ProfileUnity Log Cleanup

Manual · Version 1.1 · November 2026

ProfileUnity writes log files for every logon and logoff. On persistent desktops, physical machines and RDS / session hosts these logs are never removed automatically, so they pile up over time. The **ProfileUnity Log Cleanup** PowerShell package removes log files older than a retention window you choose and records every run in the Application event log.

Package contents

File	Purpose
`Clear-ProfileUnityLogs.ps1`	The cleanup script. Can be run manually or by the Scheduled Task, Application Launcher or User Defined Script.
`Install-ProfileUnityLogCleanupTask.ps1`	Installs (or removes) a Scheduled Task that runs the cleanup as SYSTEM.
`ProfileUnity Log Cleanup - Manual.pdf`	This manual.

Contents

- What the cleanup does
- Requirements
- Choose a deployment method
- Option 1: Scheduled Task
- Option 2: ProfileUnity Application Launcher
- Option 3: ProfileUnity User Defined Script
- Parameters
- Custom log locations
- Checking the results
- Safety measures
- Troubleshooting

What the cleanup does

- Cleans the ProfileUnity **system** log folder on **single-session** machines (desktops). Default: `C:\Windows\Temp\ProfileUnity`. On multi-session hosts this folder is left alone (see below).
- Cleans the ProfileUnity **user** log folder. Default: `%TEMP%\ProfileUnity`. With `-AllUsers` this folder is cleaned in every user profile on the machine, for example `C:\Users\<user>\AppData\Local\Temp\ProfileUnity`.
- Removes files whose *last write time* is older than the retention window. Newer files are never touched.
- Skips files that are in use (for example the log of a session that is logging on) and reports them.
- Removes empty subfolders, but never the ProfileUnity log folder itself.
- Writes a transcript of every removed or skipped file, and one entry per run to the **Application** event log (source **ProfileUnity Log Cleanup**).

> **⚠ Default log locations only**
> The script cleans the **default** ProfileUnity log locations: `C:\Windows\Temp\ProfileUnity` and `%TEMP%\ProfileUnity`. If you changed the log location in the ProfileUnity logging settings, change it in the script as well. Otherwise the cleanup reports *folder not found* and your logs are not cleaned. To check the user log location, edit the configuration in the Management Console, open **Main** and look at **Client Logging Path** (shown when **Change Client Log Settings** is enabled). Edit `$SystemLogPath` and `$UserLogPath` at the top of `Clear-ProfileUnityLogs.ps1`, or pass `-SystemLogPath` / `-UserLogPath`. See Custom log locations.

> **Multi-session hosts: user logs only**
> On an RDS Session Host or Windows 10 / 11 Enterprise multi-session, the script detects the multi-session host automatically and cleans only the user log folders. The system log folder `C:\Windows\Temp\ProfileUnity` is not cleaned there. Add `-CleanSystemOnMultiSession` only if you do want it cleaned. User logs on a session host are written to the same `%TEMP%\ProfileUnity` folder as on a desktop, so the same settings work on both.

The retention window accepts days or weeks:

Value	Keeps
`7d`, `7 days` or `7`	7 days (default)
`14d` or `2w`	2 weeks
`4w` or `4 weeks`	4 weeks (28 days)
`0`	Nothing: all log files that are not in use are removed

> **⚠ The cleanup permanently deletes files**
> Removed log files do not go to the Recycle Bin. Always preview a run with `-WhatIf` first, and test in a non-production environment. If you need older logs for an open support case, collect them first with the Diagnostic Tool (see How to Collect Logs Using the Diagnostic Tool) before you enable the cleanup.

On **non-persistent** desktops the logs are discarded when the desktop is refreshed, so the cleanup is usually not needed there. If those desktops stay up for a long time, install the Scheduled Task in the golden image.

Requirements

- Windows 10 / 11, or Windows Server 2016 or later.
- Windows PowerShell 5.1 (included in Windows).
- Administrative rights (or SYSTEM) to install the Scheduled Task, clean `C:\Windows\Temp\ProfileUnity`, clean other users' folders and write to the Application event log.
- PowerShell must run in Full Language Mode. The script compiles a small helper with `Add-Type`, which AppLocker / WDAC Constrained Language Mode blocks (see Troubleshooting).

Choose a deployment method

There are three ways to run the cleanup automatically. Pick one per machine; you do not need more than one.

