

Bootstrap Token Management on macOS

Introduction

macOS 10.15 Catalina introduces a new method of SecureToken enablement called Bootstrap Token. This is an MDM-based management feature to automatically provide a [SecureToken](#) on all mobile account logins. This does not affect how local accounts get SecureTokens.

What is SecureToken?

Apple introduced Secure Token as a method of creating a “chain of trust” on a machine. The result was that only a trusted user could be created from another trusted user, and only those trusted users could leverage FileVault. This ensured that - from Apple’s perspective - the machine and users on it are secure.

For many Mac admins, the Secure Token feature introduced with macOS High Sierra has been a cause for great consternation. It has forced IT teams to adjust their management workflows to be able to administer and manage devices differently, given that Secure Tokens are critical in order to safely create Mac users and manage their FileVault full disk encryption (FDE) credentials.

Why the Consternation with SecureToken?

The challenge presented by the introduction of Secure Token was that the chain of trust ultimately made user and FileVault management much more challenging. An organization’s mobile and network accounts with Apple devices did not have the ability to create users that would be granted Secure Tokens.

Only the original user on the machine was granted a Secure Token and only that account could go on to create subsequent users that would properly be granted a Secure Token. This overhead severely impacted the ability of Mac admins to remotely manage their fleet of Mac systems.

What is the Benefit of Bootstrap Token?

Catalina can give the first mobile account to log in a SecureToken if no other accounts have a SecureToken yet, but the benefit of Bootstrap Token comes when multiple users log into an encrypted machine. All mobile accounts that log in automatically get a SecureToken without having to hand one off manually.

What Happens Under the Hood?

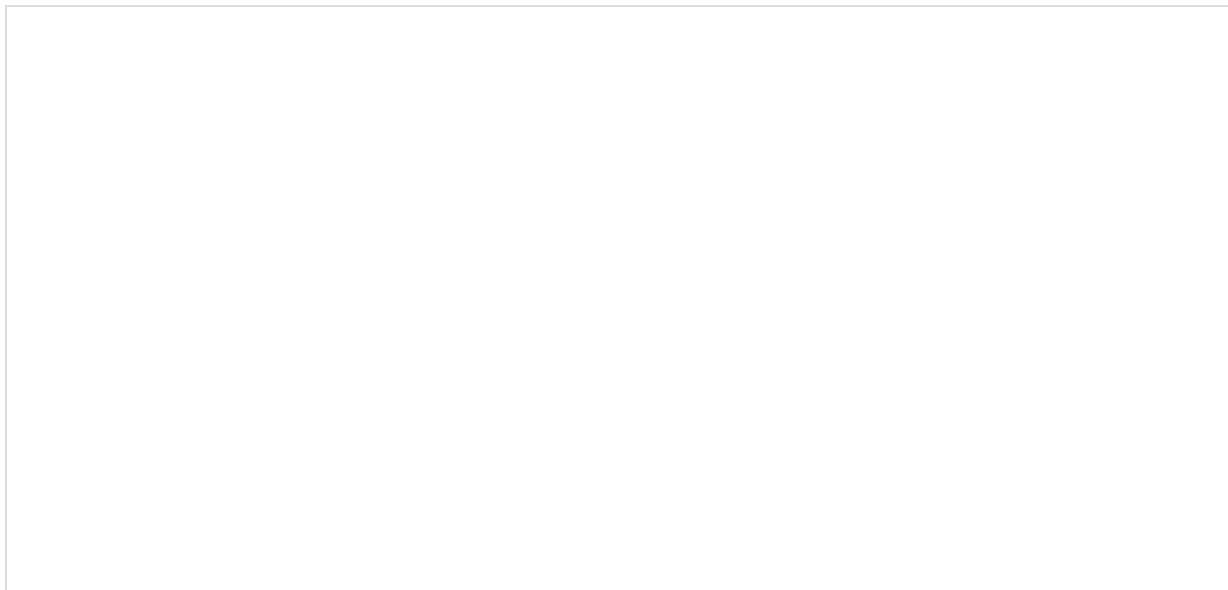
If support of Bootstrap Token management is implemented on the MDM server-side, during device enrollment command Settings / MDMOptions with flag AllowBootstrapToken = True should be sent to the device. In this case, the device will send SetBootstrapToken and GetBootstrapToken requests to the MDM server later at the appropriate time.

