JtR

¿Qué es John the Ripper?

John the Ripper es una herramienta de crackeo de contraseñas escrita en C y muy utilizada por los analistas de seguridad para comprobar la robustez de una clave frente a ataques de fuerza bruta. Este programa es capaz de romper los hashes MD5, SHA-1 y otros muchos ampliamente utilizados en el mundo informático. Este programa es capaz de detectar de forma automática el tipo de hash que estamos crackeando, con el objetivo de facilitar al usuario su crackeo sin necesidad de preocuparse por el tipo de hash que está intentando «romper».

Algunas características muy importantes de este programa es que está optimizado para muchos modelos de procesadores, funciona en muchas arquitecturas de PC y también en diferentes sistemas operativos, no obstante, generalmente se utiliza en sistemas operativos basados en Linux, de hecho, las principales distribuciones Linux orientadas al pentesting y a la seguridad informática ya incorporan de forma predeterminada este programa.

Este programa es muy personalizable, nos permite definir la longitud a probar de una contraseña, para generar todas las combinaciones posibles y lograr el objetivo de crackear el hash. También nos permite configurar qué rango de letras, números o símbolos podemos probar para descifrar la contraseña, además, también permite incluir reglas para decidir cómo deben hacerse las diferentes variaciones.

John the Ripper permite pausar el crackeo de la contraseña y continuarlo en otro momento, esto es algo ideal por si tenemos que apagar nuestro PC o servidor, además, se puede automatizar para empezar a crackear una determinada contraseña al arrancar nuestro ordenador, y todo ello de forma automática sin necesidad de intervención del administrador de sistemas.

Algunos de sus usos, es en el ámbito de administración, este se usa para evitar que los usuarios establezcan contraseñas con bajos niveles de seguridad, o fáciles. Para esto, con la versión gratuita tendremos de sobra, pues nos indicará si las contraseñas son lo suficientemente seguras, sin necesidad de que ningún administrador sepa cuál es la propia contraseña.

¿Qué versiones existen?

Actualmente, nos podemos encontrar tres versiones de este software. En primer lugar, tenemos la versión gratuita, que es la más utilizada, luego tenemos John the Ripper Pro, y la versión John The Ripper Jumbo. Cabe destacar, que se trata de software libre, distribuido bajo licencia GPL, que si bien permite que algunas partes se usen con otras licencias, otras están bajo el dominio público. En un principio fue creado para funcionar en sistemas Unix, pero actualmente podemos usarlo en unos 15 sistemas operativos diferentes. Entre ellos, once tipos de Unix, MS-DOS, Windows, BeOS y OpenVMS. Pero lo más habitual es encontrarlo en las distribuciones de Linux.

Si hablamos de la versión Pro, obviamente tendremos algunas ventajas frente a la gratuita. Por ejemplo, en esta versión de pago nos detectará de forma automática cualquier mejora en la tecnología que soporte el procesador de la máquina en la que se instaló. También tendremos un diccionario con más de 4 millones de entradas, donde detectará las mejoras de forma automática. De este modo nos ahorrará tiempo al evitarnos tener que realizar actualizaciones.

John the Ripper Jumbo, se trata de un parche que permite trabajar con una mayor cantidad de algoritmos. Como se trata de la versión que está en desarrollo, muchas de sus funciones no están a su máximo rendimiento, por lo cual puede llegar a presentar algún inconveniente.

Similitudes entre John the Ripper y Hashcat:

John the Ripper y **Hashcat** son dos herramientas ampliamente utilizadas para la auditoría de contraseñas, especialmente en el ámbito de la **seguridad informática** y las **pruebas de penetración**. Aunque ambas herramientas tienen el mismo objetivo, que es descifrar hashes de contraseñas, tienen diferencias clave en términos de funcionalidades, velocidad, compatibilidad y uso. A continuación tienes un resumen detallado de sus similitudes y diferencias:

Similitudes entre John the Ripper y Hashcat:

1. **Objetivo Común**:
    a. Ambas herramientas se utilizan para **romper o auditar contraseñas** a través de técnicas de descifrado, como ataques de diccionario, fuerza bruta y combinaciones híbridas.

b. Están diseñadas para **analizar la seguridad** de los sistemas mediante la identificación de contraseñas débiles o mal gestionadas.

2. **Compatibilidad con múltiples algoritmos de hashing**:
    a. Ambas soportan una amplia gama de **algoritmos de hash** usados para almacenar contraseñas, como MD5, SHA-1, SHA-256, NTLM, bcrypt, entre otros.

3. **Técnicas de ataque**:
    a. Ambas admiten varios tipos de ataques como:
        i. **Fuerza bruta** (brute force).
        ii. **Ataques de diccionario**.
        iii. **Ataques híbridos**, combinando diccionarios y ataques de fuerza bruta.
        iv. **Ataques por regla**, que permiten modificar las entradas del diccionario para generar variaciones (e.g., cambiando letras por números o aplicando mayúsculas/minúsculas).

4. **Código abierto**:
    a. **John the Ripper** es completamente de código abierto, mientras que **Hashcat** tiene una versión de código abierto llamada **oclHashcat** (anteriormente), aunque la versión principal actual de Hashcat también es gratuita y abierta en gran medida.

5. **Extensible y personalizable**:
    a. Ambas herramientas permiten agregar o definir nuevas reglas y ajustar configuraciones para personalizar los ataques según las necesidades del usuario.

Diferencias entre John the Ripper y Hashcat:

1. **Velocidad y rendimiento**:
    a. **Hashcat** es generalmente **más rápido** que John the Ripper, especialmente en la ejecución de ataques a gran escala. Esto se debe principalmente a su capacidad para aprovechar de manera más efectiva el hardware moderno, especialmente las **GPUs** (tarjetas gráficas).
    b. **John the Ripper** puede funcionar en **CPU**, y aunque soporta aceleración por GPU (con versiones específicas como **John the Ripper Jumbo**), su rendimiento en esta área es menos optimizado que Hashcat.

2. **Compatibilidad de hardware**:

    a. **Hashcat** está diseñado para aprovechar al máximo las **GPUs** de marcas como **NVIDIA** y **AMD**, lo que le otorga una ventaja significativa en términos de rendimiento en ataques de fuerza bruta.

    b. **John the Ripper** funciona principalmente con **CPU**, pero también tiene versiones optimizadas para GPUs, aunque no tan eficientes como Hashcat.

3. **Facilidad de uso y configuración**:

    a. **John the Ripper** tiene una interfaz relativamente más sencilla para usuarios novatos, siendo más accesible para aquellos que se inician en la seguridad informática.

    b. **Hashcat**, aunque muy potente, puede ser más complicado de configurar y usar inicialmente, ya que requiere un conocimiento más profundo de las configuraciones de GPU y los algoritmos de hash que se quieren romper.

4. **Compatibilidad con sistemas operativos**:

    a. **John the Ripper** es multiplataforma y funciona en una amplia variedad de sistemas operativos como **Linux**, **macOS** y **Windows**.

    b. **Hashcat** también es multiplataforma, pero tiene dependencias más específicas para sistemas operativos, particularmente cuando se utilizan GPUs.

