Chuco Gives a FAQ 2024: Deploy Version Control Guide

Version: 1.0

Last Update: Oct 22nd, 2024

Table of Contents

Overview

This guide details a version control strategy to be used in Tanium Deploy for software packages that do not have a queryable attribute to be used for Install Verification.

Example

Say you have a set of desktop icons that need to be deployed to every workstation's desktop in the environment and sometimes these icons need to be updated for a URL change or a policy document update, etc. We could copy these icons via a platform package, but there's no way to know what "version" of the icon set is currently on the desktop of a given machine or easily track that the deployment of the package was successful.

Solution

The solution detailed below works by adding a registry value via a command in a Deploy package with the current date formatted as yyyyMMdd. Then, using that registry value for install verification of the Deploy package by checking if the date is equal to or larger then the date when the package was created or updated. This gives us a dynamic version control strategy that is easy to update for future versions of the package.

Implementation

Formatting the Run Commands

Install and Update Operations

This process uses a one line PowerShell command to set the registry value to the current date.

Here's the full command in one line:

```
Unset
powershell.exe -ExecutionPolicy Bypass -WindowStyle Hidden -NoProfile -Command
"$regKeyPath = 'HKLM:\SOFTWARE\customer_name\TaniumDeploy'; $regKeyName =
'package_name'; if (-not (Test-Path $regKeyPath)) { New-Item -Path $regKeyPath
-Force }; New-ItemProperty -Path $regKeyPath -Name $regKeyName -Value
((Get-Date).ToString('yyyyMMdd')) -PropertyType String -Force"
```

Let's break this down:

First we call PowerShell and specify a few parameters.

```
Unset
powershell.exe -ExecutionPolicy Bypass -WindowStyle Hidden -NoProfile -Command
```

Next we have the PowerShell command itself that starts with defining variables. Change these variables in the one line command above to fit your needs.

```
Unset
$regKeyPath = 'HKLM:\SOFTWARE\customer_name\TaniumDeploy'
$regKeyName = 'package_name'
```

Now we set the registry value data to the current date in the format of 'yyyyMMdd' and create the key if it does not exist.

```
Unset
if (-not (Test-Path $regKeyPath)) { New-Item -Path $regKeyPath -Force }
New-ItemProperty -Path $regKeyPath -Name $regKeyName -Value
((Get-Date).ToString('yyyyMMdd')) -PropertyType String -Force"
```

Remove Operation

For the remove operation, we need to delete the key.

Here's the full command in one line:

```
Unset
powershell.exe -ExecutionPolicy Bypass -WindowStyle Hidden -NoProfile -Command
"$regKeyPath = 'HKLM:\SOFTWARE\customer_name\TaniumDeploy'; $regKeyName =
'package_name'; Remove-ItemProperty -Path $regKeyPath -Name $regKeyName -Force;
if (!(Get-ChildItem -Path $regKeyPath)) { Remove-Item -Path $regKeyPath -Force
}"
```

Let's break this down:

First we call PowerShell and specify a few parameters.

```
Unset
powershell.exe -ExecutionPolicy Bypass -WindowStyle Hidden -NoProfile -Command
```

Next we have the PowerShell command itself that starts with defining variables. Change these variables in the one line command above to match the "Install and Update Operations" command.

```
Unset
$regKeyPath = 'HKLM:\SOFTWARE\customer_name\TaniumDeploy'
$regKeyName = 'package_name'
```