To check the current status see "checking status" below

There are two main communications for SecureToken: SetBootStrapToken and GetBootstrapToken.

SetBootstrapToken request

When the first admin account is created (see Figure 1.1), MDM request SetBootstrapToken (requires DEP enrolled client) is sent from the device to the FileWave MDM server. The MDM server handles this request by storing the Bootstrap Token for the device in the appropriate place (FileWave encrypts and stores this in the inventory database).



A ABC ↕

Create a Computer Account


Fill out the following information to create your computer account.

Full name:

Account name:
This will be the name of your home folder.

Password:

Hint:



←
Back
→
Continue

Figure 1.1

Warning: There is one limitation - If you automatically create an admin during enrollment, the SetBootstrapToken request will not be sent.

DEP Profile
Profile Name

Sent

A human-readable name for the profile.

required

The URL of the MDM server.

https://preview.filewave.com:20443/ios/dep_enrollment

Information
Options
Setup Assistant
Account
Anchor Certs
Supervising Certs
Device Naming
Activation Lock Settings

MacOS Primary Account Setup

☒ Prompt user to create an account of type:

☐ Standard
☒ Administrator

☒ Pre-fill primary account full name and account name

☒ Use device owner's details to pre-fill primary account full name and account name

Full Name:

Account Name:

☒ Allow user to modify primary account full name and account name

Managed macOS Administrator Account

☐ Create managed macOS Administrator Account

Full Name:

Account Name:

Password:

Verify:

☒ Show administrator account in Users & Groups

Cancel
OK

Figure 1.2 - Not Suppressed (Token sent)

This is Default DEP

DEP Profile
Profile Name

Not Sent

A human-readable name for the profile.

required

The URL of the MDM server.

https://preview.filewave.com:20443/ios/dep_enrollment

Information
Options
Setup Assistant
Account
Anchor Certs
Supervising Certs
Device Naming
Activation Lock Settings

MacOS Primary Account Setup

☒ Prompt user to create an account of type:

☐ Standard
☒ Administrator

☒ Pre-fill primary account full name and account name

☒ Use device owner's details to pre-fill primary account full name and account name

Full Name:

Account Name:

☒ Allow user to modify primary account full name and account name

Managed macOS Administrator Account

☒ Create managed macOS Administrator Account

Full Name:

Account Name:

Password:

Verify:

☒ Show administrator account in Users & Groups

Cancel
OK

Figure 1.3 - Suppressed (Token will not be sent)

