Apple General Info

The Apple General Info section serves as a catch-all resource for a wide range of Apple-related information that doesn't fit within specific categories. From tips and tricks for maximizing productivity on your Apple devices to discussions on Apple services, accessories, and general troubleshooting guidance, this section covers it all. Stay up to date with the latest Apple announcements, explore lesser-known features and functionalities, and find solutions to common issues you may encounter. Discover insights and helpful advice that will enhance your overall Apple experience and make the most of your devices and services. The Apple General Info section is your go-to destination for all things Apple beyond the scope of specific topics.

- Apple MDM
  - Apple MDM Lost Mode
  - Apple MDM Command History

- Adapting to Apple's TLS Server Certificate Validity Limits
- Apple App Store and Automatic Updates
- Apple Content Caching service
- Bypassing DPI for Apple Traffic in MDM Communication
- Hardware Encryption Capabilities for Apple Hardware
- macOS Sonoma / iOS 17 support in FileWave 15.1.0+
- Microsoft Enterprise SSO plug-in for Apple devices
- Understanding Similar and Identical Software Update Names in FileWave for Apple Devices

Apple MDM

With the exception of macOS, all Apple devices are managed purely through MDM. macOS devices on the other hand, relies upon the FileWave Client, however, when MDM enrolled, benefit from the additional features

Apple MDM Lost Mode

What

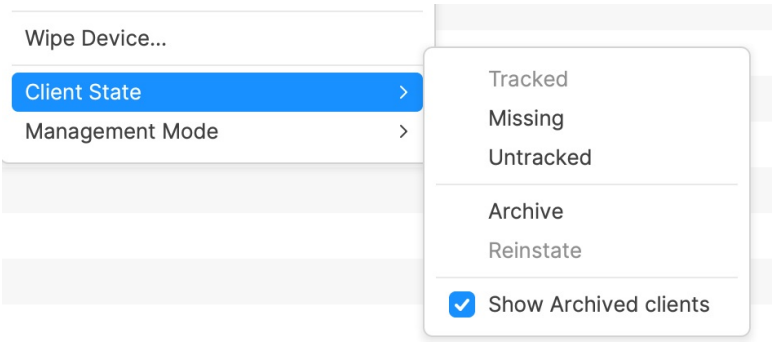
Lost Mode locks devices from use until Lost Mode is disabled. This enhances the security whilst devices are in an unknown location.

Why

On occasion, users can misplace devices. To assist the protection of data and prevent anyone from using the device until relocated, an MDM command may be sent to Lock the device.

Information

Lost Mode is enabled by setting a device into the 'Missing' Client State, via the right click contextual menu:



To disable Lost Mode, select any other Client State, Tracked or Untracked. A Model Update is required after altering the Client State before the MDM command will be sent to the device.

Devices require network to receive the command to enable or disable Lost Mode.

If a device is Locked and unable to receive network due to the surrounding W-Fi networks not yet being configured, consider connecting the device to ethernet (adaptor may be required).

Locating Devices

Whilst locked, as well as devices being unusable, additional features help locate the device

Location

After being set as Lost, location data will be sent back to the FileWave Server, where the device has a network connection. Location may be viewed on a map from the Client Info view as well as related location data being available from Inventory Queries.

Sound

Location is all very well, but what if the device is in a bag, cupboard or similar. To further assist retrieval of the device, when in the Lost Mode 'missing' state, an additional option will be available from the right click menu: 'Play Lost Mode Sound'. Previous set volume is not a consideration and the device will make an audible set of tones.

Apple MDM Command History

What

Any Apple devices that are MDM enrolled should receive MDM Requests. Each Client Info lists the requests for that device.

Why

MDM Command History exists to display all requests queued and sent to devices, along with the result of those requests; additionally showing if the request was designed for the User of the device or the System

