Penetration Testing Cheat Sheet

First Edition

Network Enumeration

Nmap

Switches:
-oA output all formats
-O Operating System enumeration
-p Port specification (-p- for all ports)
-sC Script Enumeration
-sV Version Enumeration (full tcp connect)
-sU UDP enumeration
–script= Script selection
-Pn Disable ping probes
-iL Include hosts from file
–script-args Provide arguments to NMAP scripts

Useful Scripts:
default - Default Scripts
vuln - Enumerate vulnerabilities
*ftp-** - All FTP Scripts
*http-** - All HTTP Scripts
*smb-** - All SMB Scripts
*nfs-** - All NFS Scripts
ldap-search - Performs ldap search
vulners Searches for CVEs on returned services

Windows commands

arp -a Show ARP table
ipconfig /all Shows IP configuration
ping ICMP Echo requests

Linux commands

arp -a Show ARP table
ping ICMP Echo requests
ip address Show IP configuration
ifconfig Show interface configuration

Packet Capture

- **wireshark**
- **tshark**
- **tcpdump**
- **netsh trace start capture=yes; netsh trace stop**

Service Enumeration

FTP

Checklist:

- Anonymous Access
- Vulnerabilities
  Service name be taken from banner grabs
- Default Credentials

Tools:

- Filezilla
- ftp (inbuilt linux)

DNS

Checklist:

- look for PTR records for the DNS server IP address
- look for PTR records for whole IP range
- Check for zone transfers
- Enumerate Windows AD DNS entries
- Enumerate MX records
- Enumerate SPF records (Email Security)
- Enumerate DKIM records (Email Security)
- Enumerate DMARC records

Tools:

- **dig**
- **host**
- **nslookup**
- **gobuster**

Group Policy

Checklist:

- Stored Credentials
- Automated Scripts (privesc)

Tools:

- **Active Directory Group Policy Manager**
- **SYSVOL**

HTTP/HTTPS

Checklist:

- Check common files (robots.txt .htaccess)
- Identify Web technologies:
- HTTP Server (IIS, Apache, Nginx)
- Preprocessors (Ruby on Rails, PHP, ASP)
- Web Applications (Wordpress, Drupal, Sharepoint)
- SSL Certificate Enumeration (Domain names, locations)
- Web Application Firewalls
- Check for VHOSTS (DNS enumeration can help)
- Check authentication technologies (JWT, Cookies, SAML, SSO, AD)
- Check for XSS
- Check for SQLi
- Check for LFI/RFI
- Check for shellshock

Tools:

- **Gobuster** DNS/Directory brute forcer
- **dirb** Directory brute forcer
- **Burp Suite** HTTP Proxy
- **Zed Attack Proxy** HTTP Proxy
- **wpscan** WordPress vulnerability scanner
- **drupalgeddon** Drupal vulnerability scanner
- **curl** HTTP request tool
- **wget** HTTP downloader
- **sqlmap** SQL Injection scanner
- **nmap –script=http-shellshock**

KERBEROS

Checklist:

- Enumerate Domain Users
- Pass the Ticket
- ASREPRoast
- Pass the Key
- Silver Ticket
- Golden Ticket

Tools:

- **Mimikatz**
- **Impacket**
- **PsExec**
- **kerbrute** Kerberos password brute forcing
- **GetNPUsers.py** Locate ASREPRoastable users

LDAP

Checklist:

- Users and Groups
- Privileged Groups
- Company layout
- Computers

Tools:

- **Active Directory Users and Computers**
- **python3 ldap3 module**
- **powershell**

MYSQL/MariaDB

Checklist:

- Check for CVEs
- Enumerate interesting databases

Tools:

- **mysql**

NETBIOS

Checklist:

- Discover Hosts on network segment

Tools:

- **nbtscan**

NFS

Checklist:

- Anonymous/Guest access
- Sensitive file disclosure

Tools:

- **showmount**
- **nmap –script=nfs***

NTP

Checklist:

- Synchronise time with host
- Enumerate connected clients
- Enumerate version information

Tools:

- **nmap –script=ntp-info**
- **ntpq**

- **ntpdc**

RDP

Checklist:

- Screenshots
- User Enumeration
- Check for RDP Vuln
- OS Version check

Tools:

- **mstsc.exe**
- **rdesktop**
- **remmina**
- **nmap**
- **crowbar**

RPC

Checklist:

- Anonymous login
- Enumerate Domain information
- Enumerate System information

Tools:

- **rpcclient**

SMB

Checklist:

- Anonymous/null session access
- Sensitive file disclosure
- Exposed shares
- SMB1 Vulnerability

Tools:

- **smbclient**
- **smbenum**
- **smbmap**
- **crackmapexec.py**
- **enum4linux**

SMTP

Checklist:

- Verify email address existence
- Send malicious emails

Tools:

- **nc**

SNMP

Checklist:

- Default Credentials
- Network configuration

Tools:

- **snmpwalk**
- **snmpenum**
- **snmpcheck**

SSH

Checklist:

- Reused SSH Keys
- Version Information
- Host information

Tools:

- **SSH**
- **nmap –script=ssh2-enum-algos**

VNC

Checklist:

- Enumerate running OS
- Screenshots
- Check CVEs

Tools:

- **vncviewer**
- **remmina**
- **nmap –script=vnc-info**

WinRM

Checklist:

- WMI Queries
- Pass the Hash

Tools:

- **winrs**
- **evil-winrm**
- **test-wsman**

Buffer Overflows

Workflow:

1. Fuzz the application by sending larger buffers of valid Characters
2. Once a buffer size has been confirmed, append full ASCII table to buffer
3. Confirm bad characters in payload, remove them from the buffer and repeat step.
4. Confirm the crash overwrites EIP and ESP with debugger running
5. Use msf_pattern_create to create new buffer of correct size
6. Confirm the crash with new patterned buffer and extract EIP
7. use msf_pattern_find to find offset of EIP buffer
8. use debugging tool to find a JMP ESP instruction
9. use location of JMP ESP (Little Endian) to route execution to ESP
10. place breakpoint on JMP ESP instruction and run exploit again
11. Single step through the code to ensure that it is loading to the correct address (you may need to adjust the ESP)
12. Create a malicious payload and insert into buffer (keeping buffer size the same) and ensuring that a sufficiently sized NOP sled has been included.
13. Test execution on local copy
14. Exploit

Windows Debugging Tools

Immunity Debugger:
Immunity debugger contains mona.py which can be used to enumerate a vulnerable binary, it has the following commands:

- *!mona modules*
  Lists all loaded modules
- *!mona noaslr*
  Lists all loaded modules without ASLR
- *!mona nosafesehasl*
  Lists all loaded modules without safe SEH or ASLR
- *!mona nosafeseh*
  Lists all loaded modules without safe SEH
- *!mona find -s "\xFF\xFF" -m "module.dll"*
  finds all addresses that match search term

Linux Debugging Tools

- EDB
  Use the OpCodeSearcher plugin to find ESP - EIP ensuring that the memory region has execute permissions.
- GDB
  You are on your own.

Payload Generation

msfvenom Payloads
List payloads with: msfvenom -l payloads

- shell/reverse_tcp
- shell/bind_tcp

msfvenom formats
List formats with msfvenom -l formats

- **asp(x)**
- **dll**
- **elf(-so)**
- **exe(-service)**
- **hta-psh** hypertext application with embedded powershell
- **vbs**
- **c**
- **ps1**
- **python**
- **raw**
- **rb**
- **sh**

Windows Privesc

Generic

- **Windows Exploit Suggester**
- **PSEXEC /s**
- Search for passwords in config files
- Search for passwords in Registry
- Search for sysprep.inf, sysprep.xml and Unattended.xml
- Group Policy Preference (Hardcoded encryption key)
- Install MSI as system user
- Check permissions for running services
- Check scheduled tasks
- Windows Version Specific Exploits
- Dump Password hashes

JuicyPotato PrivEsc

Requirements
Ran in the context of a Windows Service Account

Affected Versions

- Windows 7 Enterprise
- Windows 8.1 Enterprise
- Windows 10 Professional <1809
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2
- Windows Server 2019

RoguePotato PrivEsc

Requirements
Ran in the context of a Windows Service Account

Affected Versions

- Windows 10 Professional >=1809
- Windows Server 2019

Windows x86 ’afd.sys’ PrivEsc (MS11-046)

Requirements
Local Shell on host
Affected Versions

- Windows XP Pro (x86)
- Windows Vista (x86)
- Windows 7 (x86)