Setup assistant user is standard

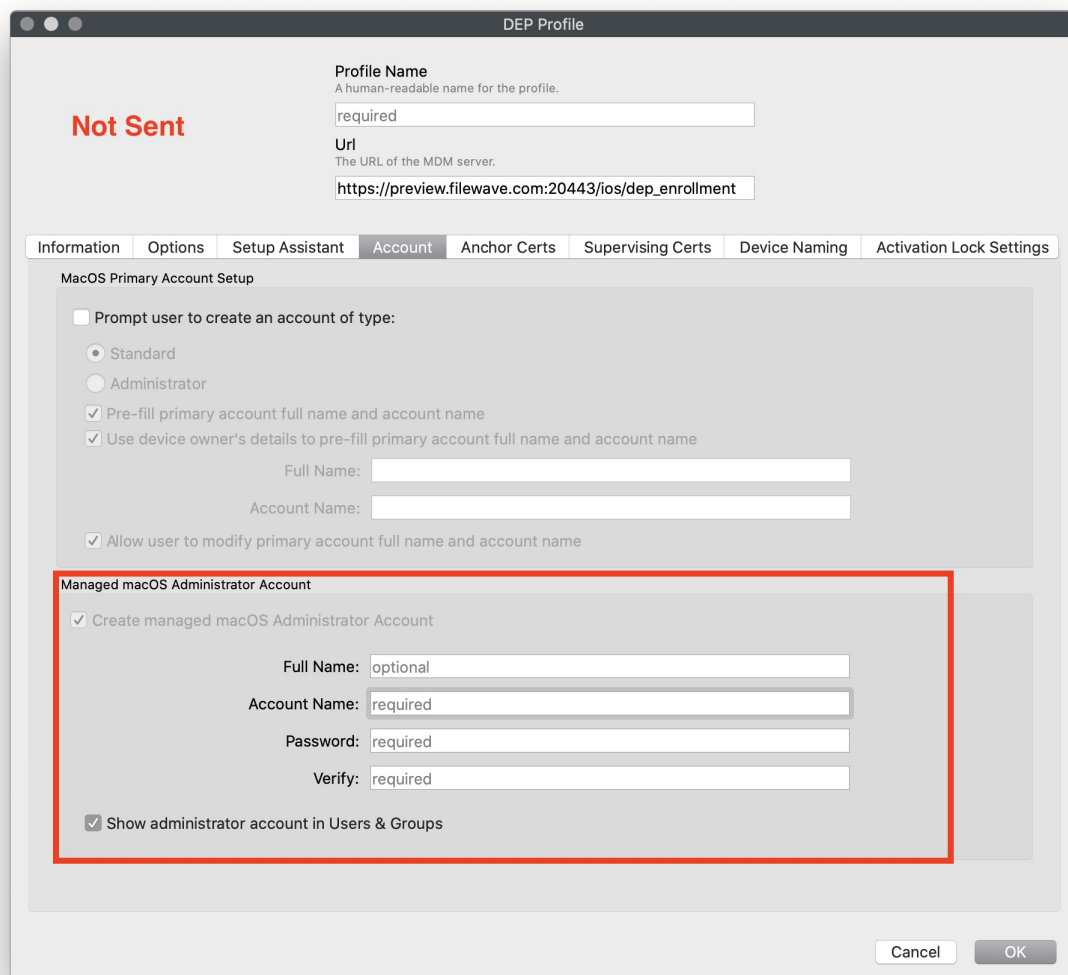


Figure 1.4 - Suppressed (Token will not be sent)

No user during setup assistant

Trigger Escrow

Escrowing of Bootstrap Token to server can be called with:

```
profiles install -type bootstrap token
```

If the SetBootstrapToken request was properly sent to your MDM server, the output will be

```
profiles: Create Bootstrap Token created
profiles: Bootstrap Token created
profiles: Bootstrap Token escrowing to server...
profiles: Bootstrap Token escrowed
```

If you got this error after the initial command:

```
Error: The profile type option was not recognized.
```

The FileWave server may not be able to handle the request; make sure it is running 13.2 or greater

GetBootstrapToken request

When a new mobile account is created (manually or automatically, see "Examples" section) MDM request GetBootstrapToken (requires DEP enrolled client) is sent from the device to MDM server (it requires a Device Enrollment Program enrolled client). The MDM server should handle this request properly by returning the stored Bootstrap Token of the appropriate device. As a result, when a mobile account logs in a SecureToken will be enabled on the account. A list of which accounts can unlock the FileVault disk can be shown by the next command:

```
diskutil apfs listcryptousers /
```