	Scheduled Task	Application Launcher	User Defined Script
Setup	Run the installer on each machine, in the golden image, or with a software deployment tool	Once, in the ProfileUnity Management Console	Once, in the ProfileUnity Management Console
Script location	`C:\Program Files\Liquidware\ProfileUnityLogCleanup` (copied by the installer)	Network share or protected local folder	Network share, or a folder in the Azure Blob container that holds the ProfileUnity configuration
When it runs	At a fixed time, also when nobody is logged on	At the first user logon of the day on each computer	At every user logon
Settings	Installer parameters	**Arguments** field	Defaults at the top of the script
Cleans	System folder (desktops only) and all user profiles	System folder (desktops only) and all user profiles	System folder (desktops only) and the folder of the user who logs on (optionally all profiles)

Option 1: Scheduled Task

The installer copies the cleanup script to `C:\Program Files\Liquidware\ProfileUnityLogCleanup` (which standard users cannot change), writes your settings to `Run-ProfileUnityLogCleanup.ps1` in the same folder, registers the event log source, and creates the Scheduled Task **\Liquidware\ProfileUnity Log Cleanup** running as SYSTEM.

1. Unblock and extract the package

Download the package ZIP. Right-click it, choose **Properties**, tick **Unblock** and click **OK**. Then extract it to a folder, for example `C:\Temp\ProfileUnityLogCleanup`.

2. Open an elevated PowerShell in that folder

Start **Windows PowerShell** with **Run as administrator** and change to the extracted folder:

```
cd C:\Temp\ProfileUnityLogCleanup
```

3. Preview what would be removed

This run deletes nothing. It lists the files that are older than the retention window:

```
.\Clear-ProfileUnityLogs.ps1 -Retention 2w -AllUsers -WhatIf
```

4. Install the Scheduled Task

For example, daily at 03:00 keeping 2 weeks of logs, and run it once right away:

```
.\Install-ProfileUnityLogCleanupTask.ps1 -Retention 2w -Time 03:00 -RunNow
```

Weekly on Sunday at 22:00, keeping 4 weeks:

```
.\Install-ProfileUnityLogCleanupTask.ps1 -Schedule Weekly -DayOfWeek Sunday -Time 22:00 -R
etention 4w
```

To deploy silently with SCCM, Intune or a GPO startup script (running as SYSTEM or administrator):

```
powershell.exe -NoProfile -ExecutionPolicy Bypass -File .\Install-ProfileUnityLogCleanupTa
sk.ps1 -Retention 2w
```

5. Check the result

Open **Event Viewer > Windows Logs > Application** and look for source **ProfileUnity Log Cleanup**, or open the transcript folder `C:\ProgramData\Liquidware\ProfileUnityLogCleanup`. See Checking the results.

Changing settings: run the installer again with the new parameters; the existing task and settings are replaced. An administrator can also edit `Run-ProfileUnityLogCleanup.ps1` in the install folder directly.

Uninstalling: this removes the task and the installed scripts. Transcripts and the event log source are kept for auditing.

```
.\Install-ProfileUnityLogCleanupTask.ps1 -Uninstall
```

To remove the transcripts and the event log source as well:

```
.\Install-ProfileUnityLogCleanupTask.ps1 -Uninstall -RemoveTranscripts -RemoveEventSource
```

Option 2: ProfileUnity Application Launcher

Instead of installing a task on every machine, ProfileUnity can start the cleanup itself through an Application Launcher item. It runs once per day per computer, at the first user logon of the day.

