# Commandeering Context Menu Entries

vx-underground collection // by smelly__vx and Ethereal

*This entry in the series derives from a proof-of-concept illustrated by Hexacorn, initially published July 29th, 2018 (Beyond good ol' Run Key, Part 82).*

# The Windows Context Menu:

The Context Menu is a menu in the Windows GUI (although present in any modern Operating System) that appears upon user interaction - usually in the event of the mouse right-click (VK_RBUTTON, 0x02). On the Windows Operating System the context menu will look something like the image below

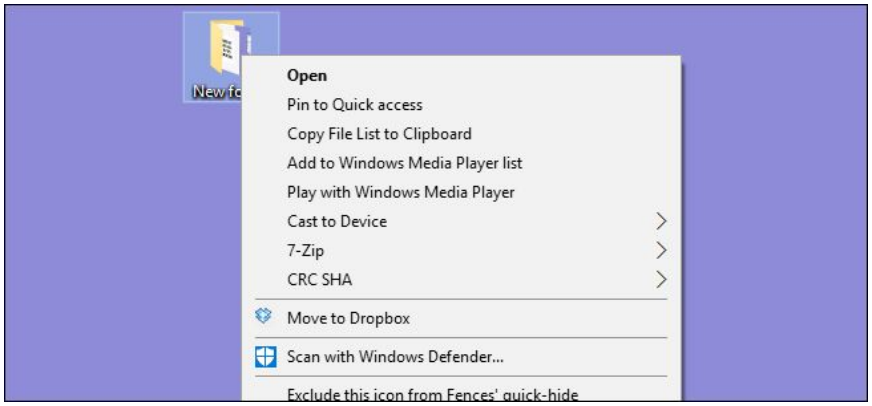



*Image courtesy of HowToGeek*

Entries for the context menu are present in **HKEY_CLASSES_ROOT (HKCR)**. Values in HKCR is a merged view from both **HKCU (HKEY_CURRENT_USER)** and **HKLM (HKEY_LOCAL_MACHINE)**. Because of this, a majority of HKCR is read-only. However, some keys allow a non-elevated user (medium integrity, user-mode) to both read and write. Additionally, in reference to MSDN documentation about HKCR: The **HKEY_CLASSES_ROOT** (HKCR) key contains file name extension associations and COM class registration information such as ProgIDs, CLSIDs, and IIDs. It is primarily intended for compatibility with the registry in 16-bit Windows.

Beside the brief description above - this registry hive also contains keys and subkeys responsible for the context menu which can be found in:

```
HKEY_CLASSES_ROOT\Directory\Background\Shell\*
```

Fortunately, the context menu and its associated functionality is heavily documented by Microsoft. This can give a better insight into the context menu, how it operates, and other vulnerabilities which may be present for malcode authors. However, due to the robustness of the context menu, its subcomponents, its general functionality, API invocations, and more - we will be skipping this segment. If you are interested in learning more about the context menu and its associated Shell functionality you can check out the related documentation here.

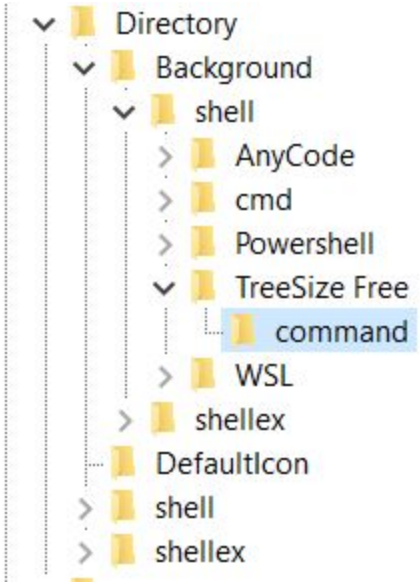
# The Objective:

Our approach to this persistence method will be simple. We will enumerate the target registry path listed previously with our predefined target in mind. Our predefined path:

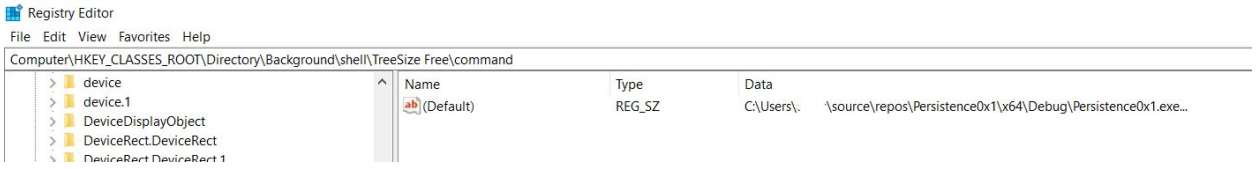
```
HKEY_CLASSES_ROOT\Directory\Background\shell\*
```

When we enumerate the predefined path we will be looking for the target registry path which will look as follows:

```
HKEY_CLASSES_ROOT\Directory\Background\shell\TreeSize Free\command
```




In this particular scenario we will be taking a relatively simple approach to hijacking the context menu - if in the event the target path is located we will query the registry to determine if it is our malicious payload. If it is not, we will replace it with ours.




The result of this code will be if and/or when the user opens the context menu (via right-click) and select TreeSize Free - it should execute our malicious application.