's request, the server must respond to the client with XML data containing:

Key	Type	Value
`SerialNumber`	String	The serial number of the client computer. This value must be the same as the one sent in the request.
`RecordNumber`	Short string	This value must be nonempty but otherwise is up to the site to define it. This value will be displayed to the user along with the serial number on the EFI login screen when the user is asked to enter the recovery key. As an example, this could be a value to assist the site administrator in locating or verifying the user's recovery key in a database.

Font Payload

A Font payload lets you add an additional font to an iOS device. Font payloads are designated by specifying `com.apple.font` as the `PayloadType` value. You can include multiple Font payloads, as needed.

A Font payload contains the following keys:

Key	Type	Value
`Name`	String	Optional. The user-visible name for the font. This field is replaced by the actual name of the font after installation.
`Font`	Data	The contents of the font file.

Each payload must contain exactly one font file in TrueType (.ttf) or OpenType (.otf) format. Collection formats (.ttc or .otc) are not supported.

> **Important:** Fonts are identified by their embedded PostScript names. Two fonts with the same PostScript name are considered to be the same font even if their contents differ. Installing two different fonts with the same PostScript name is not supported, and the resulting behavior is undefined.

Global HTTP Proxy Payload

The Global HTTP Proxy payload is designated by specifying `com.apple.proxy.http.global` as the `PayloadType`.

This payload allows you to specify global HTTP proxy settings.

There can only be one of this payload at any time. This payload can only be installed on a supervised device.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`ProxyType`	String	If you choose manual proxy type, you need the proxy server address including its port and optionally a username and password into the proxy server. If you choose auto proxy type, you can enter a proxy autoconfiguration (PAC) URL.
`ProxyServer`	String	The proxy server's network address.
`ProxyPort`	Number	The proxy server's port
`ProxyUsername`	String	Optional. The username used to authenticate to the proxy server.
`ProxyPassword`	String	Optional. The password used to authenticate to the proxy server.
`ProxyPACURL`	String	Optional. The URL of the PAC file that defines the proxy configuration.
`ProxyPACFallback-Allowed`	Boolean	Optional. If `false`, prevents the device from connecting directly to the destination if the PAC file is unreachable. Default is `false`. **Availability:** Available in iOS 7 and later.
`ProxyCaptiveLogin-Allowed`	Boolean	Optional. If `true`, allows the device to bypass the proxy server to display the login page for captive networks. Default is `false`. **Availability:** Available in iOS 7 and later.

If the `ProxyType` field is set to `Auto` and no `ProxyPACURL` value is specified, the device uses the web proxy autodiscovery protocol (WPAD) to discover proxies.

Identification Payload

The Identification payload is designated by specifying `com.apple.configurationprofile.identification` value as the `PayloadType` value.

This payload allows you to save names of the account user and prompt text. If left blank, the user has to provide this information when he or she installs the profile.

The Identification payload is not supported in iOS.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`FullName`	String	The full name of the designated accounts.
`EmailAddress`	String	The address for the accounts.
`UserName`	String	The UNIX user name for the accounts.
`Password`	String	You can provide the password or choose to have the user provide it when he or she installs the profile.
`Prompt`	String	Custom instruction for the user, if needed.

Home Screen Layout Payload

The Home Screen Layout Payload is designated by specifying `com.apple.homescreenlayout` as the `PayloadType` value. It can contain only one payload, which must be supervised. It is supported on the User Channel.

This payload defines a layout of apps, folders, and web clips for the Home screen. It is supported on iOS 9.3 and later.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`Dock`	Array	Optional. An array of dictionaries, each of which must conform to the icon dictionary format.
`Pages`	Array	Required. An array of dictionaries, each of which must conform to the icon dictionary format.

Icon format dictionaries are defined as follows:

Key	Type	Value
`Type`	String	Required. Must be one of the following: • Application • Folder • WebClip
`DisplayName`	String	Optional. Human-readable string to be shown to the user.
`BundleID`	String	Required if App type. The bundle identifier of the app.
`Pages`	Array	Optional. Array of arrays of dictionaries. Each of the dictionaries complies to the icon dictionary format.
`URL`	String	Required if WebClip type. Provides the web address of the web clip to be shown.

LDAP Payload

The LDAP payload is designated by specifying `com.apple.ldap.account` as the `PayloadType` value.

An LDAP payload provides information about an LDAP server to use, including account information if required, and a set of LDAP search policies to use when querying that LDAP server.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`LDAPAccount-Description`	String	Optional. Description of the account.
`LDAPAccountHostName`	String	The host.
`LDAPAccountUseSSL`	Boolean	Whether or not to use SSL.
`LDAPAccountUserName`	String	Optional. The username.
`LDAPAccountPassword`	String	Optional. Use only with encrypted profiles.

Key	Type	Value
LDAPSearchSettings	Dictionary	Top level container object. Can have many of these for one account. Should have at least one for the account to be useful. Each LDAPSearchSettings object represents a node in the LDAP tree to start searching from, and tells what scope to search in (the node, the node plus one level of children, or the node plus all levels of children).
LDAPSearchSetting–Description	String	Optional. Description of this search setting.
LDAPSearchSetting–SearchBase	String	Conceptually, the path to the node where a search should start. For example: ou=people,o=example corp
LDAPSearchSetting–Scope	String	Defines what recursion to use in the search. Can be one of the following 3 values: LDAPSearchSettingScopeBase: Just the immediate node pointed to by SearchBase LDAPSearchSettingScopeOneLevel: The node plus its immediate children. LDAPSearchSettingScopeSubtree: The node plus all children, regardless of depth.

Network Usage Rules Payload

The Network Usage Rules payload is designated by specifying `com.apple.networkusagerules` as the `PayloadType` value.

Network Usage Rules allow enterprises to specify how managed apps use networks, such as cellular data networks. These rules only apply to managed apps.

In addition to the settings common to all payloads, this payload defines this key:

Key	Type	Value
ApplicationRules	Array of disctionaries	Required.

Each entry in the `ApplicationRules` array must be a dictionary containing these keys:

Key	Type	Value
AppIdentifierMatches	Array	Optional. A list of managed app identifiers, as strings, that must follow the associated rules. If this key is missing, the rules will apply to all managed apps on the device. Each string in the AppIdentifierMatches array may either be an exact app identifier match, e.g. com.mycompany.myapp, or it may specify a prefix match for the Bundle ID by using the * wildcard character. The wildcard character, if used, must appear after a period character (.), and may only appear once, at the end of the string, e.g. com.mycompany.*.
AllowRoamingCellular-Data	Boolean	Optional. Default true. If set to false, matching managed apps will not be allowed to use cellular data when roaming.
AllowCellularData	Boolean	Optional. Default true. If set to false, matching managed apps will not be allowed to use cellular data at any time.

Notifications Payload

The Notifications Payload is designated by specifying com.apple.notificationsettings as the PayloadType value. It can contain only one payload, which need not be supervised. It is supported on the User Channel.

This payload specifies the restriction enforced notification settings for apps, using their bundle identifiers. It is supported on iOS 9.3 and later.

In addition to the settings common to all payloads, this payload defines the following key:

Key	Type	Value
NotificationSettings	Array	Required. An array of dictionaries, each of which specifies notification settings for one bundle identifier.

Each entry in the NotificationSettings field contains the following dictionary:

Key	Type	Value
BundleIdentifier	String	Required. Bundle identifier of app to which to apply these notification settings.
NotificationsEnabled	Boolean	Optional. Whether notifications are allowed for this app. Default is true.

Key	Type	Value
`ShowInNotification-Center`	Boolean	Optional. Whether notifications can be shown in notification center. Default is true.
`ShowInLockScreen`	Boolean	Optional. Whether notifications can be shown in the lock screen. Default is true.
`AlertType`	Integer	Optional. The type of alert for notifications for this app: • 0: None • 1: Banner • 2: Modal Alert Default is 1.
`BadgesEnabled`	Boolean	Optional. Whether badges are allowed for this app. Default is true.
`SoundsEnabled`	Boolean	Optional. Whether sounds are allowed for this app. Default is true.

Passcode Policy Payload

The Passcode Policy payload is designated by specifying `com.apple.mobiledevice.passwordpolicy` as the `PayloadType` value.

The presence of this payload type prompts an iOS or OS X device to present the user with an alphanumeric passcode entry mechanism, which allows the entry of arbitrarily long and complex passcodes.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`allowSimple`	Boolean	Optional. Default `true`. Determines whether a simple passcode is allowed. A simple passcode is defined as containing repeated characters, or increasing/decreasing characters (such as 123 or CBA). Setting this value to `false` is synonymous to setting minComplexChars to "1".
`forcePIN`	Boolean	Optional. Default NO. Determines whether the user is forced to set a PIN. Simply setting this value (and not others) forces the user to enter a passcode, without imposing a length or quality.

Key	Type	Value
`maxFailedAttempts`	Number	Optional. Default 10 (iOS only). Allowed range [1...10]. Specifies the number of allowed failed attempts to enter the passcode at the device's lock screen. Once this number is exceeded, the device is locked and must be connected to its designated iTunes in order to be unlocked.
`maxInactivity`	Number	Optional. Default Infinity. Specifies the number of minutes for which the device can be idle (without being unlocked by the user) before it gets locked by the system. Once this limit is reached, the device is locked and the passcode must be entered. In OS X, this will be translated to screensaver settings.
`maxPINAgeInDays`	Number	Optional. Default Infinity. Specifies the number of days for which the passcode can remain unchanged. After this number of days, the user is forced to change the passcode before the device is unlocked.
`minComplexChars`	Number	Optional. Default 0. Specifies the minimum number of complex characters that a passcode must contain. A "complex" character is a character other than a number or a letter, such as &%$#.
`minLength`	Number	Optional. Default 0. Specifies the minimum overall length of the passcode. This parameter is independent of the also optional minComplexChars argument.
`requireAlphanumeric`	Boolean	Optional. Default NO. Specifies whether the user must enter alphabetic characters ("abcd"), or if numbers are sufficient.
`pinHistory`	Number	Optional. When the user changes the passcode, it has to be unique within the last N entries in the history. Minimum value is 1, maximum value is 50.
`maxGracePeriod`	Number	Optional. The maximum grace period, in minutes, to unlock the phone without entering a passcode. Default is 0, that is no grace period, which requires a passcode immediately. In OS X, this will be translated to screensaver settings.
`allowFingerprint-Modification`	Boolean	Optional. Supervised only. Not supported on OS X. Allows the user to modify Touch ID. Default NO.