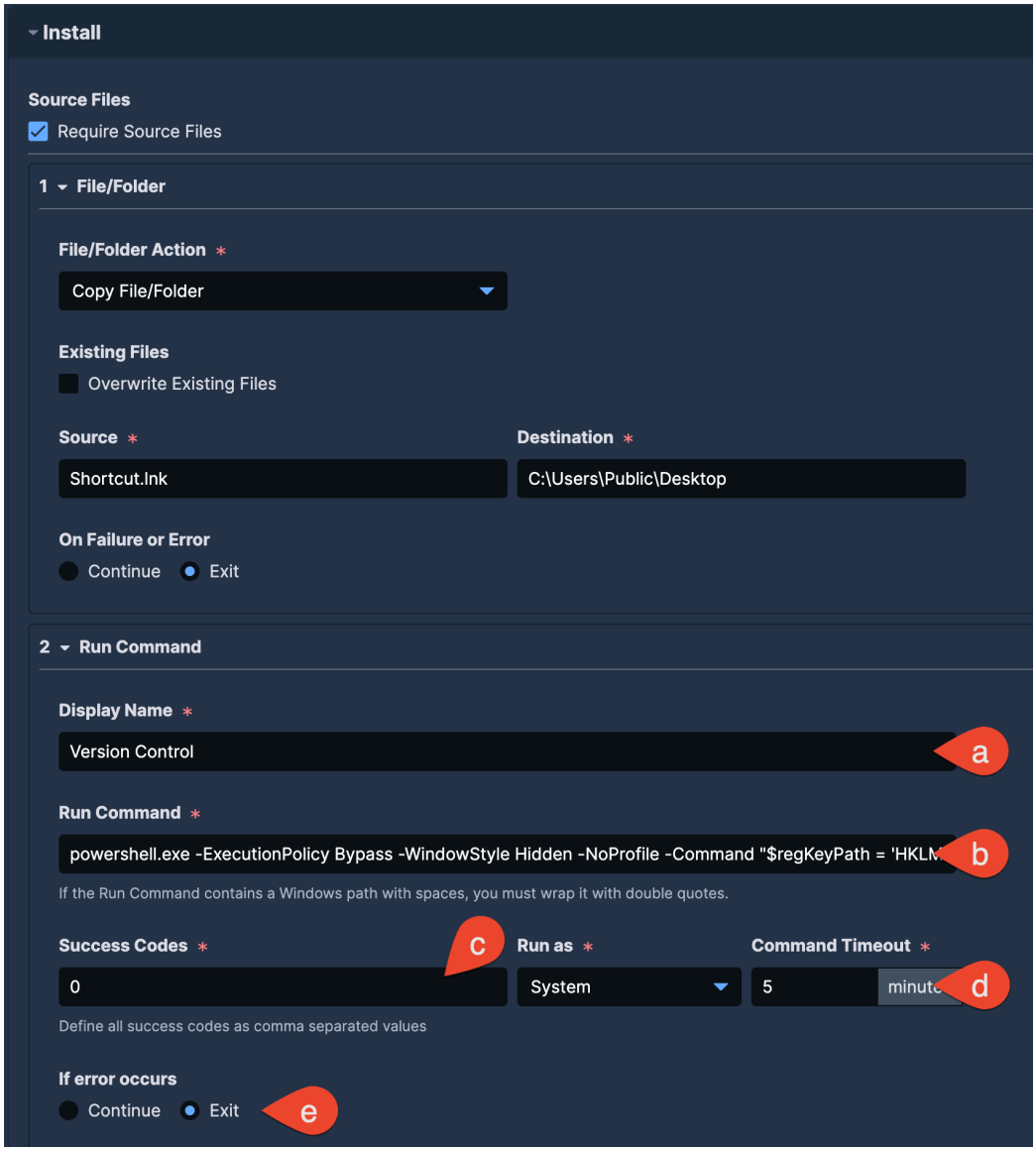
Now we remove the registry value and the parent key if it is now empty to avoid leaving an orphaned key.

```
Unset
Remove-ItemProperty -Path $regKeyPath -Name $regKeyName -Force; if
(!(Get-ChildItem -Path $regKeyPath)) { Remove-Item -Path $regKeyPath -Force }
```

