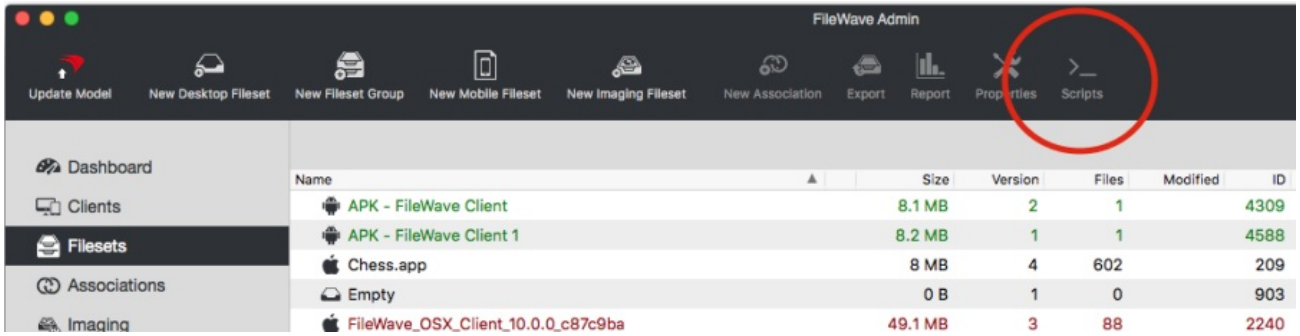
Fileset Scripts

Fileset Scripts Overview

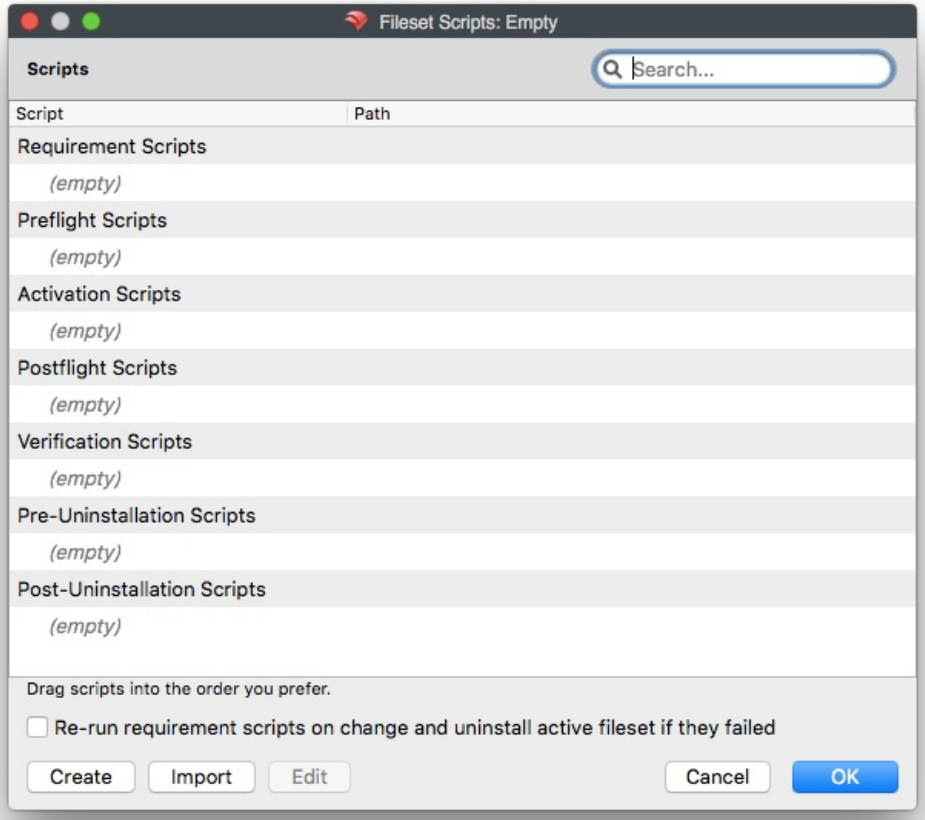
FileWave 11+ provides the ability to run a script at any of seven stages of Fileset deployment (called Activation States):

- Requirements
- Preflight
- Activation
- Postflight
- Verification
- Pre-Uninstallation
- Post-Uninstallation

In FileWave Admin, while in the Filesets view, the toolbar now contains a Scripts icon.