Profile Removal Password Payload

The Removal Password payload is designated by specifying `com.apple.profileRemovalPassword` value as the `PayloadType` value.

A password removal policy payload provides a password to allow users to remove a locked configuration profile from the device. If this payload is present and has a password value set, the device asks for the password when the user taps a profile's Remove button. This payload is encrypted with the rest of the profile.

Key	Type	Value
`RemovalPassword`	String	Optional. Supervised only. Specifies the removal password for the profile.

Restrictions Payload

The Restrictions payload is designated by specifying `com.apple.applicationaccess` as the `PayloadType` value.

A Restrictions payload allows the administrator to restrict the user from doing certain things with the device, such as using the camera.

> **Note:** You can specify additional restrictions, including maximum allowed content ratings, by creating a profile using Apple Configurator 2 or Profile Manager.

The Restrictions payload is supported in iOS; some keys are also supported in OS X, as noted below.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`allowAccount-Modification`	Boolean	Optional. Supervised only. If set to `false`, account modification is disabled. **Availability:** Available only in iOS 7.0 and later.
`allowAddingGame-CenterFriends`	Boolean	Optional. When `false`, prohibits adding friends to Game Center. This key is deprecated on unsupervised devices.
`allowAirDrop`	Boolean	Optional. Supervised only. If set to `false`, AirDrop is disabled. **Availability:** Available only in iOS 7.0 and later.

Key	Type	Value
`allowAppCellularData-Modification`	Boolean	Optional. Supervised only. If set to `false`, changes to cellular data usage for apps are disabled. **Availability:** Available only in iOS 7.0 and later.
`allowAppInstallation`	Boolean	Optional. Supervised only. When `false`, the App Store is disabled and its icon is removed from the Home screen. Users are unable to install or update their applications. This key is deprecated on unsupervised devices.
`allowAppRemoval`	Boolean	Optional. When `false`, disables removal of apps from iOS device. This key is deprecated on unsupervised devices.
`allowAssistant`	Boolean	Optional. When `false`, disables Siri. Defaults to `true`.
`allowAssistantUser-GeneratedContent`	Boolean	Optional. Supervised only. When `false`, prevents Siri from querying user-generated content from the web. **Availability:** Available in iOS 7 and later.
`allowAssistant-WhileLocked`	Boolean	Optional. When `false`, the user is unable to use Siri when the device is locked. Defaults to `true`. This restriction is ignored if the device does not have a passcode set. **Availability:** Available only in iOS 5.1 and later.
`allowBookstore`	Boolean	Optional. Supervised only. If set to `false`, iBookstore will be disabled. This will default to `true`. **Availability:** Available in iOS 6.0 and later.
`allowBookstore-Erotica`	Boolean	Optional. Supervised only prior to iOS 6.1. If set to `false`, the user will not be able to download media from the iBookstore that has been tagged as erotica. This will default to `true`. **Availability:** Available in iOS 6.0 and later.
`allowCamera`	Boolean	Optional. When `false`, the camera is completely disabled and its icon is removed from the Home screen. Users are unable to take photographs. **Availability:** Available in iOS and in OS X 10.11 and later.
`allowChat`	Boolean	Optional. When `false`, disables the use of the Messages app with supervised devices. **Availability:** Available in iOS 6.0 and later.

Key	Type	Value
allowCloudBackup	Boolean	Optional. When false, disables backing up the device to iCloud. **Availability:** Available in iOS 5.0 and later.
allowCloudDocument-Sync	Boolean	Optional. When false, disables document and key-value syncing to iCloud. This key is deprecated on unsupervised devices. **Availability:** Available in iOS 5.0 and later and in OS X 10.11 and later.
allowCloudKeychain-Sync	Boolean	Optional. If false, disables Cloud keychain synchronization. Default is true. **Availability:** Available only in iOS 7.0 and later.
allowDiagnostic-Submission	Boolean	Optional. When false, this prevents the device from automatically submitting diagnostic reports to Apple. Defaults to true. **Availability:** Available only in iOS 6.0 and later.
allowExplicitContent	Boolean	Optional. When false, explicit music or video content purchased from the iTunes Store is hidden. Explicit content is marked as such by content providers, such as record labels, when sold through the iTunes Store. This key is deprecated on unsupervised devices.
allowFindMyFriends-Modification	Boolean	Optional. Supervised only. If set to false, changes to Find My Friends are disabled. **Availability:** Available only in iOS 7.0 and later.
allowFingerprint-ForUnlock	Boolean	Optional. If false, prevents Touch ID from unlocking a device. **Availability:** Available in iOS 7 and later.
allowGameCenter	Boolean	Optional. Supervised only. When false, Game Center is disabled and its icon is removed from the Home screen. Default is true. **Availability:** Available only in iOS 6.0 and later.
allowGlobal-BackgroundFetchWhen-Roaming	Boolean	Optional. When false, disables global background fetch activity when an iOS phone is roaming.

Key	Type	Value
`allowInAppPurchases`	Boolean	Optional. When `false`, prohibits in-app purchasing.
`allowLockScreen-ControlCenter`	Boolean	Optional. If `false`, prevents Control Center from appearing on the Lock screen. **Availability:** Available in iOS 7 and later.
`allowHostPairing`	Boolean	Supervised only. If set to `false`, host pairing is disabled with the exception of the supervision host. If no supervision host certificate has been configured, all pairing is disabled. Host pairing lets the administrator control which devices an iOS 7 device can pair with. **Availability:** Available only in iOS 7.0 and later.
`allowLockScreen-NotificationsView`	Boolean	Optional. If set to `false`, the Notifications view in Notification Center on the lock screen is disabled. **Availability:** Available only in iOS 7.0 and later.
`allowLockScreen-TodayView`	Boolean	Optional. If set to `false`, the Today view in Notification Center on the lock screen is disabled. **Availability:** Available only in iOS 7.0 and later.
`allowMultiplayer-Gaming`	Boolean	Optional. When `false`, prohibits multiplayer gaming. This key is deprecated on unsupervised devices.
`allowOpenFromManaged-ToUnmanaged`	Boolean	Optional. If `false`, documents in managed apps and accounts only open in other managed apps and accounts. Default is `true`. **Availability:** Available only in iOS 7.0 and later.
`allowOpenFrom-UnmanagedToManaged`	Boolean	Optional. If set to `false`, documents in unmanaged apps and accounts will only open in other unmanaged apps and accounts. Default is `true`. **Availability:** Available only in iOS 7.0 and later.
`allowOTAPKIUpdates`	Boolean	Optional. If `false`, over-the-air PKI updates are disabled. Setting this restriction to false does not disable CRL and OCSP checks. Default is `true`. **Availability:** Available only in iOS 7.0 and later.
`allowPassbook-WhileLocked`	Boolean	Optional. If set to `false`, Passbook notifications will not be shown on the lock screen.This will default to `true`. **Availability:** Available in iOS 6.0 and later.

Key	Type	Value
`allowPhotoStream`	Boolean	Optional. When `false`, disables Photo Stream. **Availability:** Available in iOS 5.0 and later.
`allowSafari`	Boolean	Optional. When `false`, the Safari web browser application is disabled and its icon removed from the Home screen. This also prevents users from opening web clips. This key is deprecated on unsupervised devices.
`safariAllowAutoFill`	Boolean	Optional. When `false`, Safari auto-fill is disabled. Defaults to true.
`safariForceFraud-Warning`	Boolean	Optional. When `true`, Safari fraud warning is enabled. Defaults to false.
`safariAllow-JavaScript`	Boolean	Optional. When `false`, Safari will not execute JavaScript. Defaults to true.
`safariAllowPopups`	Boolean	Optional. When `false`, Safari will not allow pop-up tabs. Defaults to true.
`safariAcceptCookies`	Integer	Optional. Determines conditions under which the device will accept cookies. Following are allowed values: • 0: Never • 1: From visited sites only • 1.5: From websites I visit (enter `'<real>1.5</real>'`) • 2: Always Defaults to 2.
`allowSharedStream`	Boolean	Optional. If set to `false`, Shared Photo Stream will be disabled. This will default to `true`. **Availability:** Available in iOS 6.0 and later.
`allowUIConfiguration-ProfileInstallation`	Boolean	Optional. Supervised only. If set to `false`, the user is prohibited from installing configuration profiles and certificates interactively. This will default to `true`. **Availability:** Available in iOS 6.0 and later.

Key	Type	Value
allowUntrusted-TLSPrompt	Boolean	Optional. When false, automatically rejects untrusted HTTPS certificates without prompting the user. **Availability:** Available in iOS 5.0 and later.
allowVideo-Conferencing	Boolean	Optional. When false, disables video conferencing. This key is deprecated on unsupervised devices.
allowVoiceDialing	Boolean	Optional. When false, disables voice dialing if the device is locked with a passcode. Default is true.
allowYouTube	Boolean	Optional. When false, the YouTube application is disabled and its icon is removed from the Home screen. This key is ignored in iOS 6 and later because the YouTube app is not provided.
allowiTunes	Boolean	Optional. When false, the iTunes Music Store is disabled and its icon is removed from the Home screen. Users cannot preview, purchase, or download content. This key is deprecated on unsupervised devices.
autonomousSingleApp-ModePermittedAppIDs	Array of strings	Optional. Supervised only. If present, allows apps identified by the bundle IDs listed in the array to autonomously enter Single App Mode. **Availability:** Available only in iOS 7.0 and later.
forceAssistant-ProfanityFilter	Boolean	Optional. Supervised only. When true, forces the use of the profanity filter assistant.
forceEncryptedBackup	Boolean	Optional. When true, encrypts all backups.
forceITunesStore-PasswordEntry	Boolean	Optional. When true, forces user to enter their iTunes password for each transaction. **Availability:** Available in iOS 5.0 and later.
forceLimitAdTracking	Boolean	Optional. If true, limits ad tracking. Default is false. **Availability:** Available only in iOS 7.0 and later.
forceAirPlayOutgoing-RequestsPairing-Password	Boolean	Optional. If set to true, forces all devices receiving AirPlay requests from this device to use a pairing password. Default is false. **Availability:** Available only in iOS 7.1 and later.