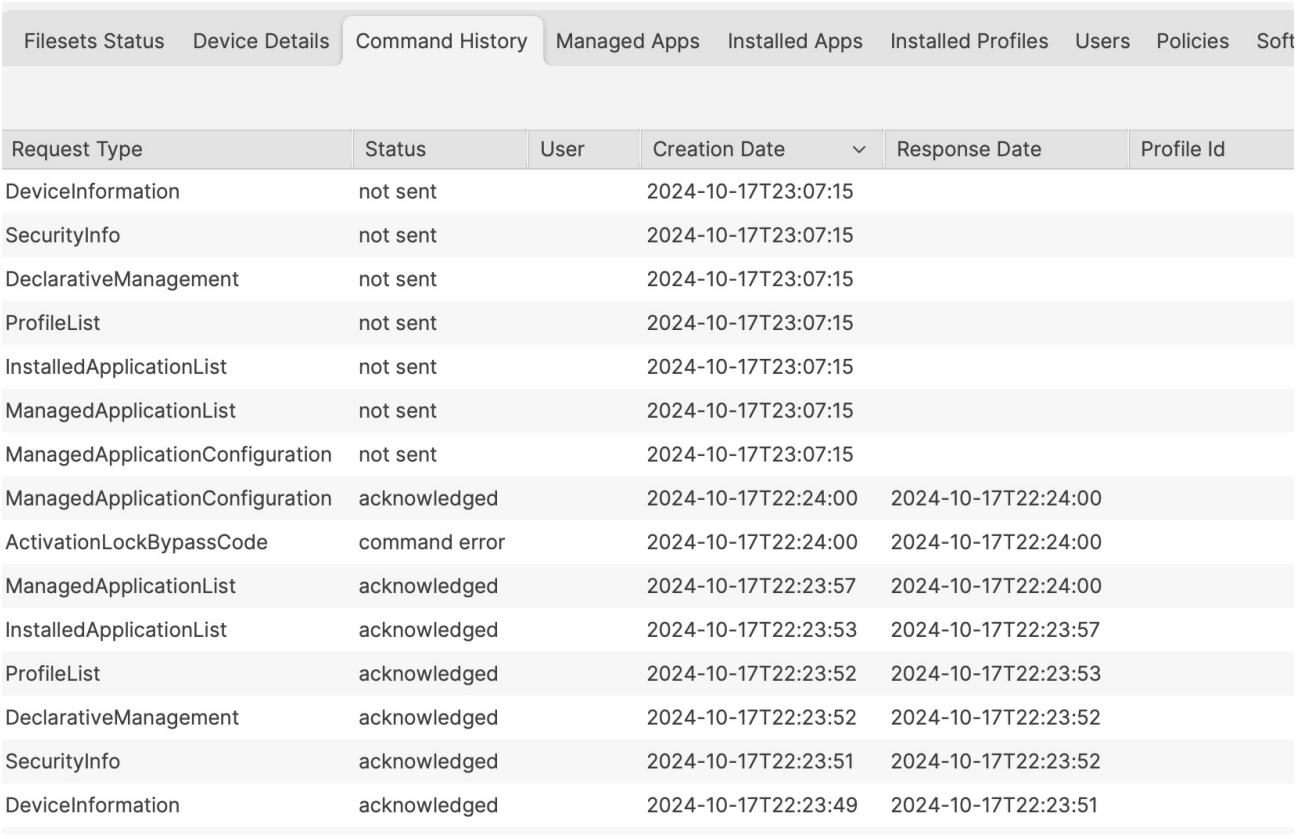
> When referring to Users and MDM, Apple allow for any amount of directory users to be managed, but only one local user is considered to be managed. This is the first local user to log into the device after enrolment. System requests impact all users.

The Request Types include:

- Inventory
- Profile instals and uninstalls
- VPP App instals and uninstalls
- Commands, e.g. erasing a device, renaming the device, etc.

Information

Opening the Client Info for an Apple MDM enrolled device and selecting the Command History tab should show something similar to the below image:



Request Type	Status	User	Creation Date	Response Date	Profile Id
DeviceInformation	not sent		2024-10-17T23:07:15		
SecurityInfo	not sent		2024-10-17T23:07:15		
DeclarativeManagement	not sent		2024-10-17T23:07:15		
ProfileList	not sent		2024-10-17T23:07:15		
InstalledApplicationList	not sent		2024-10-17T23:07:15		
ManagedApplicationList	not sent		2024-10-17T23:07:15		
ManagedApplicationConfiguration	not sent		2024-10-17T23:07:15		
ManagedApplicationConfiguration	acknowledged		2024-10-17T22:24:00	2024-10-17T22:24:00	
ActivationLockBypassCode	command error		2024-10-17T22:24:00	2024-10-17T22:24:00	
ManagedApplicationList	acknowledged		2024-10-17T22:23:57	2024-10-17T22:24:00	
InstalledApplicationList	acknowledged		2024-10-17T22:23:53	2024-10-17T22:23:57	
ProfileList	acknowledged		2024-10-17T22:23:52	2024-10-17T22:23:53	
DeclarativeManagement	acknowledged		2024-10-17T22:23:52	2024-10-17T22:23:52	
SecurityInfo	acknowledged		2024-10-17T22:23:51	2024-10-17T22:23:52	
DeviceInformation	acknowledged		2024-10-17T22:23:49	2024-10-17T22:23:51	

Request Types are defined by Apple and details may be viewed on the Apple Development Pages

Status and Error messages are those reported by Apple, with the exception of 'not sent'. The possible values are:

Status	Details	Apple Request Response
not sent	The command is queued and awaiting for the device to reply to the APNs request	-

acknowledged	Device has processed the request	Acknowledged
not now	Device has received the request, but is unable to process the request at this time	NotNow
command error	An error has occurred	Error or CommandFormatError

As of FileWave 15.5, the status of Command Queue requests are now accessible from within standard Inventory Queries

'not sent'

Before any requests may be sent to a device, the FileWave Server sends an Apple Push Notification (APN) request to Apple. Apple queue these APNs requests and only after the device checks in with Apple and pulls the APNs request, will the device then check-in with the defined server.

APNs request is nothing more that a notification for the device to check-in with the defined server.

Whilst waiting for the device to receive the APNs request and check-in, the Command History will display 'not sent'

Where requests are 'not sent' or 'not now', new requests will not be added to the queue for the same Request Types, since there is already a queued request waiting. The 'Creation Date' displays the time and day the request was added to the queue.

