Universidad de los Andes

Proyecto1-Password Crackers
Nicolás Camilo Moreno Arias

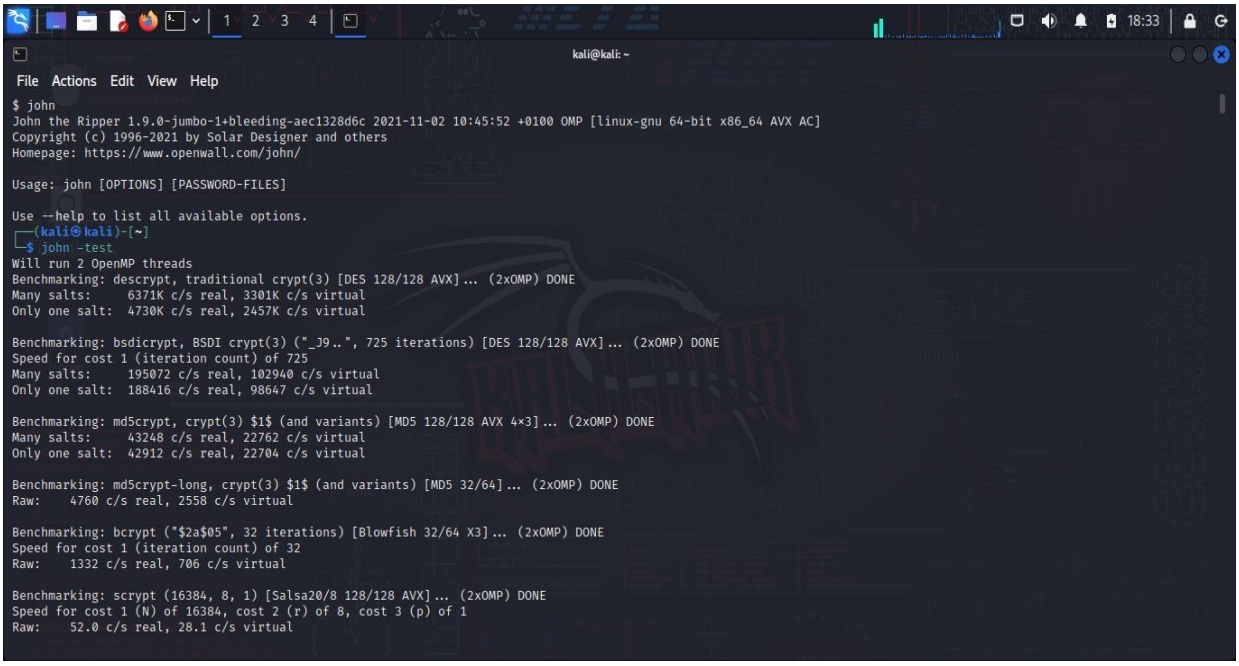
1. Solución de Actividades.

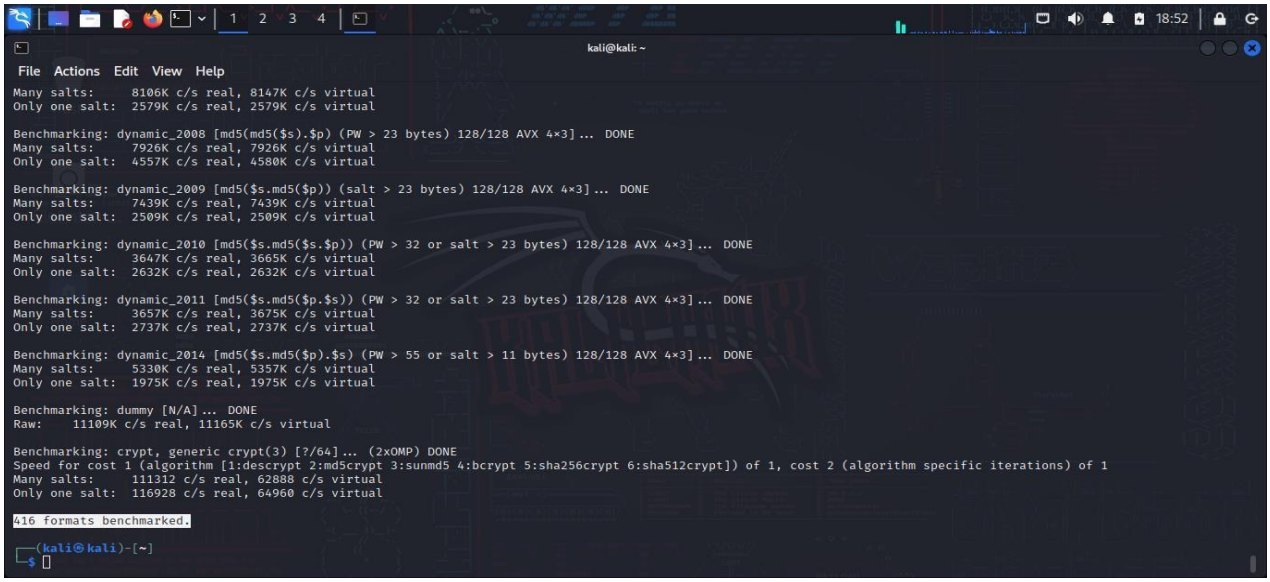
---

2 Evaluación de Hardware.

Al ejecutar el comando **'john –test'** en la terminal de Linux, podemos observar cómo se realizan varias evaluaciones de distintos componentes. Después de cada evaluación, se muestra una palabra en mayúsculas al final que dice **'DONE'**, que en español significa **'hecho'**. Es posible que este procedimiento tarde un poco en finalizar.