When you select a given Fileset, then click on the Scripts icon, the Scripts dialog opens



The dialog shows the scripts that will be executed for the given Fileset and the activation state in which they will be executed. The order in which scripts of the same activation state and Fileset are executed is the same as they appear in the list (i.e. from top to bottom). You can drag & drop scripts in order to change the execution order.

You can create and import scripts by clicking the corresponding buttons. Editing a script is also possible, so is dragging and dropping a script from Finder in order to import it.

Any changes to the Fileset will be applied when you click OK. If you click Cancel, the current changes will be lost and the Fileset will not be modified.

Scripts in the list can be double-clicked, which causes the file property dialog to appear. You can change most of the attributes of the script in the same way as in the open Fileset dialog. There are, however, certain attributes you cannot change. For instance, you cannot unset the Execute flag; therefore, it is disabled. For requirement scripts, it is not possible to change the interactive/non-interactive option, since the exit code of the script is required to decide whether the Fileset should be downloaded. Therefore, this field is also disabled.

The checkbox "Re-run requirement scripts on change and uninstall active Fileset if they failed" controls the same internal setting as the checkbox "Evaluate requirements on change and uninstall active Fileset if they failed" in the Requirements tab of the Fileset properties. If checked, when a Fileset needs to be updated, the Client checks the requirements of the Fileset again. This includes executing requirement scripts. If any of the requirements or requirement scripts fail, the Fileset will be uninstalled.

Fileset Scripts Types

- Requirements Scripts – A requirements script checks the requirement on the Fileset before any dependencies are downloaded. If any requirement script fails (return non-zero), then the Fileset and its dependencies will not be downloaded nor installed.
- Preflight Scripts – A preflight script checks the needs of the Fileset before the Fileset downloads, but after dependencies have been installed. If any preflight script fails (returns non-zero), then the Fileset won't be downloaded or installed.
- Activation Scripts – An activation script is executed upon activation of the Fileset.
- Postflight Scripts – A postflight script is executed after the installation of the Fileset has completed.
- Verification Scripts – A verification script is executed after postflight scripts and upon every "verification of the Fileset."
- Pre-Uninstallation Scripts – A pre-uninstallation script is executed on inactivation of a Fileset and right before a Fileset is uninstalled. (Useful if the script needs to reference a file that will subsequently be removed due to self-healing).
- Post-Uninstallation Scripts – A post-uninstallation script is executed right after uninstalling/removing the Fileset from a client and its dependencies.

Related Content

- Fileset / Payload Script Exit Code Status
- Script Best Practices
- How the Client Communicates

Fileset / Payload Script Exit Code Status

Script Exit Codes

When a script runs, one of the below errors may be the outcome.

Status Value	Status Description	Severity	Status Details
220	Failed! (Will Not Retry)	ERROR	Script exited with a failure, do not automatically retry
210	Success (Skipped Install)	ERROR	Script exited successfully, report fileset as installed but skip actual installation
0	Success	OK	Script exited successfully
-1000	Crashed	ERROR	Script crashed during execution
-1001	Time Out Exceeded	ERROR	Script execution time took longer than Get Info > Executable > 'Wait for executable to finish' > 'Wait for:'
-1002	No Logged In User	ERROR	Script could not run - no user currently logged in
-1003	Failed To Start	ERROR	Script could not run - failure to start the script

Expected behavior

Requirements scripts processing rules

- If any of the requirements scripts returns 220, we stop executing scripts and also stop trying to install the fileset. No further action will be done, unless manually requested by an administrator, or unless a newer version of the fileset is available. The fileset status will be reported as "Requirements Not Met: Will Not Retry".
- If any of the scripts returns non-zero and ≠210, we stop executing scripts. Requirement scripts will be executed again 2 minutes later.
- If any of the scripts returns 210 when all other scripts return 0, the fileset status will be reported as "Skipped". The fileset will not be installed.
- Only if all scripts return 0, then we will install the fileset.