Response Commands

Once the APNs request has been received by the device, on check-in, queued requests may then be sent to the device.

'acknowledged'

The request has been received by the device, processed and reported back to the FileWave Server as completed.

'not now'

Some requests will not be accepted, until the device is in a certain state. For example, a user may need to be logged into the device to process the provided request.

Hence, the request has been received and the device has responded, but the request is awaiting the desired state before it will finish processing the request and report back as much.

'command error'

In some instances, the request may not be able to complete, due to an error. Apple have two defined error status values:

- 'Error' - An error occurred (the device will report error details in the response to the request)
- 'CommandFormatError' - The request protocol was incorrect, e.g. a malformed request

The 'Error Msg' column shown in the Command History view reports information provided by the device back to FileWave Server and contains information as set out by Apple.

Apple's developer page demonstrates greater depth on MDM requests:

Sending MDM Commands to a Device

User vs System

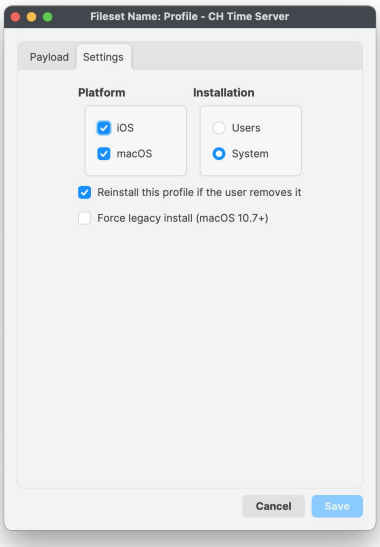
The user column contains the channel used for the request. Where the user column is blank, this implies the request is a System Channel request. Otherwise the name of any managed users existing on the device will be displayed for those requests.

Some Request Types are required for both User and System, e.g.

DeviceInformation may report inventory regarding the System and the User. As such there are multiple requests, one per Managed User and one for System.

Profile Installation

Profile Settings defines whether a Profile should be Installed for the System or User:



As with other Request Types, if the Profile is configured for System Installation, the User channel column should be blank, whilst those set as User should show a Request Type of 'InstallProfile' per managed user.

The below image shows installation of two differing Profiles, one set as System and the other set as User:

Filesets Status | Device Details | Command History | Managed Apps | Installed Apps | Installed Profiles | Users | Policies | Software Updates | DDM Declarations

Request Type	Status	User	Creation Date	Settings Items	Response Date	Profile Id
InstallProfile	acknowledged		2024-10-17T23:55:25		2024-10-17T23:55:26	fw1063.home.22e28158-118c-4956-8640-1f4119(
InstallProfile	acknowledged	sholden	2024-10-09T22:14:08		2024-10-09T22:14:08	ml1063.local.9645a261-8941-4827-b0b8-21b97a

There was a period of time where all Profiles could be set as either User or System regardless. Apple enforced 'correct' Installation channels several major versions back. As such, if Profiles were delivered before such change, it may be possible that the User column may show a User despite the Profile being set as System. This should no longer be the case for newly delivered requests

Although altered values in a Profile Payload will just cause the Profile to be updated on a device, changing the Installation channel from User to System or vice versa, will cause the Profile to be uninstalled and then re-installed using the newly defined Installation channel.

Only some Profile Payloads may be defined as either User or System. If one option is greyed out, the Profile Setting cannot be changed.

Commands

Some requests are commands designed to action an event, e.g. rename a device, erase a device, etc. If the request is altering a setting, the Request Type reported is 'Settings' and the 'Settings Items' column will display details regarding the setting to be altered.

The below image shows a request to alter the name of the device:

Filesets Status | Device Details | Command History | Managed Apps | Installed Apps | Installed Profiles | Users | Policies

Request Type	Status	User	Creation Date	Settings Items
Settings	not sent		2024-09-17T21:00:59	[{"Item": "DeviceName", "DeviceName": "macOS12VM"}]

An Update Model will be required before the Setting request is added to the device's request queue.

Further details regarding 'Command Policy' Fileset Payloads can be viewed in the following KB:

Profile Editor Command Policy

Troubleshooting

Some typical items to consider when reviewing Command History:

- The Command History tab will not be present if the device is not MDM enrolled (i.e. a macOS device with only the FileWave Client installed)
- A profile does not show on the device, yet command is acknowledged. This is typically seen where a Profile is set as User, but the currently logged in user is a local user, but not the local managed user. Compare the Command History User column with the currently logged in user.
- Where possible, it only might be prudent to set Profiles as System rather than user, if all local users require management (Note: this may also impact Administrators logging into systems, when attempting to fix a user's issue)
- Where an error has occurred, review the 'Error Msg' column. If the error message does not appear clear, consider raising a ticket with the FileWave Support team.
- As noted above, when a Profile Setting is changed, the existing Profile is removed before being re-delivered through the newly specified Installation channel (User or System). If the Profile is essential for network connectivity, the device will lose its network connection, making it impossible to receive the updated Profile.