Una vez completado, podremos verificar la cantidad de evaluaciones realizadas en los diferentes componentes de la computadora para asegurarnos de que todo esté en orden.





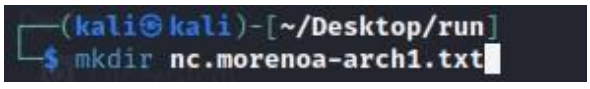
2 Ejecución 1.

---

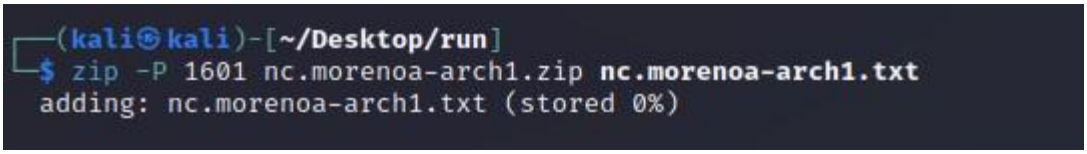
1. Crear archivo1.zip con contraseña.

En primer lugar, creamos un archivo llamado **'nc.morenoa-arch1.txt'**. Para crear este archivo, utilizamos el comando '**mkdir**', que tiene como objetivo crear un nuevo directorio.

La extensión **'.txt'** al final del nombre indica que se trata de un archivo de texto vacío, el cual será utilizado posteriormente para crear el archivo **'.zip'**.



Utilizamos el comando **'zip -P'** para crear el archivo zip con contraseña. En esta ocasión, hemos elegido una contraseña de 4 dígitos: **'1601'** para llevar a cabo esta actividad.

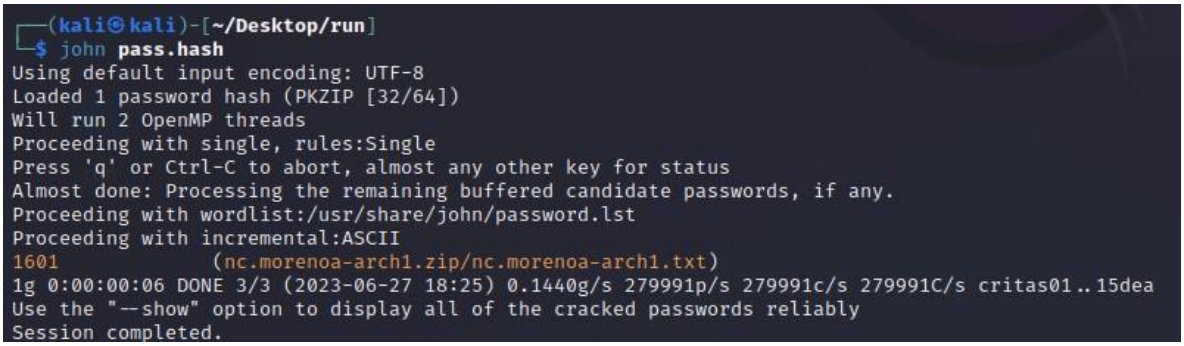


Una vez creado el archivo, debemos ejecutar el comando **'zip2john nc.morenoaarch1.zip > pass.hash'**. Este comando tiene como objetivo extraer la información del archivo zip y redirigirla hacia el archivo **'pass.hash'**

Ahora vamos a revisar el contenido del archivo **'pass.hash'** para asegurarnos de que todo esté correcto. Podemos visualizar el contenido utilizando el comando **'cat',** el cual nos permite ver el contenido del archivo.

Ahora vamos a ejecutar el siguiente comando: **'John pass.hash'**. Este comando puede tardar hasta 6 segundos en mostrar la contraseña, la cual se resaltará en color naranja.

Este comando indica que se está utilizando **John the Ripp**er para intentar descifrar las contraseñas almacenadas en el archivo **'pass.hash'**, empleando diferentes técnicas de ataque de fuerza bruta y diccionario. El éxito del descifrado depende de la configuración y opciones que se especifiquen con el comando.



2. Crear archivo2.zip con contraseña.

Realizamos los mismos comandos de la actividad anterior, como la creación del archivo.txt utilizando el comando **'mkdir',** pero con la única diferencia de reemplazar el número **"1"** por **"2".** Una vez ejecutado el comando y creado el archivo, procederemos a realizar las siguientes acciones.

Vamos a crear el archivo .zip utilizando el comando **'zip -P'** con una contraseña de 8 dígitos. Para ello, seguiremos los siguientes pasos:

1. Ejecutaremos el comando **'zip -P contraseña'** seguido del nombre del archivo.zip que deseamos crear.
2. Ingresaremos la contraseña de 8 dígitos que hayamos elegido.
3. El comando generará el archivo .zip protegido con la contraseña especificada.