5. **Algoritmos soportados**:

    a. Aunque ambos soportan una amplia gama de algoritmos de hash, **Hashcat** tiende a soportar **más algoritmos** y tiene mejor optimización para los hash modernos y complejos (como los de criptomonedas y ciertos tipos de autenticación).

    b. **John the Ripper** también soporta muchos algoritmos, pero puede estar un poco más limitado en algunos casos, a menos que se use la versión **Jumbo**, que añade más soporte de algoritmos.

6. **Modos de ataque adicionales**:

    a. **Hashcat** ofrece algunos **modos de ataque** adicionales, como ataques basados en máscaras y combinaciones que son extremadamente poderosos y optimizados.

    b. **John the Ripper**, aunque flexible, no es tan especializado en algunos de estos modos avanzados.

7. **Comunidad y soporte**:

    a. **Hashcat** tiene una comunidad activa que proporciona actualizaciones frecuentes y soporte en foros y GitHub.

b. **John the Ripper** también tiene una comunidad activa, pero las actualizaciones pueden no ser tan frecuentes como en Hashcat.

Conclusión:

- **John the Ripper** es ideal para usuarios que buscan una herramienta **fácil de usar**, con **buen soporte para CPU** y adecuada para ataques más simples o entornos sin necesidad de un hardware especializado.
- **Hashcat** es la elección preferida para aquellos que buscan **mayor rendimiento** y están dispuestos a utilizar **GPUs** para romper hashes de forma más rápida y eficiente, especialmente en entornos de pruebas a gran escala o con algoritmos de hash más modernos y complejos.

Ambas herramientas son esenciales en el arsenal de un profesional de seguridad y se pueden usar de forma complementaria dependiendo de las necesidades del ataque o auditoría.

Feature	Hashcat	John the Ripper
Hardware Acceleration	Supports GPU, CPU or both	Supports GPU, CPU or just CPU
GPU Algorithms Supported	All supported algorithms	Limited algorithm support on GPU
Multi-GPU Support	Automatic task splitting across GPUs	Requires manual config for multi-GPU support
Installation	Pre-built binaries or building from source	Building from source recommended for optimal performance
Hash Type Specification	Requires explicit hash type	Auto-detects hash type, or can specify

Modos de funcionamiento

John The Ripper tiene un funcionamiento que se basa en generar códigos hash de miles de palabras que estén incluidas en un archivo de texto plano, el cual se conoce como diccionario. No obstante, para tener una idea más clara de cómo funciona, hay que conocer los diferentes modos de funcionamiento de esta herramienta de código abierto que viene instalada por defecto en sistemas Kali Linux. Por tanto, los modos son:

- **Single crack**

En este caso se trata de un modo en el que la herramienta usa el nombre del usuario con la finalidad de poder generar distintas combinaciones y así conseguir un código hash que se pueda comparar con el que está intentando crackear.

- **Diccionario**

Por otro lado, esta este segundo modo en el que se usa una lista de palabras ya predeterminadas, las cuales se pueden adquirir de distintas maneras. Hay que tener en cuenta que hay ya diccionarios más conocidos, mientras que otros usuarios se basan en crear listas de palabras a partir de los datos que se van filtrado de diferentes bases.

- **Incremental**

Mediante este modo, la herramienta de código abierto intenta crackear la contraseña con un ataque por fuerza bruta. En cuanto a las reglas que usa, son previamente definidas por el software en sí.

- **Externo**

Por último, pero no menos importante está esta alternativa en la que se usa una herramienta externa a John. Este puede servir para ejecutar ataques de fuerza bruta por ejemplo.

¿Cuáles son los archivos más importantes de John?

john.conf	**Archivo de configuración en Linux**
john.ini	**Archivo de configuración en Windows**
john.pot	**En este archivo se almacenan los passwords que John a crackeado**
john.rec	**En caso de estar realizando un proceso,John almacena su estado en este archivo, lo actualiza cada 10 minutos y en caso de que suceda algo inesperado, John utiliza este archivo para continuar donde había quedado**

El archivo john.conf

Aquí se definen las opciones generales

```
[Options]
# Wordlist file name, to be used in batch mode
Wordlist = $JOHN/password.lst
# Use idle cycles only
Idle = Y
# Crash recovery file saving delay in seconds
Save = 600
# Beep when a password is found (who needs this anyway?)
Beep = N
```

- Wordlist = PATH

En esta opción puede configurar la ruta hacia su diccionario o Worlist

- Idle = Y

Esta opción puede ser configurada con “Y” (si) o “N” (no), viene por defecto configurada como “Y”, lo cual permitirá que John trate de usar solo los ciclos en el que el procesador está inactivo

- Save = 600

Intervalo de tiempo en el cual se grabará el estado en el archivo john.rec, 600 segundos equivalen a 10 minutos,

- Beep = N

Si está configurado con “Y”, emite un “sonido” al encontrar una contraseña

Opciones y reglas para el modo Single crack

```
# "Single crack" mode rules
[List.Rules:Single]
```

Opciones y reglas para el modo Wordlist

```
# Wordlist mode rules
[List.Rules:Wordlist]
# Try words as they are
```

Instalación y prueba de rendimiento

Actualmente este programa está disponible en todos los repositorios de las principales distribuciones de sistemas operativos Linux, por tanto, lo podremos instalar de forma fácil y rápida sin problemas. En caso de que no esté disponible para tu distribución, siempre vas a poder añadirlo posteriormente editando el archivo de los repositorios

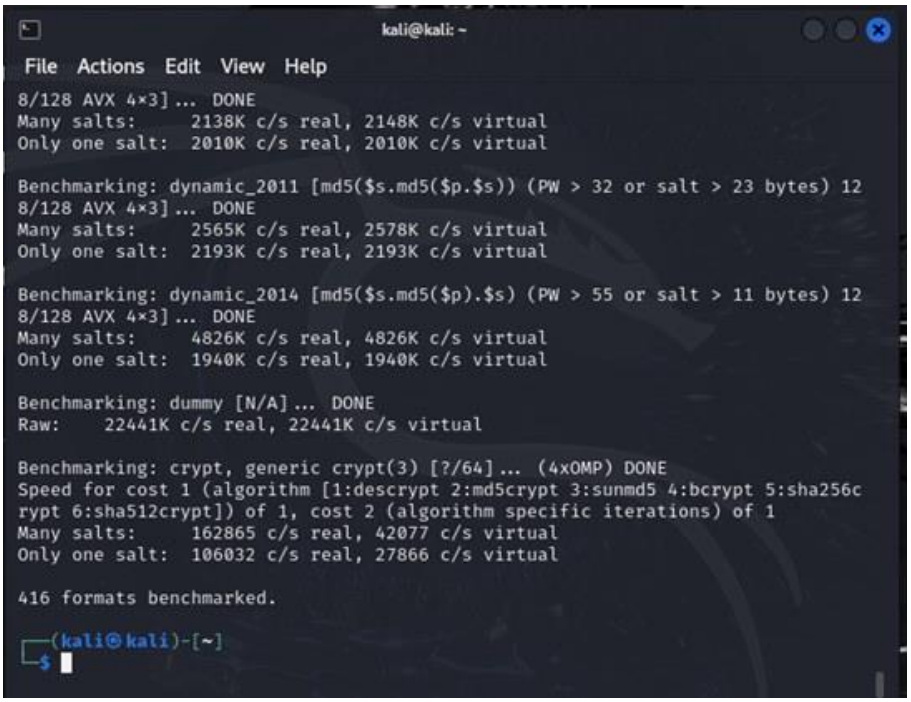
sudo apt install john

Una vez instalada la herramienta en nuestro sistema operativo, ya podremos utilizarla y empezar a crackear contraseñas, no obstante, sería muy recomendable antes de empezar a crackear claves, realizar una prueba rápida de velocidad de nuestro ordenador.

