Configuration Profile Reference

Developer

Contents

Beta Software

This documentation contains preliminary information about an API or technology in development. This information is subject to change, and software implemented according to this documentation should be tested with final operating system software.

> **Note**
>
> This document was previously titled *iPhone Configuration Profile Reference*. It now supports both iOS and macOS.

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 macOS 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 information about Apple Configurator, go to the Mac App Store description at Apple Configurator 2.

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.
`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.
`PayloadExpirationDate`	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.

Key	Type	Content
`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 macOS, you can use *uuidgen* to generate reasonable UUIDs.
`PayloadRemovalDisallowed`	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`	Integer	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 macOS 10.7 and later.
`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. A dictionary containing these keys and values: • For each language in which a consent or license agreement is available, a key consisting of the IETF BCP 47 identifier for that language (for example, `en` or `jp`) and a value consisting of the agreement localized to that language. The agreement is displayed in a dialog to which the user must agree before installing the profile. • The optional key `default` with its value consisting of the unlocalized agreement (usually in `en`). The system chooses a localized version in the order of preference specified by the user (macOS) 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 value if possible. No warning will be displayed if the user's locale does not match any localization 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

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.
`PayloadVersion`	Integer	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 macOS, 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), 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 https://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.
`CertificateAcquisitionMechanism`	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>.
`CertificateRenewalTimeInterval`	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 macOS 10.11 and later.
`EnableAutoRenewal`	Boolean	Optional. If set to `true`, the certificate obtained with this payload will attempt auto-renewal. Only applies to device Active Directory certificate payloads. **Availability:** Available in macOS 10.13.4 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 macOS 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 macOS).
`Password`	String	The password for the AirPlay destination.

AirPlay Security Payload

The AirPlay Security payload locks the Apple TV to a particular style of AirPlay Security. The AirPlay Security payload is designated by specifying `com.apple.airplay.security` as the `PayloadType` vaue.

This payload is supported on tvOS 11.0 and later.

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

Key	Type	Value
`SecurityType`	String	Required. Must be one of the defined values: `PASSCODE_ONCE`, `PASSCODE_ALWAYS`, or `PASSWORD`. `PASSCODE_ONCE` will require an on-screen passcode to be entered on the first connection from a device. Subsequent connections from the same device will not be prompted. `PASSCODE_ALWAYS` will require an on-screen passcode to be entered upon every AirPlay connection. `PASSWORD` will require a passphrase to be entered as specified in the Password key. The Password key is required if this `SecurityType` is selected. `NONE` was deprecated in tvOS 11.3. Existing profiles using `NONE` will get the `PASSWORD_ONCE` behavior.
`AccessType`	String	Required. Must be one of the defined values: `ANY` or `WIFI_ONLY`. `ANY` allows connections from both Ethernet/WiFi and AWDL. `WIFI_ONLY` allows connections only from devices on the same Ethernet/WiFi network as the Apple TV.
`Password`	String	Optional. The AirPlay password. Required if `SecurityType` is `PASSWORD`.

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 macOS 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`
`Port`	Integer	Listening port of the AirPrint destination. If this key is not specified AirPrint will use the default port. **Availability:** Available only in iOS 11.0 and later.
`ForceTLS`	Boolean	If `true` AirPrint connections are secured by Transport Layer Security (TLS). Default is `false`. **Availability:** Available only in iOS 11.0 and later.

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.

> **Note**
>
> You can't update any app while the device is locked in Single App Mode. You need to exit Single App Mode long enough to update apps as needed. During that time you should restrict the visible apps as much as possible, except for Settings and Phone and any other apps that cannot be blacklisted.

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`. Also, available in tvOS 10.2 and later to lock the touch pad on the remote.
`DisableDeviceRotation`	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.
`DisableSleepWakeButton`	Boolean	Optional. If `true`, the sleep/wake button is disabled. Default is `false`.

Key	Type	Value
`DisableAutoLock`	Boolean	Optional. If `true`, the device will not automatically go to sleep after an idle period. Also, available in tvOS 10.2 and later.
`EnableVoiceOver`	Boolean	Optional. If `true`, VoiceOver is turned on. Default is `false`. Also, available in tvOS 10.2 and later.
`EnableZoom`	Boolean	Optional. If `true`, Zoom is turned on. Default is `false`. Also, available in tvOS 10.2 and later.
`EnableInvertColors`	Boolean	Optional. If `true`, Invert Colors is turned on. Default is `false`. Also, available in tvOS 10.2 and later.
`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`. Also, available in tvOS 10.2 and later.
`Zoom`	Boolean	Optional. If `true`, allow Zoom adjustment. Default is `false`. Also, available in tvOS 10.2 and later.
`InvertColors`	Boolean	Optional. If `true`, allow Invert Colors adjustment. Default is `false`. Also, available in tvOS 10.2 and later.
`AssistiveTouch`	Boolean	Optional. If `true`, allow AssistiveTouch adjustment. Default is `false`.

AppStore Payload

The AppStore payload is designated by specifying `com.apple.appstore` as the `PayloadType` value.

It establishes macOS AppStore restrictions and is supported on the User channel.

The payload contains the following keys:

Key	Type	Value
`restrict-store-require-admin-to-install`	Boolean	Optional. Restrict app installations to admin users. Available on macOS 10.9 and later.
`restrict-store-softwareupdate-only`	Boolean	Optional. Restrict app installations to software updates only. Available on macOS 10.10 and later.
`restrict-store-disable-app-adoption`	Boolean	Optional. Disable App Adoption by users. Available on macOS 10.10 and later.
`DisableSoftwareUpdateNotifications`	Boolean	Optional. Disable software update notifications. Available on macOS 10.10 and later.

Autonomous Single App Mode

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

This payload grants Autonomous Single App Mode capabilities for specific applications. Available in macOS 10.13.4 and later.

It must be installed as a device profile. Only one payload of this type can be installed on a system. This payload can only be installed via a "user approved" MDM server.

> **Note**
>
> Applications listed in this payload will have low-level access to the system, including, but not limited to, key logging and user interface manipulation outside of the application's context.

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

Key	Type	Value
`AllowedApplications`	Array	Array of dictionaries that specify applications that are to be granted access to Assessment APIs.

Each dictionary in the `AllowedApplications` array consists of:

Key	Type	Value
`BundleIdentifier`	String	The application's bundle identifier. `BundleIdentifier` must be unique. If two dictionaries contain the same `BundleIdentifier` but different `TeamIdentifiers`, this will be considered a hard error and the payload will not be installed.
`TeamIdentifier`	String	The developer's team identifier used to sign the application.

To be granted access, applications must be signed with the specified bundle identifier and team identifier using an Apple-issued production developer certificate. Applications must specify the `com.apple.developer.assessment` entitlement with a value of `true`.

CalDAV Payload

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

This payload configures a CalDAV account.

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

Key	Type	Value
`CalDAVAccountDescription`	String	Optional. The description of the account.
`CalDAVHostName`	String	The server address. In macOS, this key is required.
`CalDAVUsername`	String	The user's login name. In macOS, this key is required.
`CalDAVPassword`	String	Optional. The user's password.
`CalDAVUseSSL`	Boolean	Whether or not to use SSL. In macOS, this key is optional.
`CalDAVPort`	Integer	Optional. The port on which to connect to the server.
`CalDAVPrincipalURL`	String	Optional. The base URL to the user's calendar. In macOS 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 macOS.

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 macOS 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`	Integer	Optional. The port on which to connect to the server.
`CardDAVPrincipalURL`	String	Optional. Not supported on macOS. 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 macOS. 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.
`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`	Integer	Optional. The proxy server's port.
`DefaultProtocolMask`	Integer	**Deprecated.** Default Internet Protocol versions. Set to the same value as `AllowedProtocolMask`. Possible values are: 1 = IPv4, 2 = IPv6, and 3 = Both. **Availability:** Available in iOS 10.3 and later.
`AllowedProtocolMask`	Integer	Optional. Supported Internet Protocol versions. Possible values are: 1 = IPv4, 2 = IPv6, and 3 = Both. **Availability:** Available in iOS 10.3 and later.

Key	Type	Value
`AllowedProtocolMaskInRoaming`	Integer	Optional. Supported Internet Protocol versions while roaming. Possible values are: 1 = IPv4, 2 = IPv6, and 3 = Both. **Availability:** Available in iOS 10.3 and later.
`AllowedProtocolMaskInDomesticRoaming`	Integer	Optional. Supported Internet Protocol versions while domestic roaming. Possible values are: 1 = IPv4, 2 = IPv6, and 3 = Both. **Availability:** Available in iOS 10.3 and later.

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.
`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
`PayloadCertificateFileName`	String	Optional. The file name of the enclosed certificate.
`PayloadContent`	Data	Mandatory. The base64 representation of the payload with a line length of 52.
`Password`	String	Optional. For PKCS#12 certificates, contains the password to the identity.

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

Certificate Preference Payload

Certificate Preference payloads are designated by specifying `com.apple.security.certificatepreference` as the `PayloadType` value. See also Identity Preference Payload for setting up identity preferences.

A Certificate Preference payload lets you identify a Certificate Preference item in the user's keychain that references a certificate payload included in the same profile. It can only appear in a user profile, not a device profile. You can include multiple Certificate Preference payloads as needed.

Available in MacOS 10.12 and later.

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

Key	Type	Value
`Name`	String	Required. An email address (RFC822) or other name for which a preferred certificate is requested.
`PayloadCertificateUUID`	String	The UUID of another payload within the same profile that installed the certificate; for example, a 'com.apple.security.root' payload.

Conference Room Display Payload

The Conference Room Display payload is designated by specifying `com.apple.conferenceroomdisplay` as the `PayloadType`.

It configures an Apple TV to enter Conference Room Display mode and restricts exit from that mode. It is supported on supervised devices running tvOS 10.2 or later.

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

Key	Type	Value
`Message`	String	Optional. A custom message displayed on the screen in Conference Room Display mode.

> **Note**
>
> When Conference Room Display mode and Single App mode are both enabled, Conference Room Display mode is active and the user can't access the app.

Content Caching Payload

The Content Caching payload is designated by specifying `com.apple.AssetCache.managed` as the `PayloadType`.

It configures the Content Caching service.

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

Key	Type	Value
`AllowPersonalCaching`	Boolean	Optional. If set to `true`, caches the user's iCloud data. Clients may take some time (hours, days) to react to changes to this setting; it does not have an immediate effect. Default is `true`. At least one of the `AllowPersonalCaching` or `AllowSharedCaching` keys must be `true`. **Availability:** Available in macOS 10.13.4 and later.
`AllowSharedCaching`	Boolean	Optional. If set to `true`, caches non-iCloud content, such as apps and software updates. Clients may take some time (hours, days) to react to changes to this setting; it does not have an immediate effect. Default is `true`. At least one of the `AllowPersonalCaching` or `AllowSharedCaching` keys must be `true`. **Availability:** Available in macOS 10.13.4 and later.
`AutoActivation`	Boolean	Optional. If set to `true`, automatically activate the Content Cache when possible and prevent disabling of the Content Cache. Default is `false`. **Availability:** Available in macOS 10.13.4 and later.
`CacheLimit`	Integer	Optional. Defines the maximum number of bytes of disk space that will be used for the Content Cache. A `CacheLimit` of 0 means unlimited disk space. Default is `0`. **Availability:** Available in macOS 10.13.4 and later.
`DataPath`	String	Optional. The path to the directory used to store Cached Content. Changing this setting manually does not automatically move cached content from the old to the new location. To move content automatically, use the Sharing preference's Content Caching pane. The value must be, or end with, `/Library/Application Support/Apple/AssetCache/Data`. A directory (and its intermediates) will be created for the given `DataPath` if it does not already exist. The directory will be owned by `_assetcache:_assetcache` and have mode 0750. Its immediate parent directory (`.../Library/Application Support/Apple/AssetCache`) will be owned by `_assetcache:_assetcache` and have mode 0755. Default is `/Library/Application Support/Apple/AssetCache/Data`. **Availability:** Available in macOS 10.13.4 and later.