Recuerda que es importante elegir una contraseña segura para garantizar la protección de tus archivos

```
┌──(kali㉿kali)-[~/Desktop/run]
└─$ zip -P 16012004 nc.morenoa-arch2.zip nc.morenoa-arch2.txt
  adding: nc.morenoa-arch2.txt (stored 0%)
```

Ahora que hemos creado el archivo .zip, procederemos a crear el archivo **'pass.hash'** utilizando el comando 'zip2john'.

1. Ejecutaremos el comando **'zip2john archivo.zip > pass.hash'**.
2. El comando **'zip2john'** extraerá la información del archivo .zip y la redirigirá hacia el archivo **'pass.hash'**.

Es importante tener en cuenta que **'pass.hash'** contendrá la información necesaria para realizar futuros procesos relacionados con el archivo .zip y las contraseñas asociadas.

```
┌──(kali㉿kali)-[~/Desktop/run]
└─$ zip2john nc.morenoa-arch2.zip > pass.hash
ver 1.0 efh 5455 efh 7875 nc.morenoa-arch2.zip/nc.morenoa-arch2.txt PKZIP Encr: 2b chk, TS_chk, cmplen=17, decmplen=5, crc=D1DBAD78 ts=8046 cs=8046 type=0
```

Ahora procederemos a verificar la información contenida en el archivo **'pass.hash'** utilizando el comando **'cat'.**

1. Ejecutaremos el comando **'cat pass.hash'.**
2. El comando **'cat'** mostrará en la pantalla el contenido del archivo **'pass.hash'**, permitiéndonos revisar la información contenida en él.

Este paso es importante para asegurarnos de que la información en **'pass.hash'** sea correcta y esté lista para ser utilizada en futuros procesos relacionados con el archivo .zip y las contraseñas asociadas.

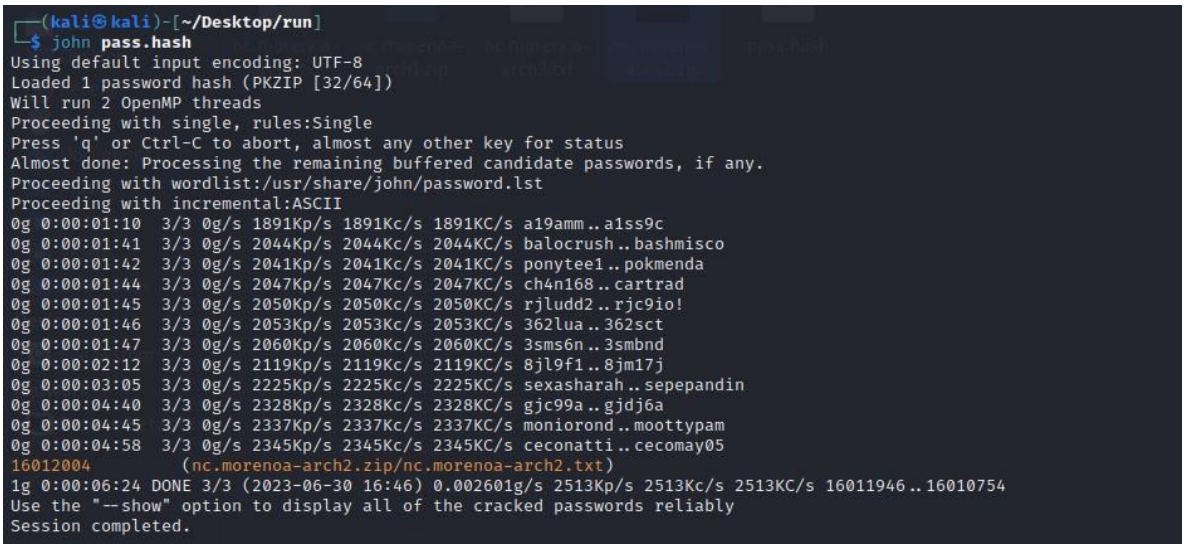
```
└─$ cat pass.hash
nc.morenoa-arch2.zip/nc.morenoa-arch2.txt:$pkzip$1*2*2*0*11*5*d1dbad78*0*4e*0*11*8046*b5d132b2174e70f594d114fd8d561cafe0*$/pkzip$:nc.morenoa-arch2.txt:nc.morenoa-arch2.zip::nc.morenoa-arch2.zip
```

Ahora procederemos a ejecutar el programa **'John'** para descubrir la contraseña y así finalizar la actividad. Además, mediremos el tiempo que tarda en encontrar la contraseña.

1. Ejecutaremos el comando **'John pass.hash'.**
2. El programa **'John'** utilizará diferentes técnicas de ataque, como fuerza bruta y diccionario, para intentar descifrar las contraseñas almacenadas en el archivo **'pass.hash'**.

3.Durante este proceso, el programa mostrará el tiempo transcurrido hasta encontrar la contraseña.

Es importante tener en cuenta que el tiempo que demora en encontrar la contraseña puede variar dependiendo de la complejidad de la contraseña y de las técnicas de ataque utilizadas.