Kiosk

If a requirements script returns 210 or 220, the fileset will not be available in Kiosk.

If the dependency of a fileset has a requirements script that returns 210, it will not affect the availability of the main fileset. However, if it returns 220, the dependencies will fail so the main fileset will not be available in Kiosk.

Dependencies

- If the main fileset is at "Skipped" state, its dependencies will still be processed and installed.
- If a dependency is at "Requirements Not Met: Will Not Retry", that does not propagate up the tree. This means the installation of the main fileset will still fail due to failed requirements, but the main fileset will simply be reported as Download/Activation/Update of dependency fileset failure.

Other scripts

Whenever a requirements script returns 210 or 220, the fileset does not get installed, so other types of scripts (preflight, activation, etc) will not be executed. Only in case the requirements scripts are run again and they change, then the fileset might get installed and in that case other types of scripts will get run as usual.

Inventory

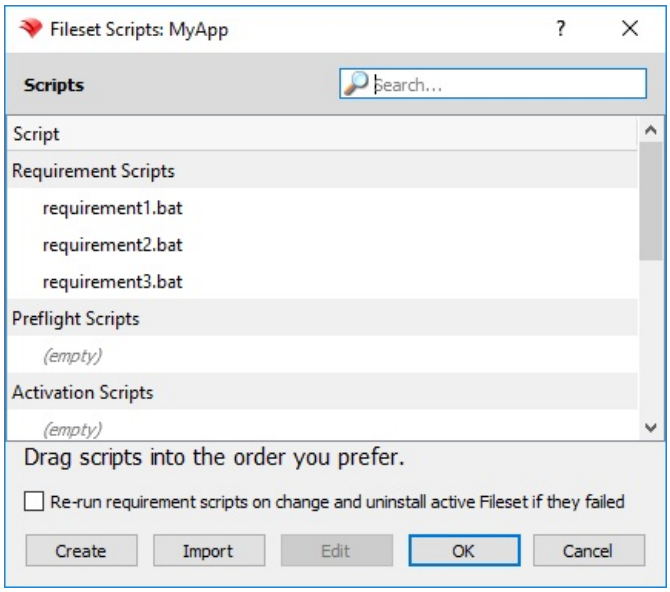
- Filesets in "Skipped" status are reported to inventory, like if they were actually installed.
- Filesets in "Requirements Not Met: Will Not Retry" status are not reported to inventory at all.

Windows Requirement Script Examples

Requirement scripts are executed on client devices with each tickle interval, 2 minutes by default, to check if the installation conditions outlined have been met. For example, it can used to verify if some file, registry key, service, or process is or is not present and decide whether to proceed with fileset activation or not, because you may want to

1. Block redundant installations if the app is already present or
2. Ensure prerequisites are present or
3. Enforce a particular installation order.

If the requirement script returns any other exit code but 0 (e.g. 1 or -1) it is considered a failure and will be reported as a "Requirements Failure: Script" in the Client Info window and Fileset Report. This prevents the contents of the fileset from downloading and installing. As mentioned previously, requirement scripts will be executed every tickle interval and the fileset will be installed only when they all return 0. To check for multiple conditions simply specify multiple requirement scripts.



Below are some sample requirement scripts for Windows that can be customized for your own use. If you want the opposite condition, then flip the exit error codes in the examples.

Install if registry key present

```
::Replace HKLM\path\to\registry\key with the actual path to the registry key, e.g.
HKLM\SOFTWARE\Macromedia\FlashPlayer

reg query "HKLM\path\to\registry\key"
if %ERRORLEVEL% EQU 0 (
    exit 0
) else (
    exit 1
)
```

Install if registry value present

```
::Replace HKLM\path\to\registry\key with the actual path to the registry key containing your value, e.g.
HKLM\SOFTWARE\Macromedia\FlashPlayer
::Replace <value> with the actual name of the value, e.g. CurrentVersion in this example

reg query "HKLM\path\to\registry\key" /v <value>
if %ERRORLEVEL% EQU 0 (
    exit 0
) else (
    exit 1
)
```