Probar el rendimiento en nuestro PC

Antes de empezar con el crackeo de las contraseñas podemos lanzar un sencillo test de rendimiento donde se pondrá a prueba nuestro hardware. De esta manera, podremos saber la velocidad con la que la herramienta probará claves con diferentes tipos de cifrado utilizando el 100% de nuestra CPU. Para ello simplemente abrimos un terminal Linux y tecleamos:

john --test



Como podemos ver, se llevan a cabo una serie de tests donde se medirá el rendimiento, esto nos puede dar una idea en general de la potencia de procesamiento de nuestro ordenador, con el objetivo de probar todas las combinaciones de letras, números y símbolos en el menor tiempo posible.

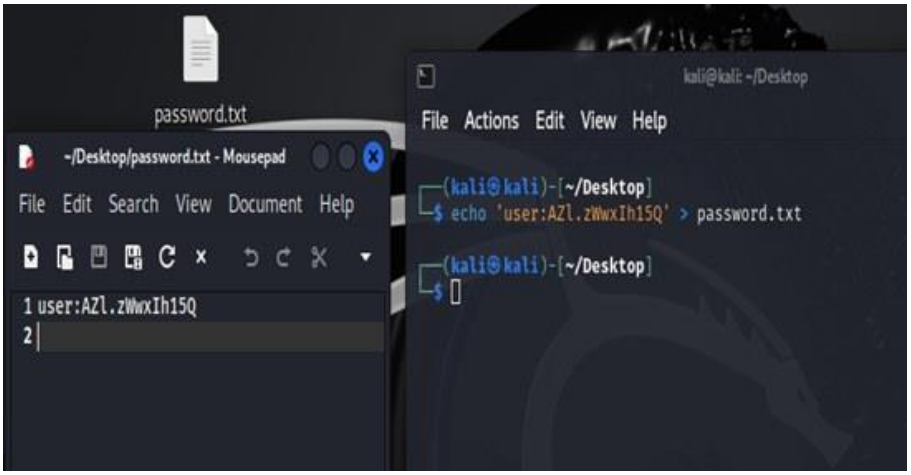
Para empezar a familiarizarnos con John vamos a crear un documento manualmente con un usuario y una contraseña y le indicaremos a John que lo crackee. Vamos a hacer esto para obtener los resultados lo más rápidamente posible (vamos a utilizar una clave muy simple como ejemplo), y para tener una primera toma de contacto con la herramienta y familiarizarnos con ella.

Para ello creamos un nuevo archivo de texto llamado «password.txt», por ejemplo, con el siguiente contenido:

user:AZl.zWwxIh15Q

La forma de crear en Linux el archivo password.txt es escribiendo en la terminal el siguiente comando:

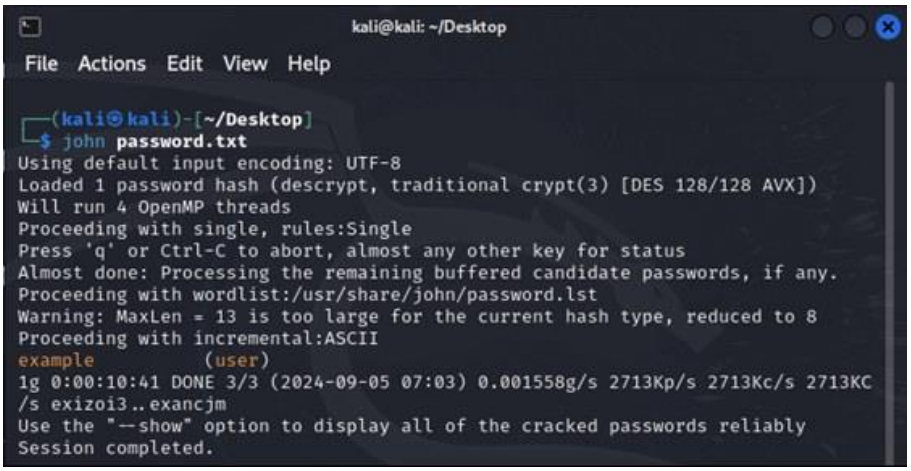
echo 'user:AZl.zWwxIh15Q' > password.txt



Una vez creado el archivo password.txt, vamos a indicar a John que empiece a trabajar para crackear la contraseña del archivo. Para ello tecleamos:

john password.txt

Debemos esperar a que esta herramienta consiga crackear la contraseña del anterior archivo. Este proceso puede tardar horas e incluso días según la dificultad de la misma.



Nuestra contraseña está crackeada. En este caso nos la muestra directamente, pero si no fuese así debemos utilizar el comando --show de la siguiente manera:

john --show password.txt

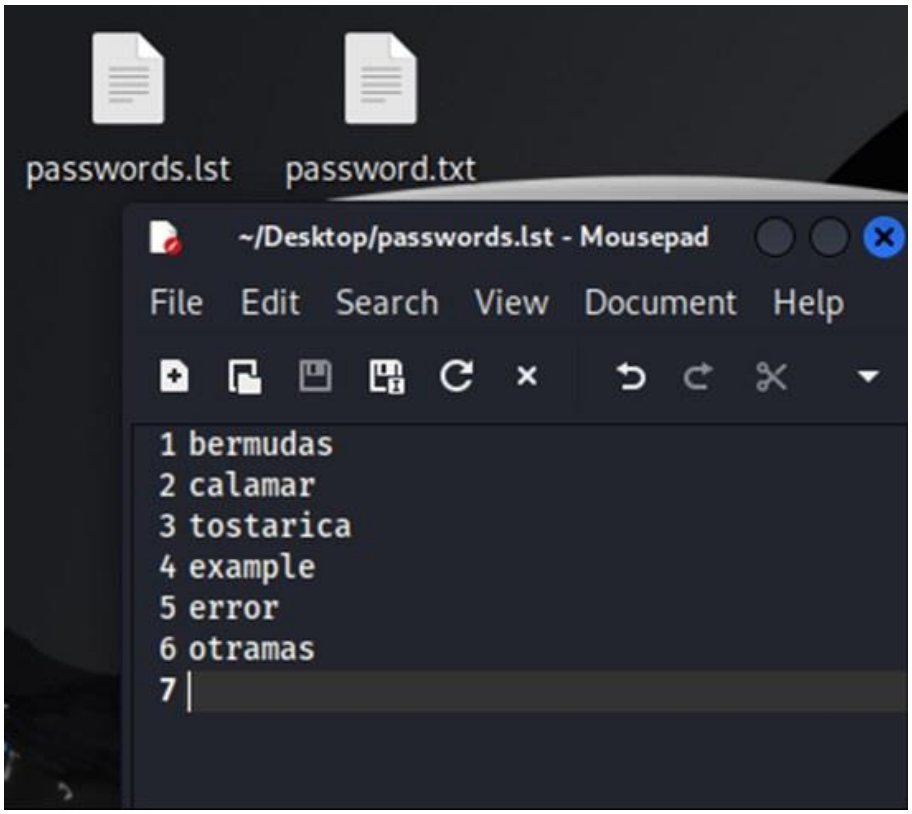
Nuestra contraseña era «example». Ya podemos intentar iniciar sesión en el sistema con el usuario «user» y la contraseña «example», o por lo menos, podríamos hacerlo si hubiéramos trabajado directamente con el fichero /etc/shadow, aunque el tiempo de crackeo hubiera tardado seguramente mucho más que varios minutos.