Ahora que hemos finalizado el proceso de búsqueda de la contraseña, podemos observar que el sistema utilizó un archivo llamado **'password.lst'**. Este archivo contiene una lista de contraseñas populares o frecuentemente utilizadas por las personas. La idea de este archivo es probar primero contraseñas débiles o comunes durante el proceso de cracking, ya que muchas personas tienden a utilizar contraseñas fáciles de adivinar.

A continuación, veremos una serie de registros que nos indican el progreso del proceso de búsqueda de contraseñas. Al final de cada registro, en texto de color naranja, podremos ver la contraseña y el nombre del archivo asociado.

En nuestro caso, el tiempo total para encontrar la contraseña fue **de 6 minutos y 24 segundos.**

3. Ejecución 2.

---

1. Crear archivo3.zip con contraseña.

Vamos a seguir los procedimientos realizados en actividades anteriores, los cuales incluyen la creación de un archivo .txt seguido por la creación de un archivo .zip.

1. Comenzaremos creando un archivo de texto utilizando el comando **'mkdir nombre_archivo.txt'.**
2. Reemplazaremos **'nombre_archivo'** con el nombre que deseemos darle al archivo. Este comando creará un archivo de texto vacío con la extensión **.txt**.

A continuación, pasaremos a la creación del archivo **.zip**:

1. Ejecutaremos el comando **'zip -r archivo.zip nombre_archivo.txt'.**
2. Nuevamente, reemplazaremos **'nombre_archivo'** con el nombre que hayamos elegido para el archivo de texto. Esto creará un archivo .zip que contendrá el archivo de texto.

Recuerda asegurarte de utilizar nombres y rutas de archivo adecuados según tus necesidades.

```
┌──(kali㉿kali)-[~/Desktop/run]
└─$ zip -P astronomia nc.morenoa-arch3.zip nc.morenoa-arch3.txt
  adding: nc.morenoa-arch3.txt (stored 0%)
```

Ahora vamos a realizar el siguiente paso, que consiste en crear un archivo **'s.hash'** utilizando el archivo .zip que hemos creado anteriormente. Utilizaremos el comando **'zip2john'** para llevar a cabo esta tarea.

1. Ejecutaremos el comando **'zip2john archivo.zip > s.hash'.**
2. El comando 'zip2john' extraerá la información del archivo .zip y la redirigirá hacia el archivo **'s.hash'.**

El archivo **'s.hash'** contendrá la información necesaria para realizar futuros procesos relacionados con el archivo .zip y las contraseñas asociadas.

```
┌──(kali㉿kali)-[~/Desktop/run]
└─$ zip2john nc.morenoa-arch3.zip > s.hash
ver 1.0 efh 5455 efh 7875 nc.morenoa-arch3.zip/nc.morenoa-arch3.txt PKZIP Encr: 2b chk, TS_chk, cmplen=17, decmplen=5, crc=D1DBAD78 ts=8EDA cs=8eda type=0
```

Una vez creado el archivo, procederemos a crear un archivo vacío con la extensión .txt. Le daremos el nombre **'claves'.**

1. Utilizaremos el comando **'nano nc.morenoa- claves.txt'** para crear el archivo vacío.
2. El comando '**nano'** se utiliza para crear un archivo sin contenido.

Una vez completados estos pasos, el archivo **'claves.txt'** estará listo para ser utilizado.

```
┌──(kali㉿kali)-[~/Desktop/run]
└─$ nano nc.morenoa-claves.txt
```

Ahora que hemos creado el archivo **'claves.txt'** y hemos agregado diferentes palabras y números para crear nuestro propio diccionario, vamos a proceder al siguiente paso.

1. Ejecutaremos el comando **'john s.hash -wordlist=claves.txt'.**
2. El comando **'john'** utilizará el archivo **'s.hash'** y el diccionario que hemos creado (**'claves.txt'**) para buscar la contraseña asociada al archivo.

Durante el proceso, el programa **'john**' intentará descifrar la contraseña utilizando las palabras y números de nuestro diccionario personalizado.

```
┌──(kali㉿kali)-[~/Desktop/run]
└─$ john s.hash -wordlist=nc.morenoa-claves.txt
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
astronomia       (nc.morenoa-arch3.zip/nc.morenoa-arch3.txt)
1g 0:00:00:00 DONE (2023-07-03 17:17) 25.00g/s 2325p/s 2325c/s 2325C/s ana ..astronomia
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

Podremos observar cómo el programa encuentra la contraseña de manera instantánea, ya que se mostrará resaltada en color naranja junto con el nombre del archivo.

Durante el proceso de búsqueda, el programa **'john'** utilizará el diccionario personalizado que hemos creado (**'claves.txt'**) para probar diferentes combinaciones y encontrar la contraseña asociada al archivo. Una vez encontrada, se mostrará en color naranja, junto con el nombre del archivo correspondiente.

Este método nos permite identificar rápidamente la contraseña descubierta y el archivo al que pertenece.

4. Ejecución 3.

---

1. Los archivos etc/passwd y etc/shadow.

Vamos a crear una cuenta con una contraseña débil. Para ello, ejecutaremos el comando **'sudo adduser hola2'.**

1. Ejecuta el comando **'sudo adduser hola2'** en la terminal.
2. El comando **'adduser'** se utiliza para crear una nueva cuenta de usuario en el sistema operativo.
3. **'hola2'** es el nombre de usuario que se asignará a la cuenta.

Recuerda que utilizar contraseñas débiles puede comprometer la seguridad de la cuenta y los datos asociados. Se recomienda utilizar contraseñas fuertes que contengan una combinación de letras mayúsculas y minúsculas, números y caracteres especiales.