Key	Type	Value
`forceAirPlayIncoming-RequestsPairing-Password`	Boolean	Optional. If set to `true`, forces all devices sending AirPlay requests to this device to use a pairing password. Default is `false`. **Availability:** Available only in Apple TV 6.1 and later.
`allowManagedApps-CloudSync`	Boolean	Optional. If set to `false`, prevents managed applications from using cloud sync.
`allowEraseContent-AndSettings`	Boolean	Supervised only. If set to `false`, disables the “Erase All Content And Settings” option in the Reset UI.
`allowSpotlight-InternetResults`	Boolean	Supervised only. If set to `false`, Spotlight will not return Internet search results. **Availability:** Available in iOS and in OS X 10.11 and later.
`allowEnabling-Restrictions`	Boolean	Supervised only. If set to `false`, disables the "Enable Restrictions" option in the Restrictions UI in Settings.
`allowActivity-Continuation`	Boolean	If set to `false`, Activity Continuation will be disabled. Defaults to `true`.
`allowEnterprise-BookBackup`	Boolean	If set to `false`, Enterprise books will not be backed up. Defaults to `true`.
`allowEnterpriseBook-MetadataSync`	Boolean	If set to `false`, Enterprise books notes and highlights will not be synced. Defaults to `true`.
`allowPodcasts`	Boolean	Supervised only. If set to `false`, disables podcasts. Defaults to `true`. **Availability:** Available in iOS 8.0 and later.
`allowDefinition-Lookup`	Boolean	Supervised only. If set to `false`, disables definition lookup. Defaults to `true`. **Availability:** Available in iOS 8.1.3 and later and in OS X 10.11.2 and later.
`allowPredictive-Keyboard`	Boolean	Supervised only. If set to `false`, disables predictive keyboards. Defaults to `true`. **Availability:** Available in iOS 8.1.3 and later.

Key	Type	Value
`allowAutoCorrection`	Boolean	Supervised only. If set to `false`, disables keyboard auto-correction. Defaults to `true`. **Availability:** Available in iOS 8.1.3 and later.
`allowSpellCheck`	Boolean	Supervised only. If set to `false`, disables keyboard spell-check. Defaults to `true`. **Availability:** Available in iOS 8.1.3 and later.
`forceWatchWrist-Detection`	Boolean	If set to `true`, a paired Apple Watch will be forced to use Wrist Detection. Defaults to `false`. **Availability:** Available in iOS 8.2 and later.
`allowMusicService`	Boolean	Supervised only. If set to `false`, Music service is disabled and Music app reverts to classic mode. Defaults to `true`. **Availability:** Available in iOS 9.0 and later.
`allowCloudPhoto-Library`	Boolean	If set to `false`, disables iCloud Photo Library. Any photos not fully downloaded from iCloud Photo Library to the device will be removed from local storage. **Availability:** Available in iOS 9.0 and later.
`allowNews`	Boolean	Supervised only. If set to `false`, disables News. Defaults to `true`. **Availability:** Available in iOS 9.0 and later.
`forceAirDrop-Unmanaged`	Boolean	Optional. If set to `true`, causes AirDrop to be considered an unmanaged drop target. Defaults to `false`. **Availability:** Available in iOS 9.0 and later.
`allowUIApp-Installation`	Boolean	Supervised only. When `false`, the App Store is disabled and its icon is removed from the Home screen. However, users may continue to use Host apps (iTunes, Configurator) to install or update their apps. Defaults to `true`. **Availability:** Available in iOS 9.0 and later.
`allowScreenShot`	Boolean	Optional. If set to `false`, users can't save a screenshot of the display and are prevented from capturing a screen recording as well. Defaults to `true`. **Availability:** Updated in iOS 9.0 to include screen recordings.

Key	Type	Value
`allowKeyboard-Shortcuts`	Boolean	Supervised only. If set to `false`, keyboard shortcuts cannot be used. Defaults to `true`. **Availability:** Available in iOS 9.0 and later.
`allowPairedWatch`	Boolean	Supervised only. If set to `false`, disables pairing with an Apple Watch. Any currently paired Apple Watch is unpaired and erased. Defaults to `true`. **Availability:** Available in iOS 9.0 and later.
`allowPasscode-Modification`	Boolean	Supervised only. If set to `false`, prevents device passcode from being added, changed, or removed. Defaults to `true`. **Availability:** Available in iOS 9.0 and later.
`allowDeviceName-Modification`	Boolean	Supervised only. If set to `false`, prevents device name from being changed. Defaults to `true`. **Availability:** Available in iOS 9.0 and later.
`allowWallpaper-Modification`	Boolean	Supervised only. If set to `false`, prevents wallpaper from being changed. Defaults to `true`. **Availability:** Available in iOS 9.0 and later.
`allowAutomatic-AppDownloads`	Boolean	Supervised only. If set to `false`, prevents automatic downloading of apps purchased on other devices. Does not affect updates to existing apps. Defaults to `true`. **Availability:** Available in iOS 9.0 and later.
`allowEnterprise-AppTrust`	Boolean	If set to `false`, prevents trusting enterprise apps. Defaults to `true`. **Availability:** Available in iOS 9.0 and later.
`allowEnterpriseApp-TrustModification`	Boolean	Supervised only. If set to `false`, prevents the enterprise app trust settings from being changed. Defaults to `true`. **Availability:** Available in iOS 9.0 and later.
`allowRadioService`	Boolean	Supervised only. If set to `false`, iTunes Radio is disabled. Defaults to `true`. **Availability:** Available in iOS 9.3 and later.

Key	Type	Value
`blacklistedApp-BundleIDs`	Array of strings	Supervised only. If present, prevents bundle IDs listed in the array from being shown or launchable. **Availability:** Available in iOS 9.3 and later.
`whitelistedApp-BundleIDs`	Array of strings	Supervised only. If present, allows only bundle IDs listed in the array from being shown or launchable. **Availability:** Available in iOS 9.3 and later.
`allowNotifications-Modification`	Boolean	Supervised only. If set to `false`, notification settings cannot be modified. Defaults to `true`. **Availability:** Available in iOS 9.3 and later.

SCEP Payload

The SCEP (Simple Certificate Enrollment Protocol) payload is designated by specifying `com.apple.security.scep` as the `PayloadType` value.

An SCEP payload automates the request of a client certificate from an SCEP server, as described in *Over-the-Air Profile Delivery and Configuration* .

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`URL`	String	The SCEP URL. See *Over-the-Air Profile Delivery and Configuration* for more information about SCEP.
`Name`	String	Optional. Any string that is understood by the SCEP server. For example, it could be a domain name like `example.org`. If a certificate authority has multiple CA certificates this field can be used to distinguish which is required.
`Subject`	Array	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 would translate to: `[ [ ["C", "US"] ], [ ["O", "Apple Inc."] ], ..., [ [ "1.2.5.3", "bar" ] ] ]` OIDs can be represented as dotted numbers, with shortcuts for country (`C`), locality (`L`), state (`ST`), organization (`O`), organizational unit (`OU`), and common name (`CN`).
`Challenge`	String	Optional. A pre-shared secret.

Key	Type	Value
`Keysize`	Number	Optional. The key size in bits, either 1024 or 2048.
`Key Type`	String	Optional. Currently always "RSA".
`Key Usage`	Number	Optional. A bitmask indicating the use of the key. 1 is signing, 4 is encryption, 5 is both signing and encryption. Some certificate authorities, such as Windows CA, support only encryption or signing, but not both at the same time. **Availability:** Available only in iOS 4 and later.
`Retries`	Integer	Optional. The number of times the device should retry if the server sends a PENDING response. Defaults to 3.
`RetryDelay`	Integer	Optional. The number of seconds to wait between subsequent retries. The first retry is attempted without this delay. Defaults to 10.

SubjectAltName Dictionary Keys

The SCEP payload can specify an optional `SubjectAltName` dictionary that provides values required by the CA for issuing a certificate. You can specify a single string or an array of strings for each key.

The values you specify depend on the CA you're using, but might include DNS name, URL, or email values. For an example, see Sample Configuration Profile (page 83) or read *Over-the-Air Profile Delivery and Configuration* .

GetCACaps Dictionary Keys

If you add a dictionary with the key `GetCACaps`, the device uses the strings you provide as the authoritative source of information about the capabilities of your CA. Otherwise, the device queries the CA for `GetCACaps` and uses the answer it gets in response.If the CA doesn't respond, the device defaults to `GET 3DES` and `SHA-1` requests. For more information, read *Over-the-Air Profile Delivery and Configuration* . This feature is not supported in OS X.

Shared Device Configuration Payload

The Shared Device Configuration Payload is designated by specifying `com.apple.shareddeviceconfiguration` as the `PayloadType` value. It can contain only one payload, which must be supervised. It is not supported on the User Channel.

The Shared Device Configuration Payload allows admins to specify optional text displayed on the login window and lock screen (i.e. a "If Lost, Return To" message and Asset Tag Information). It is supported on iOS 9.3 and later.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`IfLostReturnToMessage`	String	Optional. An "If Lost, Return To" message, displayed on the login window and lock screen.
`AssetTagInformation`	String	Optional. Asset tag information for the device, displayed on the login window and lock screen.

Single Sign-On Account Payload

The Single Sign-On Account payload is designated by specifying `com.apple.sso` as the `PayloadType`.

