System security - lab 3

In this week we will expand on what we learned in lab 1 and 2,

Scanning an environment for CVE’s // updates that don’t have patches yet, you might be able to find a kernel exploit. They’re good for a last resort, but might break everything.

Rubeus token manipulation

Always install elevated; (2 registry keys, they both have to be set, what it does ((if theyre set)) one is in HKEY local machine and the other in HKEY local user(?), allows a normal ordinary users to install software only using an msi installer, this msi installer will elevate itself to administrator so that it can run the installation.

Msi abuse via installation repair; when you create an msi installer, you can set it to do custom actions; those custom actions might reveal something to us that could be useful to us as hacker. Everytime u install something with an MSIinstaller, a cache of the msi is kept on the computer

Driver weaknesses (kernel level exploit?); drivers are a good thing to attack in windows because they bridge between user-mode and kernel mode, making them pretty powerful. If you can install software, that can implement a driver, you can implement your own driver.

Dll hijacking; given the fact we’re not an admin on the client pc yet, it might be worth looking at dll hijacking once we tried all the other typical service exploits. For a DLL hijack to be possible, we’re looking at a program that’s trying to call a DLL which is missing from the system.

We’ve got this suspect service that is running, so we need to investigate to see; is it missing a dll, does it even run a program etc. Usually we’d use ProcMon to do this type of investigation (it can listen to a program when it starts to see if it makes any calls for which there are no answers). The problem is, we need to be an admin.

We can create a shared folder between the client VM and my own desktop, and drag the .exe that belongs to ‘DLL Hijacking Service’ into the shared folder. From there, I can go to my own desktop and create a service with: sc.exe create dllsvc binPath= “\Users\user\Desktop\shared\dllhijackservice.exe” which returns [SC] CreateService SUCCESS

Now, from my desktop, I can start Procmon as admin. This lets me create a few filters;

* Path ends with .dll
* Result contains not found
* Process name contains dll

Then, I start the service and check procmon. It shows me that the DLL is hijackme.dll, but its in a few locations; Users\user\desktop\shared\hijackme.dll, its in system32, in system, and in the windows folder.

DLL’s have a search order, as far as I remember. I did some searching and found it has the following order:

1. HKEY\_LOCAL\_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\KnownDLLs
2. Application directory
3. C:\Windows\System32
4. C:\Windows\System
5. C:\Windows
6. Current directory
7. PATH variables directory ((after it searches through the formal places, 1-5)) it searches through all folders which are in the path.

I could possibly do DLL search order Hijacking, because there’s no dll in the Registry Editor, or any of those other places listed. So; we could see if we have write access in any of those folders.

We can run: echo %PATH% in the cmd which shows us paths. This does show that c:\temp is in the paths variables. It also shows appdata/local/Microsoft/windowsapps.

We could compile our own DLL on the local machine and put it in the shared folder of the client machine, and then put it in the temp folder.

**((see Brightspace for the dll and how to compile it))**

I’m also able to run net user /domain, which show me all the users that exist in this domain. I can also run net group /domain, which shows me all the groups within this domain. Domain admins is obviously the most interesting.

After running net group “domain admins” /domain it returns the members of domain admins group, like Administrator (built in admin system), chantalle.karol, domad and sqlserver.

Bloodhound is a tool used to find the shortest path between me and the domain admin.

We can use psloggedon from the systinternalssuite to see who is currently logged on to the local machine. We can run psloggedon, showing us that \normaluser and \win10client administrator is logged on.

I can also run psloggedon [\\win10adm](file:///\\win10adm) to show me that the admin is logged on to their account. I’m going to re-do the mimikatz exploit to be able to open an admin cmd, and now psexec \\(ip) cmd

This doesn’t work > mike broke it, so we need to rename PsExec so that we can run it (because PsExec is broken/blocked, so if we change the name it will work.

I tried to use RENAME PsExec.exe ExecPs.exe, and then ran ExecPs.exe [\\192.168.56.30](file:///\\192.168.56.30) cmd which didn’t work.

I can however run PsExec -r ExecPs [\\ip](file:///\\ip) cmd which does work. This lets me run PsExec with a different name, so ha. Now I can type whoami and it shows win10adm/administrator

Whats next?