Conclusion and Closing Notes

Last Updated 07/08/22: Alex Farr

- Return to Index

Abstract

This article contains a retrospective analysis of the Firefox: ESR packaged solution after completion.

Content

Process Description

The challenge as stated in Preparation and Environment Notes was to create a PowerShell script which is capable of silent installation and uninstallation of Firefox ESR. The script also needs to make the necessary changes to the Firefox installation to set the default homepage for all users to the UNG Main Page.

The Firefox MSI package was downloaded into `res` in the project directory. The challenge recommends investigating PSAppDeployToolkit for the solution, which has been downloaded to `res` as well. PSADT contains a folder structure named `Toolkit` with a set of dependencies, configs, and supporting scripts, as well as a main logic script `Deploy-Application.ps1` and an exe wrapper by the same name.

According to the documentation for PSADT, the toolkit folder is a portable framework that will be the skeleton of the final packaged deployment. The toolkit folder was copied into `src\Firefox ESR`, then `src\FirefoxESR\Files` and `src\FirefoxESR\SupportFiles` directories were added to the package, and the Firefox ESR MSI package was copied from `res` into `src\FirefoxESR\Files`. All external resources required by the initial scope are assembled in the workspace at this point and work can begin.

Following these setup procedures, the modifications to `Deploy-Application.ps1` that are described in Main Process Worknotes can be carried out, the details of which will be omitted here for organizational clarity.

Supplemental Configuration

Version detection built into the PowerShell script would likely limit the ability to keep the deployment environment in compliance if it is only executed when the package is deployed. A supporting PowerShell script should be built into the Firefox deployment package in SCCM as a custom script detection method so that the version number can be polled from the client independently of the install process. This allows for more highly responsive compliance management in the case of a user initiated rollback for instance.

In practice, the script would pull the value of `CurrentVersion` from `HKLM:SOFTWARE\Mozilla\Mozilla Firefox ESR\` on the target host and process a regular expression match such as `([0-9])\d+` into a new comparable value consisting of only the major version number. This can be logically compared to ensure that the installed version is greater-than or equal-to the target deployment version, and the script can return a response to the SCCM Server if the device is compliant. This will allow for updates to be installed by the user without being rolled back into compliance with the deployed version. Alternatively, strict matching can be enforced to force rollbacks from unsupported updates initiated by the user.

Notes

As pointed out in the Issues heading of Main Process Worknotes, this particular MSI has a problem with uninstallation. These issues appear to be rooted in how the MSI sets up it's registry values during installation and would likely require modification of the MSI itself. When attempting to uninstall Firefox ESR using:

```
Execute-MSI -Action Uninstall -Path 'Firefox Setup 102.0esr.msi'
```

the following error is generated in the output logs:

```
[Uninstallation] :: Skipping removal of application [Mozilla Firefox ESR (x64
en-US)] because unable to discover MSI ProductCode from application's registry
Uninstall subkey [Mozilla Firefox 102.0 ESR (x64 en-US)]
```

This appears to stem from the fact that PSADT expects an MSI code to be in `UninstallString` instead of a path to a helper exe. Currently PSADT is passing a filepath to `msiexec.exe -x` instead of the MSI code it needs to run correctly.

Likewise, when attempting to target the `UninstallString` key of Firefox's Uninstall entry in the registry using:

```
$uninstallPath = Get-ItemPropertyValue -Path
'HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\Mozilla Firefox
102.0 ESR (x64 en-US)' -Name UninstallString

Execute-Process -Path $uninstallPath -Parameters '/S'
```

the following error is generated in the output logs:

```
[Uninstallation] :: Function failed, setting exit code to [60002].
Error Record:
-------------At C:\Users\leToads\Firefox-ESR-
Deployment\src\FirefoxESR\AppDeployToolkit\AppDeployToolkitMain.ps1:3055 char:8
+ ...         If (([IO.Path]::IsPathRooted($Path)) -and ([IO.Path]::HasExte ...
+                 ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
Error Inner Exception(s):
-------------------------
```

This appears to be an error internal to the PSADT toolkit, and would take more time to research a solution for. However it could be the basis of a more reliable implementation if resolved.