Windows XP

- **upnphost service**
  Writeable by Authenticated users

Windows Vista

Windows 7

Windows 8

Windows 8.1

Windows 10

Windows Server 2003

Windows Server 2008/R2

Windows Server 2012

Windows Server 2016

Windows Server 2019

Linux Privesc

checklist

- SUID Binaries
- Saved credentials
- Vulnerable scripts
- Service path vulnerabilities
- Sudo config
- Kernel Exploits
- SSH keys
- Crack shadow password hashes

Scripts

- **linenum.sh**
- **linuxprivchecker.py**
- **unix-privesc-check**
- **lynis**

Shell escape sequences

Python
import pty; pty.spawn("/bin/bash")
Vim
:set shell=/bin/sh
:shell
nmap - Versions 2.02 - 5.21 Inclusive
nmap –interactive
awk
awk 'BEGIN system("/bin/sh")'

Password Cracking

john

simple crack
john –wordlist=wordlist.txt hashfile
Zip file
zip2john file.zip > file.john
john –format=zip file.john
NTLMv2
john –format=netntlmv2 hash.txt
Kerberoast
kirbi2john hashes.kirbi > kirbi.john
john –wordlist=password.lst kirbi.john
Show john results
john –show target.txt

hashcat

Simple crack
hashcat -m <hashmode> <hashfile> <wordlist>
Common hashcat modes

- **0** - MD5
- **100** - SHA1
- **1000** - NTLM
- **1100** - Domain Cached Credentials
- **1700** - SHA512
- **13100** - Kerberos TGS-REP etype 23

Show hashcat results
hashcat –show <hashfile> -m <hash mode>

Linux shadow hashes

- **$1** - MD5
- **$2** - Blowfish
- **$2a** - eksblowfish
- **$5** - SHA-256
- **$6** - SHA-512

Generating a Linux Shadow hash
openssl passwd -1 -salt <salt> <password>

online hash crackers

- Crackstation

Kerberoasting

Notes

Kerberoasting is a post-exploitation attack which extracts service account credential hashes for offline cracking. Kerberoasting requires being within a domain context (Machine account, Domain User account).

1. Gain Domain Credentials
2. Enumerate SPNs on network
3. Request TGS for Service
4. Dump ticket from memory
5. Crack ticket offline to get plaintext credentials for service account
6. Use credentials to:

    Generate fake tickets

    Log in as service account

Windows Tools and Scripts

- Invoke-Kerberoast.ps1[a]
- setSPN.exe
- Mimikatz.exe

[a] https://github.com/EmpireProject/Empire/blob/master/data/module source/credentials/Invoke-Kerberoast.ps1

Linux Tools and Scripts

- Impacket/GetNPUsers.py
- hashcat
- john
- tgsrepcrack.py

Script to convert from Invoke-Kerberoast csv to Hashcat if not copy/pasted

```
cat kerberoast.csv | iconv -f UTF-16 -t ASCII | cut -d ',' -f 2 | tail -n +3 | tr -d '\n'| sed -e 's/\"\"/\n/g'| sed -e
↪  's/\"//g' > tickets.hashcat
```

Reference

IIS Versions

- **IIS 1.0** Windows NT 3.51
- **IIS 2.0** Windows NT 4.0
- **IIS 3.0** Windows NT 4.0 SP 2
- **IIS 4.0** Windows NT 4.0 Option Pack
- **IIS 5.0** Windows 2000
- **IIS 5.1** Windows XP Professional
- **IIS 6.0** Windows Server 2003 / Windows XP Pro x64
- **IIS 7.0** Windows Server 2008 / Windows Vista
- **IIS 7.5** Windows Server 2008 R2 / Windows 7
- **IIS 8.0** Windows Server 2012 / Windows 8
- **IIS 8.5** Windows Server 2012 R2 / Windows 8.1
- **IIS 10.0** Windows Server 2016/ Windows 10 and onwards

Creating Powershell encoded payloads

1. Convert to UTF-16LE
2. base64 encode

Common MAC Address vendors

VMWare 00:50:56
VMWare 00:0C:29
VMWare 00:05:69
VMWare 00:1C:14
Virtualbox 08:00:27
Virtualbox 52:54:00
Virtualbox 00:21:F6

Active Directory SRV records

- **_gc._tcp.<domain fqdn>** - Global Catalogue Server
- **_ldap._tcp.<domain fqdn>** - LDAP server
- **_kerberos._tcp.<domain fqdn>** - Kerberos Server (KDC)
- **_kpasswd._tcp.<endpoint fqdn>** - Kerberos Server ()

Wireshark Filters

Capture Filters:
Packet Filters:

tcpdump flags