Crackear contraseñas usando un diccionario de claves

Al igual que en el anterior ejemplo, en este caso vamos a partir de la clave de ejemplo que hemos guardado a mano en el documento llamado «**password.txt**«:

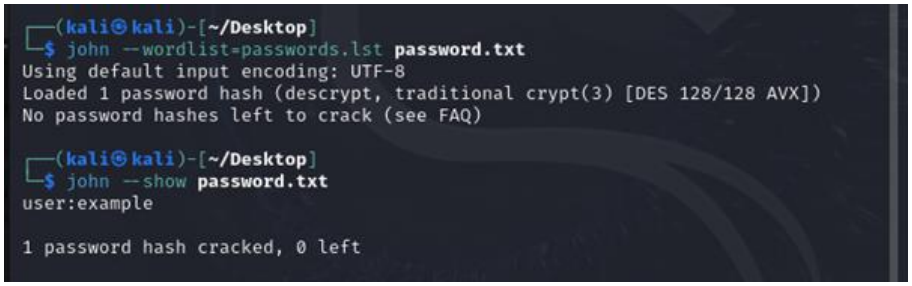
user:AZl.zWwxIh15Q

A continuación, lo que tenemos que hacer es tener o crear un diccionario de claves personalizado. Podemos descargar estos diccionarios de Internet, pero para hacer las primeras pruebas del programa vamos a crear nosotros un diccionario sencillo, al que llamaremos «**passwords.lst**» y en el que introduciremos varios valores, cada uno en una línea, pero siendo uno de ellos la palabra «example» (ya que corresponde con nuestra contraseña).



A continuación, simplemente debemos ejecutar John the Ripper con el parámetro --wordlist= seguido de la ruta de nuestro archivo.

john --wordlist=passwords.lst password.txt



Al tener una clave sencilla y pocas entradas del diccionario, el proceso será prácticamente instantáneo. Ya hemos crackeado, o descifrado, la contraseña. Lo único que nos queda por hacer es utilizar el parámetro --show para que nos muestre el resultado.

Tareas

En este proyecto, aprenderás a usar John the Ripper, una poderosa herramienta de craqueo de contraseñas, para romper contraseñas. Cracking password hash es una habilidad esencial para los hackers éticos, ya que ayuda a identificar contraseñas débiles y mejorar la seguridad general. Al final de este proyecto, podrás realizar crackeos básicos de contraseña y entender la importancia de las contraseñas fuertes.

Pre-requisitos

- Comprensión básica de conceptos de hashing y cifrado.
- Familiaridad con el uso de la interfaz de línea de comandos (CLI).
- Conocimiento básico de prácticas de seguridad de contraseñas.

Configuración de laboratorio y herramientas

Diagrama

Aquí hay un simple diagrama de red para esta configuración de laboratorio:

Atacante - Sistema de la Máquina Kali - (192.168.1.xxx)

Herramientas

- **Kali Linux:** Distribución Linux derivado de Debian diseñada para pruebas forenses digitales y de penetración.
- **John the Ripper:** Una herramienta incluida con Kali Linux.
- **Archivos Hash** : Archivos de muestra que contienen hashes de contraseña.

Instalación

John the Ripper está preinstalado en Kali Linux. Puedes verificar la instalación o actualizarlo usando el siguiente comando:

sudo apt-get update && sudo apt-get install john

Tarea 1: Basic Password Cracking

La mayoría de las aplicaciones web necesitan verificar la contraseña de un usuario en algún momento. Almacenar estas contraseñas en texto plano sería una mala práctica. Tampoco se podrían cifrar, ya que alguien que tenga acceso a la llave de cifrado (administradores o atacantes) podrían obtener todas las contraseñas almacenadas.

Aquí es donde entra el hash. ¿Qué pasaría si, en lugar de almacenar la contraseña, simplemente almacena el hash? Esto significa que nunca se tendría que almacenar la contraseña del usuario, y si se filtra la base de datos, un atacante tendría que comparar todos los hashes contra un diccionario para averiguar cuál era la contraseña.

Aunque los algoritmos de hash son muy difíciles de revertir. Eso no significa los hashes sean infalibles. Si tienes el algoritmo empleado y la cadena final, ejemplo:

Hash: SHA256

Cadena: 20bc360e9603edb25a6692d319d1e97444c74464699d09e1461ce08b60aa6e20

Lo que puedes hacer es tomar el diccionario y "hashear" todas las palabras de esta forma podrías deducir que el password empleado en este caso es: **window**

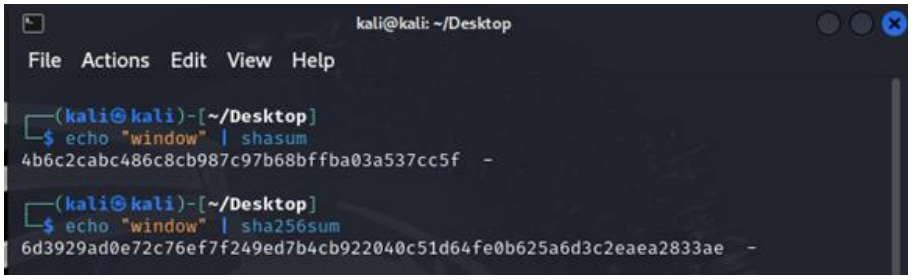
Este proceso es un ejemplo del tipo denominado ataque de diccionario y John the Ripper, o John, es una herramienta que te permite realizar este tipo de ataques, así como de fuerza bruta en una gran variedad de diferentes tipos de hash.

Con esta web podemos sacar el hash256 de una palabra online o también podemos usar la terminal de Kali para sacarlo. En este ejemplo usaremos la palabra "**window**"

https://hash.online-convert.com/es/generador-sha256

echo "window" | sha256sum

echo "window" | shasum



En esta imagen podemos apreciar la diferencia de complejidad entre un Sha y un Sha256