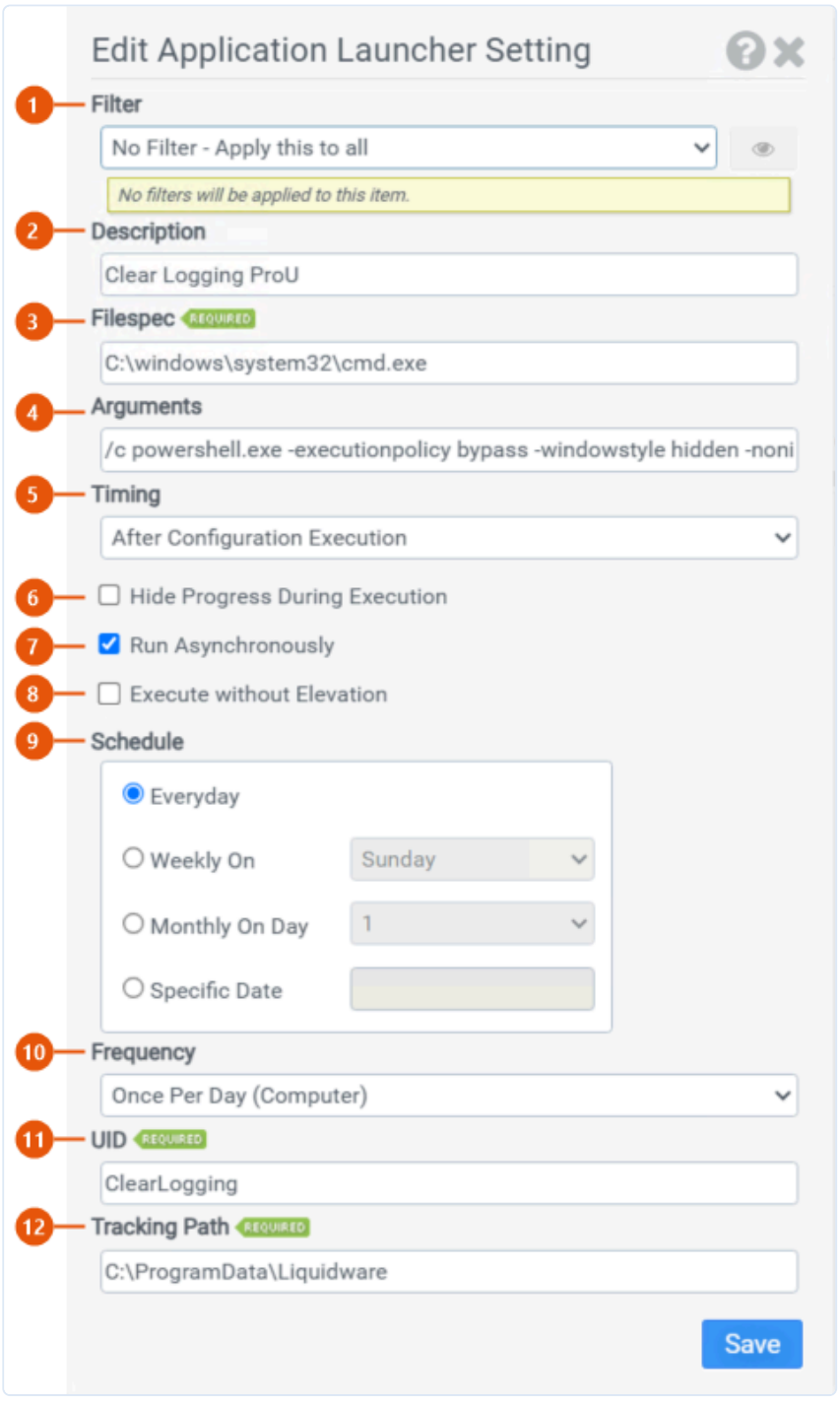
1. Place the script in a protected location

The launcher runs the cleanup **elevated**, so store `Clear-ProfileUnityLogs.ps1` where users can read it but **cannot change it**:

- The ProfileUnity network share, for example `\\fileserver\ProfileUnity\resources\scripts\Clear-ProfileUnityLogs.ps1`. Users and computer accounts need **Read** on the `scripts` folder; only administrators get **Modify**.
- Or a local folder only administrators can change, such as `C:\Program Files\Liquidware\ProfileUnityLogCleanup`, deployed in the golden image or with a software deployment tool.

2. Add an Application Launcher item

In the **ProfileUnity Management Console**, edit the configuration assigned to the machines you want to clean, open the **Application Launcher** module and add a new item with these values:



Field	Value	Notes
Filter	No Filter - Apply this to all	Or a filter that limits the cleanup to, for example, persistent desktops or RDS hosts.
Description	Clear Logging ProU	Free text.
Filespec	`C:\Windows\System32\cmd.exe`	`cmd.exe /c` starts PowerShell with the script.
Arguments	See below	Starts the script hidden, with the retention window and `-AllUsers`.
Timing	After Configuration Execution	Runs after ProfileUnity has applied the rest of the configuration.
Hide Progress During Execution	Off	Optional; the window is already hidden by `-WindowStyle Hidden`.
Run Asynchronously	**On**	ProfileUnity does not wait for the cleanup, so logon is not delayed.
Execute without Elevation	**Off**	Must stay off. The cleanup needs elevation for the system folder, other users' folders and the event log.
Schedule	Everyday	Use **Weekly On** or **Monthly On Day** to clean less often.
Frequency	Once Per Day (Computer)	Because `-AllUsers` cleans every profile, once per computer is enough.
UID	`ClearLogging`	Must be unique within the configuration.
Tracking Path	`C:\ProgramData\Liquidware`	Use a per-computer folder, not one inside a user profile, so *Once Per Day (Computer)* works for all users.