Install if file or folder present

```
::Replace <drive>:\path\to\file\or\folder with the actual path to the file or folder, e.g.
%ProgramFiles(x86)%\Mozilla Firefox\firefox.exe
```

```
if exist "<drive>:\path\to\file\or\folder" (
    exit 0
) else (
    exit 1
)
```

Install if application present in Programs & Features

```
::Replace <AppName> with the name of your app, e.g. Adobe Acrobat Reader DC
::Be as specific as possible because partial app names may provide a match when you don't necessarily want it to,
e.g. Adobe will match both Adobe Acrobat Reader DC and Adobe Flash

reg export HKLM\Software\Microsoft\Windows\CurrentVersion\Uninstall %temp%\applist1.txt
reg export HKLM\Software\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall %temp%\applist2.txt
copy %temp%\applist1.txt + %temp%\applist2.txt %temp%\applisttemp.txt
find "DisplayName" %temp%\applisttemp.txt > %temp%\applist.txt

find "<AppName>" %temp%\applist.txt
if %ERRORLEVEL% EQU 0 (
    del %temp%\applist*
    exit 0
) else (
    del %temp%\applist*
    exit 1
)
```

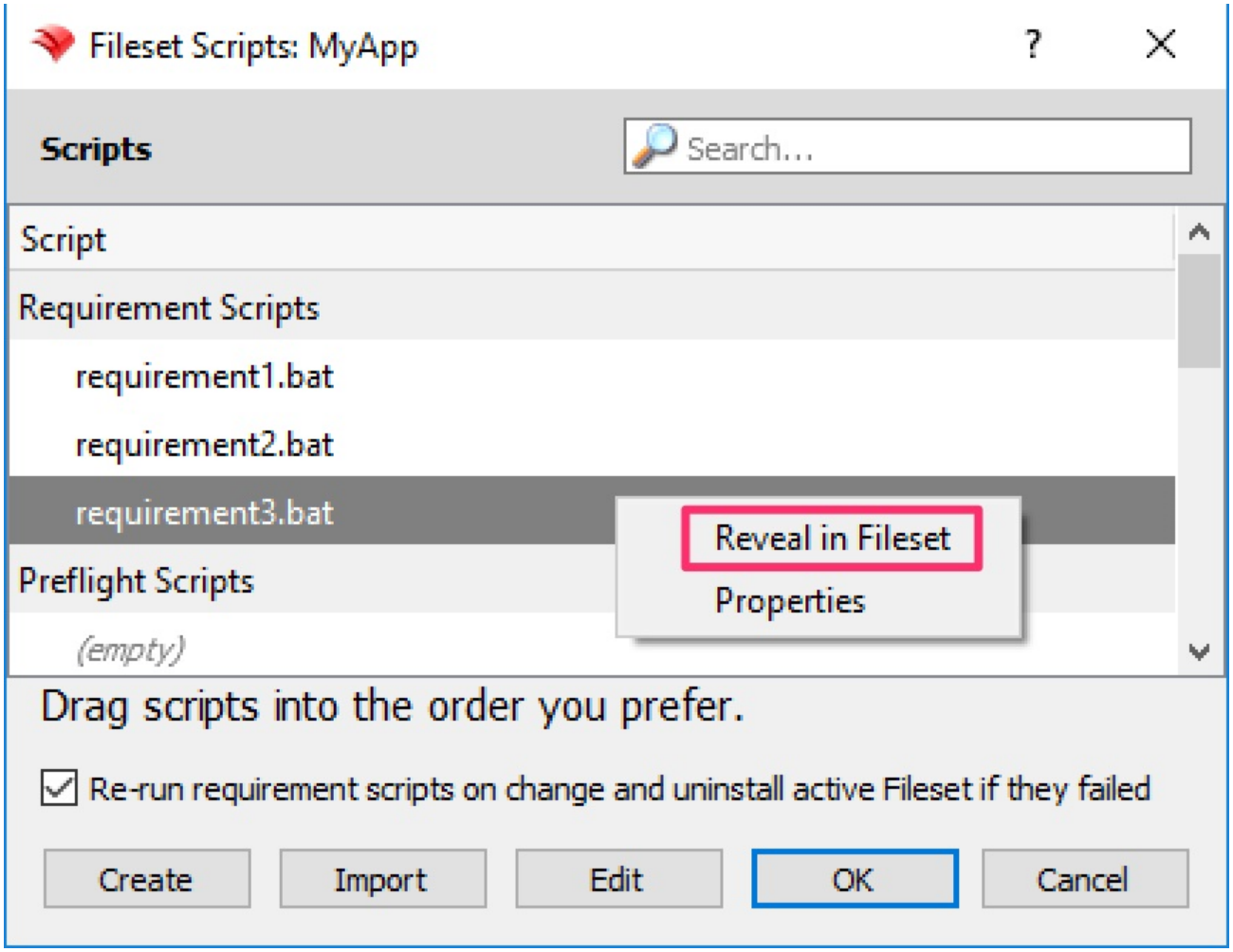
Install if service present

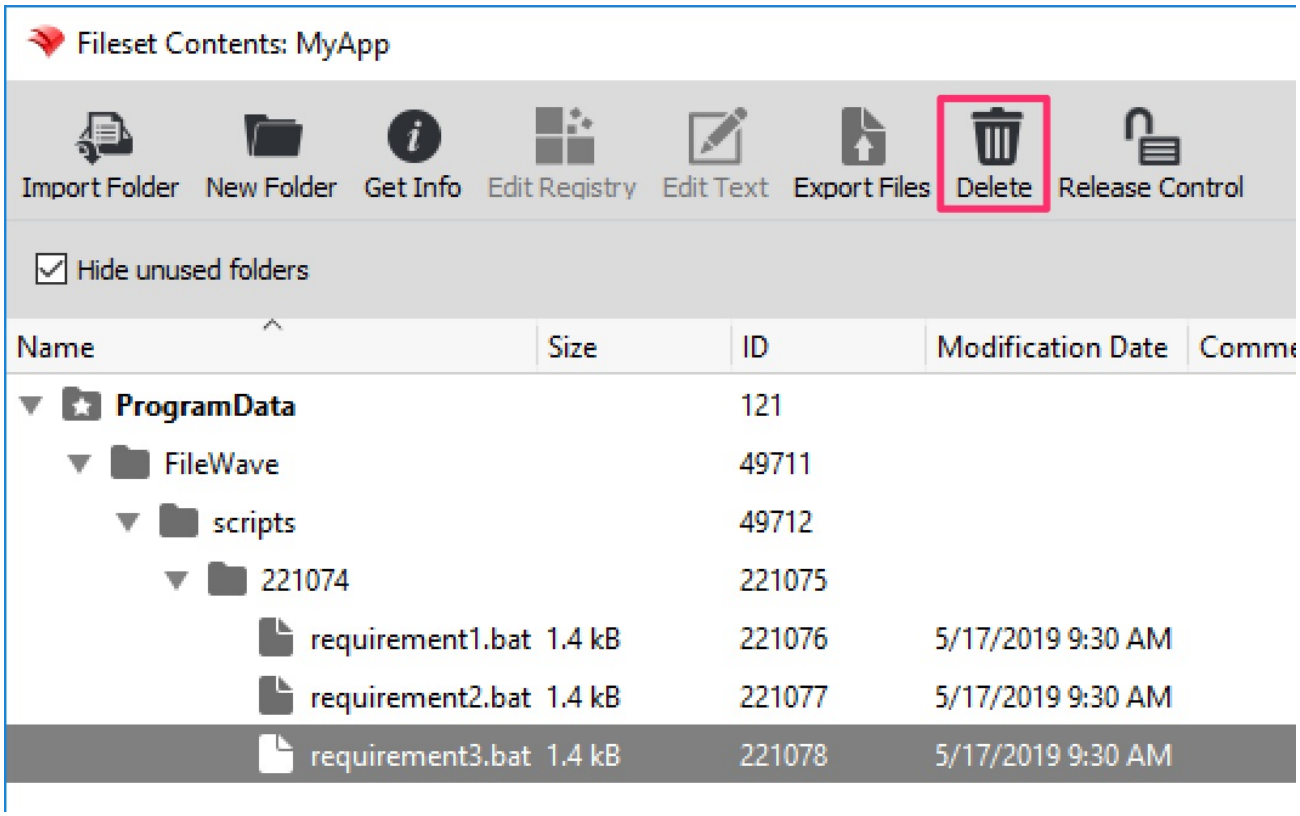
```
::Replace <service> with the name of your service from the Services control panel, e.g. Adobe Acrobat Update
Service
::Be as specific as possible because partial service names may provide a match when you don't necessarily want it
to, e.g. FileWave will match both FileWave Client and FileWave UltraVNC Server

sc query | find "DISPLAY_NAME" | find "<service>"
if %ERRORLEVEL% EQU 0 (
    exit 0
) else (
    exit 1
)
```