Key	Type	Value
`DenyTetheredCaching`	Boolean	Optional. If set to `true`, tethered caching is disabled. Default is `false`. **Availability:** Available in macOS 10.13.4 and later.
`ListenRanges`	Array of Dictionaries	Optional. Array of dictionaries describing a range of client IP addresses to serve. **Availability:** Available in macOS 10.13.4 and later.
`ListenRangesOnly`	Boolean	Optional. If set to `true`, the Content Cache provides content only to clients in the ranges specified by the `ListenRanges` key. To use the `ListenRangesOnly` key, the `ListenRanges` key must also be specified. Default is `false`. **Availability:** Available in macOS 10.13.4 and later.
`ListenWithPeersAndParents`	Boolean	Optional. If set to `true`, the Content Cache provides content to the clients in the union of the ListenRanges, PeerListenRanges and Parents ranges. Default is `true`. **Availability:** Available in macOS 10.13.4 and later.
`LocalSubnetsOnly`	Boolean	Optional. If set to `true`, the Content Cache offers content to clients only on the same immediate local network as the Content Cache. No content would be offered to clients on other networks reachable by the Content Cache. Default is `true`. If `LocalSubnetsOnly` is set to true, `ListenRanges` will be ignored. **Availability:** Available in macOS 10.13.4 and later.
`LogClientIdentity`	Boolean	Optional. If set to `true`, the Content Cache will log the IP address and port number of the clients that request content. Default is `false`. **Availability:** Available in macOS 10.13.4 and later.
`Parents`	Array of Strings	Optional. Array of the local IP addresses of other Content Caches that this cache should download from or upload to, instead of downloading from or uploading to Apple directly. Invalid addresses and addresses of computers that are not Content Caches are ignored. Parent caches that become unavailable are skipped. If all parent Content Caches become unavailable, the Content Cache will download from or upload to Apple directly until a parent Content Cache becomes available again. **Availability:** Available in macOS 10.13.4 and later.

Key	Type	Value
`ParentSelectionPolicy`	String	Optional. The policy to use when choosing among more than one configured parent Content Cache. With every policy, parent caches that are temporarily unavailable are skipped. • `first-available`: Always use the first parent in the Parents list that is available. This is useful for designating permanent primary, secondary, and subsequent parents. • `url-path-hash`: Hash the path part of the requested URL so that the same parent is always used for the same URL. This is useful for maximizing the size of the combined caches of the parents. • `random`: Choose a parent at random. This is useful for load balancing. • `round-robin`: Rotate through the parents in order. This is useful for load balancing. • `sticky-available`: Starting with the first parent in the Parents list, always use the first parent that is available. Use that parent until it becomes unavailable, then advance to the next one. This is useful for designating floating primary, secondary, and subsequent parents. Default is `round-robin`. **Availability:** Available in macOS 10.13.4 and later.
`PeerFilterRanges`	Array of Dictionaries	Optional. Array of dictionaries describing a range of peer IP addresses that the Content Cache will use to filter its list of peers to query for content. The Content Cache only queries peers that are in the `PeerFilterRanges`. When `PeerFilterRanges` is an empty array the Content Cache will not query any peers. **Availability:** Available in macOS 10.13.4 and later.
`PeerListenRanges`	Array of Dictionaries	Optional. Array of dictionaries describing a range of peer IP addresses the Content Cache will respond to peer cache queries from. When `PeerListenRanges` is an empty array, the Content Cache will respond with an error to all cache queries. **Availability:** Available in macOS 10.13.4 and later.
`PeerLocalSubnetsOnly`	Boolean	Optional. If set to `true`, the Content Cache will only peer with other Content Caches on the same immediate local network, rather than with Content Caches that use the same public IP address as the device. When `PeerLocalSubnetsOnly` is true, it overrides the configuration of `PeerFilterRanges` and `PeerListenRanges`. If the network changes, the local network peering restrictions update appropriately. If set to `false`, the Content Cache defers to `PeerFilterRanges` and `PeerListenRanges` for configuring the peering restrictions. Default is `true`. **Availability:** Available in macOS 10.13.4 and later.

Key	Type	Value
`Port`	Integer	Optional. The TCP port number on which the Content Cache accepts requests for uploads or downloads. Port set to 0 picks a random, available port. Default is `0`. **Availability:** Available in macOS 10.13.4 and later.
`PublicRanges`	Array of Dictionaries	Optional. Array of dictionaries describing a range of public IP addresses that the cloud servers should use for matching clients to Content Caches. **Availability:** Available in macOS 10.13.4 and later.

The dictionary used to define ranges used by the Content Cache uses the following keys:

Key	Type	Value
`type`	String	Optional. The IP address type (`IPv4` or `IPv6`). Default is `IPv4`.
`first`	String	Required. First IP address in the range.
`last`	String	Required. Last IP address in the range.

Desktop Payload

The Desktop payload is designated by specifying `com.apple.desktop` as the `PayloadType`.

This payload sets up macOS Desktop settings and restrictions. It is supported on the user channel and on macOS 10.10 and later.

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

Key	Type	Value
`locked`	Boolean	Optional. If `true`, the desktop picture is locked. Default is `false`.
`override-picture-path`	String	Optional. If supplied, it sets the path to the desktop picture.

DNS Proxy Payload

The DNS Proxy payload is designated by specifying `com.apple.dnsProxy.managed` as the `PayloadType`. This payload can be installed only on a Supervised device.

This payload sets up iOS DNS Proxy settings. It is supported on iOS 11 and later.

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

Key	Type	Value
`AppBundleIdentifier`	String	Required. Bundle identifer of the app containing the DNS proxy network extension.
`ProviderBundleIdentifier`	String	Optional. Bundle identifier of the DNS proxy network extension to use. Useful for apps that contain more than one DNS proxy extension.
`ProviderConfiguration`	Dictionary	Optional. Dictionary of vendor-specific configuration items.

Dock Payload

The Dock payload is designated by specifying `com.apple.dock` as the `PayloadType`.

The Dock payload is supported on the user channel and, except for `AllowDockFixupOverride`, on all version of macOS. The key `AllowDockFixupOverride` is supported on macOS 10.12 and later.

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

Key	Type	Value
`orientation`	String	Optional. Orientation of the dock. Values may be `bottom`, `left`, or `right`.
`position-immutable`	Boolean	Optional. If `true`, the position is locked.
`autohide`	Boolean	Optional. If `true`, automatically hide and show the dock.
`autohide-immutable`	Boolean	Optional. If `true`, the Automatically Hide checkbox is disabled.
`minimize-to-application`	Boolean	Optional. If `true`, enable the minimize-to-application feature.
`magnification`	Boolean	Optional. If `true`, magnification is active.
`magnify-immutable`	Boolean	Optional. If `true`, the magnification checkbox is disabled.
`largesize`	Integer	Optional. The size of the largest magnification. Values must be in range 16 to 128.
`magsize-immutable`	Boolean	Optional. If `true`, the magnify slider is disabled.
`show-process-indicators`	Boolean	Optional. If `true`, show the process indicator.
`launchanim`	Boolean	Optional. If `true`, animate opening applications.
`launchanim-immutable`	Boolean	Optional. If `true`, the Animate Opening Applications checkbox is disabled.

Key	Type	Value
mineffect	String	Optional. Set minimize effect. Values may be genie or scale.
mineffect-immutable	Boolean	Optional. If true, the Minimize Using popup is disabled.
tilesize	Integer	Optional. The tile size. Values must be in range 16 to 128.
size-immutable	Boolean	Optional. If true, the size slider will be disabled.
MCXDockSpecialFolders	Array of Strings	Optional. One or more special folders that may be created at user login time and placed in the dock. Values may be AddDockMCXMyApplicationsFolder, AddDockMCXDocumentsFolder, AddDockMCXSharedFolder, or AddDockMCXOriginalNetworkHomeFolder. The "My Applications" item is only used for Simple Finder environments. The "Original Network Home" item is only used for mobile account users.
AllowDockFixupOverride	Boolean	Optional. If true, use the file in /Library/Preferences/com.apple.dockfixup.plist when a new user or migrated user logs in. The format of this file currently has no documentation. This option has no effect for existing users.
static-only	Boolean	Optional. If true, the device will use the static-apps and static-others dictionaries for the dock and ignore any items in the persistent-apps and persistent-others dictionaries. If false, the contents will be merged with the static items listed first.
static-others	Array of Dictionaries	Optional. Dock items in the Documents side that cannot be removed from the dock.
static-apps	Array of Dictionaries	Optional. Dock items in the Applications side that cannot be removed from the dock.
contents-immutable	Boolean	Optional. If true, the user cannot remove any item from or add any item to the dock.

The static-others and static-apps dictionaries define the following keys:

Key	Type	Value
tile-data	Dictionary	Required. Information about a dock item.
tile-type	String	Required. The type of the tile. Values may be file-tile, directory-tile, or url-tile. If you are unsure whether the file item is a file or a directory, set this key to file-tile.

The tile-data dictionary defines the following keys:

Key	Type	Value
label	String	Required. Label of a dock item.
url	String	Optional. For URL tiles, the URL string.
file-type	Integer	Required. The type of the tile expressed as a number. 3 = directory, 0 = URL, 1 = file.

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. This can be any valid UUID. All teacher and student devices that need to communicate with one another must have the same OrganizationUUID, particularly if they originated from different Device Enrollment Programs.
OrganizationName	String	Required. The organization's display name. This name will be shown in the iOS login screen.
PayloadCertificateUUID	String	Required. The UUID of an identity certificate payload that will be used to perform client authentication with other devices.
LeaderPayloadCertificateAnchorUUID	Array	Optional. An array of UUIDs referring to certificate payloads that will be used to authorize leader peer certificate identities. This array must contain all certificates needed to validate the entire chain of trust. Leader certificates must have the common name prefix leader (case insensitive).
MemberPayloadCertificateAnchorUUID	Array	Optional. An array of UUIDs referring to certificate payloads that will be used to authorize group member peer certificate identities. This array must contain all certificates needed to validate the entire chain of trust. Member certificates must have the common name prefix member (case insensitive).
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.

Key	Type	Value
`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. With one-to-one member devices, this key should include only the device user and the leader but not other class members.
`DeviceGroups`	Array	Optional. Leader: An array of dictionaries that define the device groups to which the leader can assign devices. This key is not included in member payloads.
`ScreenObservationPermissionModificationAllowed`	Boolean	Optional. If set to `true`, students enrolled in managed classes can modify their teacher's permissions for screen observation on this device. Defaults to `false`.

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

Key	Type	Value
`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	Value
`BeaconID`	Integer	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	**Deprecated in iOS 9.3.1 and later.** 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.

Key	Type	Value
DeviceGroupIdentifiers	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	Value
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: A string containing a URL pointing to an image of the user. This image will be displayed in the iOS login screen and in the Classroom app. The recommended resolution is 256 x 256 pixels (512 x 512 pixels on a 2x device). The recommended formats are JPEG, PNG, and TIFF. The ResourcePayloadCertificateUUID identity certificate or the MDM client identity will be used to perform authentication when fetching the image.
FullScreenImageURL	String	**Deprecated in iOS 9.3.1 and later.** 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.
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, which must be unique in the organization.
SerialNumbers	Array	Required: strings containing the serial numbers of the devices in the group.

Notes:

- All identities must be configured as both SSL clients and servers.
- Leader certificates must have the common name prefix leader (case insensitive).
- Member certificates must have the common name prefix member (case insensitive).

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
`EmailAccountDescription`	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`, `EmailAuthCRAMMD5`, `EmailAuthNTLM`, `EmailAuthHTTPMD5`, and `EmailAuthNone`.
`IncomingMailServer HostName`	String	Designates the incoming mail server host name (or IP address).
`IncomingMailServer PortNumber`	Integer	Optional. Designates the incoming mail server port number. If no port number is specified, the default port for a given protocol is used.
`IncomingMailServerUseSSL`	Boolean	Optional. Default `false`. Designates whether the incoming mail server uses SSL for authentication.
`IncomingMailServerUsername`	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.
`OutgoingPasswordSameAs IncomingPassword`	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`, `EmailAuthCRAMMD5`, `EmailAuthNTLM`, `EmailAuthHTTPMD5`, and `EmailAuthNone`.
`OutgoingMailServer HostName`	String	Designates the outgoing mail server host name (or IP address).

Key	Type	Value
`OutgoingMailServer PortNumber`	Integer	Optional. Designates the outgoing mail server port number. If no port number is specified, ports 25, 587 and 465 are used, in this order.
`OutgoingMailServerUseSSL`	Boolean	Optional. Default `false`. Designates whether the outgoing mail server uses SSL for authentication.
`OutgoingMailServerUsername`	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.
`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. As of iOS 10.0, this key is ignored. **Availability:** Available only in iOS 5.0 through iOS 9.3.3.
`SMIMESigningEnabled`	Boolean	Optional. Default `true`. If set to `true`, S/MIME signing is enabled for this account. **Availability:** Available only in iOS 10.3 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.
`SMIMEEncryptionEnabled`	Boolean	Optional. Default `false`. If set to `true`, S/MIME encryption is on by default for this account. **Availability:** Available only in iOS 10.3 and later. As of iOS 12.0, this key is deprecated. It is recommended to use `SMIMEEncryptByDefault` instead.
`SMIMEEncryption CertificateUUID`	String	Optional. The `PayloadUUID` of the identity certificate used to decrypt messages sent to this account. The public certificate is attached to outgoing mail to allow encrypted mail to be sent to this user. When the user sends encrypted mail, the public certificate is used to encrypt the copy of the mail in their Sent mailbox. **Availability:** Available only in iOS 5.0 and later.