Enter this complete line in the **Arguments** field, with your own script path and retention window:

```
/c powershell.exe -ExecutionPolicy Bypass -WindowStyle Hidden -NonInteractive -NoLogo -NoP
rofile -File "\\fileserver\ProfileUnity\resources\scripts\Clear-ProfileUnityLogs.ps1" -Ret
ention 2w -AllUsers
```

> **⚠ -AllUsers is required in the Arguments field**
> The launcher runs the script elevated as the user who logs on. Without `-AllUsers` only that user's log folder is cleaned. Because the item runs once per day per computer, the log folders of all other users on the machine would never be cleaned.

- Keep `-File` and the double quotes around the path: this makes paths with spaces work and passes the exit code back.
- Other parameters can be added to the end of the line, for example `-SystemLogPath "D:\Logs\ProfileUnity"`. Use **double quotes**; single quotes do not work in this field.
- `cmd.exe` expands variables such as `%TEMP%` before the script starts, so do not use them in this field. For a custom user log location with variables, see Custom log locations.

3. Deploy and test

Click **Save**, then update and deploy the configuration as usual. Log on to a machine that receives the configuration and check the Application event log. The event's *Run as* line shows the user who logged on, and its folder list must show every user profile; if it shows only that user, `-AllUsers` is missing from the Arguments field. The item runs once per day per computer; to test again on the same day, use another computer or change the **UID**.

Option 3: ProfileUnity User Defined Script

Point a User Defined Script at the cleanup script on a file share or, when your ProfileUnity configuration is stored in Azure Blob storage, in a folder of the configuration container. No installation on the machines is needed. ProfileUnity runs the script at **every logon**, as the logged-on user **with elevation**. With the default settings each run cleans the system log folder and the log folder of the user who logs on, keeping 7 days.

1. (Optional) Adjust the defaults in the script

A User Defined Script has no parameters field, so the script uses the defaults in its `param(` block. Before uploading, open the script in a text editor and change only the value, keeping the quotes and the trailing comma:

Setting	Default line	Example change
Retention window	`[string]$Retention = '7d',`	`[string]$Retention = '2w',`
Also clean all other profiles at every logon	`[switch]$AllUsers,`	`[switch]$AllUsers = $true,`
System log folder	`[string]$SystemLogPath = 'C:\Windows\Temp\ProfileUnity',`	`[string]$SystemLogPath = 'D:\Logs\ProfileUnity',`
User log folder (variables allowed)	`[string]$UserLogPath = '%TEMP%\ProfileUnity',`	`[string]$UserLogPath = '%LOCALAPPDATA%\ProfileUnityLogs',`
How long transcripts are kept	`[string]$TranscriptRetention = '4w',`	`[string]$TranscriptRetention = '1w',`
No transcripts at all	`[switch]$NoTranscript,`	`[switch]$NoTranscript = $true,`