This payload is supported only in iOS 7.0 and later.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`Name`	String	Human-readable name for the account.
`Kerberos`	Dictionary	Kerberos-related information, described below.

The Kerberos `dictionary` can contain the following keys:

Key	Type	Value
`PrincipalName`	String	Optional. The Kerberos principal name. If not provided, the user is prompted for one during profile installation. This field must be provided for MDM installation.
`PayloadCertificate-UUID`	String	Optional. The PayloadUUID of an identity certificate payload that can be used to renew the Kerberos credential without user interaction. The certificate payload must have either the `com.apple.security.pkcs12` or `com.apple.security.scep` payload type. Both the Single Sign On payload and the identity certificate payload must be included in the same configuration profile
`Realm`	String	The Kerberos realm name. This value should be properly capitalized.

Key	Type	Value
URLPrefixMatches	Array of strings	List of URLs prefixes that must be matched to use this account for Kerberos authentication over HTTP. **Note** that the URL postfixes must match as well.
AppIdentifierMatches	Array of strings	Optional. List of app identifiers that are allowed to use this login. If this field missing, this login matches all app identifiers. This array, if present, may not be empty.

Each entry in the URLPrefixMatches array must contain a URL prefix. Only URLs that begin with one of the strings in this account are allowed to access the Kerberos ticket. URL matching patterns must include the scheme—for example, http://www.example.com/. If a matching pattern does not end in /, a / is appended to it.

The URL matching patterns must begin with either http:// or https://. A simple string match is performed, so the URL prefix http://www.example.com/ does not match http://www.example.com:80/. With iOS 9.0 or later, however, a single wildcard * may be used to specify missing subdomains. For example, http://*.example.com/ will match http://store.example.com/.

The patterns http://.com and https://.com match all HTTP and HTTPS URLs in the .com top-level domain (TLD). The same is true for other TLDs, which must be listed individually.

The AppIdentifierMatches array must contain strings that match app bundle IDs. These strings may be exact matches (com.mycompany.myapp, for example) or may specify a prefix match on the bundle ID by using the * wildcard character. The wildcard character must appear after a period character (.), and may appear only at the end of the string (com.mycompany.*, for example). When a wildcard is given, any app whose bundle ID begins with the prefix is granted access to the account.

System Policy Control Payload

The System Policy Control payload is designated by specifying com.apple.systempolicy.control as the PayloadType.

This payload allows control over configuring the “Allowed applications downloaded from:” option in the “General” tab of “Security & Privacy” in System Preferences.

This payload must only exist in a device profile. If the payload is present in a user profile, an error will be generated during installation and the profile will fail to install.

This payload is supported only on OS X v10.8 and later.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`EnableAssessment`	Boolean	Optional. If the key is present and has a value of `YES`, Gatekeeper is enabled. If the key is present and has a value of `NO`, Gatekeeper is disabled.
`AllowIdentified-Developers`	Boolean	Optional. If the key is present and has a value of `YES`, Gatekeeper's "Mac App Store and identified developers" option is chosen. If the key is present and has a value of `NO`, Gatekeeper's "Mac App Store" option is chosen. If `EnableAssessment` is not `true`, this key has no effect.

System Policy Rule Payload

The System Policy Rule payload is designated by specifying `com.apple.systempolicy.rule` as the `PayloadType`. This is one of three payloads that allows control of various GateKeeper settings.

This payload allows control over Gatekeeper's system policy rules. The keys and functionality are tightly related to the `spctl` command line tool. You should be read the manual page for `spctl`.

This payload must only exist in a device profile. If the payload is present in a user profile, an error will be generated during installation and the profile will fail to install.

This payload is supported only on OS X v10.8 and later.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`Requirement`	String	The policy requirement. This key must follow the syntax described in Code Signing Requirement Language.
`Comment`	String	Optional. This string will appear in the System Policy UI. If it is missing, "PayloadDisplayName" or "PayloadDescription" will be put into this field before the rule is added to the System Policy database.
`Expiration`	Date	Optional. An expiration date for rule(s) being processed.
`OperationType`	String	Optional. One of `operation:execute`, `operation:install`, or `operation:lsopen`. This will default to `operation:execute`.

The client has no way to display information about what certificate is being accepted by the signing requirement if the requirement keys is specified as:

```
certificate leaf = H"7696f2cbf7f7d43fceb879f52f3cdc8fadfccbd4"
```

You can embed the certificate within the payload itself, allowing the Profiles preference pane and System Profile report to display information about the certificate(s) being used. To do so, specify the `Requirement` key using a payload variable of the form $HASHCERT_xx$ where "xx" is the name of an additional key within the same payload that contains the certificate data in DER format.

For example, if you specify:

```
<key>Requirement</key>
<string>certificate leaf = $HASHCERT_Cert1Data$</string>
```

and then provide:

```
<key>Cert1Data</key>
<data>
MIIFTDCCBDSgAwIBAgIHBHXzxGzq8DANBgkqhkiG9w0BAQUFADCByjELMAkGA1UEBhMC
...
z1I6yBET5qaGhpWexEp3baLbXLcrtgufmDSUtUnImavGyw==
</data>
```

The client will get the value of `Cert1Data` key, perform a SHA1 hash on it and use the resulting requirement string of:

```
certificate leaf = H"7696f2cbf7f7d43fceb879f52f3cdc8fadfccbd4"
```

If you want, you may reference multiple $HASHCERT_xx$ within the requirement string.

System Policy Managed Payload

The System Policy Managed payload is designated by specifying `com.apple.systempolicy.managed` as the `PayloadType`. This is one of three payloads that allows control of various GateKeeper settings.

This payload allows control to disable the Finder's contextual menu that allows bypass of System Policy restrictions.

This payload is supported only on OS X v10.8 and later.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
DisableOverride	Boolean	Optional. If YES, the Finder's contextual menu item will be disabled.

VPN Payload

The VPN payload is used for traditional systemwide VPNs based on L2TP, PPTP, and IPSec. This payload should not be confused with the Per-App VPN, described in Per-App VPN Payload (page 13).

The VPN payload is designated by specifying `com.apple.vpn.managed` as the `PayloadType` value. In addition to the settings common to all payload types, the VPN payload defines the following keys:

Key	Type	Value
UserDefinedName	String	Description of the VPN connection displayed on the device.
OverridePrimary	Boolean	Specifies whether to send all traffic through the VPN interface. If `true`, all network traffic is sent over VPN. Defaults to `false`.
VPNType	String	Determines the settings available in the payload for this type of VPN connection. It can have one of the following values: • L2TP • PPTP • IPSec (Cisco) • IKEv2 (see IKEv2 Dictionary Keys (page 63)) • AlwaysOn (see AlwaysOn VPN Keys (page 69)) • VPN (solution uses a VPN plugin or NetworkExtension, so the VPNSubType key is required (see below)).

Key	Type	Value
`VPNSubType`	String	Optional. If `VPNType` is `VPN`, this field is required. If the configuration is targeted at a VPN solution that uses a VPN plugin, then this field contains the bundle identifier of the plugin. Here are some examples: • Cisco AnyConnect: `com.cisco.anyconnect.applevpn.plugin` • Juniper SSL: `net.juniper.sslvpn` • F5 SSL: `com.f5.F5-Edge-Client.vpnplugin` • SonicWALL Mobile Connect: `com.sonicwall.SonicWALL-SSLVPN.vpnplugin` • Aruba VIA: `com.arubanetworks.aruba-via.vpnplugin` If the configuration is targeted at a VPN solution that uses a `NetworkExtension` provider, then this field contains the bundle identifier of the app that contains the provider. Contact the VPN solution vendor for the value of the identifier. If `VPNType` is `IKEv2`, then the `VPNSubType` field is optional and is reserved for future use. If it is specified, it must contain the empty string.
`ProviderBundle-Identifier`	String	Optional. If the `VPNSubType` field contains the bundle identifier of an app that contains multiple VPN providers of the same type (`app-proxy` or `packet-tunnel`), then this field is used to specify which provider to use for this configuration.
If `VPNType` is `VPN`, `IPSec`, or `IKEv2`, the following keys may be defined in the corresponding `VPN`, `IPSec`, or `IKEv2` dictionaries to configure VPN On Demand:		
`OnDemandEnabled`	Integer	`1` if the VPN connection should be brought up on demand, else `0`.

Key	Type	Value
`OnDemandMatchDomains-Always`	Array of Strings	*Deprecated.* A list of domain names. In versions of iOS prior to iOS 7, if the hostname ends with one of these domain names, the VPN is started automatically. In iOS 7 and later, if this key is present, the associated domain names are treated as though they were associated with the `OnDemandMatchDomainsOnRetry` key. This behavior can be overridden by `OnDemandRules`.
`OnDemandMatch-DomainsNever`	Array of Strings	*Deprecated.* A list of domain names. If the hostname ends with one of these domain names, the VPN is *not* started automatically. This might be used to exclude a subdomain within an included domain. This behavior can be overridden by `OnDemandRules`. In iOS 7 and later, this key is deprecated (but still supported) in favor of `EvaluateConnection` actions in the `OnDemandRules` dictionaries.
`OnDemandMatchDomains-OnRetry`	Array of Strings	*Deprecated.* A list of domain names. If the hostname ends with one of these domain names, if a DNS query for that domain name fails, the VPN is started automatically. This behavior can be overridden by `OnDemandRules`. In iOS 7 and later, this key is deprecated (but still supported) in favor of `EvaluateConnection` actions in the `OnDemandRules` dictionaries.
`OnDemandRules`	Array of Dictionaries	Determines when and how an on-demand VPN should be used. See On Demand Rules Dictionary Keys (page 59) for details.
If `VPNType` is not `AlwaysOn`, the following key may be defined:		
`VendorConfig`	Dictionary	A dictionary for configuration information specific to a given third-party VPN solution.

There are two possible dictionaries present at the top level, under the keys "PPP" and "IPSec". The keys inside these two dictionaries are described below, along with the VPNType value under which the keys are used.