Key	Type	Value
SMIMEEnablePerMessage Switch	Boolean	Optional. Default false. If set to true, displays the per-message encryption switch in the Mail Compose UI. **Availability:** Available only in iOS 8.0 and later. As of iOS 12.0, this key is deprecated. It is recommended to use SMIMEEnableEncryptionPerMessageSwitch instead.
disableMailRecentsSyncing	Boolean	If true, this account is excluded from address Recents syncing. This defaults to false. **Availability:** Available only in iOS 6.0 and later.
allowMailDrop	Boolean	Optional. If true, this account is allowed to use Mail Drop. The default is false. **Availability:** Available in iOS 9.2 and later.
SMIMESigningUser Overrideable	Boolean	Optional. Default false. If set to true, the user can toggle S/MIME signing on or off in Settings. **Availability:** Available only in iOS 12.0 and later.
SMIMESigningCertificate UUIDUserOverrideable	Boolean	Optional. Default false. If set to true, the user can select the signing identity. **Availability:** Available only in iOS 12.0 and later.
SMIMEEncryptByDefault	Boolean	Optional. Default false. If set to true, S/MIME encryption is enabled by default. If SMIMEEnableEncryptionPerMessageSwitch is false, this default cannot be changed by the user. **Availability:** Available only in iOS 12.0 and later.
SMIMEEncryptByDefault UserOverrideable	Boolean	Optional. Default false. If set to true, the user can toggle the encryption by default setting. **Availability:** Available only in iOS 12.0 and later.
SMIMEEncryptionCertificate UUIDUserOverrideable	Boolean	Optional. Default false. If set to true, the user can select the S/MIME encryption identity and encryption is enabled. **Availability:** Available only in iOS 12.0 and later.
SMIMEEnableEncryptionPer MessageSwitch	Boolean	Optional. Default false. If set to true, display the per-message encryption switch in the Mail Compose UI. **Availability:** Available only in iOS 12.0 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`
- `com.apple.globalethernet.managed`

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

Key	Type	Value
`Interface`	String	The `com.apple.globalethernet.managed` payload uses the value `AnyEthernet`. The values for the other payloads are derived from their name; for example the `com.apple.firstethernet.managed` value would be `FirstEthernet`.

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.

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 Contacts account on the device. Mail and Calendar are not configured using this payload on iOS.

In macOS, 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 macOS		
`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 macOS, this key is required.
`Host`	String	Specifies the Exchange server host name (or IP address). In macOS 10.11 and later, this key is optional.
`SSL`	Boolean	Optional. Default YES. Specifies whether the Exchange server uses SSL for authentication.
`UserName`	String	This string specifies the user name for this Exchange account. If missing, the devices prompts for it during profile installation. In macOS, this key is required.
`Password`	String	Optional. The password of the account. Use only with encrypted profiles.
`OAuth`	Boolean	Optional. Specifies whether the connection should use OAuth for authentication. If enabled, a password should not be specified. This defaults to `false`. **Availability:** Available only in iOS 12.0 and macOS 10.14 and later.
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.

Key	Type	Value
`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.
`PayloadCertificateUUID`	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 `true`, this account supports S/MIME. As of iOS 10.0, this key is ignored. **Availability:** Available only in iOS 5.0 through 9.3.3.
`SMIMESigningEnabled`	Boolean	Optional. Default `true`. If set to `true`, S/MIME signing is enabled for this account. **Availability:** Available only in iOS 10.3 and later.
`SMIMESigningCertificateUUID`	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.
`SMIMEEncryptionEnabled`	Boolean	Optional. Default `false`. If set to `true`, S/MIME encryption is on by default for this account. **Availability:** Available only in iOS 10.3 and later. As of iOS 12.0, this key is deprecated. It is recommended to use `SMIMEEncryptByDefault` instead.
`SMIMEEncryption CertificateUUID`	String	Optional. The `PayloadUUID` of the identity certificate used to decrypt messages sent to this account. The public certificate is attached to outgoing mail to allow encrypted mail to be sent to this user. When the user sends encrypted mail, the public certificate is used to encrypt the copy of the mail in their Sent mailbox. **Availability:** Available only in iOS 5.0 and later.
`SMIMEEnablePerMessageSwitch`	Boolean	Optional. Default `false`. If set to `true`, displays the per-message encryption switch in the Mail Compose UI. **Availability:** Available only in iOS 8.0 and later. As of iOS 12.0, this key is deprecated. It is recommended to use `SMIMEEnableEncryptionPerMessageSwitch` instead.
`SMIMESigningUserOverrideable`	Boolean	Optional. Default `false`. If set to `true`, the user can toggle S/MIME signing on or off in Settings. **Availability:** Available only in iOS 12.0 and later.
`SMIMESigningCertificateUUID UserOverrideable`	Boolean	Optional. Default `false`. If set to `true`, the user can select the signing identity. **Availability:** Available only in iOS 12.0 and later.

Key	Type	Value
SMIMEEncryptByDefault	Boolean	Optional. Default false. If set to true, S/MIME encryption is enabled by default. If SMIMEEnableEncryptionPerMessageSwitch is false, this default cannot be changed by the user. **Availability:** Available only in iOS 12.0 and later.
SMIMEEncryptByDefaultUserOverrideable	Boolean	Optional. Default false. If set to true, the user can toggle the encryption by default setting. **Availability:** Available only in iOS 12.0 and later.
SMIMEEncryptionCertificateUUIDUserOverrideable	Boolean	Optional. Default false. If set to true, the user can select the S/MIME encryption identity and encryption is enabled. **Availability:** Available only in iOS 12.0 and later.
SMIMEEnableEncryptionPerMessageSwitch	Boolean	Optional. Default false. If set to true, displays the per-message encryption switch in the Mail Compose UI. **Availability:** Available only in iOS 12.0 and later.
disableMailRecentsSyncing	Boolean	If true, this account is excluded from address Recents syncing. This defaults to false. **Availability:** Available only in iOS 6.0 and later.
MailNumberOfPastDaysToSync	Integer	The number of days since synchronization.
CommunicationServiceRules	Dictionary	Optional. The communication service handler rules for this account. The CommunicationServiceRules dictionary currently contains only a DefaultServiceHandlers key; its value is a dictionary which contains an AudioCall key whose value is a string containing the bundle identifier for the default application that handles audio calls made to contacts from this account.
Available in macOS Only		
Path	String	Optional.
Port	Integer	Optional.
ExternalHost	String	Optional.
ExternalSSL	Boolean	Optional.
ExternalPath	String	Optional.
ExternalPort	Integer	Optional.
OAuthSignInURL	String	Optional. Specifies the URL to load into a webview for authentication via OAuth when auto-discovery is not used. Requires a Host value. **Availability:** Available only in macOS 10.14 and later.

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 macOS 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.
`DeferForceAtUserLogin MaxBypassAttempts`	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 macOS 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 macOS 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.

FDE Recovery Key Escrow Payload

FileVault Full Disk Encryption (FDE) recovery keys are, by default, sent to Apple if the user requests it. Starting with macOS 10.13, recovery key escrow payloads are designated by specifying com.apple.security.FDERecoveryKeyEscrow as the PayloadType value. Only one payload of this type is allowed per system.

If FileVault is enabled after this payload is installed on the system, the FileVault PRK will be encrypted with the specified certificate, wrapped with a CMS envelope and stored at /var/db/FileVaultPRK.dat. The encrypted data will be made available to the MDM server as part of the SecurityInfo command. Alternatively, if a site uses its own administration software, it can extract the PRK from the foregoing location at any time. Because the PRK is encrypted using the certificate provided in the profile, only the author of the profile can extract the data.

Note these cautions:

- The payload must exist in a system-scoped profile.
- Installing more than one payload of this type per machine will cause an error.
- The previous payload (com.apple.security.FDERecoveryRedirect) is no longer supported. It can still be installed, but it will be ignored. This lets servers send out the same profile to old and new clients.
- If only an old-style redirection payload is installed at the time FileVault is turned on (by means of the Security Preferences pane), an error will be displayed and FileVault will not be enabled.
- No warning or error will be provided if FileVault is already enabled and an old-style payload is installed. In this case, it's assumed that the recovery key has already been escrowed with the server.

This payload contains these keys:

Key	Type	Value
Location	String	Required. A short description of the location where the recovery key will be escrowed. This text will be inserted into the message the user sees when enabling FileVault.
EncryptCertPayloadUUID	String	Required. The UUID of a payload within the same profile that contains the certificate that will be used to encrypt the recovery key. The referenced payload must be of type com.apple.security.pkcs1.
DeviceKey	String	Optional. An optional string that will be included in help text if the user appears to have forgotten the password. Can be used by a site admin to look up the escrowed key for the particular machine. Replaces the RecordNumber key used in previous escrow mechanism. If missing, the device serial number will be used instead.

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`	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.

Firewall Payload

Available in macOS 10.12 and later. A Firewall payload manages the Application Firewall settings accessible in the Security Preferences pane. Note these restrictions:

- The payload must exist in a system-scoped profile.
- If more than one profile contains this payload, the most restrictive union of settings will be used.
- The "Automatically allow signed downloaded software" and "Automatically allow built-in software" options are not supported, but both will be forced ON when this payload is present.

This payload is designated by specifying `com.apple.security.firewall` as the `PayloadType` value.

The Firewall payload contains the following keys:

Key	Type	Value
`EnableFirewall`	Boolean	Required. Whether the firewall should be enabled or not.
`BlockAllIncoming`	Boolean	Optional. Corresponds to the "Block all incoming connections" option.
`EnableStealthMode`	Boolean	Optional. Corresponds to "Enable stealth mode."
`Applications`	Array of Dictionaries	Optional. The list of applications. Each dictionary contains these keys: • `BundleID` (string) : identifies the application • `Allowed` (Boolean) : specifies whether or not incoming connections are allowed

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.

> **Note**
>
> 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.
`ProxyServerPort`	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.
`ProxyPACFallbackAllowed`	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.
`ProxyCaptiveLoginAllowed`	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.

Google Account Payload

The Google account payload is designated by specifying `com.apple.google-oauth` as the `PayloadType` value. You can install multiple Google payloads.

Each Google payload sets up a Google email address as well as any other Google services the user enables after authentication. Google accounts must be installed via MDM or by Apple Configurator 2 (if the device is supervised). The payload never contains credentials; the user will be prompted to enter credentials shortly after the payload has been successfully installed.

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

Key	Type	Value
`AccountDescription`	String	Optional. A user-visible description of the Google account, shown in the Mail and Settings apps. **Availability:** Available in iOS 9.3 and later.
`AccountName`	String	Optional. The user's full name for the Google account. This name will appear in sent messages. **Availability:** Available in iOS 9.3 and later.
`EmailAddress`	String	Required. The full Google email address for the account. **Availability:** Available in iOS 9.3 and later.
`CommunicationServiceRules`	Dictionary	Optional. The communication service handler rules for this account. The `CommunicationServiceRules` dictionary currently contains only a `DefaultServiceHandlers` key; its value is a dictionary which contains an `AudioCall` key whose value is a string containing the bundle identifier for the default application that handles audio calls made to contacts from this account. **Availability:** Available in iOS 10 and later.

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.

> **Note**
>
> If a home screen layout puts more than six items in the iPad dock the location of the seventh and succeeding items may be undefined but they will not be omitted.

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. If it is not present, the user's dock will be empty.
`Pages`	Array	Required. An array of arrays 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. Valid only if `Folder` type.
`BundleID`	String	Required if `Application` type. The bundle identifier of the app.
`Pages`	Array	Optional. An array of arrays of dictionaries, each of which must conform to the icon dictionary format. Valid only if `Folder` type.
`URL`	String	Required if `WebClip` type. URL of the WebClip being referenced. If more than one WebClip exists with the same URL, the behavior is undefined. **Availability:** Available in iOS 11.3 and later.

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.

Identity Preference Payload

Available in macOS 10.12 and later. An Identity Preference payload lets you identify an Identity Preference item in the user's keychain that references a identity payload included in the same profile. It can only appear in a user profile, not a device profile. See also Certificate Preference Payload for setting up certificate preferences.

You can include multiple Identity Preference payloads as needed. Identity Preference payloads are designated by specifying `com.apple.security.identitypreference` as the `PayloadType` value.

An Identity Preference payload contains the following keys:

Key	Type	Value
`Name`	String	Required. An email address (RFC822), DNS hostname, or other name that uniquely identifies a service requiring this identity.
`PayloadCertificateUUID`	String	The UUID of another payload within the same profile that installed the identity; for example, a 'com.apple.security.pkcs12' or 'com.apple.security.scep' payload.