Adapting to Apple's TLS Server Certificate Validity Limits

What

This article provides guidance on adapting to Apple's updated policy regarding the maximum allowed lifetimes of TLS server certificates. Effective from September 1, 2020, 00:00 GMT/UTC, TLS server certificates must have a validity period no greater than 398 days. This policy, part of Apple's efforts to enhance web security, affects TLS server certificates issued from Root CAs preinstalled with iOS, iPadOS, macOS, watchOS, and tvOS.

When/Why

The policy is critical for administrators using FileWave to manage Apple devices. It ensures that device profiles and their associated TLS server certificates comply with the new security standards. Non-compliance results in network and application failures, and can prevent websites from loading on affected Apple devices.

How

To comply with Apple's policy:

1. Certificate Issuance and Renewal: Certificates should be issued with a maximum validity of 397 days to avoid edge case issues.
2. Check Existing Certificates: Certificates issued before September 1, 2020, are not affected by this change. However, their renewal must comply with the 398-day limit.
3. Profile Deployment in FileWave: Ensure all TLS server certificates embedded in profiles for Apple devices meet these validity requirements.
4. Monitoring and Planning: Regularly monitor certificate expiration dates and plan renewals accordingly.

Related Links

- Apple's Certificate Policy Announcement - Details on the TLS server certificate validity limit.
- RFC 5280, Section 4.1.2.5 - Reference for certificate validity period definition.

Digging Deeper

This policy shift reflects a broader move towards enhancing digital security and trustworthiness in online environments. By reducing certificate lifetimes, Apple aims to mitigate risks such as certificate compromise and mis-issuance. For FileWave users, adapting to these new requirements is essential for maintaining secure, reliable, and compliant management of Apple devices across various environments.

Apple App Store and Automatic Updates

Developers are frequently updating applications and submitting new versions to Apple App Store. FileWave is automatically sending requests to devices to upgrade applications when a new version is available.

How does FileWave detect if there is an update?

For iOS 11.3 and later and macOS 10.13.4 and later:

Apple's MDM protocol returns the information when FileWave requests the list of installed applications: each application will provide, via the HasUpdateAvailable flag, if an update is available, and in this case, FileWave send the device a command to upgrade the application (InstallApplication command).

For previous OS versions:

Every hour, FileWave is contacting Apple's iTunes database and updates metadata about applications used in your environment. When a device checks in for Verify, we compare the application versions (on the device, from iTunes) and if the version from iTunes is higher we know there is an update.

The automatic update process does not work as expected, what could be the reasons?

If the flag is not provided by the device (older iOS / macOS version for instance), it may happen that the version from iTunes and the version reported by the device are not accurate (they are filled by hand by the developer and they may be incorrect). Apple introduced the hasUpdateAvailable flag exactly to solve this problem.

From Apple's documentation:

> If `true`, the app has an update available. This key is present only for App Store apps. In macOS, this key is present only for Volume Purchase Program (VPP) apps. This status updates daily and isn't always up-to-date when installing an app.

Unfortunately, the flag provided by the device is not 100% reliable. FileWave will only update the application once a day. The InstallApplication command will not be sent if the device reports HasUpdateAvailable=True and the same command has been already sent that same day. But if the flag is not properly updated by the device the next day, FileWave will request an update.

We have seen reports in the past that some iOS versions would not update the flag properly even if the application is up to date. And because the flag is not updated within 24h, FileWave will request the application to be reinstalled every day, which may be an issue for the end-user or for network bandwidth. In this situation, the best is to contact FileWave support and Apple Care. We will help you configuring both FileWave and your device to gather all the logs Apple will require to investigate the issue. Most of the time, upgrading iOS or macOS to the most recent available version solves the problem.

When looking at FileWave's Client Info for impacted devices, the Managed Applications tab will show you the value of the flag - if it is always "True", there is likely something wrong on at the device level.

Apple Content Caching service

What

Every device will individually reach out to Apple to pull Software Updates and VPP Apps and their respective updates. To alleviate the amount of external traffic, Apple provide Content Caching.

When/Why

Any macOS device may be configured to provide Content Caching. Once configured, devices residing on the same network will be told to use the Content Caching device to pull all updates. This means each update is only pulled from Apple once, for that network, and devices will pull any cached updates from the device sharing this cache.

How

When a device is informed to instal an OS update or VPP App (or App update), the device reaches out to the App Store. Typically the device would then pull that update directly from Apple. However, for any registered Content Caching devices on that network, the response from Apple would be to pull the update from the Content Caching device instead. For any update, the initial request for any one update will cause the Content Caching device to download the update first. Once downloaded all requesting devices may then receive the update from the local caching server instead.

