

Verifying macOS Firmware with EFlgy

Background:

Duo discovered that Macs have been failing (on occasion) to update their firmware for awhile now, and they detailed it in a blog post on their website. The TL;DR is that firmware is supposed to be updated as necessary during security patches, combo updates, dot releases of macOS, or new versions of macOS like High Sierra, but sometimes it fails to download the firmware updates. The fix is to re-run the installer for the macOS version you're already running on, and ensure you have an active internet connection so that the computer can download the required firmware. There is absolutely no way to download only the firmware updates - they are only packaged with the previously listed macOS updates.

The Problem:

Having out-of-date firmware can be detrimental to the security of computers. Unfortunately, Jamf Pro has no built-in mechanism to report on whether the firmware is valid or not, so we don't know what machines need to have the installer re-run.

The Solution:

Duo wrote a nifty little [Python script](#) that detects whether a Mac has the firmware to match the hardware and the current installed version of macOS. This script doesn't integrate directly with the JSS, but there is a workflow to utilize the script alongside an Extension Attribute to get accurate reporting for enrolled machines. We need a Policy that deploys a Package and then runs a Script, along with an Extension Attribute to throw the data into. Admins can then use that Extension Attribute in Smart Groups to base scoping criteria on.

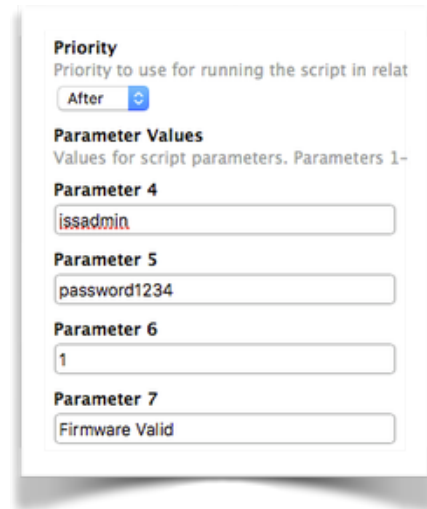
At this point, EFlgy does not work on High Sierra.

The Workflow:

1. [Download this package](#). It contains the Python script and a .pem file necessary for the script to reach out and get the information, and puts both of them into /tmp/EFlgy
 - a. Upload the package to the JSS
2. [Download this script](#). It will run the Python script, create a log file, parse the log file, and then update an Extension Attribute for the computer it was run on
 - a. Upload the script to the JSS
3. Create a Computer Extension Attribute in the JSS. Call it whatever you like ("Firmware Valid" makes good sense) and set the Data Type to String, and the Input Type to Text Field.
 - a. Make note of the Display Name, as well as the ID of the EA after you save it. We'll need this for the script parameters

4. Create a Policy in the JSS. I recommend triggering this at Recurring Check-In, Once per Month. (There's really no need to continually verify firmware - it would only change when there's some sort of macOS update, which usually isn't all that often)
5. Add the Package payload and select the previously uploaded package containing the Python script and .pem file. Set this to Install.
6. Add the Scripts payload and select the previously uploaded script. Set the Priority to After, so the package will install first.
7. Configure values for the following parameter / variable pairs:

Parameter #	Variable
4	API Username
5	API Password
6	Extension Attribute ID
7	Extension Attribute Display Name



Priority
Priority to use for running the script in relat

After

Parameter Values
Values for script parameters. Parameters 1-

Parameter 4
issadmin

Parameter 5
password1234

Parameter 6
1

Parameter 7
Firmware Valid

8. Configure the scope. I would suggest scoping the Policy to All Computers and Excluding a Smart Group that detects computers running High Sierra, given that EFIgy does not currently support it.

What Happens During the Workflow:

1. The Policy is triggered
2. The package downloads to the machine and installs itself into /tmp, creating an EFIgy folder with the script and .pem inside of it
3. The script comes along and runs a command to kick off the EFIgy Python script and write a log to /tmp/EFIgy/log
4. The script parses the log to see whether the firmware check found a good result or not
5. The script makes an API call to update the Extension Attribute for the computer's Inventory Record
 - a. Running a recon is not necessary here, the API call updates the EA instantly
 - b. If the firmware is good, the EA will report "True." If the firmware is bad, it will report "False"