PPP Dictionary Keys

The following elements are for VPN payloads of type PPP.

Key	Type	Value
`AuthName`	String	The VPN account user name. Used for L2TP and PPTP.
`AuthPassword`	String	Optional. Only visible if TokenCard is `false`. Used for L2TP and PPTP.
`TokenCard`	Boolean	Whether to use a token card such as an RSA SecurID card for connecting. Used for L2TP.
`CommRemoteAddress`	String	IP address or host name of VPN server. Used for L2TP and PPTP.
`AuthEAPPlugins`	Array	Only present if RSA SecurID is being used, in which case it has one entry, a string with value "EAP-RSA". Used for L2TP and PPTP.
`AuthProtocol`	Array	Only present if RSA SecurID is being used, in which case it has one entry, a string with value "EAP". Used for L2TP and PPTP.
`CCPMPPE40Enabled`	Boolean	See discussion under `CCPEnabled`. Used for PPTP.
`CCPMPPE128Enabled`	Boolean	See discussion under `CCPEnabled`. Used for PPTP.
`CCPEnabled`	Boolean	Enables encryption on the connection. If this key and `CCPMPPE40Enabled` are `true`, represents automatic encryption level; if this key and `CCPMPPE128Enabled` are `true`, represents maximum encryption level. If no encryption is used, then none of the CCP keys are `true`. Used for PPTP.

IPSec Dictionary Keys

The following elements are for VPN payloads of type IPSec.

Key	Type	Value
`RemoteAddress`	String	IP address or host name of the VPN server. Used for Cisco IPSec.
`AuthenticationMethod`	String	Either `SharedSecret` or `Certificate`. Used for L2TP and Cisco IPSec.
`XAuthEnabled`	Integer	`1` if Xauth is on, `0` if it is off. Used for Cisco IPSec.

Key	Type	Value
XAuthName	String	User name for VPN account. Used for Cisco IPSec.
XAuthPassword	String	Required for VPN account user authentication. Used for Cisco IPSec.
LocalIdentifier	String	Present only if AuthenticationMethod is SharedSecret. The name of the group to use. If Hybrid Authentication is used, the string must end with [hybrid]. Used for Cisco IPSec.
LocalIdentifierType	String	Present only if AuthenticationMethod is SharedSecret. The value is KeyID. Used for L2TP and Cisco IPSec.
SharedSecret	Data	The shared secret for this VPN account. Only present if AuthenticationMethod is SharedSecret. Used for L2TP and Cisco IPSec.
PayloadCertificate-UUID	String	The UUID of the certificate to use for the account credentials. Only present if AuthenticationMethod is Certificate. Used for Cisco IPSec.
PromptForVPNPIN	Boolean	Tells whether to prompt for a PIN when connecting. Used for Cisco IPSec.

On Demand Rules Dictionary Keys

The OnDemandRules key in a VPN payload is associated with an array of dictionaries that define the network match criteria that identify a particular network location.

In typical use, VPN On Demand matches the dictionaries in the OnDemandRules array against properties of your current network connection to determine whether domain-based rules should be used in determining whether to connect, then handles the connection as follows:

- If domain-based matching is enabled for a matching OnDemandRules dictionary, then for each dictionary in that dictionary's EvaluateConnection array, VPN On Demand compares the requested domain against the domains listed in the Domains array.
- If domain-based matching is not enabled, the specified behavior (usually Connect, Disconnect, or Ignore) is used if the dictionary otherwise matches.

> **Note:** For backwards compatibility, VPN On Demand also allows you to specify the `Allow` action, in which case the domains to match are determined by arrays in the VPN payload itself (`OnDemandMatchDomainsAlways`, `OnDemandMatchDomainsOnRetry`, and `OnDemandMatchDomainsNever`). However, this is deprecated in iOS 7.

Whenever a network change is detected, the VPN On Demand service compares the newly connected network against the match network criteria specified in each dictionary (in order) to determine whether VPN On Demand should be allowed or not on the newly joined network. The matching criteria can include any of the following:

- DNS domain or DNS server settings (with wildcard matching)
- SSID
- Interface type
- reachable server detection

Dictionaries are checked sequentially, beginning with the first dictionary in the array. A dictionary matches the current network only if *all* of the specified policies in that dictionary match. You should always set a default behavior for unknown networks by specifying an action with no matching criteria as the last dictionary in the array.

If a dictionary matches the current network, a server probe is sent if a URL is specified in the profile. VPN then acts according to the policy defined in the dictionary (for example, allow VPNOnDemand, ignore VPNOnDemand, connect, or disconnect).

> **Important:** Be sure to set a catch-all value. If you do not, the current default behavior is to allow the connection to occur, but this behavior is not guaranteed.

The `OnDemandRules` dictionaries can contain one or more of the following keys:

Key	Type	Value
`Action`	String	The action to take if this dictionary matches the current network. Possible values are: `Allow`—*Deprecated.* Allow VPN On Demand to connect if triggered. `Connect`—Unconditionally initiate a VPN connection on the next network attempt. `Disconnect`—Tear down the VPN connection and do not reconnect on demand as long as this dictionary matches. `EvaluateConnection`—Evaluate the `ActionParameters` array for each connection attempt. `Ignore`—Leave any existing VPN connection up, but do not reconnect on demand as long as this dictionary matches.
`ActionParameters`	Array of dictionaries	A dictionary that provides rules similar to the `OnDemandRules` dictionary, but evaluated on each connection instead of when the network changes. These dictionaries are evaluated in order, and the behavior is determined by the first dictionary that matches. The keys allowed in each dictionary are described in Table 1-1 (page 62). **Note:** This array is used only for dictionaries in which `EvaluateConnection` is the `Action` value.
`DNSDomainMatch`	Array of strings	An array of domain names. This rule matches if any of the domain names in the specified list matches any domain in the device's search domains list. A wildcard '*' prefix is supported. For example, `*.example.com` matches against either `mydomain.example.com` or `yourdomain.example.com`.

Key	Type	Value
DNSServerAddress-Match	Array of strings	An array of IP addresses. This rule matches if any of the network's specified DNS servers match any entry in the array. Matching with a single wildcard is supported. For example, 17.* matches any DNS server in the class A 17 subnet.
InterfaceTypeMatch	String	An interface type. If specified, this rule matches only if the primary network interface hardware matches the specified type. Supported values are Ethernet, WiFi, and Cellular.
SSIDMatch	Array of strings	An array of SSIDs to match against the current network. If the network is not a Wi-Fi network or if the SSID does not appear in this array, the match fails. Omit this key and the corresponding array to match against any SSID.
URLStringProbe	String	A URL to probe. If this URL is successfully fetched (returning a 200 HTTP status code) without redirection, this rule matches.

The keys allowed in each ActionParameters dictionary are described in Table 1-1.

Table 1-1 Keys in the ActionParameters dictionary

Key	Type	Value
Domains	Array of strings	*Required.* The domains for which this evaluation applies.

Key	Type	Value
`DomainAction`	String	*Required.* Defines the VPN behavior for the specified domains. Allowed values are: ConnectIfNeeded—The specified domains should trigger a VPN connection attempt if domain name resolution fails, such as when the DNS server indicates that it cannot resolve the domain, responds with a redirection to a different server, or fails to respond (timeout). NeverConnect—The specified domains will not trigger a VPN connection nor be accessible through an existing VPN connection.
`RequiredDNSServers`	Array of strings	*Optional.* An array of IP addresses of DNS servers to be used for resolving the specified domains. These servers need not be part of the device's current network configuration. If these DNS servers are not reachable, a VPN connection is established in response. These DNS servers should be either internal DNS servers or trusted external DNS servers. **Note:** This key is valid only if the value of `DomainAction` is `ConnectIfNeeded`.
`RequiredURLString-Probe`	String	*Optional.* An `HTTP` or `HTTPS` (preferred) URL to probe, using a `GET` request. If no HTTP response code is received from the server, a VPN connection is established in response. **Note:** This key is valid only if the value of `DomainAction` is `ConnectIfNeeded`.

IKEv2 Dictionary Keys

If `VPNType` is `IKEv2`, the following keys may be provided in a dictionary:

Key	Type	Value
`RemoteAddress`	String	Required. IP address or hostname of the VPN server.

Key	Type	Value
LocalIdentifier	String	Required. Identifier of the IKEv2 client in one of the following formats: • FQDN • UserFQDN • Address • ASN1DN
RemoteIdentifier	String	Required. Remote identifier in one of the following formats: • FQDN • UserFQDN • Address • ASN1DN
AuthenticationMethod	String	Required. One of the following: • SharedSecret • Certificate
PayloadCertificate–UUID	String	Optional. The UUID of the identity certificate as the account credential. If AuthenticationMethod is Certificate, and extended authentication (EAP) is not used, this certificate will be sent out for IKE client authentication. If extended authentication is used, this certificate can be used for EAP-TLS.
CertificateType	String	Optional. This key applies to the PayloadCertificateUUID, if specified. Value is one of the following: • RSA (Default) • ECDSA256 • ECDSA384 • ECDSA521 If this key is included, the ServerCertificate–IssuerCommonName key is required.

Key	Type	Value
`SharedSecret`	String	Optional. If `AuthenticationMethod` is `SharedSecret`, this value is used for IKE authentication.
`ExtendedAuthEnabled`	Integer	Optional. Set to 1 to enable extended authentication (EAP). Default to 0.
`AuthName`	String	Optional. Username used for authentication.
`AuthPassword`	String	Optional. Password used for authentication.
`DeadPeerDetection-Rate`	String	Optional. One of the following: • `None` (Disable) • `Low` (`keepalive` sent every 30 minutes) • `Medium` (`keepalive` sent every 10 minutes) • `High` (`keepalive` sent every 1 minute) Defaults to Medium.
`ServerCertificate-IssuerCommonName`	String	Optional. Common Name of the server certificate issuer. If set, this field will cause IKE to send a certificate request based on this certificate issuer to the server. This key is required if the `CertificateType` key is included.
`ServerCertificate-CommonName`	String	Optional. Common Name of the server certificate. This name is used to validate the certificate sent by the IKE server. If not set, the Remote Identifier will be used to validate the certificate.