Related Content

- Intro to content caching – Apple Support (UK)
- macOS User Guide – Apple Support (UK)
- Set up content caching on Mac – Apple Support (UK)

Digging Deeper

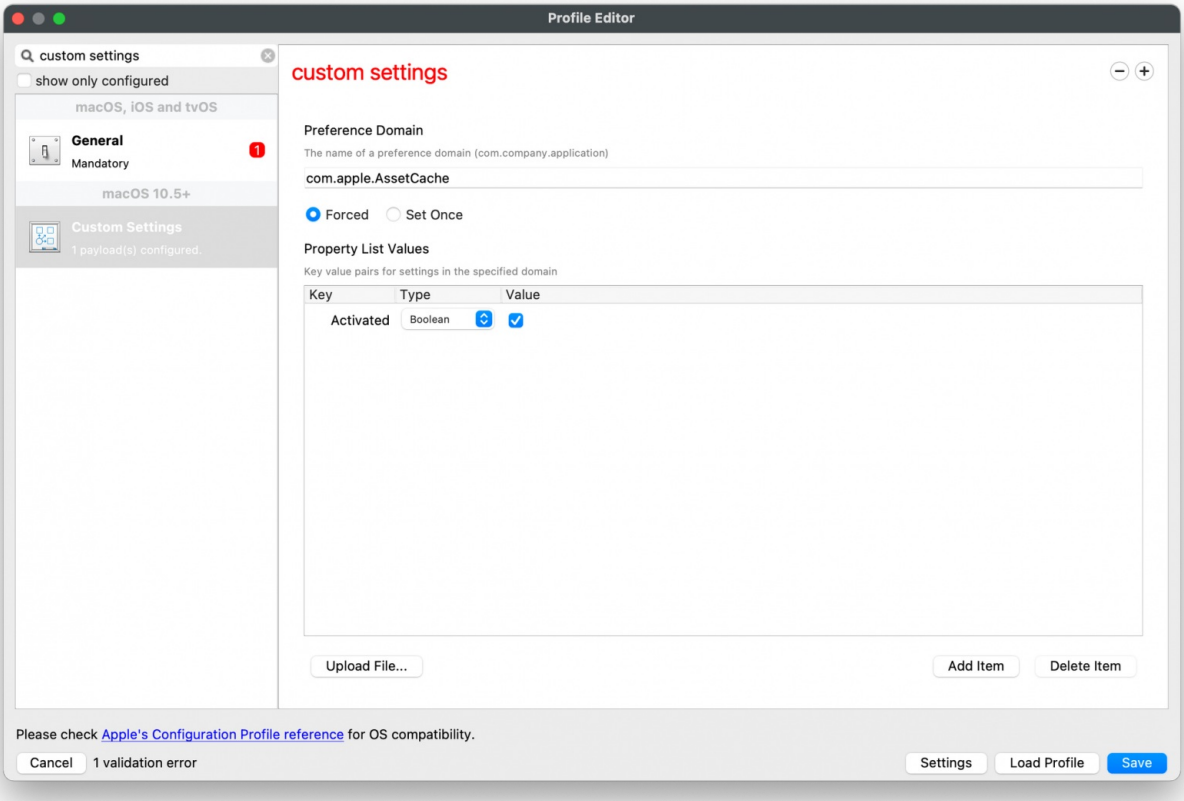
Caching configuration is stored in the following file:

```
/Library/Preferences/com.apple.AssetCache.plist
```

A Custom Field could be created, using the following code, to determine devices which have Content Caching Enabled – Boolean 0(False) 1(True)

```
defaults read /Library/Preferences/com.apple.AssetCache.plist Activated
```

A device could be set to cache, by way of a Custom Settings Payload:


Profile Editor
custom settings
show only configured
macOS, iOS and tvOS
General
Mandatory
1
macOS 10.5+
Custom Settings
1 payload(s) configured.
custom settings
Preference Domain
The name of a preference domain (com.company.application)
com.apple.AssetCache
Forced
Set Once
Property List Values
Key value pairs for settings in the specified domain
Key
Type
Value
Activated
Boolean
Upload File...
Add Item
Delete Item
Please check Apple's Configuration Profile reference for OS compatibility.
Cancel
1 validation error
Settings
Load Profile
Save

Bypassing DPI for Apple Traffic in MDM Communication

What

This article explains the importance of bypassing Deep Packet Inspection (DPI) on network traffic directed to Apple's IP range (17.0.0.0/8) to ensure seamless communication between Apple devices and the FileWave Mobile Device Management (MDM) solution.

When/Why

Deep Packet Inspection is a network packet filtering technique that examines the data part (and possibly also the header) of a packet as it passes an inspection point, to determine what to do with the packet based on its content. This is often employed in firewalls, intrusion prevention systems, and content filters to scrutinize traffic for security and compliance purposes.

However, when managing Apple devices via an MDM solution like FileWave, it's crucial to ensure uninterrupted communication with Apple's network. The DPI can interfere with the SSL traffic to and from Apple's servers, thus hindering the communication between your managed devices and the MDM server. This is particularly vital for the initial device setup, software updates, and continuous management operations.

How

To prevent any interference with the communication between Apple devices and FileWave MDM, it's advised to configure your network's firewall and content filters to bypass or disable Deep Packet Inspection for traffic destined to or originating from the IP range 17.0.0.0/8. Here are general steps:

1. Access Firewall/Content Filter Settings:
   - Log in to your firewall or content filter management interface.
2. Create a Bypass Rule:
   - Navigate to the section where you can create rules or policies.
   - Create a new rule to bypass DPI for the IP range 17.0.0.0/8.
3. Verify Configuration:
   - After setting the rule, verify the configuration by testing the communication between your MDM and an Apple device.
   - You can also check the logs to ensure traffic is flowing correctly without any SSL manipulation.