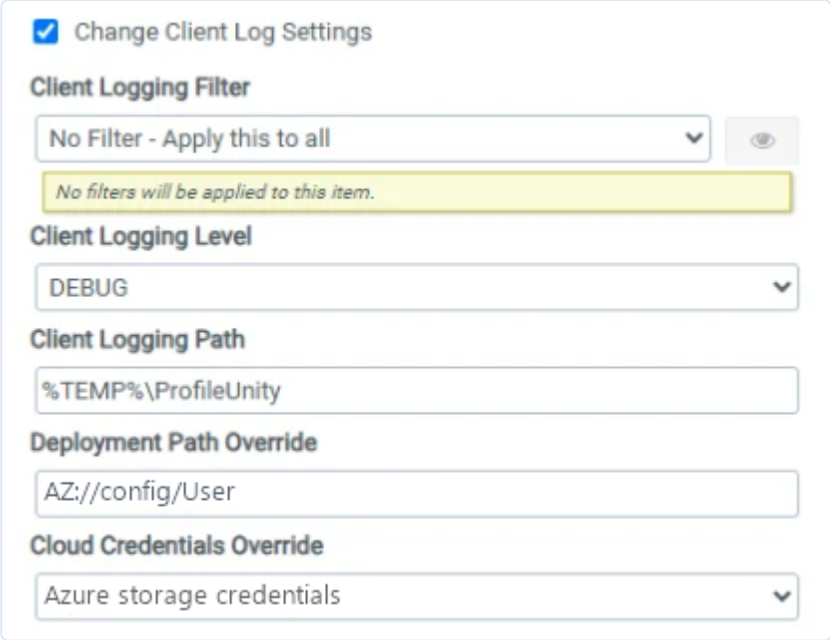
Because the script runs at every logon, it writes one transcript per logon. On busy session hosts, consider a shorter transcript retention or no transcripts; the event log entry is always written. Keep a copy of your changes so you can reapply them to a newer version of the script. Logs in profiles of users who no longer log on are only cleaned if you set `$AllUsers = $true`.

2. Place the script on a share or in the configuration container

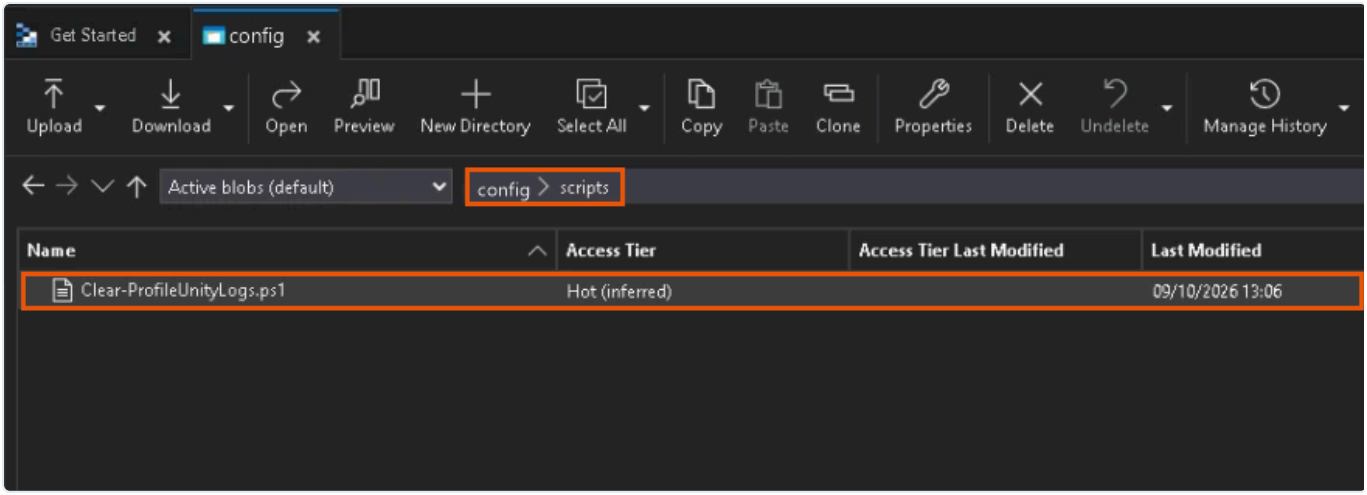
File share: copy the script to the ProfileUnity share, for example `\\fileserver\ProfileUnity\resources\scripts\`. Users and computer accounts need **Read** on that folder; only administrators get **Modify**.

Azure Blob storage: upload it to a folder in the configuration container, for example with **Microsoft Azure Storage Explorer**: open the storage account, expand **Blob Containers**, open the ProfileUnity configuration container, click **New Directory** to create a folder such as `scripts`, open it and click **Upload > Upload Files**.

> **Tip: which container?**
> In the **ProfileUnity Management Console**, edit the configuration and open **Main**. **Deployment Path Override** shows the configuration path, for example `AZ://<container>/User`. The part after `AZ://` is the container to put the script in.



The same page also shows the **Client Logging Path**: the user log location the cleanup must match (see [Default log locations only]).