There you can see the UUID of the mobile account as well as the Bootstrap Token External Key:

```
Cryptographic users for disk1s5 (9 found)
|
+-- JJJJJJJJ-UUUU-IIII-0000-AAAAAAAAAAAA
| Type: Local Open Directory User
|
+-- KKKKKKKK-LLLL-MMMM-NNNN-BBBBBBBBBBBB
| Type: MDM Bootstrap Token External Key
|
+-- GGGGGGGG-WWWW-QQQQ-ZZZZ-CCCCCCCCCCCC
| Type: Local Open Directory User
|
+-- BBBBBBBB-AAAA-MMMM-BBBB-AAAAAAAAAAAA
| Type: iCloud Recovery User
|
+-- DDDDDDDD-0000-EEEE-DDDD-EEEEEEEEEEEE
| Type: iCloud Recovery External Key
|
+-- TTTTTTTT-0000-WWWW-TTTT-WWWWWWWWWWWW
| Type: Personal Recovery User
|
+-- NNNNNNNN-EEEE-SSSS-NNNN-EEEEEEEEEEEE
| Type: Institutional Recovery User
|
+-- JJJJJJJJ-0000-CCCC-JJJJ-CCCCCCCCCCCC
| Type: Institutional Recovery External Key
|
+-- AAAAAAAA-EEEE-FFFF-CCCC-DDDDDDDDDDDD
| Type: Personal Recovery Key
```

Compare that list with

```
sudo fdesetup list
```

to show the same UUIDs of the accounts that have SecureTokens:

```
localadmin, JJJJJJJJ-UUUU-IIII-0000-AAAAAAAAAAAA
mobileaccount,GGGGGGGG-WWWW-QQQQ-ZZZZ-CCCCCCCCCCCC
```

Check Escrow/Server Status

To check if Bootstrap Token was escrowed to the server, the same command as above can be used:

```
profiles status -type bootstraptoken
```

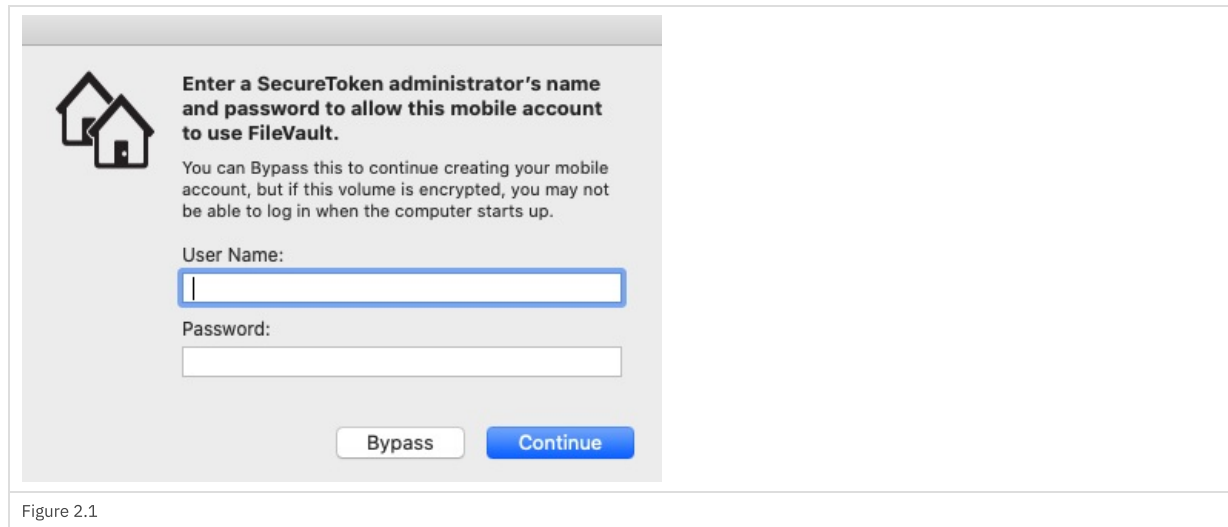
Server Supported	Server Does not support
<pre>profiles: Bootstrap Token supported on server: YES</pre>	<pre>profiles: Bootstrap Token supported on server: NO</pre>
<div>Not Escrowed</div> <div>If the Bootstrap Token was not escrowed on the server (SetBootstrapToken request was not sent to MDM server or was not handled by MDM server properly), the output will be:</div> <pre>profiles: Bootstrap Token escrowed on server: NO</pre> <div>You can manually trigger (see "Trigger Escrow")</div>	<div>Escrowed Done</div> <div>If the Bootstrap Token is already on the server the output will be:</div> <pre>profiles: Bootstrap Token escrowed on server: YES</pre>