Key	Type	Value
`NATKeepAliveOffload-Enable`	Integer	Optional. Set to 1 to enable or 0 to disable NAT Keepalive offload for Always On VPN IKEv2 connections. Keepalive packets are sent by the device to maintain NAT mappings for IKEv2 connections that have a NAT on the path. Keepalive packets are sent at regular interval when the device is awake. If `NATKeepAliveOffloadEnable` is set to 1, Keepalive packets will be offloaded to hardware while the device is asleep. NAT Keepalive offload has an impact on the battery life since extra workload is added during sleep. The default interval for the Keepalive offload packets is 20 seconds over WiFi and 110 seconds over Cellular interface. The default NAT Keepalive works well on networks with small NAT mapping timeouts but imposes a potential battery impact. If a network is known to have larger NAT mapping timeouts, larger Keepalive intervals may be safely used to minimize battery impact. The Keepalive interval can be modified by setting the `NATKeepAliveInterval` key. Default value for `NATKeepAliveOffloadEnable` is 1.
`NATKeepAliveInterval`	Integer	Optional. NAT Keepalive interval for Always On VPN IKEv2 connections. This value controls the interval over which Keepalive offload packets are sent by the device. The minimum value is 20 seconds. If no key is specified, the default is 20 seconds over WiFi and 110 seconds over a Cellular interface.
`EnablePFS`	Integer	Optional. Set to 1 to enable Perfect Forward Secrecy (PFS) for IKEv2 Connections. Default is 0.
`IKESecurity-Association-Parameters`	Dictionary	Optional. See table below. Applies to child Security Association unless `ChildSecurityAssociation-Parameters` is specified.
`ChildSecurity-Association-Parameters`	Dictionary	Optional. See table below.

The `IKESecurityAssociationParameters` and `ChildSecurityAssociationParameters` dictionaries may contain the following keys:

Key	Type	Value
`EncryptionAlgorithm`	String	Optional. One of: • `DES` • `3DES` (Default) • `AES-128` • `AES-256` • `AES-128-GCM (16-octet ICV)` • `AES-256-GCM (16-octet ICV)`
`IntegrityAlgorithm`	String	Optional. One of: • `SHA1-96` (Default) • `SHA1-160` • `SHA2-256` • `SHA2-384` • `SHA2-512`
`DiffieHellmanGroup`	Integer	Optional. One of: 1, 2 (Default), 5, 14, 15, 16, 17, 18, 19, 20, or 21.
`LifeTimeInMinutes`	Integer	Optional SA lifetime (rekey interval) in minutes. Valid values are 10 through 1440. Defaults to 1440 minutes.
`UseConfiguration-AttributeInternal-IPSubnet`	Integer	Optional. If set to 1, negotiations should use IKEv2 Configuration Attribute INTERNAL_IP4_SUBNET and INTERNAL_IP6_SUBNET. Defaults to 0. **Availability:** Available in iOS 9.0 and later.
`DisableMOBIKE`	Integer	Optional. If set to 1, disables MOBIKE. Defaults to 0. **Availability:** Available in iOS 9.0 and later.
`DisableRedirect`	Integer	Optional. If set to 1, disables IKEv2 redirect. If not set, the IKEv2 connection would be redirected if a redirect request is received from the server. Defaults to 0. **Availability:** Available in iOS 9.0 and later.

Key	Type	Value
NATKeepAliveOffload-Enable	Integer	Optional. Set to 1 to enable and 0 to disable NAT Keepalive offload for Always On VPN IKEv2 connections. Keepalive packets are used to maintain NAT mappings for IKEv2 connections. These packets are sent at regular interval when the device is awake. If NATKeepAliveOffloadEnable is set to 1, Keepalive packets would be sent by the chip even while the device is asleep. The default interval for the Keepalive packets for Always On VPN is 20 seconds over WiFi and 110 seconds over Cellular interface. The interval could be changed by setting the desired value in NATKeepAliveInterval. Defaults to 1. **Availability:** Available in iOS 9.0 and later.
NATKeepAliveInterval	Integer	Optional. Controls the interval over which Keepalive packets are sent by the device. The minimum value is 20 seconds. If no key is specified, the default is 20 seconds. **Availability:** Available in iOS 9.0 and later.
EnablePFS	Integer	Optional. Set to 1 to enable Perfect Forward Secrecy for IKEv2 EnablePFS connections. Default value is 0. **Availability:** Available in iOS 9.0 and later.

Proxies Dictionary Keys

The Proxies dictionary may contain the following keys:

Key	Type	Value
ProxyAutoConfig-Enable	Integer	Optional. Set to 1 to enable automatic proxy configuration. Defaults to 0.
ProxyAutoConfig-URLString	String	Optional. URL to the location of the proxy auto-configuration file. Used only when ProxyAutoConfigEnable is 1.
SupplementalMatch-Domains	Array of strings	Optional. If set, then only connections to hosts within one or more of the specified domains will use the proxy settings

If ProxyAutoConfigEnable is 0, the dictionary may also contain the following keys:

Key	Type	Value
HTTPEnable	Integer	Optional. Set to 1 to enable proxy for HTTP traffic. Defaults to 0.
HTTPProxy	String	Optional. The host name of the HTTP proxy.
HTTPPort	Integer	Optional. The port number of the HTTP proxy. This field is required if HTTPProxy is specified.
HTTPProxyUsername	String	Optional. The username used for authentication.
HTTPProxyPassword	String	Optional. The password used for authentication.
HTTPSEnable	Integer	Optional. Set to 1 to enable proxy for HTTPS traffic. Defaults to 0.
HTTPSProxy	String	Optional. The host name of the HTTPS proxy.
HTTPSPort	Integer	Optional. The port number of the HTTPS proxy. This field is required if HTTPSProxy is specified.

AlwaysOn Dictionary Keys

If VPNType is AlwaysOn, the following keys may be provided in a dictionary:

Key	Type	Value
UIToggleEnabled	Integer	Optional. If set to 1, allows the user to disable this VPN configuration. Defaults to 0.
TunnelConfigurations	Array of dictionaries	Required. See below.
ServiceExceptions	Array of dictionaries	Optional. See below.
AllowCaptiveWebSheet	Integer	Optional. Set to 1 to allow traffic from Captive Web Sheet outside the VPN tunnel. Defaults to 0.
AllowAllCaptive-NetworkPlugins	Integer	Optional. Set to 1 to allow traffic from all Captive Networking apps outside the VPN tunnel to perform Captive network handling. Defaults to 0.

Key	Type	Value
`AllowedCaptive-NetworkPlugins`	Array of dictionaries	Optional. Array of Captive Networking apps whose traffic will be allowed outside the VPN tunnel to perform Captive network handling. Used only when `AllowAllCaptiveNetworkPlugins` is 0. Each dictionary in the `AllowedCaptive-NetworkPlugins` array must contain a `BundleIdentifier` key of type string, the value of which is the app's bundle identifier. Captive Networking apps may require additional entitlements to operate in a captive environment.

Each dictionary in a `TunnelConfigurations` array may contain the following keys:

Key	Type	Value
`ProtocolType`	String	Must be IKEv2.
`Interfaces`	Array of strings	Optional. Specify the interfaces to which this configuration applies. Valid values are `Cellular` and `WiFi`. Defaults to `Cellular, WiFi`.

In addition, all keys defined for the IKEv2 dictionary, such as `RemoteAddress` and `LocalIdentifier` may be present in a `TunnelConfigurations` dictionary.

Each dictionary in a ServiceExceptions array may contain the following keys:

Key	Type	Value
`ServiceName`	String	Required. The name of a system service which is exempt from Always On VPN. Must be one of: • `VoiceMail` • `AirPrint`
`Action`	String	Required. One of the following: • `Allow` • `Drop`

Web Clip Payload

The Web Clip payload is designated by specifying `com.apple.webClip.managed` as the `PayloadType` value.

A Web Clip payload provides a web clipping on the user's home screen as though the user had saved a bookmark to the home screen.

In addition to the settings common to all payloads, this payload defines the following keys:

Key	Type	Value
`URL`	String	The URL that the Web Clip should open when clicked. The URL must begin with HTTP or HTTPS or it won't work.
`Label`	String	The name of the Web Clip as displayed on the Home screen.
`Icon`	Data	Optional. A PNG icon to be shown on the Home screen. Should be 59 x 60 pixels in size. If not specified, a white square will be shown.
`IsRemovable`	Boolean	Optional. If false, the web clip is unremovable. Defaults to true. Not available in OS X.

Web Content Filter Payload

The Web Content Filter payload allows you to whitelist and blacklist specific web URLs. This payload is supported only on supervised devices.

Web content filtering is designated by specifying `com.apple.webcontent-filter` as the `PayloadType` value and adding a `FilterType` string with one of these values:

- `BuiltIn` (Default)
- `PlugIn`

On OS X, `FilterType` must be `PlugIn`.

If `FilterType` is `BuiltIn`, this payload defines the following keys in addition to the settings common to all payloads:

Key	Type	Value
`AutoFilterEnabled`	Boolean	Optional. If `true`, automatic filtering is enabled. This function evaluates each web page as it is loaded and attempts to identify and block content not suitable for children. The search algorithm is complex and may vary from release to release, but it is basically looking for adult language, i.e. swearing and sexually explicit language. The default value is `false`.
`PermittedURLs`	Array of strings	Optional. Used only when `AutoFilterEnabled` is `true`. Otherwise, this field is ignored. Each entry contains a URL that is accessible whether the automatic filter allows access or not.
`WhitelistedBookmarks`	Array of dictionaries	Optional. If present, these URLs are added to the browser's bookmarks, and the user is not allowed to visit any sites other than these.
`BlacklistedURLs`	Array of strings	Optional. Access to the specified URLs is blocked.