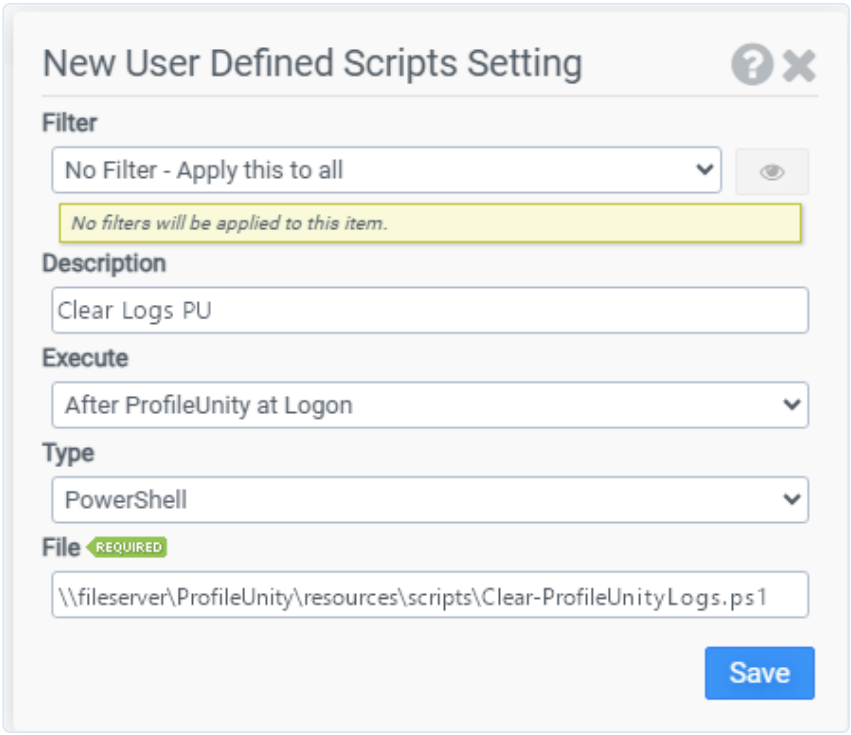


> **⚠ Protect the script location**
> The script runs elevated on every machine that receives the configuration. Make sure only administrators can write to this share or container, just like for the configuration itself. To update the script later, replace the file with the new version.

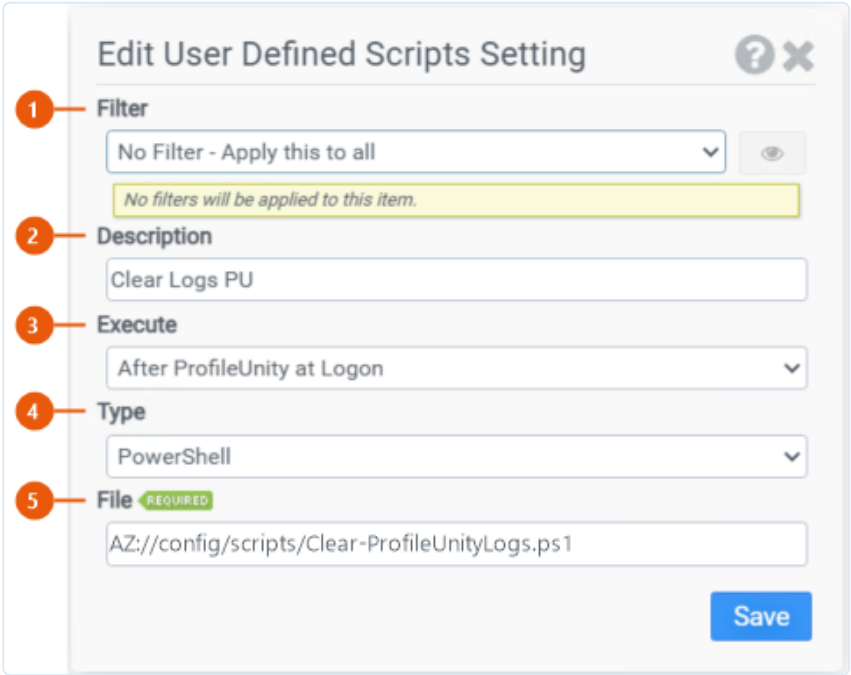
3. Add the User Defined Script

In the **ProfileUnity Management Console**, edit the configuration, open the **User Defined Scripts** module and add a new script:

Script on a file share:



Script in Azure Blob storage:



Field	Value	Notes
Filter	No Filter - Apply this to all	Or a filter for persistent desktops or RDS hosts.
Description	Clear Logs PU	Free text.
Execute	After ProfileUnity at Logon	Runs after ProfileUnity has processed the logon.
Type	PowerShell	

File	`\\fileserver\ProfileUnity\resources\scripts\Clear-ProfileUnityLogs.ps1`
	or
	`AZ://<container>/scripts/Clear-ProfileUnityLogs.ps1`
		Full path on the share, or in Azure Blob storage (replace `<container>` with your configuration container name).