Window

hex: 44ff7b02c80d38b26dd6aa31d9470aed81b32e10331a3c994fb1a9945fd847ba

HEX:44FF7B02C80D38B26DD6AA31D9470AED81B32E10331A3C994FB1A9945FD847BA

h:e:x:44:ff:7b:02:c8:0d:38:b2:6d:d6:aa:31:d9:47:0a:ed:81:b3:2e:10:33:1a:3c:99:4f:b1:a9:94:5f:d8:47:ba

base64: RP97AsgNOLJt1qox2UcK7YGzLhAzGjyZT7GplF/YR7o=

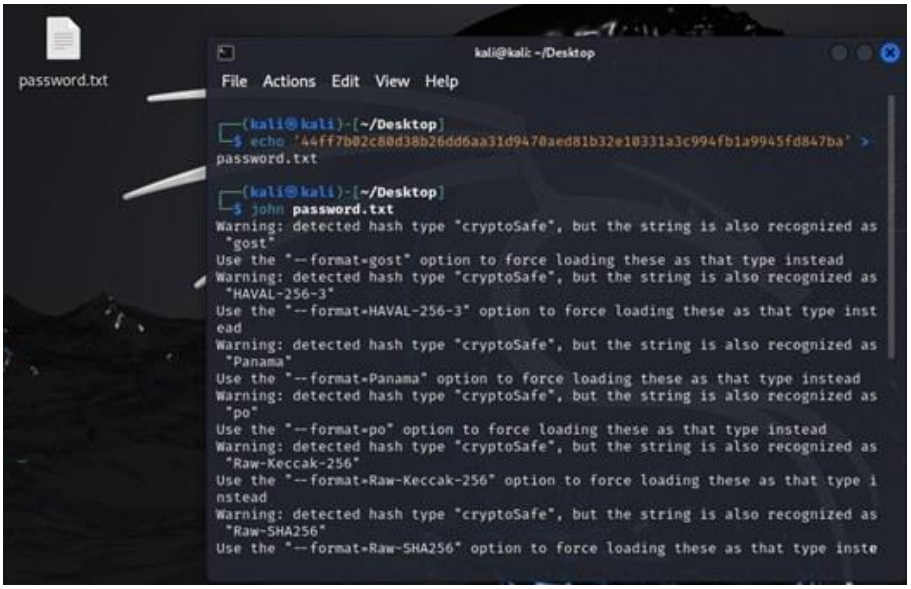
Paso1: Crear un archivo de texto llamado password.txt que contiene un hash de contraseña. Por ejemplo:

echo '44ff7b02c80d38b26dd6aa31d9470aed81b32e10331a3c994fb1a9945fd847ba' > password.txt

Paso2: Abre una terminal en tu máquina Kali Linux. Ejecuta John el Destripador para romper los hahs de contraseña en passwords.txt:

john password.txt

Al ejecutar el comando john password.txt nos indica que es un formato sha256 y nos pide que usemos el formato como parte de la instrucción:



sudo john --format=Raw-SHA256 –wordlist=/usr/share/john/password.lst password.txt

Salida esperada: John el Destripador debe comenzar el proceso de crackeadas y eventualmente mostrar las contraseñas crackeadas. Podemos comprobar como tarda 0 segundos en descifrar el hash. John ha encontrado la palabra que se esconde tras el hash porque la palabra estaba en su password.lst, Pero, ¿qué ocurre si la palabra no está en la lista? Entonces tendremos que usar un diccionario más extenso o crear un propio.

Para crear diccionarios extensos personalizados podemos utilizar Crunch

Tarea 2: Usando un Wordlist personal

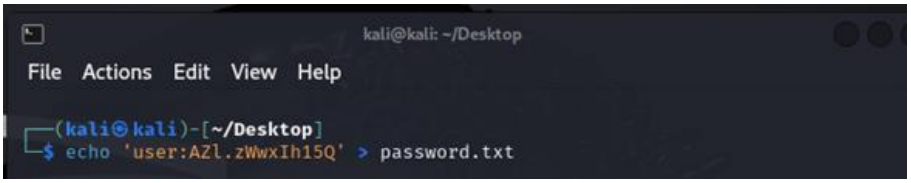
Paso1: Vamos a descargar un diccionario muy común llamado rockyou.txt que contiene contraseñas potenciales.

Lo podemos descargar desde github.

Vamos a utilizar el ejemplo del principio y volveremos a meter nuestra contraseña dentro de un archivo llamado password.txt:

user:AZl.zWwxIh15Q

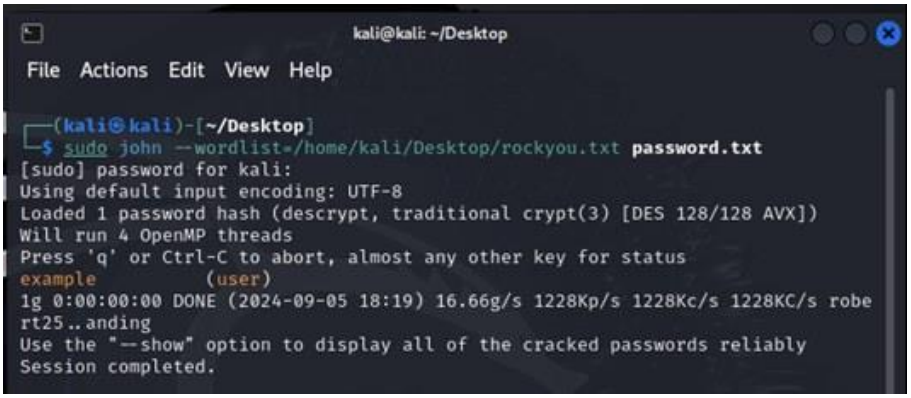
echo 'user:AZl.zWwxIh15Q' > password.txt



Paso2: Utiliza John el Destripador con el diccionario rockyou.txt para descifrar los hashs de contraseña:

sudo john --format=Raw-SHA256 --wordlist=/home/kali/Desktop/rockyou.txt password.txt

Salida esperada: John the Ripper debe usar el diccionario personalizado (en este caso rockyou.txt) para descifrar las contraseñas y mostrar los resultados.

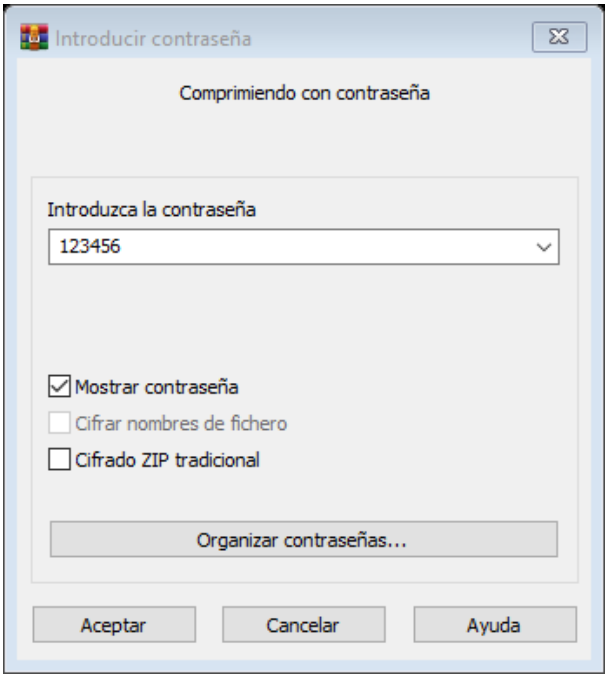


Tarea 3: Descifrar la Contraseña de un archivo Comprimido

Vamos a descifrar una contraseña de un archivo zip. Para hacer eso, primero tenemos que obtener el hash de la contraseña del archivo zip.

Como unshadow, John tiene otra utilidad llamada zip2john.

Paso1: Comprimiremos un archivo .zip protegido por contraseña:



zip2john nos ayuda en obtener el hash de los archivos zip. Si estás descifrando un archivo .rar, puedes usar la utilidad rar2john como veremos más adelante.