```
┌──(kali㉿kali)-[~/Desktop/run]
└─$ sudo adduser hola2
[sudo] password for kali:
Adding user `hola2' ...
Adding new group `hola2' (1003) ...
Adding new user `hola2' (1003) with group `hola2 (1003)' ...
Creating home directory `/home/hola2' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for hola2
Enter the new value, or press ENTER for the default
        Full Name []: hola
        Room Number []: 234
        Work Phone []: 123
        Home Phone []: 2345
        Other []: 4
Is the information correct? [Y/n] y
Adding new user `hola2' to supplemental / extra groups `users' ...
Adding user `hola2' to group `users' ...
```

Una vez ejecutado el comando **'sudo adduser hola2'**, se te solicitará crear una contraseña y repetirla para confirmarla. A continuación, se te pedirá proporcionar información adicional para completar el proceso de creación de la cuenta. Esto incluirá el nombre completo, número de habitación, número de trabajo, número de teléfono y cualquier otra información relevante.

Ingresa la contraseña deseada y repítela cuando se te solicite. Luego, proporciona los datos adicionales solicitados, como el nombre completo, número de habitación, número de trabajo, número de teléfono y cualquier otra información requerida.

Asegúrate de completar toda la información de manera precisa y correcta para crear la cuenta satisfactoriamente.

2. Herramienta John the ripper.

Vamos a ejecutar el comando **'unshadow /etc/passwd /etc/shadow > output.db**'. Este comando combina los archivos **'/etc/passwd'** y **'/etc/shadow'** en un solo archivo llamado **'output.db'**.

Sin embargo, puede haber restricciones de permisos que te impidan acceder al archivo **'/etc/shadow'**. En ese caso, necesitarás utilizar el comando 'sudo' para ejecutar el comando como administrador.

Ejecuta el comando **'sudo unshadow /etc/passwd /etc/shadow > output.db'** para superar las restricciones de permisos y crear el archivo **'output.db'** que contiene la combinación de los archivos **'/etc/passwd'** y **'/etc/shadow'.**

Recuerda que el comando **'sudo'** te permite ejecutar comandos con privilegios de administrador. Asegúrate de tener los permisos adecuados y de utilizarlo con precaución.

```
┌──(kali㉿kali)-[~]
└─$ unshadow /etc/passwd /etc/shadow > output.db
fopen: /etc/shadow: Permission denied
```

Basado en la imagen proporcionada, intente en ejecutar el comando **'unshadow /etc/passwd /etc/shadow > output.db'**. Sin embargo, recibí un mensaje de negación al intentar acceder al archivo **'/etc/shadow'**.

Para superar esta restricción y acceder al archivo '**/etc/shadow'**, necesitarás utilizar el comando **'sudo'**. El comando **'sudo'** te permite ejecutar comandos con privilegios de administrador. Por lo tanto, puedes intentar ejecutar el siguiente comando:

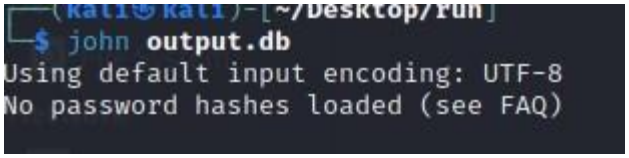
'sudo unshadow /etc/passwd /etc/shadow > output.db'

Al utilizar **'sudo',** se te solicitará la contraseña de administrador para verificar tu identidad y, luego, se ejecutará el comando con los privilegios adecuados.

Recuerda tener cuidado al utilizar comandos con privilegios de administrador y asegurarte de entender las implicaciones de tus acciones.

según la imagen proporcionada, utilizamos el comando **'sudo**' para acceder y me solicitó la clave de usuario de Kali para verificar mi identidad. Después de ingresar la clave, se muestra que el directorio fue creado exitosamente.

Ahora procederemos a ejecutar el comando **'john output.db'** para intentar encontrar las contraseñas de las cuentas. Sin embargo, hemos encontrado con algún tipo de error durante la ejecución.



Así que tuve que ejecutar el siguiente comando: **'sudo john --format=crypt output.db'.** Esto me permitió buscar la contraseña en el archivo **'output.db',** el cual estaba en formato cifrado.

Al utilizar el parámetro **'--format=crypt'**, indiqué al programa '**john'** que el archivo **'output.db'** estaba en un formato de cifrado específico. Esto era importante para que el programa pudiera aplicar los métodos de búsqueda de contraseñas adecuados.

Con este comando**, 'john'** utilizó diferentes técnicas de ataque, como fuerza bruta, diccionario y combinaciones, para intentar encontrar la contraseña en el archivo cifrado.

Recuerda que el proceso de búsqueda de contraseñas puede llevar tiempo, especialmente si se utilizan técnicas de ataque más complejas. Es posible que el programa **'john'** pruebe múltiples combinaciones antes de encontrar la contraseña correcta.