Each entry in the `WhitelistedBookmarks` field contains a dictionary with the following keys:

Key	Type	Value
`URL`	String	URL of the whitelisted bookmark.
`BookmarkPath`	String	Optional. The folder into which the bookmark should be added in Safari—`/Interesting Topic Pages/Biology/`, for example. If absent, the bookmark is added to the default bookmarks directory.
`Title`	String	The title of the bookmark.

When multiple content filter payloads are present:

- The blacklist is the union of all blacklists—that is, any URL that appears in any blacklist is inaccessible.
- The permitted list is the intersection of all permitted lists—that is, only URLs that appear in *every* permitted list are accessible when they would otherwise be blocked by the automatic filter.
- The whitelist list is the intersection of all whitelists—that is, only URLs that appear in *every* whitelist are accessible.

URLs are matched by using string-based root matching. A URL matches a whitelist, blacklist, or permitted list pattern if the exact characters of the pattern appear as the root of the URL. For example, if `test.com/a` is blacklisted, then `test.com`, `test.com/b`, and `test.com/c/d/e` will all be blocked. Matching does not discard subdomain prefixes, so if `test.com/a` is blacklisted, `m.test.com` is not blocked. Also, no attempt is made to match aliases (IP address versus DNS names, for example) or to handle requests with explicit port numbers.

If a profile does not contain an array for `PermittedURLs` or `WhitelistedBookmarks`, that profile is skipped when evaluating the missing array or arrays. As an exception, if a payload contains an `AutoFilterEnabled` key, but does not contain a `PermittedURLs` array, that profile is treated as containing an empty array—that is, all websites are blocked.

All filtering options are active simultaneously. Only URLs and sites that pass **all** rules are permitted.

If `FilterType` is `PlugIn`, this payload defines the following keys in addition to the settings common to all payloads:

Key	Type	Value
`UserDefinedName`	String	A string which will be displayed for this filtering configuration.
`PluginBundleID`	String	The Bundle ID of the plugin that provides filtering service.
`ServerAddress`	String	Optional. Server address (may be IP address, hostname, or URL).
`UserName`	String	Optional. A username for the service.
`Password`	String	Optional. A password for the service.
`PayloadCertificate-UUID`	String	Optional. UUID pointing to an identity certificate payload. This identity will be used to authenticate the user to the service.
`Organization`	String	Optional. An Organization string that will be passed to the 3rd-party plugin.
`VendorConfig`	Dictionary	Optional. Custom dictionary needed by the filtering service plugin.
`FilterBrowsers`	Integer	Optional. If set to 1, filter WebKit traffic. Defaults to 0.
`FilterSockets`	Integer	Optional. If set to 1, filter socket traffic. Defaults to 0.

At least one of `FilterBrowsers` or `FilterSockets` must be `true` for the filter to have any effect.

Wi-Fi Payload

The Wi-Fi payload is designated by specifying `com.apple.wifi.managed` as the `PayloadType` value.

In addition to the settings common to all payload types, the payload defines the following keys.

Key	Type	Value
`SSID_STR`	String	SSID of the Wi-Fi network to be used. In iOS 7.0 and later, this is optional if a `DomainName` value is provided
`HIDDEN_NETWORK`	Boolean	Besides SSID, the device uses information such as broadcast type and encryption type to differentiate a network. By default (`false`), it is assumed that all configured networks are open or broadcast. To specify a hidden network, must be `true`.
`AutoJoin`	Boolean	Optional. Default `true`. If `true`, the network is auto-joined. If `false`, the user has to tap the network name to join it. **Availability:** Available in iOS 5.0 and later and in all versions of OS X.
`EncryptionType`	String	The possible values are `WEP`, `WPA`, `WPA2`, `Any`, and `None`. `WPA` specifies WPA only; WPA2 applies to both encryption types. Make sure that these values exactly match the capabilities of the network access point. If you're unsure about the encryption type, or would prefer that it apply to all encryption types, use the value `Any`. **Availability:** Key available in iOS 4.0 and later and in all versions of OS X. The `None` value is available in iOS 5.0 and later and the `WPA2` value is available in iOS 8.0 and later.
`IsHotspot`	Boolean	Optional. Default `false`. If `true`, the network is treated as a hotspot. **Availability:** Available in iOS 7.0 and later and in OS X 10.9 and later.
`DomainName`	String	Optional. Domain Name used for Wi-Fi Hotspot 2.0 negotiation. This field can be provided instead of `SSID_STR`. **Availability:** Available in iOS 7.0 and later and in OS X 10.9 and later..

Key	Type	Value
ServiceProvider-RoamingEnabled	Boolean	Optional. If true, allows connection to roaming service providers. **Availability:** Available in iOS 7.0 and later and in OS X 10.9 and later..
RoamingConsortiumOIs	Array of strings	Optional. Array of Roaming Consortium Organization Identifiers used for Wi-Fi Hotspot 2.0 negotiation. **Availability:** Available in iOS 7.0 and later and in OS X 10.9 and later..
NAIRealmNames	Array of strings	Optional. Array of strings. List of Network Access Identifier Realm names used for Wi-Fi Hotspot 2.0 negotiation. **Availability:** Available in iOS 7.0 and later and in OS X 10.9 and later..
MCCAndMNCs	Array of strings	Optional. Array of strings. List of Mobile Country Code (MCC)/Mobile Network Code (MNC) pairs used for Wi-Fi Hotspot 2.0 negotiation. Each string must contain exactly six digits. **Availability:** Available in iOS 7.0 and later. This feature is not supported in OS X.
DisplayedOperator-Name	String	**Availability:** Available in iOS 7.0 and later and in OS X 10.9 and later..
ProxyType	String	Optional. Valid values are None, Manual, and Auto. **Availability:** Available in iOS 5.0 and later and on all versions of OS X.

If the EncryptionType field is set to WEP, WPA, or ANY, the following fields may also be provided:

Key	Type	Value
Password	String	Optional.
EAPClientConfiguration	Dictionary	Described in EAPClientConfiguration Dictionary (page 76).
PayloadCertificateUUID	String	Described in Certificates (page 79).

Note: The absence of a password does not prevent a network from being added to the list of known networks. The user is eventually prompted to provide the password when connecting to that network.

If the `ProxyType` field is set to `Manual`, the following fields must also be provided:

Key	Type	Value
`ProxyServer`	String	The proxy server's network address.
`ProxyPort`	Integer	The proxy server's port.
`ProxyUsername`	String	Optional. The username used to authenticate to the proxy server.
`ProxyPassword`	String	Optional. The password used to authenticate to the proxy server.
`ProxyPACURL`	String	Optional. The URL of the PAC file that defines the proxy configuration.
`ProxyPACFallback-Allowed`	Boolean	Optional. If `false`, prevents the device from connecting directly to the destination if the PAC file is unreachable. Default is `false`. **Availability:** Available in iOS 7 and later.

If the `ProxyType` field is set to `Auto` and no `ProxyPACURL` value is specified, the device uses the web proxy autodiscovery protocol (WPAD) to discover proxies.

For 802.1X enterprise networks, the EAP Client Configuration Dictionary must be provided.

EAPClientConfiguration Dictionary

In addition to the standard encryption types, it is possible to specify an enterprise profile for a given network via the `EAPClientConfiguration` key. If present, its value is a dictionary with the following keys.

Key	Type	Value
`UserName`	String	Optional. Unless you know the exact user name, this property won't appear in an imported configuration. Users can enter this information when they authenticate.

Key	Type	Value
`AcceptEAPTypes`	Array of integers.	The following EAP types are accepted: 13 = TLS 17 = LEAP 18 = EAP-SIM 21 = TTLS 23 = EAP-AKA 25 = PEAP 43 = EAP-FAST
`UserPassword`	String	Optional. User password. If not provided, the user may be prompted during login.
`OneTimePassword`	Boolean	Optional. If `true`, the user will be prompted for a password each time they connect to the network. Defaults to `false`.
`PayloadCertificate-AnchorUUID`	Array of strings	Optional. Identifies the certificates to be trusted for this authentication. Each entry must contain the UUID of a certificate payload. Use this key to prevent the device from asking the user if the listed certificates are trusted. Dynamic trust (the certificate dialogue) is disabled if this property is specified, unless TLSAllowTrustExceptions is also specified with the value `true`.
`TLSTrustedServer-Names`	Array of strings	Optional. This is the list of server certificate common names that will be accepted. You can use wildcards to specify the name, such as wpa.*.example.com. If a server presents a certificate that isn't in this list, it won't be trusted. Used alone or in combination with TLSTrustedCertificates, the property allows someone to carefully craft which certificates to trust for the given network, and avoid dynamically trusted certificates. Dynamic trust (the certificate dialogue) is disabled if this property is specified, unless TLSAllowTrustExceptions is also specified with the value `true`.

Key	Type	Value
`TLSAllowTrust-Exceptions`	Boolean	Optional. Allows/disallows a dynamic trust decision by the user. The dynamic trust is the certificate dialogue that appears when a certificate isn't trusted. If this is `false`, the authentication fails if the certificate isn't already trusted. See PayloadCertificateAnchorUUID and TLSTrustedNames above. The default value of this property is `true` unless either PayloadCertificateAnchorUUID or TLSTrustedServerNames is supplied, in which case the default value is `false`.
`TLSCertificateIs-Required`	`Boolean`	Optional. If `true`, allows for two-factor authentication for EAP-TTLS, PEAP, or EAP-FAST. If `false`, allows for zero-factor authentication for EAP-TLS. The default is `true` for EAP-TLS, and `false` for other EAP types. **Availability:** Available in iOS 7.0 and later.
`OuterIdentity`	`String`	Optional. This key is only relevant to TTLS, PEAP, and EAP-FAST. This allows the user to hide his or her identity. The user's actual name appears only inside the encrypted tunnel. For example, it could be set to "anonymous" or "anon", or "anon@mycompany.net". It can increase security because an attacker can't see the authenticating user's name in the clear.
`TTLSInner-Authentication`	String	Optional. Specifies the inner authentication used by the TTLS module. Possible values are PAP, CHAP, MSCHAP, MSCHAPv2, and EA. Defaults to MSCHAPv2.