4. Deploy and test

Click **Save**, then update and deploy the configuration. Each logon writes an event to the Application log; its *Run as* line shows the user who logged on. When several users log on at the same time, files already removed by another logon are ignored and not reported as skipped.

Parameters

Clear-ProfileUnityLogs.ps1

Parameter	Default	Description
`-Retention`	`7d`	Retention window.
`-SystemLogPath`	`C:\Windows\Temp\ProfileUnity`	ProfileUnity system log folder.
`-UserLogPath`	`%TEMP%\ProfileUnity`	ProfileUnity user log folder.
`-AllUsers`	Off	Clean the user log folder in every profile (requires administrator). Without it, only the current user's folder is cleaned.
`-Include`	`*`	File name patterns to clean, for example `'*.log','*.txt'`.
`-SkipSystem` / `-SkipUser`	Off	Do not clean the system / user log folders.
`-CleanSystemOnMultiSession`	Off	Also clean the system log folder on a multi-session host. By default it is only cleaned on desktops.
`-KeepEmptyFolders`	Off	Do not remove empty subfolders.
`-TranscriptPath`	`%ProgramData%\Liquidware\ProfileUnityLogCleanup`	Transcript folder.
`-TranscriptRetention`	`4w`	How long transcripts are kept.
`-NoTranscript` / `-NoEventLog`	Off	Do not write a transcript / event log entry.
`-WhatIf`		Show what would be removed without deleting anything.
`-Verbose`		Show every file on screen.

Run the script on its own, for testing or from your own automation. Clean the system folder and all user folders, keeping 14 days:

```
.\Clear-ProfileUnityLogs.ps1 -Retention 14d -AllUsers
```

Clean only the current user's logs, keeping 7 days:

```
.\Clear-ProfileUnityLogs.ps1 -SkipSystem
```

Install-ProfileUnityLogCleanupTask.ps1

The installer accepts the same `-Retention`, `-SystemLogPath`, `-UserLogPath`, `-Include`, `-TranscriptPath`, `-TranscriptRetention`, `-SkipSystem`, `-SkipUser` and `-CleanSystemOnMultiSession` parameters, and always runs the cleanup with `-AllUsers`. In addition:

Parameter	Default	Description
`-Schedule`	`Daily`	`Daily`, `Weekly` or `AtStartup`.
`-Time`	`03:00`	Time of day (24h) for Daily and Weekly.
`-DayOfWeek`	`Sunday`	Day for the Weekly schedule.
`-InstallPath`	`C:\Program Files\Liquidware\ProfileUnityLogCleanup`	Folder the scripts are installed to.
`-RunNow`		Start the task once right after installing.
`-Uninstall`		Remove the task and installed scripts. Add `-RemoveTranscripts` and/or `-RemoveEventSource` to remove those too.

Custom log locations

The script only knows the default locations. If the ProfileUnity logging settings write logs somewhere else, give the script the same location: change `$SystemLogPath` and `$UserLogPath` at the top of the script (required for a User Defined Script), or pass them as parameters to the installer, the Arguments field or the script. The variables `%TEMP%`, `%LOCALAPPDATA%`, `%APPDATA%`, `%USERPROFILE%` and `%USERNAME%` are supported. Put the path in **single quotes** so PowerShell does not change it:

```
.\Install-ProfileUnityLogCleanupTask.ps1 -SystemLogPath 'D:\Logs\ProfileUnity' -UserLogPath '%LOCALAPPDATA%\ProfileUnityLogs'
```

When the cleanup runs with `-AllUsers`, user variables are resolved for every profile, so `%LOCALAPPDATA%\ProfileUnityLogs` becomes `C:\Users\<user>\AppData\Local\ProfileUnityLogs` for each user.