Kernel Extension Policy

The Kernel Extension Policy payload is designated by specifying `com.apple.syspolicy.kernel-extension-policy` as the `PayloadType` value. It is supported on macOS 10.13.2 and later.

This profile must be delivered via a user approved MDM server.

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

Key	Type	Value
`AllowUserOverrides`	Boolean	If set to `true`, users can approve additional kernel extensions not explicitly allowed by configuration profiles.
`AllowedTeamIdentifiers`	Array of Strings	An array of team identifiers that define which validly signed kernel extensions will be allowed to load.
`AllowedKernelExtensions`	Dictionary	A dictionary representing a set of kernel extensions that will always be allowed to load on the machine. The dictionary maps team identifiers (keys) to arrays of bundle identifiers.

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
`LDAPAccountDescription`	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.
`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).
`LDAPSearchSettingDescription`	String	Optional. Description of this search setting.
`LDAPSearchSettingSearchBase`	String	Conceptually, the path to the node where a search should start. For example: `ou=people,o=example corp`
`LDAPSearchSettingScope`	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.

Loginwindow Payload

The Loginwindow payload is designated by specifying `com.apple.loginwindow` as the `PayloadType` value.

This payload creates managed preferences on all versions of macOS for system and device profiles. Multiple Loginwindow payloads may be installed together.

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

Key	Type	Value
`SHOWFULLNAME`	Boolean	Optional. Set to `true` to show the name and password dialog. Set to `false` to display a list of users.
`HideLocalUsers`	Boolean	Optional. When showing a user list, set to `true` to show only network and system users.
`IncludeNetworkUser`	Boolean	Optional. When showing a user list, set to `true` to show network users.
`HideAdminUsers`	Boolean	Optional. When showing a user list, set to `false` to hide the administrator users.
`SHOWOTHERUSERS_MANAGED`	Boolean	Optional. When showing a list of users, set to `true` to display Other... users.
`AdminHostInfo`	String	Optional. If this key is included in the payload, its value will be displayed as additional computer information on the login window. Before macOS 10.10, this string could contain only particular information (`HostName`, `SystemVersion`, or `IPAddress`). After macOS 10.10, setting this key to any value will allow the user to click the "time" area of the menu bar to toggle through various computer information values.
`AllowList`	Array of Strings	Optional. User or group GUIDs of users that are allowed to log in. An asterisk '*' string specifies all users or groups.
`DenyList`	Array of Strings	Optional. User or group GUIDs of users that cannot log in. This list takes priority over the list in the `AllowList` key.
`HideMobileAccounts`	Boolean	Optional. If set to `true`, mobile account users will not be visible in a user list. In some cases mobile users will show up as network users.
`ShutDownDisabled`	Boolean	Optional. If set to `true`, the Shut Down button item will be hidden.
`RestartDisabled`	Boolean	Optional. If set to `true`, the Restart item will be hidden.
`SleepDisabled`	Boolean	Optional. If set to `true`, the Sleep button item will be hidden.
`DisableConsoleAccess`	Boolean	Optional. If set to `true`, the Other user will disregard use of the '>console' special user name.
`LoginwindowText`	String	Optional. Text to display in the login window.
`ShutDownDisabledWhileLoggedIn`	Boolean	Optional. If set to `true`, the Shut Down menu item will be disabled when the user is logged in.
`RestartDisabledWhileLoggedIn`	Boolean	Optional. If set to `true`, the Restart menu item will be disabled when the user is logged in.

Key	Type	Value
`PowerOffDisabledWhileLoggedIn`	Boolean	Optional. If set to `true`, the Power Off menu item will be disabled when the user is logged in.
`LogOutDisabledWhileLoggedIn`	Boolean	Optional. If set to `true`, this will disable the Log Out menu item when the user is logged in. **Availability:** Available in macOS 10.13 and later.
`DisableScreenLockImmediate`	Boolean	Optional. If set to `true`, the immediate Screen Lock functions will be disabled. **Availability:** Available in macOS 10.13 and later.

An older, separate, Loginwindow payload also exists and is designated by specifying `loginwindow` as the `PayloadType` value.

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

Key	Type	Value
`DisableLoginItemsSuppression`	Boolean	Optional. If set to `true`, the user is prevented from disabling login item launching using the Shift key.

Media Management

The profile configuration keys for media management are of two kinds: those that restrict disc burning and those that restrict media mounting and ejection. All keys are available on all versions of macOS and are supported on the user channel.

Disc Burning Payloads

Disc burning restrictions require both Disc Burning and Finder payloads.

The Disc Burning payload is designated by specifying `com.apple.DiscRecording` as the `PayloadType` value.

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

Key	Type	Value
`BurnSupport`	String	Required. Set to `off` to disable disc burning. Set to `on` for normal default operation. Set to `authenticate` to require authentication. Setting this key to `on` will not enable disc burn support if it has already been disabled by other mechanisms or preferences.

The Finder payload is designated by specifying `com.apple.finder` as the `PayloadType` value.

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

Key	Type	Value
`ProhibitBurn`	Boolean	Required. Set to `false` to enable the Finder's burn support. Set to `true` to disable the Finder's burn support.

Allowed Media Payload

The Allowed Media payload is designated by specifying `com.apple.systemuiserver` as the `PayloadType` value.

This payload defines these keys:

Key	Type	Value
`logout-eject`	Dictionary	Optional. Media type dictionary to define volumes to eject when the user logs out.
`mount-controls`	Dictionary	Optional. Media type dictionary to control volume mounting.
`unmount-controls`	Dictionary	Optional. Media type dictionary to control volume unmounting.

The Media type dictionaries can contain the following keys. Not all dictionaries use all keys. Values for media action strings are given in the next table.

Key (media type)	Type	Value
`all-media`	String	Optional. Unused; set to empty string.
`cd`	String or Array of Strings	Optional. Media action string(s).
`dvd`	String or Array of Strings	Optional. Media action string(s).
`bd`	String or Array of Strings	Optional. Media action string(s).
`blankcd`	String or Array of Strings	Optional. Media action string(s).
`blankdvd`	String or Array of Strings	Optional. Media action string(s).
`blankbd`	String or Array of Strings	Optional. Media action string(s).
`dvdram`	String or Array of Strings	Optional. Media action string(s).
`disk-image`	String or Array of Strings	Optional. Media action string(s).
`harddisk-internal`	String or Array of Strings	Optional. Media action string(s).
`networkdisk`	String or Array of Strings	Optional. Media action string(s).
`harddisk-external`	String or Array of Strings	Optional. Media action string(s). Internally installed SD-Cards and USB flash drives are included in the `harddisk-external` category. This key is the default for media types that don't fall into other categories.

Media action strings are described below. You can combine some strings in arrays to create custom actions.

Key	Type	Value
`authenticate`	Boolean	Optional. The user will be authenticated before the media is mounted.
`read-only`	Boolean	Optional. The media will be mounted as read-only; this action cannot be combined with unmount controls.
`deny`	Boolean	Optional. The media will not be mounted.
`eject`	Boolean	Optional. The media will not be mounted and it will be ejected if possible. Note that some volumes are not defined as ejectable, so using the deny key may be the best solution. This action cannot be combined with unmount controls.

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 Dictionaries	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.*`.
`AllowRoamingCellularData`	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 must be installed on supervised devices. 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`.
`ShowInNotificationCenter`	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 (default) • 2: Modal Alert
`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`.
`ShowInCarPlay`	Boolean	Optional. Whether notifications can be shown in CarPlay. Default is `true`. **Availability:** Available in iOS 12 and later.
`GroupingType`	Integer	Optional. The type of grouping for notifications for this app: • 0: Automatic - group notifications into app-specified groups. (Default) • 1: By app - group notifications into one group. • 2: Off - do not group notifications. **Availability:** Available in iOS 12 and later.
`CriticalAlertEnabled`	Boolean	Optional. Whether an app can mark a notification as a critical notification that will ignore Do Not Disturb and ringer settings. Default is `false`. **Availability:** Available in iOS 12 and later.

NSExtension Management

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

This payload specifies which NSExtensions are allowed or disallowed on a system. Extensions can be managed by bundleID in whitelists and blacklists or by a blacklist of extension points.

It is supported on macOS 10.13 and later.

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

Key	Type	Value
`AllowedExtensions`	Array	Optional. Array of extension identifiers for extensions that are allowed to run on the system.
`DeniedExtensions`	Array	Optional. Array of extension identifiers for extensions that are not allowed to run on the system.
`DeniedExtensionPoints`	Array	Optional. Array of NSExtension extension points for extensions that are not allowed to run on the system.

If an array element within `DeniedExtensionPoints` is "AllPublicExtensionPoints", `DeniedExtensionPoints` will be filled with a list of extension points that the client considers to be "public". These are the extension points referenced in developer documentation and supported by the Xcode programming environment.

Expansion of "AllPublicExtensionPoints" happens at evaluation time. The list of extension points may change from release to release.

This feature is intended as a way to specify "Start with no extensions belonging to any public extension points enabled but then allow only extensions A, B, C to run". Specifying "AllPublicExtensionPoints" will disallow both Apple and third-party extensions within the "public" extension points but will still allow extensions belonging to system-critical extension points to execute.

Parental Controls Payload

Parental Control on macOS consists of many different payloads which are set individually depending on the type of control required. Parental control payloads are supported on the user channel. Each payload and its respective keys are described in the sections below.

Parental Control Web Content Filter Payload

The Parental Control Web Content Filter payload is designated by specifying `com.apple.familycontrols.contentfilter` as the `PayloadType` value.

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

Key	Type	Value
`restrictWeb`	Boolean	Required. Set to `true` to enable Web content filters.
`useContentFilter`	Boolean	Optional. Set to `true` to try to automatically filter content.

Key	Type	Value
`whiteListEnabled`	Boolean	Optional. Set to `true` to use the filterWhiteList and filterBlackList lists.
`siteWhiteList`	Array of Dictionaries	Required if `whiteListEnabled` is `true`. If specified, this key contains an array of dictionaries (see below) that define additional allowed sites besides those in the automated list of allowed and unallowed sites, including disallowed adult sites.
`filterWhiteList`	Array of URL Strings	Optional. If specified and `restrictWeb` is `true`, qn array of URLs designating the only allowed Websites.
`filterBlackList`	Array of URL Strings	Optional. If specified and `restrictWeb` is `true`, an array of URLs of Websites never to be allowed.

Each `siteWhiteList` dictionary contains these keys:

Key	Type	Value
`address`	String	Required. Site prefix, including `http(s)` scheme.
`pageTitle`	String	Optional. Site page title.

Parental Control Time Limits Payload

The Parental Control Time Limits payload is designated by specifying `com.apple.familycontrols.timelimits.v2` as the `PayloadType` value.

It consists of a dictionary containing a master enabled flag plus a dictionary of time limit specification keys.

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

Key	Type	Value
`familyControlsEnabled`	Boolean	Required. Set to `true` to use time limits.
`time-limits`	Dictionary	Required if `familyControlsEnabled` is `true`. Time limits settings.

Each `time-limits` dictionary contains these keys:

Key	Type	Value
`weekday-allowance`	Dictionary	Optional. Weekday allowance settings.
`weekday-curfew`	Dictionary	Optional. Weekday curfew settings.
`weekend-allowance`	Dictionary	Optional. Weekend allowance settings.
`weekend-curfew`	Dictionary	Optional. Weekend curfew settings.

Each `allowance` or `curfew` dictionary contains these keys:

Key	Type	Value
`enabled`	Boolean	Required. Set to `true` to enable these settings.

Key	Type	Value
rangeType	Integer	Required. Type of day range: 0 = weekday, 1 = weekend.
start	String	Optional. Curfew start time in the format %d:%d:%d.
end	String	Optional. Curfew end time in the format %d:%d:%d.
secondsPerDay	Integer	Optional. Seconds for that day for allowance.

Parental Control Application Access Payload

The Parental Control Application Access payload is designated by specifying com.apple.applicationaccess.new as the PayloadType value.

It enables application access restrictions on macOS.

To determine if an application can be launched, these rules are evaluated:

1. Certain system applications and utilities are always allowed to run.
2. The whiteList is searched to see if a matching entry is found by bundleID. If a match is found, the appID and detachedSignature (if present) are used to verify the signature of the application being launched. If the signature is valid and matches the designated requirement (in the appID key), the application is allowed to launch.
3. If the path to the binary being launched matches (or is in a subdirectory) of a path in pathBlackList, the binary is denied.
4. If the path to the binary being launched matches (or is a subdirectory) of a path in pathWhiteList, the binary is allowed to launch.
5. The binary is denied permission to launch.

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

Key	Type	Value
familyControlsEnabled	Boolean	Required. Set to true to enable application access restrictions.
whiteList	Array of Dictionaries	Optional. A list of code signatures for applications that are allowed to run.
pathBlackList	Array of Strings	Optional. Paths to disallowed applications.
pathWhiteList	Array of Strings	Optional. Paths to allowed applications.