```
┌──(kali㉿kali)-[~/Desktop/run]
└─$ john --format=crypt output.db
Using default input encoding: UTF-8
Loaded 4 password hashes with 4 different salts (crypt, generic crypt(3) [?/64])
Remaining 2 password hashes with 2 different salts
Cost 1 (algorithm [1:descrypt 2:md5crypt 3:sunmd5 4:bcrypt 5:sha256crypt 6:sha512crypt]) is 0 for all loaded hashes
Cost 2 (algorithm specific iterations) is 1 for all loaded hashes
Will run 2 OpenMP threads
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
1234             (hola2)
```

Aquí podemos ver que, con el comando que ejecutamos, encontramos una cuenta llamada **'hola2'** con la contraseña **'1234'**. El proceso de descubrimiento de esta contraseña tomó aproximadamente **48 segundos.**

Ahora procederemos a ejecutar el mismo comando para intentar encontrar una contraseña más fuerte.

3. Creación de la cuenta con contraseña fuerte.

Vamos a crear una cuenta con el nombre **'nicolasm'.** Completaremos los datos requeridos, que incluyen el nombre completo, número de habitación, número de trabajo, número de celular y cualquier otra información adicional solicitada.

Asegúrate de proporcionar la información correcta y precisa para completar el proceso de creación de la cuenta de **'nicolasm'.**

Recuerda que es importante ingresar la información de manera adecuada para mantener la precisión y la integridad de los datos.

```
NAME_REGEX in configuration.

┌──(kali㉿kali)-[~]
└─$ sudo adduser nicolasm
Adding user `nicolasm' ...
Adding new group `nicolasm' (1001) ...
Adding new user `nicolasm' (1001) with group `nicolasm (1001)' ...
Creating home directory `/home/nicolasm' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for nicolasm
Enter the new value, or press ENTER for the default
        Full Name []: niko
        Room Number []: 123
        Work Phone []: 123
        Home Phone []: 234
        Other []: 9
Is the information correct? [Y/n] y
Adding new user `nicolasm' to supplemental / extra groups `users' ...
Adding user `nicolasm' to group `users' ...
```

Ahora que hemos creado la cuenta para '**nicolasm**', vamos a ejecutar el siguiente comando: **'sudo unshadow /etc/passwd /etc/shadow > output.db'.** Esto nos permitirá combinar los archivos **'/etc/passwd'** y **'/etc/shadow'** en un solo archivo llamado **'output.db'.**

Asegúrate de ingresar la contraseña cuando se te solicite al utilizar el comando **'sudo'.** Una vez ingresada la contraseña, el comando se ejecutará y se creará el directorio necesario para su uso posterior.

Este directorio será importante para futuros pasos y procesos relacionados con la cuenta de **'nicolasm'.**

Ahora vamos a verificar que todos los datos se hayan guardado correctamente en el archivo **'output.db'** utilizando el comando **'cat'.**

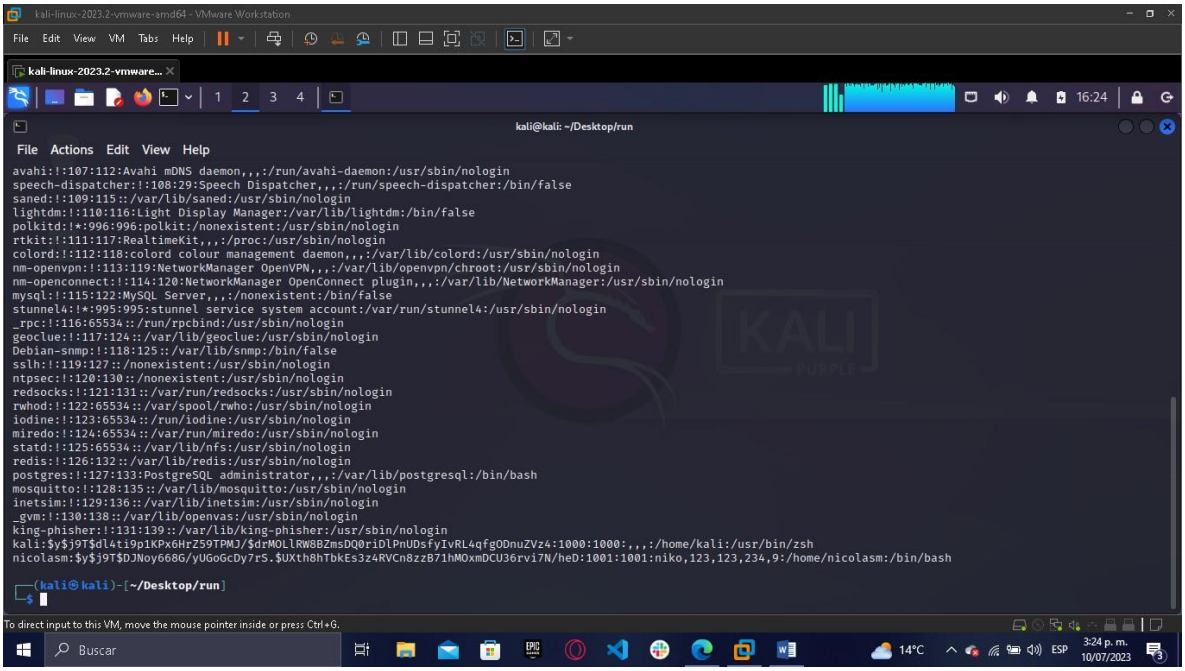
Ejecutaremos el comando **'cat output.db'** para mostrar el contenido del archivo y asegurarnos de que todos los datos estén guardados correctamente.

Recuerda que el comando **'cat'** se utiliza para mostrar el contenido de un archivo en la terminal.

Si todo está en orden, podremos ver los datos guardados en **'output.db'** en la salida del comando **'cat'**.

Universidad de los Andes