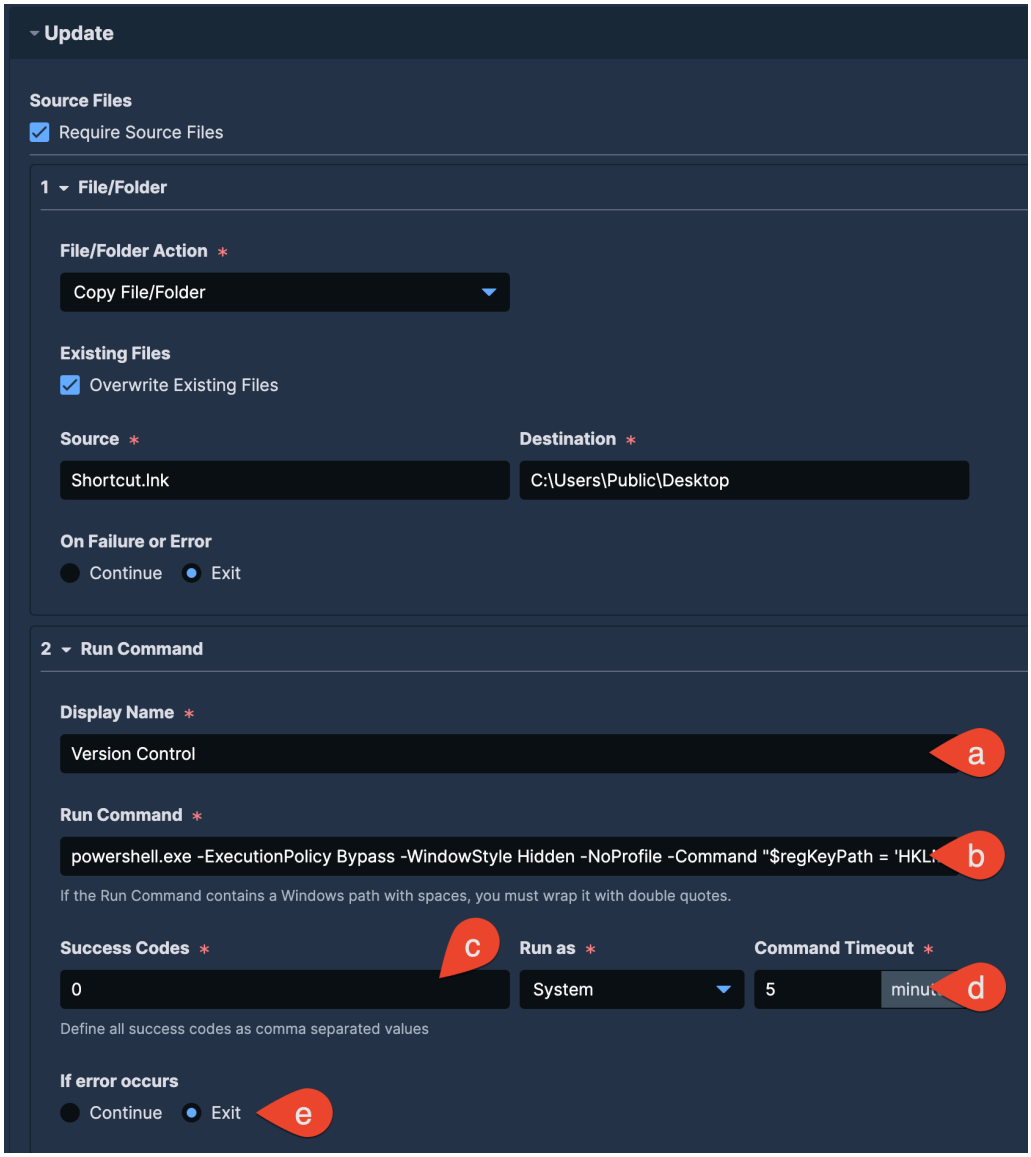
Setup Version Control in the Deploy Package

1. Add the updated “Install and Update Operations” PowerShell command from above as the last step for the Install and Update Operations (if applicable). Be sure to exit on failure for the previous steps as we don’t want to set the registry value if the package failed.
    a. **Display Name**: Version Control
    b. **Run Command**: Enter “Install and Update Operations” PowerShell Command
    c. **Success Codes**: 0
    d. **Command Timeout**: 5 minutes (Default)
    e. **If error occurs**: Exit