Install if process present

```
::Replace <process> with the name of your process, e.g. notepad.exe

tasklist /FI "IMAGENAME eq <process>" 2>NUL | find /I /N "<process>"
if %ERRORLEVEL% EQU 0 (
    exit 0
) else (
    exit 1
)
```

If you find that you need to delete a requirement script for any reason, right-click that script and choose Reveal in Fileset. That will open the Fileset Contents window with the script file highlighted. Click the Delete icon in the toolbar to delete your script.





Remember to always test your requirement scripts locally first before adding them to a fileset. With the above building blocks as examples, you'll quickly be on your way to taking advantage of requirements scripts for providing conditional intelligence to your Windows filesets.

Mitigating Privilege Escalation in Fileset Executables

What

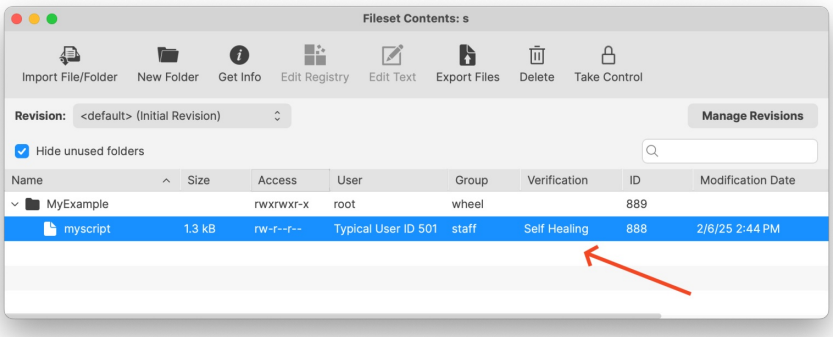
> This article reflects the behavior in FileWave 16.0.0+

When deploying Filesets in FileWave;

- It's possible to include a Verification Script that runs every 24 hours (or at system restart) on Windows and macOS.
- You might also have a Requirements Script that runs every 2 minutes to test for a condition needed to install something.

These scripts are typically used for tasks such as re-applying Group Policy settings—using tools like LGPO.exe on Windows—as an example. However, if an executable (like LGPO.exe) is replaced by a non-administrator, there's a risk that the malicious replacement could be executed with elevated privileges (SYSTEM on Windows or root on macOS).

By default, Filesets benefit from a self-healing mechanism. If an executable is modified, the Fileset will restore the original file before the Verification Script executes. An image below shows this status and picking Get Info for a file may be used to change this on a per-file basis. Properties for a Fileset can be used to change the behavior across the entire Fileset as well.



This same protection applies when the executable is deployed and then called by a Custom Field—the Fileset's verification step precedes the Custom Field execution, thereby mitigating the risk of local privilege escalation.

However, there is one scenario where this protection of Self-Healing does not apply: the Blocker Script, which by default runs every 5 minutes. Its more frequent execution window could allow a local user to replace an executable before the next Fileset verification occurs. However this too can be mitigated as we will discuss below.

When/Why

This article is particularly relevant in environments where:

- Executable files (e.g., LGPO.exe) are deployed via Filesets that also include a Verification Script or are triggered via Custom Fields.
- There is a risk of local users, who lack administrator rights, replacing these executables to attempt privilege escalation.
- The deployment involves Blocker Scripts, which run at more frequent intervals (every 5 minutes), thus presenting a window where an attacker might replace a file before it is healed by the Fileset.

Why is this important?

Ensuring that only the intended executable is run with elevated privileges is critical for system security. A malicious modification could lead to unauthorized privilege escalation. Mitigating this risk by enforcing strict file ownership and permission settings prevents non-administrative users from replacing or modifying executable files.