For a network location, use a UNC path (`\\server\share\folder`), not a mapped drive letter: mapped drives are not available to SYSTEM. The computer account needs Modify permissions on that share.

Application Launcher with a custom user path: `cmd.exe` would expand the variables in the Arguments field too early. Instead, create a small `Run-ProfileUnityLogCleanup.ps1` next to the script and point `-File` to it:

```
& (Join-Path $PSScriptRoot 'Clear-ProfileUnityLogs.ps1') -Retention '2w' -AllUsers -UserLo
gPath '%LOCALAPPDATA%\ProfileUnityLogs'
exit $LASTEXITCODE
```

Checking the results

Application event log

Every run writes one event to the **Application** log with source **ProfileUnity Log Cleanup**. The event text lists the files removed, space freed, files skipped, the retention window, the account that ran the cleanup, the result per folder and the transcript path.

Event ID	Level	Meaning
1000	✓ Information	Cleanup completed successfully.
1001	⚠ Warning	Completed, but some files were skipped because they were in use or access was denied. Usually harmless: they are removed on a later run.
1002	✗ Error	A folder was refused for safety reasons, or the cleanup failed. See the event text and transcript.

In Event Viewer, open **Windows Logs > Application**, click **Filter Current Log** and select the source **ProfileUnity Log Cleanup**. Or with PowerShell:

```
Get-WinEvent -FilterHashtable @{ LogName = 'Application'; ProviderName = 'ProfileUnity Log
Cleanup' } -MaxEvents 10 |
    Format-List TimeCreated, Id, LevelDisplayName, Message
```

Monitoring tools such as SCOM or Splunk can alert on event ID 1002 from this source.

Transcripts

Every run writes a transcript to `C:\ProgramData\Liquidware\ProfileUnityLogCleanup` , named `ProfileUnityLogCleanup_<computer>_<user>_<date_time>.log` . Preview runs get `_WhatIf` at the end of the name. Each line starts with:

- `REMOVED` : file removed, with size and last write time.

- `SKIPPED` : file not removed, with the reason (for example in use).
- `WHATIF would remove` : preview runs only.

Transcripts older than the transcript retention (default 4 weeks) are removed automatically.

Safety measures

- Only files older than the retention window are removed; files in use are skipped.
- The script refuses to clean system or profile root folders, such as a drive root, `C:\Windows` , `C:\Windows\Temp` , `C:\Program Files` , `C:\ProgramData` , or a user's profile, AppData or Temp folder. A mistyped path cannot wipe a complete folder tree.
- Junctions and symbolic links are never followed. A log folder that is (or is inside) a junction pointing elsewhere is refused, and every file is verified to really be inside the log folder at the moment it is deleted. This prevents users from redirecting the SYSTEM cleanup to other files.
- The scheduled script is installed in `C:\Program Files` , which standard users cannot modify.
- `-WhatIf` previews every action without deleting anything.

Troubleshooting

Symptom	Resolution
running scripts is disabled on this system or *is not digitally signed*	Unblock the ZIP before extracting, or run the script with `powershell.exe -ExecutionPolicy Bypass -File ...`. If Group Policy enforces `AllSigned`, sign the scripts with your organization's code signing certificate.
No event in the Application log	The event source is registered by the installer, or the first time the cleanup runs as administrator or SYSTEM. Run either once elevated. `-WhatIf` runs never write an event.
Event 1001: files skipped	The files were in use or access was denied. They are removed on a later run; the transcript lists each file and the reason.
Event 1002: *Refusing to clean*	The log path points to a protected folder, or is (or is inside) a junction / symbolic link. Check `-SystemLogPath` and `-UserLogPath`. If profile folders are redirected with a junction on purpose, configure the real target as the log path.
Other users' logs are not cleaned	Run as SYSTEM (the installer does this) or from an elevated PowerShell, and add `-AllUsers`.
Cannot add type, or the script fails under AppLocker / WDAC	The safe-delete helper is compiled with `Add-Type`, which Constrained Language Mode does not allow. Allow the script in your AppLocker / WDAC policy, for example by signing it with a trusted certificate.

If the problem persists, contact Liquidware Support and include the transcript from `C:\ProgramData\Liquidware\ProfileUnityLogCleanup` and the matching Application event.

These scripts are provided as-is, without warranty of any kind. Test them in a non-production environment and use `-WhatIf` before running them in production.