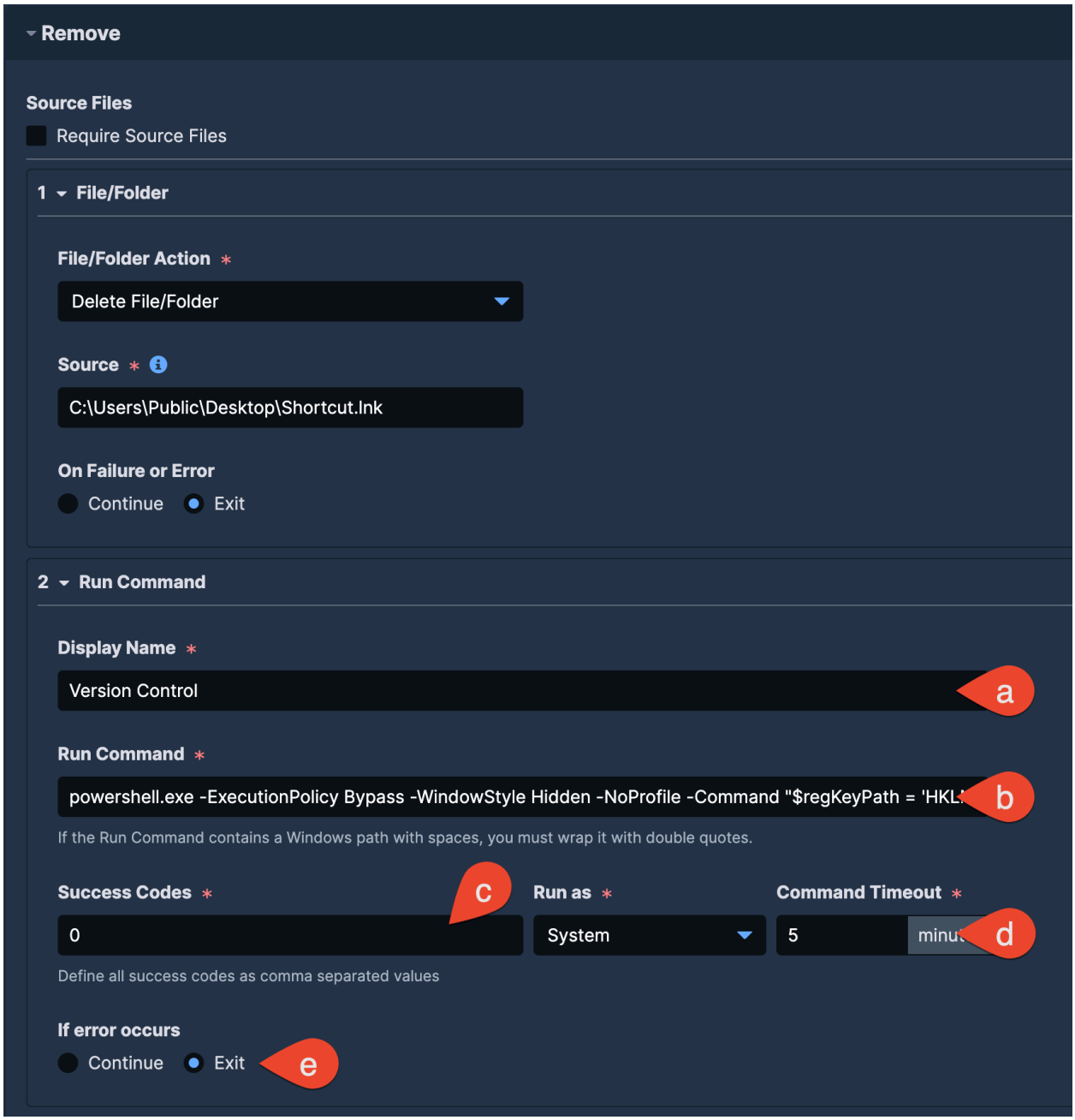
Install Operation



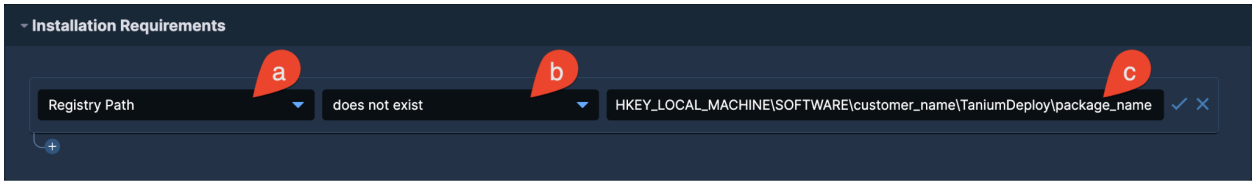
Update Operation



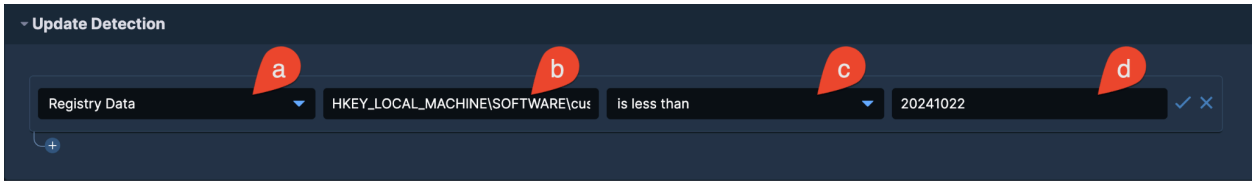
2. Add the updated “Remove Operation” PowerShell command from above as the last step for the Remove Operation (if applicable). Be sure to exit on failure for the previous steps as we don’t want to set the registry value if the package failed.
    a. **Display Name**: Version Control
    b. **Run Command**: Enter “Remove Operation” PowerShell Command
    c. **Success Codes**: 0
    d. **Command Timeout**: 5 minutes (Default)
    e. **If error occurs**: Exit



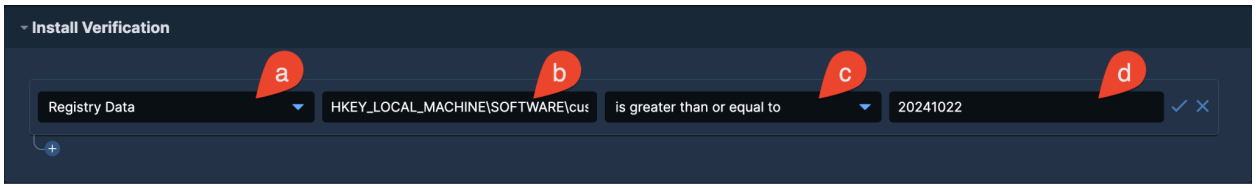
3. Add a clause to the Installation Requirements section using the registry key path from the PowerShell command.
    a. **Attribute**: Registry Path
    b. **Operator**: does not exist
    c. **Registry Value Path**: Registry path from the PowerShell command. (e.g. HKEY_LOCAL_MACHINE\SOFTWARE\customer_name\TaniumDeploy\package_name)



4. Add a clause to the Update Detection section using the registry key path from the PowerShell command and the current date.
    a. **Attribute**: Registry Data
    b. **Registry Value Path**: Registry path from the PowerShell command. (e.g. HKEY_LOCAL_MACHINE\SOFTWARE\customer_name\TaniumDeploy\package_name)
    c. **Operator**: is less than
    d. **Value**: The current date formatted as yyyyMMdd



5. Add a clause to the Install Verification section using the registry key path from the PowerShell command and the current date.
    a. **Attribute**: Registry Data
    b. **Registry Value Path**: Registry path from the PowerShell command. (e.g. HKEY_LOCAL_MACHINE\SOFTWARE\customer_name\TaniumDeploy\package_name)
    c. **Operator**: is greater than or equal to
    d. **Value**: The current date formatted as yyyyMMdd



Updating The Package

After updating the package with new content you can simply modify the date in the Update Detection section of the package to the current date. Now, clients with an “out of date” registry key will evaluate the package as updates needed.

If your package does not need an update operation, you could simplify the package by only setting the Install Verification and not including the key in the Installation Requirements section. Then when updating the content of the package, you would modify the date in the Install Verification section of the package to the current date. Now, clients with an “out of date” registry key will evaluate the package as install eligible. For this method, make sure your Install Operation is configured to update or overwrite any files or configurations, effectively using the Install Operation as both an install and update function.