Each whiteList dictionary contains these keys:

Key	Type	Value
bundleID	String	Required. Bundle ID of application.
appID	Data	Required. The designated requirement describing the code signature of this executable. This value is obtained from the Security.framework using SecCodeCopyDesignatedRequirement.

Key	Type	Value
`detachedSignature`	Data	Optional. Can be used to provide the required signature for an unsigned binary. Generate an ad-hoc signature of the unsigned binary and store the signature here.
`disabled`	Boolean	Optional. Specifies whether this application information is to be included in the `whiteList` or not. Set to `true` to keep the application off the `whiteList`. It could still be allowed to launch via `pathWhiteList`, although this behavior is discouraged. Default is `false`.
`subApps`	Array of Dictionaries	Optional. For applications that include nested helper applications, describes the signatures of embedded applications. The dictionary format is the same as for the `whiteList` key.
`displayName`	String	Optional. Display name.

Parental Control Dashboard Payload

The Parental Control Dashboard payload is designated by specifying `com.apple.dashboard` as the `PayloadType` value.

It is used to define a white list of dashboard widgets.

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

Key	Type	Value
`whiteListEnabled`	Boolean	Required. Set to `true` to enable the widget white list items.
`whiteList`	Array of Dictionaries	Required. List that defines Dashboard widgets.

Each widget `whiteList` dictionary contains these keys:

Key	Type	Value
`Type`	String	Required. Set to `bundleID` to use a widget's bundle ID as its ID.
`ID`	String	Required. The bundle ID of a widget.

Parental Control Dictionary Payload

The Parental Control Dictionary payload is designated by specifying `com.apple.Dictionary` as the `PayloadType` value.

It enables the restrictions defined in the device's Parental Controls Dictionary.

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

Key	Type	Value
`parentalControl`	Boolean	Required. Set to `true` to enable parental controls dictionary restrictions.

Parental Control Dictation and Profanity Payload

The Parental Control Dictation and Profanity payload is designated by specifying `com.apple.ironwood.support` as the `PayloadType` value.

It disables dictation and suppresses profanity on the device.

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

Key	Type	Value
`IronwoodAllowed`	Boolean	Optional. Set to `false` to disable dictation.
`ProfanityAllowed`	Boolean	Optional. Set to `false` to suppress profanity.

Parental Control Game Center Payload

The Parental Control Game Center payload is designated by specifying `com.apple.gamed` as the `PayloadType` value.

It restricts Game Center options on the device.

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

Key	Type	Value
`GKFeatureGameCenterAllowed`	Boolean	Optional. Set to `false` to disable Game Center.
`GKFeatureAccountModificationAllowed`	Boolean	Optional. Set to `false` to disable account modifications.
`GKFeatureAddingGameCenterFriendsAllowed`	Boolean	Optional. Set to `false` to disable adding Game Center friends.
`GKFeatureMultiplayerGamingAllowed`	Boolean	Optional. Set to `false` to disable multiplayer gaming.

Additional Parental Controls

Additional parental control functions can be found in the following payloads:

- System Policy Control Payload
- Email Payload
- Media Management
- AppStore Payload
- Dock Payload

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 macOS 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.
`maxFailedAttempts`	Integer	Optional. Default 11. Allowed range [2...11]. Specifies the number of allowed failed attempts to enter the passcode at the device's lock screen. After six failed attempts, there is a time delay imposed before a passcode can be entered again. The delay increases with each attempt. Once this number is exceeded, on macOS the device is locked and on iOS the device is wiped.
`maxInactivity`	Integer	Optional. Default Infinity. Specifies the maximum 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. The user can edit this setting, but the value cannot exceed the `maxInactivity` value. In macOS, this will be translated to screensaver settings.
`maxPINAgeInDays`	Integer	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`	Integer	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`	Integer	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`	Integer	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.

Key	Type	Value
`maxGracePeriod`	Integer	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 macOS, this will be translated to screensaver settings.
`allowFingerprint Modification`	Boolean	Optional. Supervised only. Not supported on macOS. Allows the user to modify Touch ID. Default NO.
`changeAtNextAuth`	Boolean	Optional. On macOS, setting this to `true` will cause a password reset to occur the next time the user tries to authenticate. If this key is set in a device profile, the setting takes effect for all users, and admin authentications may fail until the admin user password is also reset. **Availability:** Available in macOS 10.13 and later.

Privacy Preferences Policy Control Payload

The Privacy Preferences payload is designated by specifying `com.apple.TCC.configuration-profile-policy` value as the `PayloadType` value.

It controls the settings that are displayed in the "Privacy" tab of the "Security & Privacy" pane in System Preferences.

This profile must be delivered via a user approved MDM server.

Availability: Available in macOS 10.14 and later.

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

Key	Type	Value
`Services`	Dictionary	Keys are limited to the privacy service names listed below. Each key is an array of dictionaries describing the app or process to which access is given. In the case of conflicting specifications, the most restrictive setting (deny) will be used.

Privacy Service Dictionary Keys

Key	Type	Value
`AddressBook`	Array of Identity Dictionaries	Optional. Contact information managed by Contacts.app.
`Calendar`	Array of Identity Dictionaries	Optional. Calendar information managed by Calendar.app.
`Reminders`	Array of Identity Dictionaries	Optional. Reminders information managed by Reminders.app.
`Photos`	Array of Identity Dictionaries	Optional. Pictures managed by Photos.app in `~/Pictures/.photoslibrary`.

Key	Type	Value
`Camera`	Array of Identity Dictionaries	Optional. A system camera. Access to the camera cannot be given in a profile; it can only be denied.
`Microphone`	Array of Identity Dictionaries	Optional. A system microphone. Access to the microphone cannot be given in a profile; it can only be denied.
`Accessibility`	Array of Identity Dictionaries	Optional. Control the application via the Accessibility subsystem.
`PostEvent`	Array of Identity Dictionaries	Optional. Allows the application to use CoreGraphics APIs to send CGEvents to the system event stream.
`SystemPolicyAllFiles`	Array of Identity Dictionaries	Optional. Allows the application access to all protected files.
`SystemPolicySysAdminFiles`	Array of Identity Dictionaries	Optional. Allows the application access to some files used in system administration.
`AppleEvents`	Array of Identity Dictionaries	Optional. Allows the application to send a restricted `AppleEvent` to another process.

Identity Dictionary Keys

Key	Type	Value
`Identifier`	String	The bundle ID or installation path of the binary.
`IdentifierType`	String	The type of `Identifier` value. Must be either `bundleID` or `path`. Application bundles should be identified by bundle ID. Non-bundled binaries must by identified by installation path. Helper tools embedded within an application bundle will automatically inherit the permissions of their enclosing app bundle.
`CodeRequirement`	String	Obtained via the command "`codesign – display –r –`".
`StaticCode`	Boolean	Optional. If set to `true`, statically validate the code requirement. Used only if the process invalidates its dynamic code signature. Defaults to `false`.
`Allowed`	Boolean	If set to `true`, access is granted. Otherwise the process does not have access. The user is not prompted and cannot change this value.
`AEReceiverIdentifier`	String	Optional. The identifier of the process receiving an `ApplEvent` sent by the `Identifier` process. Required for `AppleEvents` service; not valid for other services.
`AEReceiverIdentifierType`	String	Optional. The type of `AEReceiverIdentifier` value. Must be either `bundleID` or `path`. Required for `AppleEvents` service; not valid for other services.

Key	Type	Value
AEReceiverCodeRequirement	String	Optional. Code requirement for the receiving binary. Required for AppleEvents service; not valid for other services.
Comment	String	Not used.

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 macOS, 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.
`allowAddingGameCenter Friends`	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.
`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.
`allowAssistantWhile Locked`	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`, Apple Books will be disabled. This will default to `true`. **Availability:** Available in iOS 6.0 and later.

Key	Type	Value
`allowBookstoreErotica`	Boolean	Optional. Supervised only prior to iOS 6.1. If set to `false`, the user will not be able to download media from Apple Books that has been tagged as erotica. This will default to `true`. **Availability:** Available in iOS and in tvOS 11.3 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 macOS 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.
`allowCloudBackup`	Boolean	Optional. When `false`, disables backing up the device to iCloud. **Availability:** Available in iOS 5.0 and later.
`allowCloudBookmarks`	Boolean	Optional. When `false`, disallows macOS iCloud Bookmark sync. **Availability:** Available in macOS 10.12 and later.
`allowCloudMail`	Boolean	Optional. When `false`, disallows macOS Mail iCloud services. **Availability:** Available in macOS 10.12 and later.
`allowCloudCalendar`	Boolean	Optional. When `false`, disallows macOS iCloud Calendar services. **Availability:** Available in macOS 10.12 and later.
`allowCloudReminders`	Boolean	Optional. When `false`, disallows iCloud Reminder services. **Availability:** Available in macOS 10.12 and later.
`allowCloudAddressBook`	Boolean	Optional. When `false`, disallows macOS iCloud Address Book services. **Availability:** Available in macOS 10.12 and later.
`allowCloudNotes`	Boolean	Optional. When `false`, disallows macOS iCloud Notes services. **Availability:** Available in macOS 10.12 and later.
`allowCloudDocumentSync`	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 macOS 10.11 and later.
`allowCloudKeychainSync`	Boolean	Optional. When `false`, disables iCloud keychain synchronization. Default is `true`. **Availability:** Available in iOS 7.0 and later and macOS 10.12 and later.
`allowContentCaching`	Boolean	Optional. When `false`, this disallows content caching. Defaults to `true`. **Availability:** Available only in macOS 10.13 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.

Key	Type	Value
`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. **Availability:** Available in iOS and in tvOS 11.3 and later.
`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.
`allowFingerprintFor Unlock`	Boolean	Optional. If `false`, prevents Touch ID from unlocking a device. **Availability:** Available in iOS 7 and later and in macOS 10.12.4 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.
`allowGlobalBackground FetchWhenRoaming`	Boolean	Optional. When `false`, disables global background fetch activity when an iOS phone is roaming.
`allowInAppPurchases`	Boolean	Optional. When `false`, prohibits in-app purchasing.
`allowLockScreenControl Center`	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 and users can't receive notifications when the screen is locked. **Availability:** Available only in iOS 7.0 and later.
`allowLockScreenToday View`	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.
`allowMultiplayerGaming`	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.
`allowOpenFromUnmanaged ToManaged`	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.

Key	Type	Value
`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.
`allowPassbookWhile Locked`	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.
`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`.
`safariForceFraudWarning`	Boolean	Optional. When `true`, Safari fraud warning is enabled. Defaults to `false`.
`safariAllowJavaScript`	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`	Real	Optional. Determines conditions under which the device will accept cookies. The user facing settings changed in iOS 11, though the possible values remain the same: • 0: Prevent Cross-Site Tracking and Block All Cookies are enabled and the user can't disable either setting. • 1 or 1.5: Prevent Cross-Site Tracking is enabled and the user can't disable it. Block All Cookies is not enabled, though the user can enable it. • 2: Prevent Cross-Site Tracking is enabled and Block All Cookies is not enabled. The user can toggle either setting. (Default) These are the allowed values and settings in iOS 10 and earlier: • 0: Never • 1: Allow from current website only • 1.5: Allow from websites visited (Available in iOS 8.0 and later); enter `'<real>1.5</real>'` • 2: Always (Default) In iOS 10 and earlier, users can always pick an option that is more restrictive than the payload policy, but not a less restrictive policy. For example, with a payload value of 1.5, a user could switch to *Never*, but not *Always Allow*.

Key	Type	Value
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.
allowUntrustedTLSPrompt	Boolean	Optional. When false, automatically rejects untrusted HTTPS certificates without prompting the user. **Availability:** Available in iOS 5.0 and later.
allowVideoConferencing	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.
allowiTunesFileSharing	Boolean	Optional. When false, iTunes application file sharing services are disabled. **Availability:** Available in macOS 10.13 and later.
autonomousSingleAppMode PermittedAppIDs	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.
forceAssistantProfanity Filter	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 RequestsPairingPassword	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 RequestsPairingPassword`	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 to tvOS 10.1. It is recommended to use the AirPlay Security Payload.
`allowManagedAppsCloud Sync`	Boolean	Optional. If set to `false`, prevents managed applications from using iCloud sync.
`allowEraseContentAnd Settings`	Boolean	Supervised only. If set to `false`, disables the "Erase All Content And Settings" option in the Reset UI.
`allowSpotlightInternet Results`	Boolean	Supervised only. If set to `false`, Spotlight will not return Internet search results. **Availability:** Available in iOS and in macOS 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`.
`allowEnterpriseBook Backup`	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.
`allowDefinitionLookup`	Boolean	Supervised only. If set to `false`, disables definition lookup. Defaults to `true`. **Availability:** Available in iOS 8.1.3 and later and in macOS 10.11.2 and later.
`allowPredictiveKeyboard`	Boolean	Supervised only. If set to `false`, disables predictive keyboards. Defaults to `true`. **Availability:** Available in iOS 8.1.3 and later.
`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 9.3 and later and macOS 10.12 and later.