Related Links

- Deep Packet Inspection (Wikipedia) - Overview of Deep Packet Inspection.
- Apple's Managed Devices - Understanding Apple's Managed Devices and their communication.
- Default TCP and UDP Port Usage - FileWave port usage.

Digging Deeper

Understanding the technical intricacies of network traffic inspection and its implications on MDM communication is crucial for ensuring a seamless operation of managed Apple devices. Disabling DPI for specified traffic ensures that the necessary communication between your FileWave MDM server and managed Apple devices remains uninterrupted, providing a stable and reliable management infrastructure.

Hardware Encryption Capabilities for Apple Hardware

What

From a security perspective, it is important to understand the encryption capabilities of devices.

When/Why

In FileWave 14.6.0 some reporting was added to report on HardWareEncryptionCaps (https://developer.apple.com/documentation/devicemanagement/securityinforesponse/securityinfo?changes=latest_minor) as reported through Apple's MDM framework.

How

- Hardware Encryption Capabilities has been added as a field for iOS 4+ and tvOS 6+ devices to report the supported encryption.
- Passcode Present had its description updated to explain how it ties to Hardware Encryption Capabilities and also is for iOS 4+ and tvOS 6+.
- Is Recovery Lock Enabled was added for macOS devices to reflect if Recovery Lock is enabled on Apple Silicon running macOS 11.5+.

Digging Deeper

HardwareEncryptionCaps is an integer that indicates the underlying hardware encryption capabilities of the device, which is one of the following values:

- `1`: Block-level encryption
- `2`: File-level encryption
- `3`: Both block-level and file-level encryption

This value is available in iOS 4 and later, and tvOS 6 and later.

> For a device to have data protection, `HardwareEncryptionCaps` must be `3` and `PasscodePresent` must `true`.

macOS Sonoma / iOS 17 support in FileWave 15.1.0+

What

iOS, iPadOS and tvOS 17 will be released on Monday, 18th of September. macOS 14 Sonoma was released on the 26th of September. FileWave 15.1.0 released on Thursday, September 21, 2023.

When/Why

As an administrator, it is important to know if these new Operating Systems are supported in order to schedule device upgrade or new purchase.

How

iOS, tvOS, iPadOS 17

Generally, devices running iOS, tvOS, iPadOS can safely be upgraded at any time. FileWave 15.1.0 will bring new management features for these devices.

macOS 14 Sonoma

Generally, a new version of macOS requires a new version of FileWave : desktop agent and server needs an update to report properly the OS version, likewise various features like SIP detection within Filesets.

In addition, macOS 14 introduces changes related to application sandboxing which prevent FileWave 15.0 desktop agent or earlier versions to upgrade without a device restart to complete the upgrade. FileWave 15.1 has been modified to adapt to these changes, therefore it is then recommended to upgrade to FileWave 15.1 before upgrading to macOS Sonoma.

In case devices are upgraded to macOS Sonoma before FileWave is upgraded to 15.1, you have to know that:

- These devices won't report the macOS version properly until they are upgraded to FileWave 15.1 or newer client.
- A device restart will be required to upgrade FileWave client to version 15.1. This can be achieved automatically by enabling the "requires reboot" option if you go to Properties for the upgrade Fileset.

As a reminder, released version of Operating Systems can be supported. Testing existing and new features is made with Beta versions which are subject to change. These OS will be supported after the release and going through our QA process.

Related Content

- FileWave Downloads
- macOS 14 Compatible Devices (Custom Field)
- iOS 17 Compatible Devices (Query)

Microsoft Enterprise SSO plug-in for Apple devices

What

The Microsoft Single Sign-On (SSO) plug-in for Apple devices is a software extension that allows users to log in to Microsoft services on their Apple devices without needing to enter their credentials each time. This plug-in enables users to authenticate once and use Microsoft services seamlessly across multiple applications and services. For more information, you may visit, Microsoft Enterprise SSO plug-in for Apple devices.

When/Why

Using the Microsoft SSO plug-in for Apple devices offers several advantages. First, it saves time by eliminating the need to enter login credentials each time a user needs to access Microsoft services. This can be particularly useful for users using Microsoft services on their Apple devices.

Second, the Microsoft SSO plug-in provides an added layer of security. Users can use multi-factor authentication (MFA) to secure their login credentials and protect their data from unauthorized access. The plug-in provides a more secure way to access Microsoft services on Apple devices than standard login credentials.

Finally, the Microsoft SSO plug-in offers a more streamlined and user-friendly experience. Users can easily switch between different Microsoft services without needing to log in again and quickly access their files and data on any device.

How

Below are the following requirements and configuration creation steps for deployment.