Esta es la sintaxis para obtener el hash de la contraseña de un archivo zip:

zip2john file.zip > zip.hashes

El comando de arriba obtendrá el hash de un archivo zip y lo almacenará en el archivo zip.hashes. Luego puedes usar John para descifrar el hash.

john zip.hashes

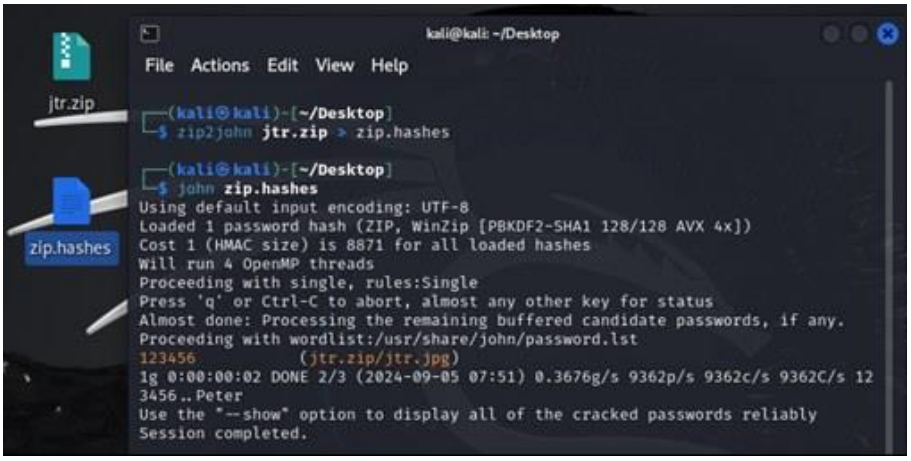
Paso2: Entonces sacamos el hash del archivo .zip:

zip2john jtr.zip > zip.hashes

Paso3: para luego descifrarlo:

john zip.hashes

El resultado es la contraseña con la que protegimos el archivo. Ya podemos abrirlo.



Paso4: Vamos a repetir la misma operación, pero esta vez con un archivo .rar y otra contraseña:

Esta es la sintaxis para obtener el hash de la contraseña de un archivo .rar:

rar2john file.rar > rar.hashes

El comando de arriba obtendrá el hash de un archivo zip y lo almacenará en el archivo zip.hashes. Luego puedes usar John para descifrar el hash.

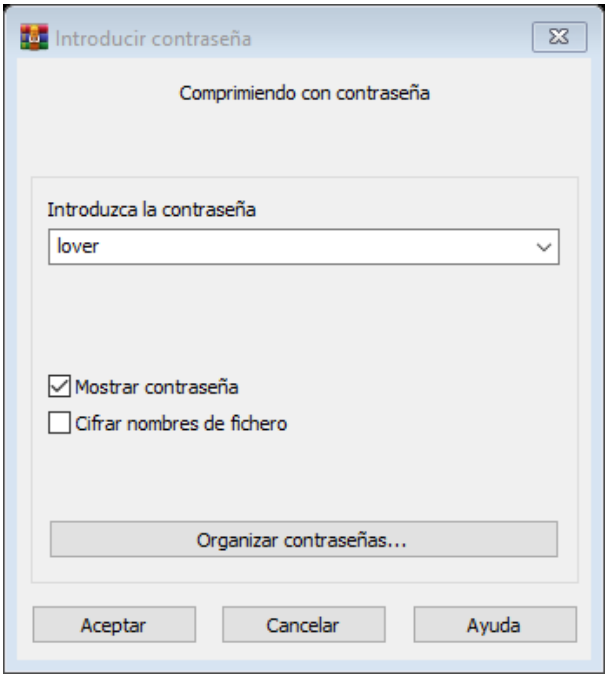
john rar.hashes

Entonces, en mi caso el nombre del .rar es JTR, luego:

rar2john JTR.rar > rar.hashes

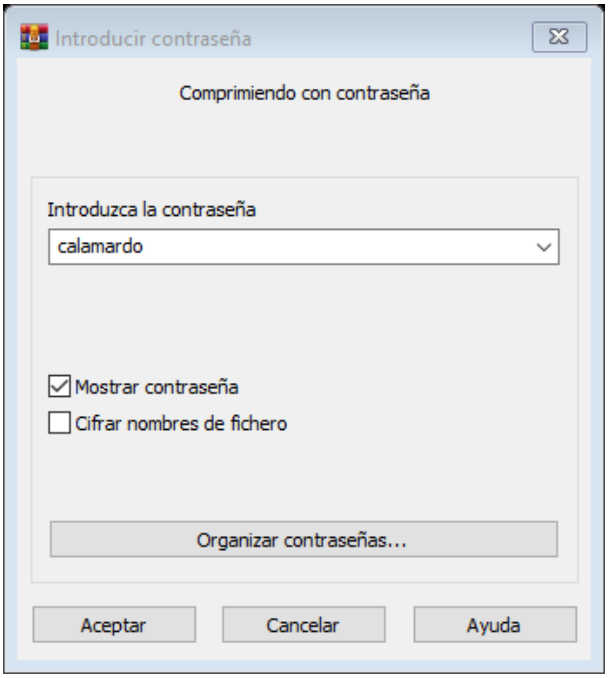
y luego:

john rar.hashes



En unos minutos descifra la contraseña utilizando el diccionario:

¿Y si la contraseña no está en el diccionario? Entonces empezaría a buscarla mediante el modo incremental. En ese caso, si la contraseña es larga podríamos tardar bastante más tiempo:



Tarea 4: Cracking Shadow File Hashes

Paso1: Conseguir un archivo de shadow de muestra que contiene hashs de contraseña.

En mi caso voy a utilizar en una prueba el fichero /etc/shadow de mi máquina virtual Kali, y en la otra prueba el fichero /etc/shadow de otra máquina.

El proceso implica dos pasos básicos; el primero se llama unshadowing mientras que el segundo es el crackeado en sí. Unshadow es un proceso en el que combinamos el archivo /etc/passwd junto con el /etc/shadow para que John pueda entender lo que le estamos alimentando. Unshadow es una herramienta que maneja esta tarea y es parte del paquete John. Para desensombrar el archivo de sombras necesitamos también tener la línea equivalente de la passwd para el usuario de nuestro interés. Por ejemplo:

/etc/passwd line

root:x:0:0:root:/root:/bin/bash

/etc/shadow line

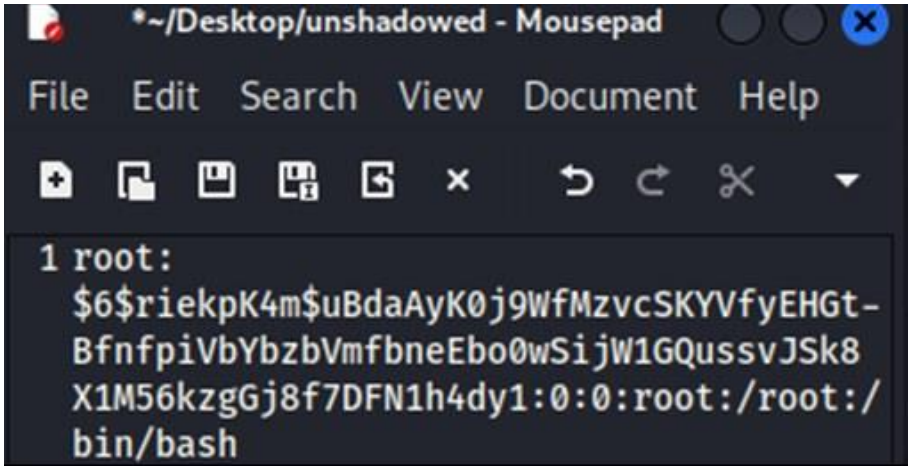
root:6riekpK4m$uBdaAyK0j9WfMzvcSKYVfyEHGtBfnfpiVbYbzbVmfbneEbo0wSijW1GQussvJSk8X1M56kzgGj8f7DFN1h4dy1:18226:0:99999:7:::

