Web mindmap

Start

- Look at the Important Section
    - For both cheatsheets
- Remember to always look at **Niche Attack Vector** and **Random** Section
- Look at file:///C:/Users/Micha/Downloads/Linux%20Privilege%20Escalation-1.pdf
    - This is the Tib3rius PDF for Linux Priv Escc
    - Also in OSCP folder

- Look at Ippsec rocks
- Look at HackTricks

Port Scanning

- **Nmap**
- do normal nmap scan and then after your done, while ur already enumerating, run a full port scan and also a UDP scan, so that they run the in the background and can be hugely impactful
    - Add this to Linux checklist too

Notes

Web Related:

- **Default Credentials**
- Look at the whole Web sections
- **Firefox and .mozilla (and firefox_decrypt)**

Wordlists:

- **Wordlists**
- **cewl**
- **cupp**

- **How to find files in Linux**
- grep
- SCP

Helpful:

- **Bypassing file extension filters**

Fix Linux Path:

- export PATH=$PATH:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
- **PATH environment**

Misc:

- How to find IP of a machine given its FQDN or hostname using nslookup

---

Reverse/Web Shells:

- Use penelope
- Look at all the reverse and webshell section!
- If you are uploading a file that you can access, try webshell first since it's easier to see if it works and then from there it's easy to turn into reverse shell
- Try Busybox!
- Try base64 encoding it!
- Try different port (try ports open on the target machine)
- Try putting it in different locations on web like /dev/shm
- Ping first to see if you can reach Kali. If not, set up listener on pivot and forward connection back to the Kali
- How to upload and execute reverse shell **upload.php and run.php** (Staged Reverse Shell via Web Execution)
- How to **bypass extension** restrictions for file upload (including rev shell) on Apache by using .htaccess
- Encoding Reverse Shells in **Base64**
- Uploading reverse shell as an image using **GIF signature**

- **rbash (restricted bash)**

Service Enumeration

General

- Searchsploit
- Hydra
- Use new credentials to recheck logins
- Password List (look at Pen Testing Notes)
-

FTP - 21

- Look at pen testing notes
- Try putting stuff to see if you have write permissions.
    - Could lead to file upload if connected via web server
    - Or maybe if you have LFI, then you can access files dropped in there if FTP connected to web server
- Maybe try just bruteforcing most common creds (seen in AuthBy PG Practice)
- **FTP**
- **Mount FTP**
- **Filezilla**
- **/var/ftp write permissions leading to reverse shell (putting in reverse shell to ftp)**

SMTP - 25

- Just a tip: If you see port 25 or SMTP on the target, always look through /var/mail/ and /var/spool/mail.
- **SMTP**
- **Phishing**
- **Mail**
- nmap -p 25 --script smtp-* $IP
- nc -vv $IP 25, try commands EXPN or VRFY to verify users
- telnet $IP 25

DNS - 53

- DNSEnum
- DNS Zone transfer - dig AXFR <domain> @<master dns server>
- nslookup - then type server $IP, try 127.0.0.1, 127.0.0.2, and given IP

HTTP/HTTPS - 80, 443

- **Apache**
- **PHP**
- Directory Busting
    - Try different wordlists
    - Look at pen testing notes
- cewl
- Web
    - Look at pen testing notes
- Wordpress
    - Look at pen testing notes
- LFI and RFI
    - **Look at pentesing notes for full list**
    - **Log poisoning**
    - Stealing SSH private keys
    - **knockd (port knocking)**
    - Try **PHP wrappers** to either view full PHP pages or get command execution
    - Look for **config** files specific to the service on the website
        - Seen in SPX, DVR4 and Fish and Mantis PG practice
    - LFI2RCE via phpinfo()
        - Hacktricks
- XSS (Cross-Site Scripting)
- XXE (XML External Entity)
- SSTI (Server Side Template Injection)
- SSRF
- File upload:
    - **Overwriting authorized_keys file using file upload**
- View Source Code
- Check cookies and JWTs
    - Look at powall's checklist
- **Nikito (pen testing notes)**
    - nikto -h $IP
    - nikto id <u:p> -h $IP
- **Web/IIS**
    - hydra -L users.txt -P pass.txt http-get://192.168.229.122/
    - davtest -url http://$IP -auth user:pass
    - cadaver http://192.168.142.122/
- **SQL injections**
    - Look at pentesting notes