How

To mitigate this risk, it is recommended to either pick properties on the file in the Fileset and properly securit it or use a Postflight Script in your Fileset deployment that sets the file permissions so that only SYSTEM (on Windows) or root (on macOS) can modify the file.

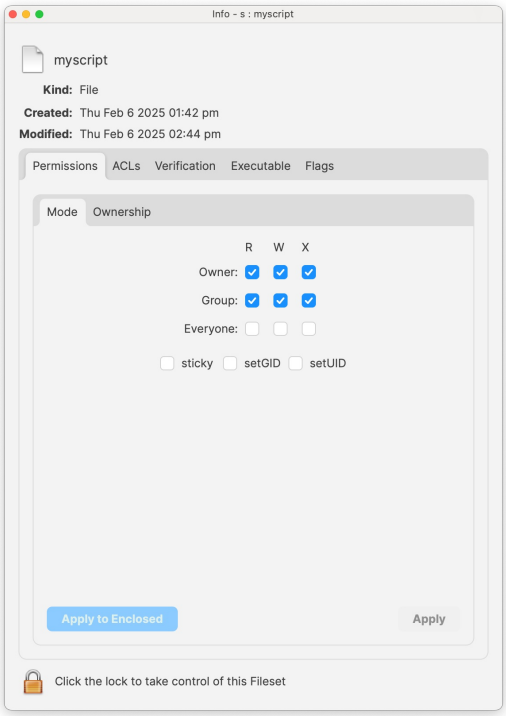
On Windows systems as of FileWave 16.0.0 the c:\ProgramData\FileWave\ directories like FWClient and the scripts directory are restricted so that a non-Administrator is unable to manipulate files located there. This is one way that FileWave will prevent a local attack by default. Files on Windows will also inherit the permissions of the directory that they are deployed in to which will help for a

more consistent permissions experience.

For some examples of setting permissions on files please take a look at the below. For the scripts you could have them run as Fileset scripts when deploying the files to adjust permissions to how you'd like.

macOS Example (Properties of file in Fileset)

In the below images you can see an example file in a Fileset where Get Info has been clicked in the toolbar when viewing the Fileset and the permissions have been adjusted so only root:wheel has access to the file. Make sure you click Apply on each screen to make the setting change.



macOS Example (zsh Script)

Below is an example zsh script that sets the ownership to root:wheel and the permissions to 770 for the target executable.

```
#!/bin/zsh
# Define the path to the executable deployed via the Fileset
TARGET_FILE="/path/to/executable"

# Change the ownership of the file to root:wheel
chown root:wheel "$TARGET_FILE"

# Set the permissions to 770 (owner and group have full permissions, others have none)
chmod 770 "$TARGET_FILE"

# Explanation:
# - chown root:wheel: sets the file owner to root and group to wheel.
# - chmod 770: grants read, write, and execute permissions to the owner and group,
#   while denying any permissions to others.
```

Windows Example (PowerShell Script)

The following PowerShell script sets the file's owner to SYSTEM and adjusts the permissions so that only Administrators and SYSTEM have full control. This ensures that no non-administrator can replace the file.

```
# Define the path to the executable deployed via the Fileset
$TargetFile = "C:\Path\To\executable.exe"

# Set the owner of the file to SYSTEM using icacls
icacls $TargetFile /setowner "SYSTEM"

# Remove inherited permissions to ensure only our defined permissions are in place
icacls $TargetFile /inheritance:r

# Grant full control (F) to SYSTEM
```

```
icacls $TargetFile /grant SYSTEM:(F)

# Grant full control (F) to Administrators
icacls $TargetFile /grant Administrators:(F)

# Explanation:
# - icacls /setowner "SYSTEM": sets the file owner to SYSTEM.
# - icacls /inheritance:r: removes inherited permissions so only our custom permissions apply.
# - icacls /grant SYSTEM:(F) and /grant Administrators:(F):
#   ensures that only SYSTEM and Administrators have full control over the file.
```

Related Content

- FileWave Version 16.0.2 Security Notices
- Verification
- Script Best Practices
- Fileset Scripts Overview