Paso2: Extraer los hashes de contraseña del archivo de sombra en un nuevo archivo llamado unshadowed. Con el fin de desensombrar a los dos archivos que necesitamos ejecutar:

sudo unshadow /etc/passwd /etc/shadow > unshadowed

Esto almacenará en el archivo unshadowed lo siguiente:

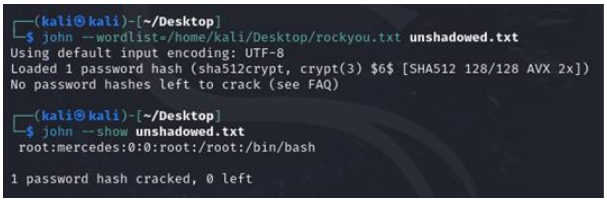
root:6riekpK4m$uBdaAyK0j9WfMzvcSKYVfyEHGtBfnfpiVbYbzbVmfbneEbo0wSijW1GQussvJSk8X1M56kzgGj8f7DFN1h4dy1:0:0:root:/root:/bin/bash



Paso3: Utiliza a John The Ripper para romper los hahs de la contraseña en unshadowed.hashes.txt:

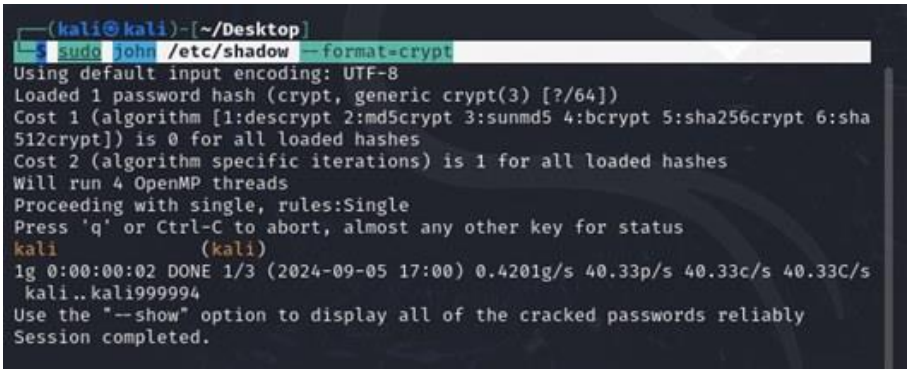
john --wordlist=/home/kali/Desktop/rockyou.txt unshadowed.txt

Salida esperada: John The Ripper debe comenzar el proceso de agrietado y eventualmente mostrar las contraseñas.



Adicionalmente podemos utilizar el siguiente comando para tratar de extraer la contraseña directamente del archivo /etc/shadow

└─$ sudo john /etc/shadow --format=crypt



Tarea 5: Cracking con modo incremental

Paso1: Utilíza a John The Ripper en modo incremental para intentar una grieta de fuerza bruta en los hahs de contraseña:

john --incremental passwords.txt

Vamos a volver al archivo llamado «**password.txt**« del principio:

user:AZl.zWwxIh15Q

Creamos el archivo password.txt:

echo 'user:AZl.zWwxIh15Q' > password.txt

Paso2: Usamos el comando --incremental:

john --incremental password.txt

Salida esperada: John The Ripper debe comenzar el proceso de fuerza bruta y mostrar las contraseñas a medida que las encuentra.

Tarea 6: Cracking contraseñas con reglas

¿Qué son las reglas en John the Ripper?

Las reglas son un conjunto de comandos o instrucciones que le dicen a JtR cómo modificar las contraseñas o las palabras de un diccionario para generar nuevas combinaciones. Estas reglas se definen en el archivo de configuración de JtR (john.conf), bajo la sección [List.Rules].

Cada regla se compone de un conjunto de acciones, como agregar, eliminar, reemplazar o modificar caracteres en las contraseñas originales.

Sintaxis básica de las reglas

Cada regla está formada por una cadena de caracteres donde cada carácter representa una acción o modificación específica que se debe aplicar a la contraseña original.

Ejemplo de una regla:

- A: Convierte la primera letra en mayúscula.
- z: Convierte la última letra en minúscula.
- "123: Añade "123" al final de la palabra.

Acciones comunes en las reglas

A continuación, se presenta una lista de los comandos más utilizados en las reglas de John the Ripper:

- **A**: Convierte la primera letra en mayúscula.
- **a**: Convierte la primera letra en minúscula.
- **l**: Convierte toda la palabra a minúsculas.
- **u**: Convierte toda la palabra a mayúsculas.
- **c**: Invierte el caso de todas las letras.
- **t**: Invierte el caso solo de la primera letra.
- **sX**: Sustituye todas las ocurrencias del carácter X por el siguiente carácter.
- **pX**: Añade el carácter X al principio de la palabra.
- **$X**: Añade el carácter X al final de la palabra.
- **r**: Reinvierte la palabra (es decir, invierte el orden de los caracteres).

Ejemplos prácticos de reglas

1. Capitalización de la primera letra:

Esta regla convierte la primera letra de cada palabra en mayúscula.

```css
A
```

- Palabra original: password
- Resultado: Password

2. Agregar números al final:

Añade un número al final de cada palabra.

```bash
$1
```

- Palabra original: password
- Resultado: password1

3. Reemplazar letras comunes por números (leet speak):

Sustituye letras por números siguiendo un patrón común de "leet speak"

```css
s[48]s[13]s[05]s[1!]
```

- s4: Reemplaza todas las ocurrencias de "a" por "4".
- s1: Reemplaza todas las ocurrencias de "l" por "1".
- s0: Reemplaza todas las ocurrencias de "o" por "0".
- s!: Reemplaza todas las ocurrencias de "i" por "!".
- Palabra original: password
- Resultado: p4ssw0rd

4. Añadir secuencias numéricas:

Esta regla genera palabras con secuencias numéricas agregadas al final.

```css
$[123]
```

- Palabra original: admin
- Resultado: admin1, admin2, admin3

5. Invertir el caso de la palabra:

Invierte el caso de todas las letras.

- Palabra original: Password
- Resultado: pASSWORD

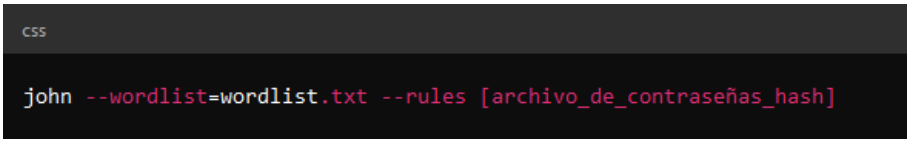
6. Invertir la palabra:

Invierte el orden de los caracteres en la palabra.