- **SQL authentication bypass**
- **Command injection (pen testing notes)**
    - **How to send pings to Kali from target machine to check if you have command execution**
- Shellshock
    - **Shellshock/Bashbug (CGI-Bin)**
- Check for LFI
    - Steal SSH keys and make sure to put trailing line at the end of private SSH keys

POP - 110
- telenet $IP 110, then USER <user> and PASS <pass> on separate lines
- nmap --script "pop3-capabilities or pop3-ntlm-info" -sV -p 110 $IP
- **POP3**

rpc - 135
- **RPC**
    - Change domain user password if IT group or has rights like GenericWrite

SMB - 139, 445
- **SMB**
- **NTLM_Theft and NTLM Hash leak (Write Access to SMB)**
- Look at Powall's Checklist

imap - 143
- nc -nv $IP 143
- **IMAP**

snmp - 161, 162
- Look at pentesting notes
    - **SNMP**
- Look at autorecon output for onesixtyone and snmpwalk
- snmp-check -p <port number> $IP

ldap - 389, 636
- **LDAP**

Oracle TNS Listener
- nmap --script "oracle-tns-version" -p 1521 -T4 -sV $IP

- use hydra if password protected - hydra -P rockyou.txt -t 32 -s 1521 $IP oracle-listener
- bruteforce SID (database name) - nmap --script +oracle-sid-brute -p 1521 $IP
- use odat to bruteforce credentials for SID

NFS - 2049
- **NFS** subsection of RPC

mysql - 3306
- Look at pentesting notes

Redis - 6379
- **Redis (VERY VULNERABLE)**
- connect - nc -nv $IP 6379
- check permissions - INFO

Exchange
- PrivExchange - https://dirkjanm.io/abusing-exchange-one-api-call-away-from-domain-admin/

General

1. Use penelope
2. Always try using the username as the password. I see that a lot in OSCP practices
3. Look in /opt/ directory since those are 3rd party non-standard files
4. If there is a website, and you get shell through a method other than rev shell via website, then it's often worth it to try uploading a rev shell to the web root directory and then trigger it using the website (ex. LFI or by accessing it if it's like /var/www/html), and then you can pivot to the user who is running apache
    a. You can also run ps -aux to see who is running apache server, which is going to be the user you are pivoting to
    b. We've seen this in Readys PG Practice and Relia (challenge lab) .246 (when we added file to /dev/shm)
5. If you have a domain like bullybox.local for a website, and you have to add it to /etc/hosts, and then you have a login page requiring email, then bullybox.local is likely the email format: michael@bullybox.local
    a. Seen in Bullybox PG Practice

6. As seen in the Marketing PG Practice, when we had a website like LimeSurvey, even after we got a shell, we googled "LimeSurvey config" file and found the file "/application/config/config.php" which had credentials for priv esc. **Always google config file locations for services!**
7. If you have LFI, look for config files specific to the service! We've seen that A LOT in Linux PG practice, like DVR4 and Fish and Mantis
8. If you find any host names or subdomains, add them to /etc/hosts and try accessing them
9. If you unzip a file but it skips a lot, then try 7z instead since it might be a .7z file instead of .zip even if it says .zip (seen in .144 in OSCP-A)

Foothold tips

Just bruteforce all services, especially things like FTP, SSH, and websites

- hydra -L users.txt -P /usr/share/wordlists/rockyou.txt **ftp**://192.168.236.157
- hydra -l users.txt -P **/usr/share/seclists/Passwords/500-worst-passwords.txt** -e nsr **ssh**://192.168.171.142
- hydra -l michael -P /usr/share/wordlists/rockyou.txt -e nsr **ssh**://192.168.171.142

Privilege Escalation

- If you can't run basic commands without specifying full path, then fix PATH variable:
  - export PATH=$PATH:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
    - **Note: this only works on current session so you have to do this again if you close current shell**

- [ ] id
- [ ] id -a
- [ ] Groups
  - Getting Root from users in Docker, lxd, disk, or adm Group
- [ ] Try logging in as root
  - [ ] su root
  - [ ] Try no password or try "root" as password
- [ ] First it's always good to check for .git directories