Examples when Bootstrap Token helps

Example 1 - mobile account is created manually by an admin which doesn't have a Secure Token

Problem:

In macOS 10.15+ with FileVault is turned on, a network account login without Secure Token escrowed will be prompted for an admin who does have a secure token.



Solution:

With the token escrowed, there is no need to enter a SecureToken administrator's name and password anymore - "Bypass" can be safely pressed and the new mobile account will be shown at the startup window after the device reboots.

- ✓ To disable SecureToken for an admin, use:

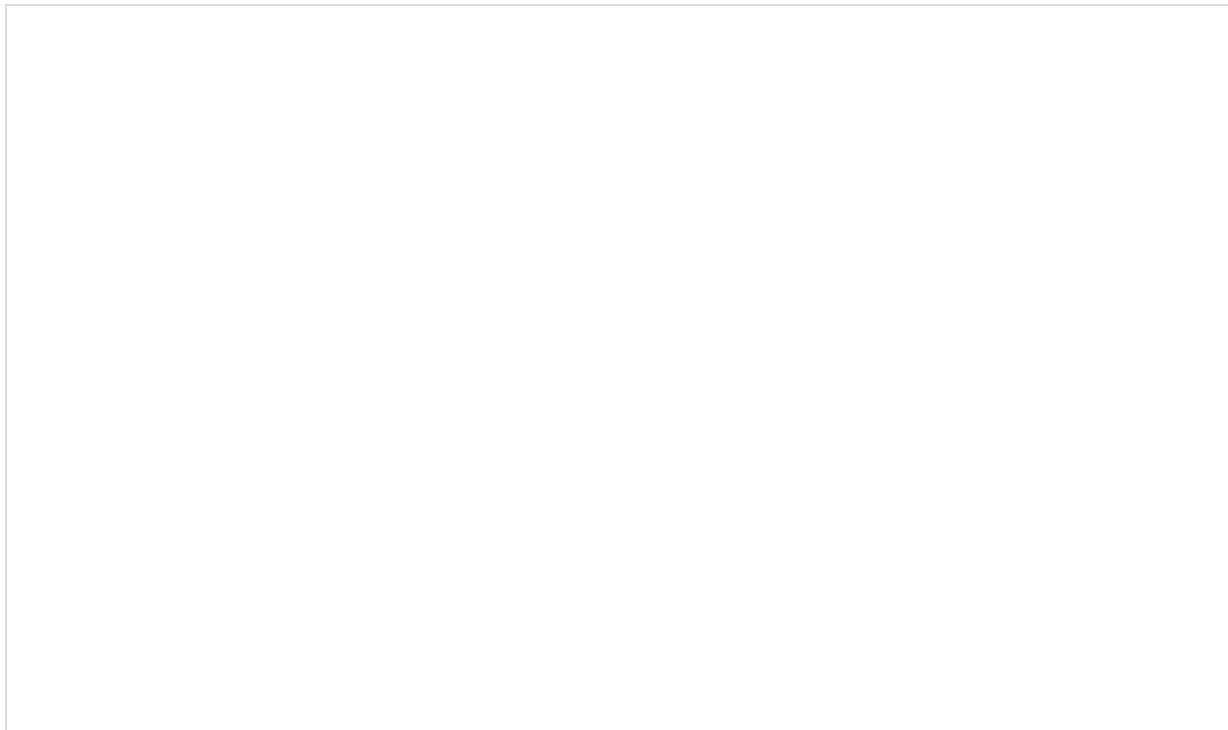
```
sysadminctl -secureTokenOff
```

- ✓ To check if Secure Token is enabled or not, use:

```
adminsysadminctl -secureTokenStatus
```

Example 2 - mobile account is created automatically when profile Mobility is used

With macOS 10.15 and FileVault turned on, profile Mobility was installed on the device with the option 'Create mobile account when user logs in to network account'.