```
┌──(kali㉿kali)-[~/Desktop/run]
└─$ cat output.db
root:*:0:0:root:/root:/usr/bin/zsh
daemon:*:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:*:2:2:bin:/bin:/usr/sbin/nologin
sys:*:3:3:sys:/dev:/usr/sbin/nologin
sync:*:4:65534:sync:/bin:/bin/sync
games:*:5:60:games:/usr/games:/usr/sbin/nologin
man:*:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:*:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:*:8:8:mail:/var/mail:/usr/sbin/nologin
news:*:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:*:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:*:13:13:proxy:/bin:/usr/sbin/nologin
www-data:*:33:33:www-data:/var/www:/usr/sbin/nologin
backup:*:34:34:backup:/var/backups:/usr/sbin/nologin
list:*:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:*:39:39:ircd:/run/ircd:/usr/sbin/nologin
_apt:*:42:65534::/nonexistent:/usr/sbin/nologin
nobody:*:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-network:!*:998:998:systemd Network Management:/:/usr/sbin/nologin
systemd-timesync:!*:997:997:systemd Time Synchronization:/:/usr/sbin/nologin
messagebus:!:100:107::/nonexistent:/usr/sbin/nologin
tss:!:101:109:TPM software stack,,,:/var/lib/tpm:/bin/false
strongswan:!:102:65534::/var/lib/strongswan:/usr/sbin/nologin
tcpdump:!:103:110::/nonexistent:/usr/sbin/nologin
usbmux:!:104:46:usbmux daemon,,,:/var/lib/usbmux:/usr/sbin/nologin
sshd:!:105:65534::/run/sshd:/usr/sbin/nologin
dnsmasq:!:106:65534:dnsmasq,,,:/var/lib/misc:/usr/sbin/nologin
```



Al verificar el contenido del archivo **'output.db'** utilizando el comando **'cat'**, observamos que se muestran los usuarios, las contraseñas sin descifrar, los identificadores de grupo, los identificadores de usuario y los nombres reales asociados a los datos que llenamos en el paso anterior.

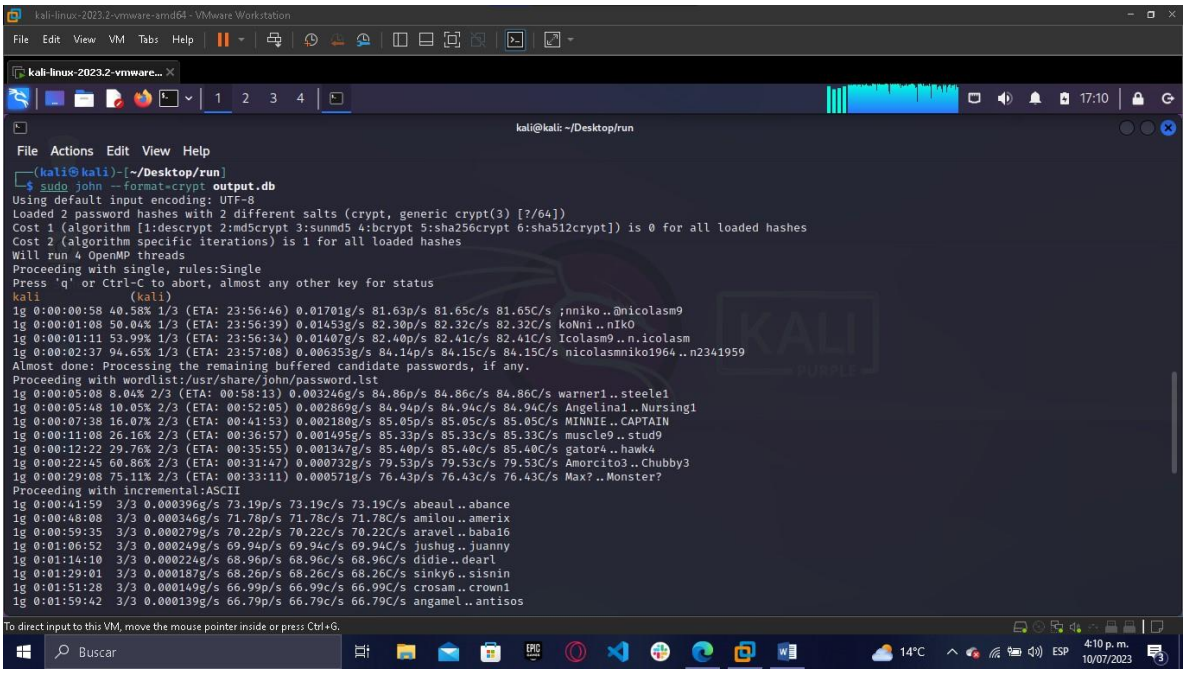
Es importante tener en cuenta que las contraseñas se muestran sin descifrar en el archivo **'output.db'.** Esto puede ser útil para analizar la seguridad de las contraseñas y evaluar su fortaleza.