Note: For information about EAP-SIM, see tools.ietf.org/html/rfc4186.

EAP-Fast Support

The EAP-FAST module uses the following properties in the EAPClientConfiguration dictionary.

Key	Type	Value
`EAPFASTUsePAC`	Boolean	Optional.If `true`, the device will use an existing PAC if it's present. Otherwise, the server must present its identity using a certificate. Defaults to `false`.

Key	Type	Value
EAPFASTProvisionPAC	Boolean	Optional. Used only if EAPFASTUsePAC is true. If set to true, allows PAC provisioning. Defaults to false. This value must be set to true for EAP-FAST PAC usage to succeed, because there is no other way to provision a PAC.
EAPFASTProvision-PACAnonymously	Boolean	Optional. If true, provisions the device anonymously. Note that there are known man-in-the-middle attacks for anonymous provisioning. Defaults to false.
EAPSIMNumberOfRANDs	Integer	Optional. Number of expected RANDs for EAPSIM. Valid values are 2 and 3. Defaults to 3.

These keys are hierarchical in nature: if EAPFASTUsePAC is false, the other two properties aren't consulted. Similarly, if EAPFASTProvisionPAC is false, EAPFASTProvisionPACAnonymously isn't consulted.

If EAPFASTUsePAC is false, authentication proceeds much like PEAP or TTLS: the server proves its identity using a certificate each time.

If EAPFASTUsePAC is true, then an existing PAC is used if present. The only way to get a PAC on the device currently is to allow PAC provisioning. So, you need to enable EAPFASTProvisionPAC, and if desired, EAPFASTProvisionPACAnonymously. EAPFASTProvisionPACAnonymously has a security weakness: it doesn't authenticate the server so connections are vulnerable to a man-in-the-middle attack.

Certificates

As with VPN configurations, it is possible to associate a certificate identity configuration with a Wi-Fi configuration. This is useful when defining credentials for a secure enterprise network. To associate an identity, specify its payload UUID via the "PayloadCertificateUUID" key.

Key	Type	Value
PayloadCertificateUUID	String	UUID of the certificate payload to use for the identity credential.

Domains Payload

This payload defines domains that are under an enterprise's management. This payload is designated by the com.apple.domains PayloadType value.

Unmarked Email Domains

Any email address that does not have a suffix that matches one of the unmarked email domains specified by the key `EmailDomains` will be considered out-of-domain and will be highlighted as such in the Mail app.

Key	Type	Value
`EmailDomains`	Array	Optional. An array of strings. An email address lacking a suffix that matches any of these strings will be considered out-of-domain.

Managed Safari Web Domains

Opening a document originating from a managed Safari web domain causes iOS to treat the document as managed for the purpose of Managed Open In.

Key	Type	Value
`WebDomains`	Array	Optional. An array of URL strings. URLs matching the patterns listed here will be considered managed. Not supported in OS X
`SafariPasswordAuto-FillDomains`	Array	Optional. An array of URL strings. Supported in iOS 9.3 and later; not supported in OS X. Users can save passwords in Safari only from URLs matching the patterns listed here. Regardless of the iCloud account that the user is using, if the device is not supervised, there can be no whitelist. If the device is supervised, there may be a whitelist, but if there is still no whitelist, note these two cases: • If the device is configured as ephemeral multi-user, no password can be saved. • If the device is not configured as ephemeral multi-user, all passwords can be saved.

The `WebDomains` and `SafariPasswordAutoFillDomains` arrays may contain strings using any of the following matching patterns:

Format	Description
`apple.com`	Any path under `apple.com` matches, but not `site.apple.com/`.

Format	Description
`foo.apple.com`	Any path under `foo.apple.com` matches, but not `apple.com/` or `bar.apple.com/`.
`*.apple.com`	Any path under `foo.apple.com` or `bar.apple.com` matches, but not `apple.com`.
`apple.com/sub`	`apple.com/sub` and any path under it matches, but not `apple.com/`.
`foo.apple.com/sub`	Any path under `foo.apple.com/sub` matches, but not `apple.com`, `apple.com/sub`, `foo.apple.com/`, or `bar.apple.com/sub`.
`*.apple.com/sub`	Any path under `foo.apple.com/sub` or `bar.apple.com/sub` matches, but not `apple.com` or `foo.apple.com/`.
`*.co`	Any path under `apple.co` or `beats.co` matches, but not `apple.co.uk` or `apple.com`.

A URL that begins with the prefix `www.` is treated as though it did not contain that prefix during matching. For example, `http://www.apple.com/store` will be matched as `http://apple.com/store`.

Trailing slashes will be ignored.

If a `ManagedWebDomain` string entry contains a port number, only addresses that specify that port number will be considered managed. Otherwise, the domain will be matched without regard to the port number specified. For example, the pattern `*.apple.com:8080` will match `http://site.apple.com:8080/page.html` but not `http://site.apple.com/page.html`, while the pattern `*.apple.com` will match both URLs.

Managed Safari Web Domain definitions are cumulative. Patterns defined by all Managed Web Domains payloads will be used to match a URL request.

`SafariPasswordAutoFillDomains` definitions are cumulative. Patterns defined by all `SafariPasswordAutoFillDomains` payloads will be used to determine if passwords can be stored for a given URL.

OS X Server Payload

This payload defines an OS X Server account. This payload is designated by the `com.apple.osxserver.account` PayloadType value.

Key	Type	Value
`HostName`	String	Mandatory. The server address.
`UserName`	String	Mandatory. The user's login name.
`Password`	String	Optional. The user's password.
`AccountDescription`	String	Optional. Description of the account.
`ConfiguredAccounts`	Array	Mandatory. An array of dictionaries containing configured account types and relevant settings. Each dictionary consists of a `Type` field plus additional settings specific to the `Type`.

The following `ConfiguredAccounts` dictionary is currently supported:

Documents Dictionary

Key	Type	Value
`Type`	String	Mandatory. The Documents account type: `com.apple.osxserver.documents`.
`Port`	Number	Optional. Designates the port number to use when contacting the server. If no port number is specified, the default port is used.

Encrypted Profiles

A profile can be encrypted so that it can only be decrypted using a private key previously installed on a device.

To encrypt a profile do the following:

1. Remove the `PayloadContent` array and serialize it as a proper plist. Note that the top-level object in this plist is an array, not a dictionary.
2. CMS-encrypt the serialized plist as enveloped data.
3. Serialize the encrypted data in DER format.
4. Set the serialized data as the value of as a Data plist item in the profile, using the key `EncryptedPayloadContent`.

Signing a Profile

To sign a profile, place the XML plist in a DER-encoded CMS Signed Data data structure.

Sample Configuration Profile

The following is a sample configuration profile containing an SCEP payload.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Inc//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
    <dict>
        <key>PayloadVersion</key>
        <integer>1</integer>
        <key>PayloadUUID</key>
        <string>Ignored</string>
        <key>PayloadType</key>
        <string>Configuration</string>
        <key>PayloadIdentifier</key>
        <string>Ignored</string>
        <key>PayloadContent</key>
        <array>
            <dict>
                <key>PayloadContent</key>
                <dict>
                    <key>URL</key>
                    <string>https://scep.example.com/scep</string>
                    <key>Name</key>
                    <string>EnrollmentCAInstance</string>
                    <key>Subject</key>
                    <array>
                        <array>
                            <array>
                                <string>O</string>
                                <string>Example, Inc.</string>
```

```
                    </array>
                </array>
                <array>
                    <array>
                        <string>CN</string>
                        <string>User Device Cert</string>
                    </array>
                </array>
            </array>
            <key>Challenge</key>
            <string>...</string>
            <key>Keysize</key>
            <integer>1024</integer>
            <key>Key Type</key>
            <string>RSA</string>
            <key>Key Usage</key>
            <integer>5</integer>
        </dict>
        <key>PayloadDescription</key>
        <string>Provides device encryption identity</string>
        <key>PayloadUUID</key>
        <string>fd8a6b9e-0fed-406f-9571-8ec98722b713</string>
        <key>PayloadType</key>
        <string>com.apple.security.scep</string>
        <key>PayloadDisplayName</key>
        <string>Encryption Identity</string>
        <key>PayloadVersion</key>
        <integer>1</integer>
        <key>PayloadOrganization</key>
        <string>Example, Inc.</string>
        <key>PayloadIdentifier</key>
        <string>com.example.profileservice.scep</string>
    </dict>
</array>
```

```
    </dict>
</plist>
```

Document Revision History

This table describes the changes to *Configuration Profile Reference*.

Date	Notes
2016-01-20	Updated to iOS 9.3 and made other updates and corrections.
2015-12-08	Minor updates and corrections.
2015-10-08	Minor revision.
2015-09-17	Update for iOS 9 and OS X 10.11.
2015-06-12	Made miscellaneous updates and corrections. Updated rules for removal of profiles installed through an MDM server. Added new section Network Usage Rules (page 36). Added new section OS X Server Payload (page 81). Added new Email, Restrictions, SCEP, and VPN Payload keys. Clarified Web Content Filter URL matching.
2015-01-31	Added new keys to the Restrictions Payload and clarified managed domain terminology.
2014-09-17	Updated for iOS 8 and OS X v10.10.
2014-03-20	Updated for iOS 7.1.
2014-01-14	Updated for iOS 7 and OS X v10.9.

Date	Notes
2013-10-22	Added information about the keychain syncing restriction.
2013-10-01	Removed unsupported keys from document.
2013-09-18	Updated with a few additional iOS 7 keys.
2012-12-13	Corrected minor technical and typographical errors.
2012-09-22	Made minor typographical fixes and clarified a few details specific to OS X.
2012-09-19	Updated document for iOS 6 and added support for OS X 10.8.
2011-10-17	Removed extraneous iCloud key.
2011-10-12	Updated for iOS 5.0.
2011-03-08	Retitled document.
2010-09-21	Fixed typographical errors.
2010-08-03	New document that describes the property list keys used in iOS configuration profiles.

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010