- [ ] find / -type d -name ".git" 2>/dev/null
- [ ] env
  - Look at the environment, like we always do in powershell
  - Useful in Fired PG Practice
- [ ] Check **sudo -l**
  - [ ] LD_PRELOAD (environment variable)
  - [ ] LD_PRELOAD (environment variable)
  - [ ] Get root shell if a file from sudo -l doesn't exist or you can overwrite it
- [ ] **LinPEAS**
  - Cron
- [ ] Check **SUID**
  - find / -perm /4000 2>/dev/null
  - **cat SUID.txt | grep -vxF -f SUIDstandard.txt | grep -v '^/snap/'**
    - Save list of SUID to a file called SUID.txt and run this to **filter out all the standard binaries**
  - cat **SUID.txt** | grep -F -f SUIDgtfoBins.txt
    - Save list of SUID to a file called SUID.txt and run this to check if the any are in **GTFOBins**
  - [ ] Check **SGID**
    - find / -perm /2000 2>/dev/null
  - [ ] Shared Object Injection
    - Run strace on SUID files to see if they are missing a .so file that you can replace
  - [ ] PATH Environment Variable
    - Run strace on SUID files to see if they call a binary without full path
  - [ ] Abusing Shell Features (#1)
    - bash --version
    - Bash < 4.2-048 is vulnerable (ex. 4.1.5)
  - [ ] Abusing Shell Features (#2)
    - bash --version
    - Bash versions below 4.4 may be vulnerable
- [ ] Check **capabilities**
  - /usr/sbin/getcap -r / 2>/dev/null
- [ ] **Scheduled Tasks (Cron) Enumeration**
  - [ ] cd /etc/cron.d
    - [ ] ls
    - [ ] Look at each file. I think "e2scrub_all" is standard file
    - [ ] cron.d is where custom cron jobs live
  - [ ] ls -lah /etc/cron*

- Purpose: Check permissions and ownership of cron-related files and directories.

- [ ] cat /etc/crontab
  - Purpose: See what commands are being run, and if any point to scripts or binaries you might modify.
  - Look at "LD_LIBRARY_PATH" and see if you can write to any directories. And if there are any missing .so files or the .so file is later in the PATH, then add ur own to the early directories in PATH
    - Seen in Sybaris PG Practice
- [ ] ls -al /etc/ | grep cron
  - Purpose: Quickly identify cron-related directories and see permissions.
- [ ] Payload
  - [ ] echo 'echo " useradm ALL=(ALL) NOPASSWD:ALL" >> /etc/sudoers' > cleanup.sh
    - Replace "useradm" with the username of your current user
    - Result: Once the cron job executes cleanup.sh as root, your user gains passwordless root privileges.
- [ ] /var/spool/cron/ or /var/spool/cron/crontabs/

- [ ] ./pspy64
- [ ] ps aux

- [ ] sudo -v

- [ ] Look in /opt since that is a folder for 3rd part non-standard software

- [ ] Run Linux Smart Enumeration
- [ ] **Kernel Enumeration**
  - Dirty Pipe Vulnerable: 5.8 → 5.16.10 (inclusive), 5.15 → 5.15.24, 5.10 → 5.10.101
    - 5.9.0 exploited in OSCP-B .149
- [ ] Check Dirtcow
- [ ] Check pwnkit

- [ ] Manually enumerate user directories and look at all hidden files
  - ls -la
- [ ] Find git, backup, zip files
  - [ ] find / \(-name ".git" -o -name "*backup*" -o -name "*.zip" \) 2>/dev/null

    - Case sensitive
  - [ ] find / \(-iname ".git" -o -iname "*backup*" -o -iname "*.zip" \) 2>/dev/null
    - Case insensitive (-iname instead of -name)
- [ ] Look for /opt/ files, since those are 3rd party softwares, and not standard.
  - Like in the PC PG Practice, there was a process running /opt/rpc.py which was vulnerable
- [ ] Recursively looking through a bunch of files for credentials:
  - grep -rinE '(password|username|user|pass|key|token|secret|admin|login|credentials)'
- [ ] Manually look for backup and config

- [ ] **Network Enumeration**
  - [ ] ss -tulnp

- [ ] history
  - Check shell history

- [ ] Check if /etc/passwd or /etc/shadow are writable
  - [ ] ls -la /etc/passwd
  - [ ] ls -la /etc/shadow
- [ ] Find writeable files and folders
  - **Writeable directories**
    - find / -type d -writable -not -path "/proc/*" -not -path "/home/**<USER>**/*" 2>/dev/null
      - Replace **<USER>** with your own username
      - Filters our proc and /home/<USER>
    - find / -writable -type d 2>/dev/null
      - The output is usually not as long so no need for filter
  - **Writeable files**
    - find / -type f -writable -not -path "/proc/*" -not -path "/sys/*" -not -path "/dev/*" -not -path "/run/*" -not -path "/home/**<USER>**/*" 2>/dev/null
      - Replace **<USER>** with your own username
      - Filters out the useless stuff that makes the output thousands of lines long
        - proc, sys, dev, run

      find / -type f -writable 2>/dev/null
      - ALL writeable files (**usually thousands of lines long**)
  - [ ] **Insecure File Enumeration**
- [ ] Check for writeable files which will get executed every time a new shell launched