Key	Type	Value
`allowCloudPhotoLibrary`	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 and in macOS 10.12 and later.
`allowNews`	Boolean	Supervised only. If set to `false`, disables News. Defaults to `true`. **Availability:** Available in iOS 9.0 and later.
`forceAirDropUnmanaged`	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.
`allowUIAppInstallation`	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; it also prevents the Classroom app from observing remote screens. Defaults to `true`. **Availability:** Updated in iOS 9.0 to include screen recordings.
`allowKeyboardShortcuts`	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 the device passcode from being added, changed, or removed. Defaults to `true`. This restriction is ignored by shared iPads. **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.
`allowAutomaticApp Downloads`	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.

Key	Type	Value
allowEnterpriseAppTrust	Boolean	If set to false removes the Trust Enterprise Developer button in Settings->General->Profiles & Device Management, preventing apps from being provisioned by universal provisioning profiles. This restriction applies to free developer accounts but it does not apply to enterprise app developers who are trusted because their apps were pushed via MDM, nor does it revoke previously granted trust. Defaults to true. **Availability:** Available in iOS 9.0 and later.
allowRadioService	Boolean	Supervised only. If set to false, Apple Music Radio is disabled. Defaults to true. **Availability:** Available in iOS 9.3 and later.
blacklistedAppBundleIDs	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.
whitelistedAppBundleIDs	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.
allowRemoteScreen Observation	Boolean	If set to false, remote screen observation by the Classroom app is disabled. Defaults to true. This key should be nested beneath allowScreenShot as a sub-restriction. If allowScreenShot is set to false, it also prevents the Classroom app from observing remote screens. **Availability:** Available in iOS 9.3 and later.
allowDiagnostic SubmissionModification	Boolean	Supervised only. If set to false, the diagnostic submission and app analytics settings in the Diagnostics & Usage pane in Settings cannot be modified. Defaults to true. **Availability:** Available in iOS 9.3.2 and later.
allowBluetooth Modification	Boolean	Supervised only. If set to false, prevents modification of Bluetooth settings. Defaults to true. **Availability:** Available in iOS 10.0 and later.
allowAutoUnlock	Boolean	If set to false, disallows macOS auto unlock. Defaults to true. **Availability:** Available only in macOS 10.12 and later.
allowCloudDesktopAnd Documents	Boolean	If set to false, disallows macOS cloud desktop and document services. Defaults to true. **Availability:** Available only in macOS 10.12.4 and later.
allowDictation	Boolean	Supervised only. If set to false, disallows dictation input. Defaults to true. **Availability:** Available only in iOS 10.3 and later.
forceWiFiWhitelisting	Boolean	Optional. Supervised only. If set to true, the device can join Wi-Fi networks only if they were set up through a configuration profile. Defaults to false. **Availability:** Available only in iOS 10.3 and later.

Key	Type	Value
forceUnpromptedManaged ClassroomScreenObservation	Boolean	Deprecated in iOS 11. Use forceClassroomUnpromptedScreenObservation instead.
allowAirPrint	Boolean	Supervised only. If set to false, disallow AirPrint. Defaults to true. **Availability:** Available in iOS 11.0 and later.
allowAirPrint CredentialsStorage	Boolean	Supervised only. If set to false, disallows keychain storage of username and password for Airprint. Defaults to true. **Availability:** Available in iOS 11.0 and later.
forceAirPrintTrustedTLS Requirement	Boolean	Supervised only. If set to true, requires trusted certificates for TLS printing communication. Defaults to false. **Availability:** Available in iOS 11.0 and later.
allowAirPrintiBeacon Discovery	Boolean	Supervised only. If set to false, disables iBeacon discovery of AirPrint printers. This prevents spurious AirPrint Bluetooth beacons from phishing for network traffic. Defaults to true. **Availability:** Available in iOS 11.0 and later.
allowProximitySetupTo NewDevice	Boolean	Supervised only. If set to false, disables the prompt to setup new devices that are nearby. Defaults to true. **Availability:** Available only in iOS 11.0 and later.
allowSystemAppRemoval	Boolean	Supervised only. If set to false, disables the removal of system apps from the device. Defaults to true. **Availability:** Available only in iOS 11.0 and later.
allowVPNCreation	Boolean	Supervised only. If set to false, disallow the creation of VPN configurations. Defaults to true. **Availability:** Available only in iOS 11.0 and later.
allowUSBRestrictedMode	Boolean	Supervised only. If set to false, device will always be able to connect to USB accessories while locked. Defaults to true. **Availability:** Available only in iOS 11.4.1 and later.
forceDelayedSoftware Updates	Boolean	Supervised only. If set to true, delays user visibility of Software Updates. Defaults to false. **Availability:** Available in iOS 11.3 and later and macOS 10.13 and later.
enforcedSoftwareUpdate Delay	Integer	Supervised only. This restriction allows the admin to set how many days a software update on the device will be delayed. With this restriction in place, the user will not see a software update until the specified number of days after the software update release date. The max is 90 days and the default value is 30. **Availability:** Available in iOS 11.3 and later and macOS 10.13.4 and later.

Key	Type	Value
`forceAuthentication BeforeAutoFill`	Boolean	Optional. Supervised only. If set to `true`, the user will have to authenticate before passwords or credit card information can be autofilled in Safari and Apps. If this restriction is not enforced, the user can toggle this feature in settings. Only supported on devices with FaceID or TouchID. Defaults to `true`. **Availability:** Available only in iOS 11.0 and later.
`forceClassroom AutomaticallyJoinClasses`	Boolean	Optional. Supervised only. If set to `true`, automatically give permission to the teacher's requests without prompting the student. Defaults to `false`. **Availability:** Available only in iOS 11.0 and later.
`forceClassroomRequest PermissionToLeaveClasses`	Boolean	Optional. Supervised only. If set to `true`, a student enrolled in an unmanaged course via Classroom will request permission from the teacher when attempting to leave the course. Defaults to `false`. **Availability:** Available only in iOS 11.3 and later.
`forceClassroomUnprompted AppAndDeviceLock`	Boolean	Optional. Supervised only. If set to `true`, allow the teacher to lock apps or the device without prompting the student. Defaults to `false`. **Availability:** Available only in iOS 11.0 and later.
`forceClassroomUnprompted ScreenObservation`	Boolean	Optional. Supervised only. If set to `true`, and `ScreenObservationPermissionModificationAllowed` is also `true` in the Education payload, a student enrolled in a managed course via the Classroom app will automatically give permission to that course's teacher's requests to observe the student's screen without prompting the student. Defaults to `false`. **Availability:** Available only in iOS 11.0 and later.
`ratingRegion`	String	This 2-letter key is used by profile tools to display the proper ratings for given region. Possible values: • au: Australia • ca: Canada • fr: France • de: Germany • ie: Ireland • jp: Japan • nz: New Zealand • gb: United Kingdom • us: United States **Availability:** Available in iOS and tvOS 11.3 and later.

Key	Type	Value
`ratingMovies`	Integer	This value defines the maximum level of movie content that is allowed on the device. Possible values (with the US description of the rating level): • 1000: All • 500: NC-17 • 400: R • 300: PG-13 • 200: PG • 100: G • 0: None **Availability:** Available only in iOS and tvOS 11.3 and later.
`ratingTVShows`	Integer	This value defines the maximum level of TV content that is allowed on the device. Possible values (with the US description of the rating level): • 1000: All • 600: TV-MA • 500: TV-14 • 400: TV-PG • 300: TV-G • 200: TV-Y7 • 100: TV-Y • 0: None **Availability:** Available only in iOS and tvOS 11.3 and later.
`ratingApps`	Integer	This value defines the maximum level of app content that is allowed on the device. Possible values (with the US description of the rating level): • 1000: All • 600: 17+ • 300: 12+ • 200: 9+ • 100: 4+ • 0: None **Availability:** Available only in iOS 5 and tvOS 11.3 and later.
`forceAutomaticDate AndTime`	Boolean	Optional. Supervised only. If set to `true`, the Date & Time "Set Automatically" feature is turned on and can't be turned off by the user. Defaults to `false`. **Note:** The device's time zone will only be updated when the device can determine its location (cellular connection or wifi with location services enabled). **Availability:** Available only in iOS 12.0 and later.

Key	Type	Value
`allowPasswordAutoFill`	Boolean	Optional. Supervised only. If set to `false`, users will not be able to use the AutoFill Passwords feature on iOS and will not be prompted to use a saved password in Safari or in apps. If set to `false`, Automatic Strong Passwords will also be disabled and strong passwords will not be suggested to users. Defaults to `true`. **Availability:** Available only in iOS 12.0 and tvOS 9.0 and later.
`allowPasswordProximityRequests`	Boolean	Optional. Supervised only. If set to `false`, a user's device will not request passwords from nearby devices. Defaults to `true`. **Availability:** Available only in iOS 12.0, macOS 10.14, and tvOS 12.0 and later.
`allowPasswordSharing`	Boolean	Optional. Supervised only. If set to `false`, users can not share their passwords with the Airdrop Passwords feature. Defaults to `true`. **Availability:** Available only in iOS 12.0 and tvOS 9.0 and later.
`allowManagedToWriteUnmanagedContacts`	Boolean	Optional. If set to `true`, managed apps can write contacts to unmanaged contacts accounts. Defaults to `false`. if `allowOpenFromManagedToUnmanaged` is `true`, this restriction has no effect. A payload that sets this to `true` must be installed via MDM. **Availability:** Available only in iOS 12.0 and later.
`allowUnmanagedToReadManagedContacts`	Boolean	Optional. Supervised only. If set to `true`, unmanaged apps can read from managed contacts accounts. Defaults to `false`. if `allowOpenFromManagedToUnmanaged` is `true`, this restriction has no effect. A payload that sets this to `true` must be installed via MDM. **Availability:** Available only in iOS 12.0 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.
`Keysize`	Integer	Optional. The key size in bits, either 1024 or 2048.
`KeyType`	String	Optional. Currently always "RSA".
`KeyUsage`	Integer	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.
`CAFingerprint`	Data	Optional. The fingerprint of the Certificate Authority certificate.
`AllowAllAppsAccess`	Boolean	Optional. If `true`, all apps have access to the private key. Default is `false`.
`KeyIsExtractable`	Boolean	Optional. If `false`, the private key cannot be exported from the keychain. Default is `true`.

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 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 macOS.

Screensaver

Screensaver payloads are designated by specifying `com.apple.screensaver` as the `PayloadType`.

The device level screensaver payload can be used to customize the screensaver and enable or disable the screen lock password function.

The Screensaver payload defines the following keys:

Key	Type	Value
`askForPassword`	Boolean	Optional. If `true`, the user will be prompted for a password when the screensaver is unlocked or stopped. When using this prompt, `askForPasswordDelay` must also be provided. **Availability:** Available in macOS 10.13 and later.
`askForPasswordDelay`	Integer	Optional. Number of seconds to delay before the password will be required to unlock or stop the screen saver (the "grace period"). A value of 2147483647 (eg 0x7FFFFFFF) can be used to disable this requirement, and on 10.13, the payload is one of the only ways of disabling the feature. Note that `askForPassword` must be set to `true` to use this option. **Availability:** Available in macOS 10.13 and later.
`loginWindowModulePath`	String	Optional. A full path to the screen saver module to be used. **Availability:** Available in macOS 10.11 and later.
`loginWindowIdleTime`	Integer	Optional. Number of seconds of inactivity before screensaver activates. (0=never activate). **Availability:** Available in macOS 10.11 and later.

User level screensaver payloads are designated by specifying `com.apple.screensaver.user` as the `PayloadType`.

The user level screensaver settings are specific to a user, instead of the device.

The Screensaver User payload defines the following keys:

Key	Type	Value
`modulePath`	String	Optional. A full path to the screen saver module to be used. **Availability:** Available in macOS 10.11 and later.
`idleTime`	Integer	Optional. Number of seconds of inactivity before screensaver activates. (0=never activate). **Availability:** Available in macOS 10.11 and later.

Setup Assistant

The Setup Assistant Payload is designated by specifying `com.apple.SetupAssistant.managed` as the `PayloadType`.

On macOS, this payload specifies Setup Assistant options for either the system or particular users.