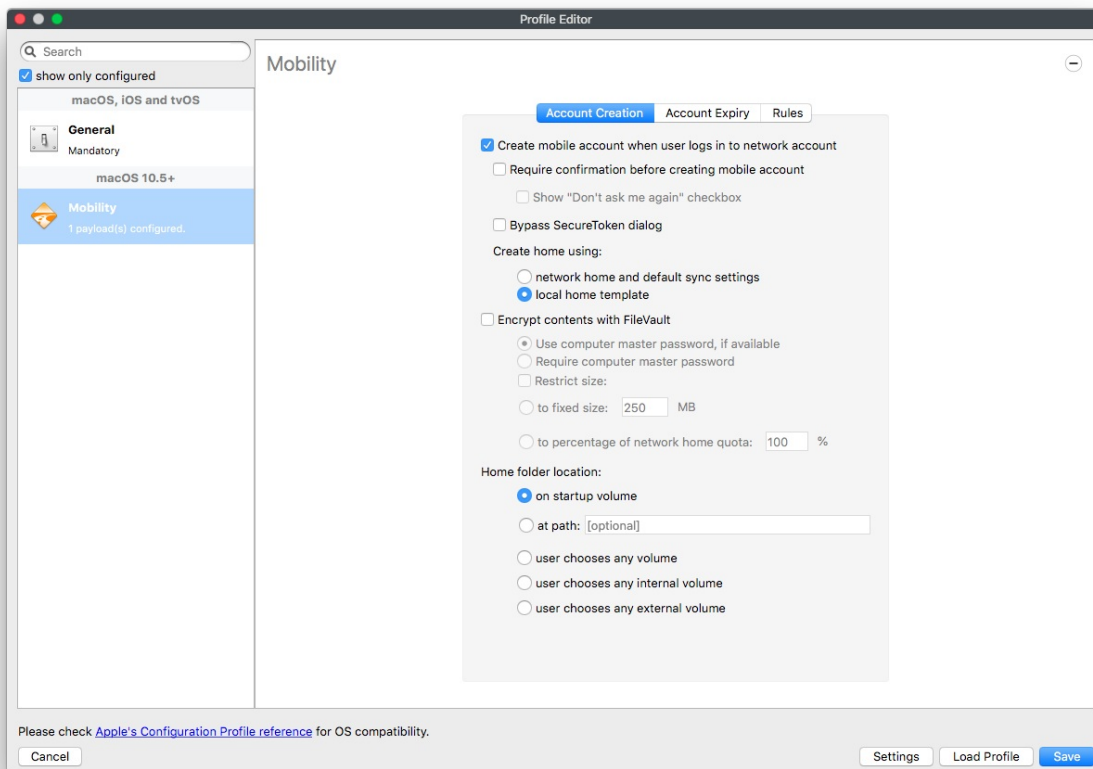


Figure 2.2

After a network account login, a mobile account is created automatically (as the "Bypass" button from the dialog "Enter a SecureToken ..." was pressed implicitly). A new mobile account will be shown at startup window after the device reboots.

In comparison: when the same actions are done on macOS 10.14, there is one additional dialog (see figure 2.1) to enter a SecureToken administrator's name and password during the auto-creation of a mobile user. If "Bypass" will be pressed, the new mobile account will not be shown at the startup window after the device reboots and FileVault should be enabled manually for each new mobile account.

Summary

If FileVault is already encrypted, the new mobile account(s) will be added to the preboot unlock user list automatically. No need to update preboot and do other additional work like past OSes.

🔄Revision #5

★Created 24 July 2023 14:12:17 by Josh Levitsky

✍Updated 23 October 2024 14:44:35 by Josh Levitsky