- [ ] ls -l /etc/profile /etc/profile.d/
    - If you find something that is writable by you, then you can inject a Bash reverse shell or something into it

- [ ] Look at /var/mail/ and /var/spool/mail for mail
    - cd /var/mail
    - cd /var/spool/mail

- [ ] sudo tokens
    - [ ] ps -e - check for sudo token with same uid
        - [ ] ps -e -o pid,uid,cmd | grep $(id -u) | grep sudo
    - [ ] cat /proc/sys/kernel/yama/ptrace_scope
        - [ ] Make sure it's 0
    - [ ] sudo id
        - [ ] Trigger sudo token
    - [ ] https://github.com/nongiach/sudo_inject

- [ ] **Installed Programs Enumeration**
    - installed applications - dpkg -l
- [ ] **Unmounted Drives Enumeration**
    - cat /etc/fstab
        - If you see this: /dev/sdb1 **/mnt/backup** ext4 rw,user,suid,exec 0 0
            - This means any user can can read, write, assign SUID, and execute files
        - Then make a copy of bash and assign SUID:
            - mount /mnt/backup
            - cp /bin/bash /mnt/backup/bashroot
            - chmod u+s /mnt/backup/bashroot
            - /mnt/backup/bashroot -p
    - unmounted disks - mountvol
- [ ] **User Trail Enumeration**
    - env
    - cat .bashrc
    - su - root
- [ ] **Service footprint enumeration**
    - watch -n 1 "ps -aux | grep pass"

- [ ] **systemctl & systemd enumeration**
    - Check systemctl --version for vulnerable versions

- [ ] /usr/share/linux-exploit-suggester/linux-exploit-suggester.sh -k
- [ ] file system - cat /etc/fstab
- [ ] users with login - grep -vE "nologin|false" /etc/passwd
    - Checks in /etc/passwd for entries with shell login
    - If you see user **sync** with "/bin:/bin/sync", then it's useless since /bin/sync is a useless shell and it doesn't do anything

- [ ] If you see user vagrant, try password vagrant
- [ ] **Finding users in Linux**

- [ ] If you are inside a shell, look at the web directory and look for config and source code for credentials

- [ ] Bruteforce SSH
    - [ ] They did in Medtech .122 using /usr/share/seclists/Passwords/500-worst-passwords.txt

- [ ] **.bash_history**
- [ ] **Steganography**
- [ ] **Impersonation (editing UID/GID) to read file/directory for NFS**

- [ ] **Firewall enumeration**
- [ ] **Service Enumeration**

- [ ] Check versions like xampp version

- [ ] If you see a binary that you don't understand and it runs with SUID or something special, then try running it with -h, --help, and -v or --version. In the Mzeeav PG Practice, it turned out to be the "find" binary, so we can use the find SUID GTFO bins exploit

- [ ] Try logging in as root using previous passwords
    - Seen in Apex PG Practice

- [ ] If you see writable files like /usr/local/bin that are high up in cron job PATH, then see if there are any commands run by root in a process (via pspy64) and see if you can replace it with a malicious binary higher up the PATH

- [ ] **PATH environment variable hijacking**
    - [ ] If a script run by root runs a command like "whoami" but doesn't specify the full path (/usr/bin.whoami), then we can add our own whoami executable to PATH and have it open rev shell

Automated Privilege Escalation

- [ ] Linux Smart Enumeration
- [ ] Automated Privilege Escalation
- [ ] LinPEAS
- [ ] linux-exploit-suggester

Helpful Checklist

- [ ] Manual Privilege Escalation

Post-exploitation

- [ ] Look for SSH keys to pivot as another user or to another machine
    - [ ] And look in authorized_keys to get hints about users and IP

- [ ] check /etc/ssh/sshd_config
- [ ] service command line: /proc/self/cmdline

- [ ] Look for history or other files in home directories

ALWAYS LOOK AT THE **NICHE ATTACK VECTOR** AND **RANDOM SECTION**. AS WELL AS THE **IMPORTANT SECTION**