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

Key	Type	Value
`SkipCloudSetup`	Boolean	Optional. If `true`, skip the Apple ID setup window. **Availability:** Available in macOS 10.12 and later.
`SkipSiriSetup`	Boolean	Optional. If `true`, skip the Siri setup window. **Availability:** Available in macOS 10.12 and later.
`SkipPrivacySetup`	Boolean	Optional. If `true`, skip the Privacy consent window. **Availability:** Available in macOS 10.13.4 and later.
`SkipiCloudStorageSetup`	Boolean	Optional. If `true`, skip the iCloud Storage window. **Availability:** Available in macOS 10.13.4 and later.
`SkipAppearance`	Boolean	Optional. If `true`, skip the Choose Your Look window. **Availability:** Available in macOS 10.14 and later.

Shared Device Configuration Payload

The Shared Device Configuration Payload is designated by specifying `com.apple.shareddeviceconfiguration` as the `PayloadType`. 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
`AssetTagInformation`	String	Optional. Asset tag information for the device, displayed on the login window and lock screen.
`LockScreenFootnote`	String	Optional. A footnote displayed on the login window and lock screen. Available in iOS 9.3.1 and later.
`IfLostReturnToMessage`	String	**Deprecated.** Use `LockScreenFootnote` instead.

ShareKit Payload

MacOS 10.9 or later only. The ShareKit Payload is designated by specifying `com.apple.com.apple.ShareKitHelper` as the `PayloadType`. It can contain only one payload. It is supported on the User Channel.

The ShareKit Payload specifies which ShareKit plugin can be accessed on client. Both allow and disallow lists can be specified.

This payload is deprecated as of macOS 10.12. For clients running macOS 10.13 or later, use the NSExtension Payload instead. If a profile contains both a `NSExtension Payload` and a `ShareKit Payload`, the ShareKit Payload will be ignored.

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

Key	Type	Value
`SHKAllowedShareServices`	Array of Strings	Optional. List of plugin IDs that will show up in the user's Share menu. If this array exists then only these items will be permitted.
`SHKDeniedShareServices`	Array of Strings	Optional. List of plugin IDs that will not show up in the user's Share menu. This key is used only if there is no `SHKAllowedShareServices` key.

The following plugin IDs are supported by this payload:

- "com.apple.share.AirDrop": AirDrop
- "com.apple.share.Facebook": Facebook
- "com.apple.share.Twitter": Twitter
- "com.apple.share.Mail": Mail
- "com.apple.share.Messages": Messages
- "com.apple.share.Video": Video Services
- "com.apple.share.addtoiphoto": Photos
- "com.apple.share.addtoaperture": Aperture
- "com.apple.share.readlater": Reading List
- "com.apple.share.SinaWeibo": Sina Weibo
- "com.apple.Notes.SharingExtension": Notes
- "com.apple.reminders.RemindersShareExtension": Reminders
- "com.apple.share.LinkedIn.post": LinkedIn

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.
`PayloadCertificateUUID`	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.
`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 all matching values. For example, http://*.example.com/ will match both `http://store.example.com/` and `http://www.example.com`.

The patterns `http://.com` and `https://.com` match all HTTP and HTTPS URLs, respectively.

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 once, at the

end of the string (`com.mycompany.*`, for example). When a wildcard is included, any app whose bundle ID begins with the prefix is granted access to the account.

SmartCard Settings Payload

The SmartCard Settings payload is designated by specifying `com.apple.security.smartcard` as the `PayloadType`.

This payload controls restrictions and settings for SmartCard pairing on macOS v10.12.4 and later.

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

Key	Type	Value
`UserPairing`	Boolean	Optional. If `false`, users will not get the pairing dialog, although existing pairings will still work. Default is `true`.
`allowSmartCard`	Boolean	Optional. If `false`, the SmartCard is disabled for logins, authorizations, and screensaver unlocking. It is still allowed for other functions, such as signing emails and web access. A restart is required for a change of setting to take effect. Default is `true`.
`checkCertificateTrust`	Integer	Optional. Valid values are 0-3: • 0: certificate trust check is turned off • 1: certificate trust check is turned on. Standard validity check is being performed but this does not include additional revocation checks. • 2: certificate trust check is turned on, plus a soft revocation check is performed. Until the certificate is explicitly rejected by CRL/OCSP, it is considered as valid. This implies that unavailable/unreachable CRL/OCSP allows this check to succeed. • 3: certificate trust check is turned on, plus a hard revocation check is performed. Unless CRL/OCSP explicitly says "this certificate is OK", the certificate is considered as invalid. The is the most secure option. Default is `0`.
`oneCardPerUser`	Boolean	Optional. If `true`, a user can pair with only one SmartCard, although existing pairings will be allowed if already set up. Default is `false`.
`enforceSmartCard`	Boolean	Optional. If `true`, a user can only login or authenticate with a SmartCard. Default is `false`.
`tokenRemovalAction`	Integer	Optional. If `1`, when the SmartCard is removed, the screensaver will be enabled. Default is `0`. **Availability:** Available in macOS 10.13.4 and later.

Software Update

The Software Update payload is designated by specifying `com.apple.SoftwareUpdate` as the `PayloadType`.

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

Key	Type	Value
`CatalogURL`	String	Optional. The URL of the software update catalog.
`AllowPreReleaseInstallation`	Boolean	Optional. If `true`, prerelease software can be installed on this computer. Default is `true`.

System Migration Payload

The System Migration payload is designated by specifying `com.apple.systemmigration` as the `PayloadType`.

System migration occurs when items are transferred to a macOS device from a Windows device by reading source and destination path pairs from plist files. This payload provides a way to customize those transfers.

This payload must be single and exist only 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 macOS 10.12.4 and later.

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

Key	Type	Value
`CustomBehavior`	Array of Dictionaries	Optional. Specifies custom behavior for the context designated in each dictionary.

Each dictionary in the `CustomBehavior` array contains these keys:

Key	Type	Value
`Context`	String	Required. The context to which custom paths apply.
`Paths`	Array of Dictionaries	Required. The custom paths to be migrated from a source system to a target system.

Each dictionary in the `Paths` array contains these keys:

Key	Type	Value
`SourcePath`	String	Required. The path to the migrating file or directory on the source system.
`SourcePathInUserHome`	Boolean	Required. If `true`, the source path is located within a user home directory.
`TargetPath`	String	Required. The path to the destination file or directory on the target system.

Key	Type	Value
`TargetPathInUserHome`	Boolean	Required. If `true`, the target path is located within a user home directory.

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 macOS 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.
`AllowIdentifiedDevelopers`	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 macOS 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.

Key	Type	Value
`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 macOS 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.

TV Remote Payload

The TV Remote payload is designated by specifying `com.apple.tvremote` as the `PayloadType` value.

This payload allows restricting the connections from the Apple TV Remote app to an Apple TV and restricting the available Apple TV devices in the Apple TV Remote app.

To lock specific Apple TVs to specific devices running Apple TV Remote app, both the Apple TVs and remote devices can be specified in the same payload.

In addition to the settings common to all payload types, the TV Remote payload defines the following keys:

Key	Type	Value
`AllowedRemotes`	Array of Dictionaries	If present, the Apple TV will only connect with the Apple TV Remote app from the devices specified. If not present, or the list is empty, any device will be allowed to connect. **Availability:** Available in tvOS 11.3 and later.
`AllowedTVs`	Array of Dictionaries	If present, the Apple TV Remote app will only connect to the specified Apple TVs. If not present, or the list is empty, the device will be able to connect to any Apple TV. **Availability:** Available in iOS 11.3 and later.

Each entry in the `AllowedRemotes` array is a dictionary that can contain the following key:

Key	Type	Value
`RemoteDeviceID`	String	The MAC address of a permitted iOS device that can control this Apple TV. Use the format "xx:xx:xx:xx:xx:xx". The field is not case sensitive. **Availability:** Available in tvOS 11.3 and later.

Each entry in the `AllowedTVs` array is a dictionary that can contain the following key:

Key	Type	Value
`TVDeviceID`	String	The MAC address of an Apple TV device that this iOS device is permitted to control. Use the format "xx:xx:xx:xx:xx:xx". The field is not case sensitive. **Availability:** Available in iOS 11.3 and later.

Time Server Payload

The Time Server payload is designated by specifying `com.apple.MCX` as the `PayloadType` value.

This payload allows devices to connect to custom time servers.

In addition to the settings common to all payload types, the Time Server payload defines the following keys:

Key	Type	Value
`timeServer`	String	The ntp server to connect to. **Availability:** Available in macOS 10.12.4 and later.
`timeZone`	String	Time zone path location string in `/usr/share/zoneinfo/`. For example, "America/Denver" or "Zulu". **Availability:** Available in macOS 10.12.4 and later.

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.

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	Optional. Description of the VPN connection displayed on the device.
`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) • `AlwaysOn` (see AlwaysOn Dictionary Keys) • VPN (solution uses a VPN plugin or `NetworkExtension`, so the `VPNSubType` key is required (see below)).
`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.
`ProviderBundleIdentifier`	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:

Key	Type	Value
OnDemandEnabled	Integer	1 if the VPN connection should be brought up on demand, else 0.
OnDemandMatchDomainsAlways	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.
OnDemandMatchDomainsNever	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.
OnDemandMatchDomainsOnRetry	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 for details.

If VPNType is not AlwaysOn, the following key may be defined:

Key	Type	Value
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.

Key	Type	Value
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.

IPv4 Dictionary Keys

The following element is for VPN payloads of type L2TP or PPTP

Key	Type	Value
OverridePrimary	Integer	Specifies whether to send all traffic through the VPN interface. If 1, all network traffic is sent over VPN. Defaults to 0.

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.
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.

Key	Type	Value
SharedSecret	Data	The shared secret for this VPN account. Only present if AuthenticationMethod is SharedSecret. Used for L2TP and Cisco IPSec.
PayloadCertificateUUID	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).

> **Note**
>
> 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 Keys in the ActionParameters dictionary. **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`.
`DNSServerAddressMatch`	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`.

Key	Type	Value
`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:

Key	Type	Value
`Domains`	Array of Strings	*Required.* The domains for which this evaluation applies.
`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`.
`RequiredURLStringProbe`	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.
`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` • `None` To enable EAP-only authentication, the authentication method should be set to `None` and the `ExtendedAuthEnabled` key should be set to 1. If this key is set to `None` and the `ExtendedAuthEnabled` key is not set, the authentication configuration defaults to `SharedSecret`.
`PayloadCertificateUUID`	String	Optional. The UUID of the identity certificate used as the account credential. If the value of `AuthenticationMethod` is `Certificate`, this certificate is sent out for IKEv2 machine authentication. If extended authentication (EAP) is used, it is sent out for EAP-TLS authentication.
`CertificateType`	String	Optional. This key specifies the type of `PayloadCertificateUUID` used for IKEv2 machine authentication. Its value is one of the following: • RSA (Default) • `ECDSA256` • `ECDSA384` • `ECDSA521` If this key is included, the `ServerCertificateIssuerCommonName` key is required.
`SharedSecret`	String	Optional. If `AuthenticationMethod` is `SharedSecret`, this value is used for IKE authentication.
`ExtendedAuthEnabled`	Integer	Optional. Set to 1 to enable EAP-only authentication (see `AuthenticationMethod`, above). Defaults to 0.
`AuthName`	String	Optional. Username used for authentication.
`AuthPassword`	String	Optional. Password used for authentication.

Key	Type	Value
DeadPeerDetectionRate	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.
ServerCertificateIssuerCommonName	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 both the CertificateType key is included and the ExtendedAuthEnabled key is set to 1.
ServerCertificateCommonName	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.
TLSMinimumVersion	String	Optional. The minimum TLS version to be used with EAP-TLS authentication. Value may be 1.0, 1.1, or 1.2. If no value is specified, the default minimum is 1.0. **Availability:** Available in iOS 11.0 and macOS 10.13 and later.
TLSMaximumVersion	String	Optional. The maximum TLS version to be used with EAP-TLS authentication. Value may be 1.0, 1.1, or 1.2. If no value is specified, the default maximum is 1.2. **Availability:** Available in iOS 11.0 and macOS 10.13 and later.
NATKeepAliveOffloadEnable	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.

Key	Type	Value
EnablePFS	Integer	Optional. Set to 1 to enable Perfect Forward Secrecy (PFS) for IKEv2 Connections. Default is 0.
EnableCertificate RevocationCheck	Integer	Optional. Set to 1 to enable a certificate revocation check for IKEv2 connections. This is a best-effort revocation check; server response timeouts will not cause it to fail. **Availability:** Available in iOS 9.0 and later.
IKESecurityAssociation Parameters	Dictionary	Optional. See table below. Applies to child Security Association unless ChildSecurityAssociationParameters is specified.
ChildSecurity AssociationParameters	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 • AES-128 • AES-256 (Default) • AES-128-GCM (16-octet ICV) • AES-256-GCM (16-octet ICV)
IntegrityAlgorithm	String	Optional. One of: • SHA1-96 • SHA1-160 • SHA2-256 (Default) • 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 AttributeInternalIPSubnet	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.