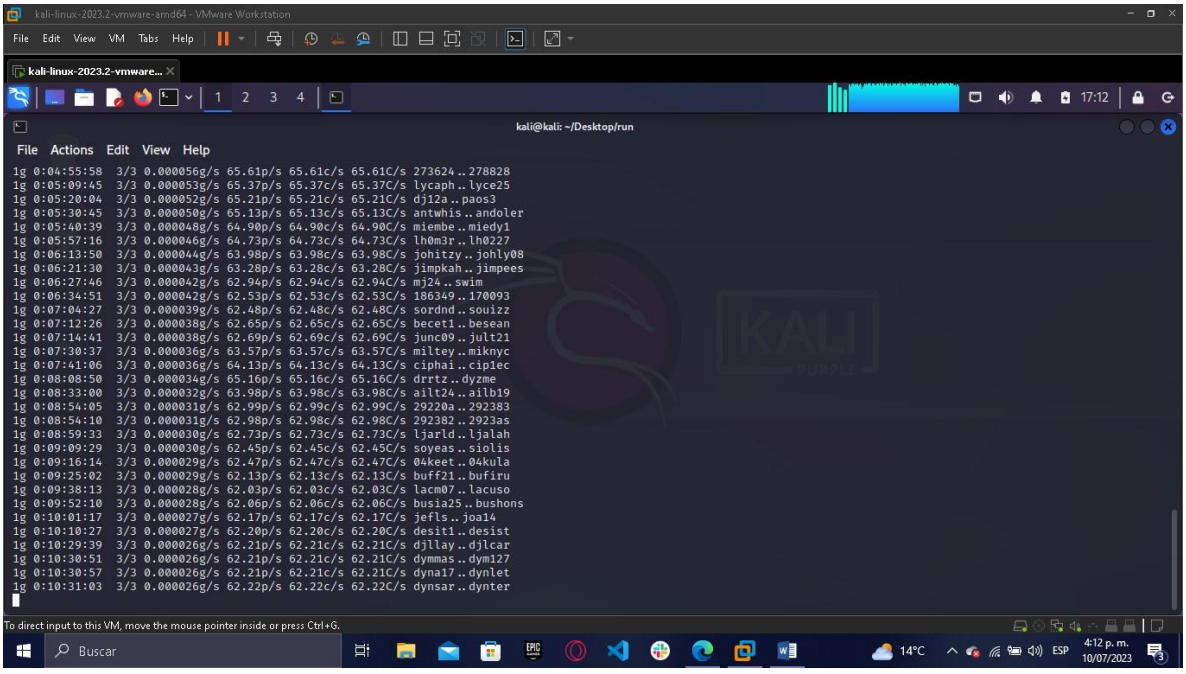
Recuerda que es esencial mantener la confidencialidad de las contraseñas y asegurarte de que sean seguras y robustas para proteger la información y las cuentas asociadas.

Continuaremos con el siguiente paso, que consiste en ejecutar el comando **'john'** para intentar encontrar la contraseña. Sin embargo, ya sabemos que simplemente ejecutar **'john'** no funcionará, ya que aparecerá una advertencia indicando que no se pudieron cargar los hashes.

Para solucionar esto, utilizaremos la ayuda del comando **'sudo'** para realizar el proceso con privilegios de administrador y agregaremos el parámetro **'--format=crypt'** para asegurarnos de que el formato de los hashes sea el correcto.

Ejecutaremos el siguiente comando: **'sudo john --format=crypt output.db'.** De esta manera, el programa **'john'** utilizará diferentes técnicas de ataque, como fuerza bruta, diccionario y combinaciones, para intentar encontrar las contraseñas almacenadas en el archivo.





En las capturas se puede observar que el proceso de encontrar la contraseña está tomando mucho tiempo. Hasta el momento, llevamos 10 horas de proceso y aún no se ha encontrado la contraseña. También hemos intentado cambiar de aplicación a VMware para ver si el proceso se aceleraba, pero aún así se está demorando.

Es importante tener paciencia y comprender que el proceso de búsqueda de contraseñas puede llevar tiempo, especialmente si se utilizan técnicas de ataque más complejas y las contraseñas son robustas.

Recuerda que la seguridad de las contraseñas es crucial para proteger la información y las cuentas asociadas. Es recomendable utilizar contraseñas seguras y seguir buenas prácticas de seguridad.

En este documento, se encuentran todas las actividades que has realizado con sus explicaciones detalladas de los comandos utilizados. Esto servirá como referencia para futuras consultas y como registro de las acciones realizadas.

Muchas gracias por leer