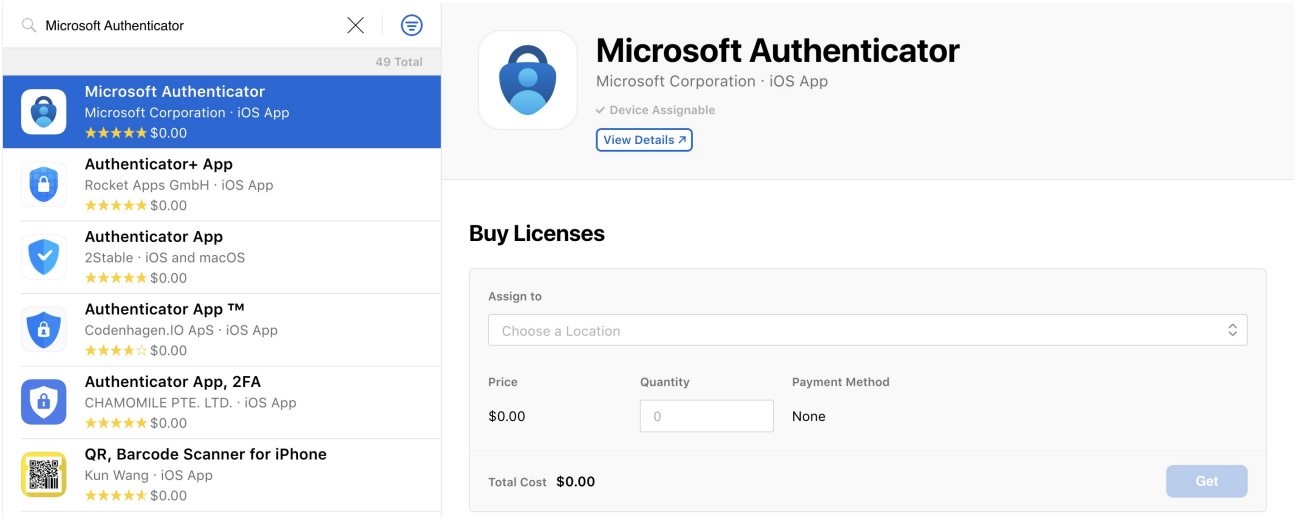
Requirements:

- The device must support and have an installed app that has the Microsoft Enterprise SSO plug-in for Apple devices:
  - iOS 13.0 and later: Microsoft Authenticator app
  - iPadOS 13.0 and later: Microsoft Authenticator app
  - macOS 10.15 and later: Intune Company Portal app
- The device must be enrolled in MDM, i.e. DEP enrolled.
- Configuration must be pushed to the device to enable the Enterprise SSO plug-in. Apple requires this security constraint.

> Please Note: On macOS devices, Apple requires the Company Portal app be installed. Users don't need to use or configure the Company Portal app, it just needs to be installed on the device. You may download here: Download the Company Portal app installer package.

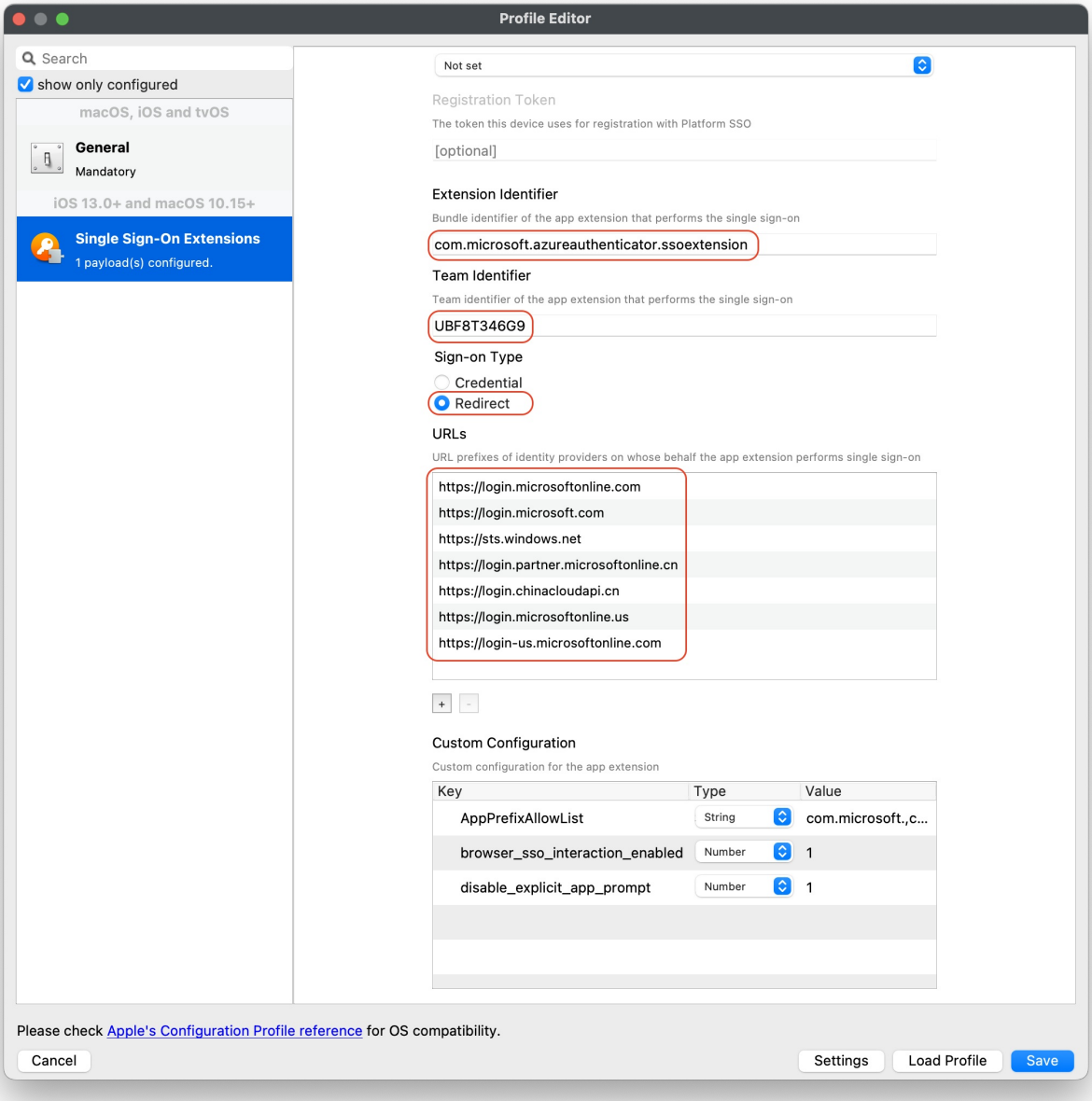
Microsoft Authenticator app deployment:

You may acquire and deploy the Microsoft Authenticator app via your ASM/ABM account. A similar method as any VPP application, search the ASM/ABM, enter in the number of licenses for the VPP application, and click on GET.



Creating the Configuration profile to be deployed to your devices:

1. Open FileWave Central
2. Select Filesets from the left side menu
3. Select New Desktop Fileset
4. Click on Profile
5. Enter in the name of the Profile, example: Microsoft Single-Sign On
6. Select the Single Sign-On Extensions payload
7. Enter in the following for specified payload:
    1. iOS settings:
        - Extension ID: `com.microsoft.azureauthenticator.ssoextension`
        - Team ID: This field isn't needed for iOS but you can use `UBF8T346G9`
    2. macOS settings:
        - Extension ID: `com.microsoft.CompanyPortalMac.ssoextension`
        - Team ID: `UBF8T346G9`
    3. Sign-On Type:
        - Type: Redirect
    4. URL identity providers:
        - `https://login.microsoftonline.com`
        - `https://login.microsoft.com`
        - `https://sts.windows.net`
        - `https://login.partner.microsoftonline.cn`
        - `https://login.chinacloudapi.cn`
        - `https://login.microsoftonline.us`
        - `https://login-us.microsoftonline.com`
    5. Optional Custom Configurations (Not required):
        - Enable SSO for all apps with specific bundle IDs or prefix IDs: Key:AppPreFixAllowList - Type:String - Value:com.microsoft., com.apple., or com.business.travelapp
        - Sign in with browser that don’t use MSAL and Safari: Key:browser_sso_interaction_enabled - Type:Number - Value:1
        - Disable OAuth 2 app prompts: Key:disable_explicit_app_prompt - Type:Number - Value:1



Deployment

Next deploy the Microsoft Authenticator app and Configuration profile on a few devices. If you're not deploying the Microsoft Authenticator app using an app policy, then users must install it manually. Users don't need to use the Authenticator app, it just needs to be installed on the device.

Users sign in to any supported app or website to bootstrap the extension.

> Bootstrap is the process of signing in for the first time, which sets up the extension. After users sign in successfully, the extension is automatically used to sign in to any other supported app or website.
>
> Meaning the end users will need to sign into their Microsoft account for their first time manually for the extension to authenticate successfully.

You can test single sign-on by opening Safari in private mode (opens Apple's web site) and opening the https://portal.office.com site. If configured successfully, no username and password will be required.

Related Content

- Microsoft SSO for macOS devices

Understanding Similar and Identical Software Update Names in FileWave for Apple Devices

What

As a Unified Endpoint Management tool, FileWave manages a wide range of devices, including Apple devices such as macOS, iOS, iPadOS, and tvOS. In the Software Updates list for these devices, you may notice updates that have similar or identical names, such as "iPadOS 16.1," "iPadOSs 16.1," and "iOS 16.1." This can be confusing, as it may seem like there are duplicate updates or that the updates are intended for different devices.

When/Why

This is due to how Apple publishes updates for its devices. Different devices, as well as different versions of macOS and iOS, may have updates with slightly different names. For example, an update for iPadOS may have the same version number as an update for iOS, but the names will be slightly different to reflect the intended device.

How

To ensure that all of your Apple devices are updated with the latest patches, it is important to enable all variations of the patch for a specific version (e.g., 16.1) that you are trying to update. This will ensure that all relevant devices receive the necessary updates. Ensuring that all of your Apple devices are up to date is crucial for maintaining the security and functionality of your organization's technology.

Related Content

- Apple MDM Software Updates
- View - Software Updates