DNS Dictionary Keys

If `VPNType` is `IKEv2`, the following DNS keys may be provided:

Key	Type	Value
`ServerAddresses`	Array of Strings	Required. An array of DNS server IP address strings. These IP addresses can be a mixture of IPv4 and IPv6 addresses. **Availability:** Available in iOS 10.0 and later and macOS 10.12 and later.
`SearchDomains`	Array of Strings	Optional. A list of domain strings used to fully qualify single-label host names. **Availability:** Available in iOS 10.0 and later and macOS 10.12 and later.
`DomainName`	String	Optional. The primary domain of the tunnel. **Availability:** Available in iOS 10.0 and later and macOS 10.12 and later.

Key	Type	Value
`SupplementalMatchDomains`	Array of Strings	Optional. A list of domain strings used to determine which DNS queries will use the DNS resolver settings contained in `ServerAddresses`. This key is used to create a split DNS configuration where only hosts in certain domains are resolved using the tunnel's DNS resolver. Hosts not in one of the domains in this list are resolved using the system's default resolver. If `SupplementalMatchDomains` contains the empty string it becomes the default domain. This is how a split-tunnel configuration can direct all DNS queries first to the VPN DNS servers before the primary DNS servers. If the VPN tunnel becomes the network's default route, the servers listed in `ServerAddresses` become the default resolver and the `SupplementalMatchDomains` list is ignored. **Availability:** Available in iOS 10.0 and later and macOS 10.12 and later.
`SupplementalMatchDomainsNoSearch`	Integer	Optional. Whether (0) or not (1) the domains in the `SupplementalMatchDomains` list should be appended to the resolver's list of search domains. Default is 0. **Availability:** Available in iOS 10.0 and later and macOS 10.12 and later.

Proxies Dictionary Keys

The `Proxies` dictionary may contain the following keys:

Key	Type	Value
`ProxyAutoConfigEnable`	Integer	Optional. Set to 1 to enable automatic proxy configuration. Defaults to 0.
`ProxyAutoConfigURLString`	String	Optional. URL to the location of the proxy auto-configuration file. Used only when `ProxyAutoConfigEnable` is 1.
`SupplementalMatchDomains`	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.

Key	Type	Value
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.
AllowAllCaptiveNetworkPlugins	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.
AllowedCaptiveNetworkPlugins	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 AllowedCaptiveNetworkPlugins 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` • `CellularServices` (Available in iOS 11.3 and later.)
`Action`	String	Required. One of the following: • `Allow` • `Drop`

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.

This payload is supported only in iOS 7.0 and later and macOS 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 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.
`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"`.
`OnDemandMatchAppEnabled`	Boolean	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.
`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 macOS 10.9 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.
`DesignatedRequirement`	String	The code signature designated requirement of the app that will use the per-app VPN.
`SigningIdentifier`	String	The code signature signing identifier of the app that will use the per-app VPN.

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 macOS.

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 macOS, `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. The number of these URLs should be limited to about 500.
`BlacklistedURLs`	Array of Strings	Optional. Access to the specified URLs is blocked. The number of these URLs should be limited to about 500.

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.

Apple domains cannot be blacklisted.

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.
`PayloadCertificateUUID`	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 macOS.
`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 macOS. 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 macOS 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 macOS 10.9 and later..
`ServiceProviderRoaming Enabled`	Boolean	Optional. If `true`, allows connection to roaming service providers. Defaults to `false`. **Availability:** Available in iOS 7.0 and later and in macOS 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 macOS 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 macOS 10.9 and later..

Key	Type	Value
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 macOS.
DisplayedOperatorName	String	The operator name to display when connected to this network. Used only with Wi-Fi Hotspot 2.0 access points. **Availability:** Available in iOS 7.0 and later and in macOS 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 macOS.
CaptiveBypass	Boolean	Optional. If set to true, Captive Network detection will be bypassed when the device connects to the network. Defaults to false. **Availability:** Available in iOS 10.0 and later.
QoSMarkingPolicy	Dictionary	Optional. When this dictionary is not present for a Wi-Fi network, all apps are whitelisted to use L2 and L3 marking when the Wi-Fi network supports Cisco QoS fast lane. When present in the Wi-Fi payload, the QoSMarkingPolicy dictionary should contain the list of apps that are allowed to benefit from L2 and L3 marking. For dictionary keys, see the table below. **Availability:** Available in iOS 10.0 and later and in macOS 10.13 and later.

The QoSMarkingPolicy dictionary contains these keys:

Key	Type	Value
QoSMarkingWhitelisted AppIdentifiers	Array of Strings	Optional. Array of app bundle identifiers that will be whitelisted for L2 and L3 marking for traffic sent to the Wi-Fi network. If the array is not present but the QoSMarkingPolicy key is present (even empty) no app gets whitelisted.
QoSMarkingAppleAudio VideoCalls	Boolean	Optional. Specifies if audio and video traffic of built-in audio/video services such as FaceTime and Wi-Fi Calling will be whitelisted for L2 and L3 marking for traffic sent to the Wi-Fi network. Defaults to true.
QoSMarkingEnabled	Boolean	Optional. May be used to disable L3 marking and only use L2 marking for traffic sent to the Wi-Fi network. When this key is false the system behaves as if Wi-Fi was not associated with a Cisco QoS fast lane network. Defaults to true.

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.
`PayloadCertificateUUID`	String	Described in Certificates.

> **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.
`ProxyServerPort`	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.
`ProxyPACFallbackAllowed`	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.
TLSTrustedServerNames	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 PayloadCertificateAnchorUUID, 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.
TLSAllowTrustExceptions	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. **Availability:** Deprecated and ignored in iOS 8.0 and later.

Key	Type	Value
`TLSCertificateIsRequired`	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.
`TLSMinimumVersion`	String	Optional. The minimum TLS version to be used with EAP authentication. Value may be 1.0, 1.1, or 1.2. If no value is specified, the default minimum is 1.0. **Availability:** Available in iOS 11.0 and macOS 10.13 and later.
`TLSMaximumVersion`	String	Optional. The maximum TLS version to be used with EAP authentication. Value may be 1.0, 1.1, or 1.2. If no value is specified, the default maximum is 1.2. **Availability:** Available in iOS 11.0 and macOS 10.13 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.
`TTLSInnerAuthentication`	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 https://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`.
`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.

Key	Type	Value
`EAPFASTProvisionPACAnonymously`	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 macOS
`SafariPasswordAutoFillDomains`	Array	Optional. An array of URL strings. Supported in iOS 9.3 and later; not supported in macOS. 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 Shared iPad, no password can be saved. • If the device is not configured as Shared iPad, 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/`.
`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.

Active Directory Payload

In macOS 10.9 and later, a configuration profile can be used to configure macOS to join an Active Directory (AD) domain. Advanced AD options available via Directory Utility or the `dsconfigad` command line tool can also be set using a configuration profile, following this procedure:

1. Start with a macOS Directory payload, created in Profile Manager.
2. Save and download the profile so you can edit it manually.

The following AD configuration keys can be added to the Directory payload, of type `com.apple.DirectoryService.managed`. Note that some settings will only be set if the associated flag key is set to `true`. For example, `ADPacketEncryptFlag` must be set to `true` to set the `ADPacketEncrypt` key to `enable`.

Key	Type	Description
`HostName`	String	The Active Directory domain to join.
`UserName`	String	User name of the account used to join the domain.
`Password`	String	Password of the account used to join the domain.
`ADOrganizationalUnit`	String	The organizational unit (OU) where the joining computer object is added.
`ADMountStyle`	String	Network home protocol to use: “afp” or “smb”.
`ADCreateMobileAccountAtLoginFlag`	Boolean	Enable or disable the `ADCreateMobileAccountAtLogin` key.
`ADCreateMobileAccountAtLogin`	Boolean	Create mobile account at login.
`ADWarnUserBeforeCreatingMAFlag`	Boolean	Enable or disable the `ADWarnUserBeforeCreatingMA` key.
`ADWarnUserBeforeCreatingMA`	Boolean	Warn user before creating a Mobile Account.
`ADForceHomeLocalFlag`	Boolean	Enable or disable the `ADForceHomeLocal` key.
`ADForceHomeLocal`	Boolean	Force local home directory.
`ADUseWindowsUNCPathFlag`	Boolean	Enable or disable the `ADUseWindowsUNCPath` key.
`ADUseWindowsUNCPath`	Boolean	Use UNC path from Active Directory to derive network home location.
`ADAllowMultiDomainAuthFlag`	Boolean	Enable or disable the `ADAllowMultiDomainAuth` key.
`ADAllowMultiDomainAuth`	Boolean	Allow authentication from any domain in the forest.
`ADDefaultUserShellFlag`	Boolean	Enable or disable the `ADDefaultUserShell` key.
`ADDefaultUserShell`	String	Default user shell; e.g. /bin/bash.
`ADMapUIDAttributeFlag`	Boolean	Enable or disable the `ADMapUIDAttribute` key.
`ADMapUIDAttribute`	String	Map UID to attribute.

Key	Type	Description
ADMapGIDAttributeFlag	Boolean	Enable or disable the ADMapGIDAttribute key.
ADMapGIDAttribute	String	Map user GID to attribute.
ADMapGGIDAttributeFlag	Boolean	Enable or disable the ADMapGGIDAttributeFlag key.
ADMapGGIDAttribute	String	Map group GID to attribute.
ADPreferredDCServerFlag	Boolean	Enable or disable the ADPreferredDCServer key.
ADPreferredDCServer	String	Prefer this domain server.
ADDomainAdminGroupListFlag	Boolean	Enable or disable the ADDomainAdminGroupList key.
ADDomainAdminGroupList	Array of Strings	Allow administration by specified Active Directory groups.
ADNamespaceFlag	Boolean	Enable or disable the ADNamespace key.
ADNamespace	String	Set primary user account naming convention: ”forest” or ”domain”; ”domain” is default.
ADPacketSignFlag	Boolean	Enable or disable the ADPacketSign key.
ADPacketSign	String	Packet signing: ”allow”, ”disable” or ”require”; ”allow” is default.
ADPacketEncryptFlag	Boolean	Enable or disable the ADPacketEncrypt key.
ADPacketEncrypt	String	Packet encryption: ”allow”, ”disable”, ”require” or ”ssl”; ”allow” is default.
ADRestrictDDNSFlag	Boolean	Enable or disable the ADRestrictDDNS key.
ADRestrictDDNS	Array of Strings	Restrict Dynamic DNS updates to the specified interfaces (e.g. en0, en1, etc).
ADTrustChangePassIntervalDaysFlag	Boolean	Enable or disable the ADTrustChangePassIntervalDays key.
ADTrustChangePassIntervalDays	Integer	How often to require change of the computer trust account password in days; ”0” is disabled.

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>KeyType</key>
            <string>RSA</string>
            <key>KeyUsage</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>
```

Revision History

Date	Notes
2018-08-06	Documented the Privacy Preferences Policy Control Pane payload. Added `allowManagedToWriteUnmanagedContacts` and `allowUnmanagedToReadManagedContacts` to the Restrictions Payload.
2018-08-06	Documented the Time Server payload. Added `tokenRemovalAction` to the SmartCard Payload. Added `AllowPreReleaseInstallation` to the Software Update Payload. Documented the ShareKit Payload deprecation; instead, use the NSExtension Management Payload.
2018-07-16	Minor updates and corrections.
2018-07-05	Added new restrictions for `allowPasswordAutoFill`, `allowPasswordProximityRequests`, and `allowPasswordSharing`. Updated `OAuth` availability in the Exchange Payload. Added `AllowAllAppsAccess` for the SCEP Payload. Added `GroupingType` to the Notifications Payload. Other miscellaneous updates and corrections.
2018-06-18	Converted to PDF format. Removed APN payload section. Instead, use the Celluar Payload.
2018-06-04	Updated for iOS 12, macOS 10.14, and tvOS 12.
2018-04-09	Updated for iOS 11.3, macOS 10.13.3, and tvOS 11.3.
2017-12-07	Updated for iOS 11.2, macOS 10.13.2, and tvOS 11.2 public release.
2017-09-19	Updated for iOS 11.0, macOS 10.13, and tvOS 11.0.
2017-03-27	Update for iOS 10.3.
2016-12-12	Added a link to "iOS Human Interface Guidelines" for current icon recommendations.
2016-09-13	Made miscellaneous updates and corrections.
2016-07-01	Updated for iOS 10.0 and macOS 10.12.
2016-06-21	Added new section "Active Directory Payload"; made minor updates and corrections throughout.
2016-03-21	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 Payload. Added new section macOS Server Payload. 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.

Copyright and Notices

Apple Inc.
One Apple Park Way
Cupertino, CA 95014
USA
408-996-1010