- Palabra original: password
- Resultado: drowssap

Uso de reglas en John the Ripper

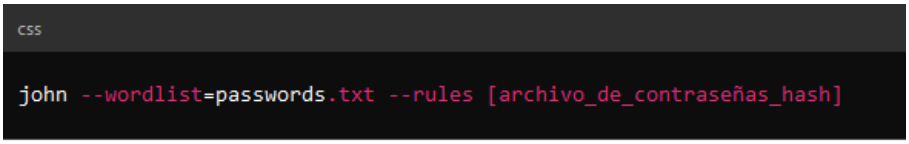
Para utilizar las reglas en JtR, estas se definen en el archivo de configuración, por lo general dentro de la sección [List.Rules:Wordlist]. Luego, puedes invocar las reglas durante el ataque con el siguiente comando:



- --wordlist=wordlist.txt: Define el diccionario que se usará.
- --rules: Aplica las reglas definidas en el archivo de configuración john.conf.

Ejemplo:

Supongamos que tienes un archivo de diccionario llamado passwords.txt y deseas probar diferentes variaciones usando las reglas de capitalización y números al final:



Esto aplicará las reglas predefinidas y generará variaciones como Password, password1, Password123, etc.

Reglas avanzadas

Puedes crear reglas más complejas combinando varias acciones en una misma línea. Por ejemplo:

```bash
Az$123r
```

- A: Capitaliza la primera letra.
- z: Minúscula en la última letra.
- $123: Añade "123" al final.
- r: Invierte la palabra.
- Palabra original: password
- Resultado: drowssap123

Conclusión

Las reglas en John the Ripper son una herramienta poderosa para incrementar las posibilidades de éxito en un ataque de fuerza bruta, ya que permiten generar múltiples variaciones de las contraseñas originales. Aprender a usarlas eficientemente, y crear reglas personalizadas basadas en los patrones comunes de contraseñas, puede optimizar significativamente los intentos de cracking.

Paso1: Utiliza a John The Ripper con un enfoque basado en reglas para romper los hahs de contraseña. Por ejemplo, utiliza las reglas por defecto proporcionadas por John The Ripper:

john --rules passwords.txt

Salida esperada: John The Ripper debe aplicar las reglas e intentar descifrar las contraseñas, mostrando cualquier resultado exitoso.

Tarea 7: Analizar contraseñas descifradas

El comando john --show en **John the Ripper** se utiliza para mostrar las contraseñas descifradas de un archivo de hashes. Este comando proporciona una manera rápida de ver cuáles contraseñas ya han sido descifradas sin tener que analizar manualmente los resultados de los ataques.

Sintaxis básica:

```css
john --show [archivo_de_hashes]
```

- --show: Muestra las contraseñas que ya han sido descifradas.
- [archivo_de_hashes]: Es el archivo que contiene los hashes de contraseñas que se están atacando.

Funcionamiento del comando john --show:

Cuando ejecutas John the Ripper en un archivo de hashes, todas las contraseñas descifradas se almacenan en el archivo john.pot (también llamado **potfile**). Este archivo es una especie de "base de datos" que John usa para registrar todas las contraseñas descifradas.

Al ejecutar john --show, el programa compara los hashes del archivo de entrada con las contraseñas almacenadas en john.pot, mostrando directamente los resultados sin necesidad de volver a realizar un ataque.

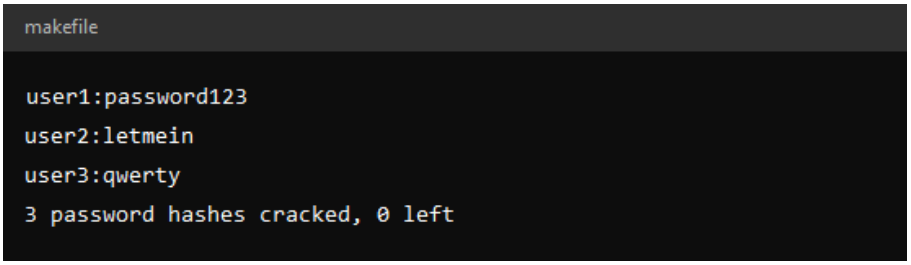
Ejemplo práctico:

Supongamos que tienes un archivo de hashes llamado hashes.txt y ya has ejecutado John the Ripper previamente para descifrar algunas contraseñas.

1. **Ejecutar el ataque de John the Ripper:**

2. **Mostrar las contraseñas descifradas:** Una vez que se han descifrado algunas contraseñas (o si ya habías ejecutado un ataque previamente), puedes usar el siguiente comando para verlas:

Salida del comando:



En este ejemplo:

- user1:password123: Se ha descifrado la contraseña del user1, que es password123.
- user2:letmein: El usuario user2 tiene como contraseña letmein.
- user3:qwerty: El usuario user3 tiene como contraseña qwerty.

Al final, John indica cuántos hashes han sido descifrados y cuántos quedan pendientes. En este caso, 3 contraseñas han sido descifradas y no quedan más por descifrar.

Detalles adicionales del comando john --show:

- **Formato de salida:** John te muestra el resultado en formato usuario:contraseña, lo que hace fácil identificar a qué usuario corresponde cada contraseña descifrada.
- **Uso del archivo** john.pot**:** Si tienes varios archivos de hashes y los has atacado en diferentes momentos, John almacena todas las contraseñas descifradas en el archivo john.pot, lo que significa que al usar john --show en cualquier momento, puedes ver las contraseñas descifradas para todos los archivos anteriores.

Opciones adicionales de john --show:

Mostrar todas las contraseñas, incluyendo las no descifradas: Si quieres ver tanto las contraseñas descifradas como las no descifradas, puedes agregar la opción --show=left:

```css
john --show=left hashes.txt
```

 Esto mostrará las contraseñas que **aún no han sido** descifradas.

 Visualizar contraseñas de otros archivos de hashes: Si quieres ver todas las contraseñas descifradas en algún archivo específico, puedes simplemente proporcionar el archivo correspondiente, tal como en el ejemplo

```css
john --show otro_archivo_de_hashes.txt
```

Restaurar un ataque: Si por alguna razón interrumpes un ataque y deseas continuarlo más tarde, puedes usar el comando:

```css
john --restore
```

Después, puedes ejecutar john --show para verificar las contraseñas descifradas hasta el momento.

Resumen:

El comando john --show es útil para obtener una vista rápida de las contraseñas descifradas y permite comprobar el progreso de los ataques realizados previamente. Recuerda que John siempre guarda las contraseñas en el archivo john.pot, por lo que no necesitas realizar el ataque nuevamente para obtener una lista de las contraseñas ya descifradas.

Paso 1: Después de descifrar las contraseñas, analiza los resultados para identificar patrones y debilidades comunes.

Paso2: Utilíza el comando john --show para mostrar todas las contraseñas descifradas:

john --show passwords.txt

Salida esperada: Una lista detallada de todas las contraseñas descifradas y sus correspondientes hashes, destacando patrones de contraseña comunes.

Recursos adicionales

Este proyecto le ayudará a entender cómo romper los hashes de contraseña usando John the Ripper en Kali Linux, destacando la importancia de políticas de contraseñas fuertes y algoritmos de hashing seguros.

John The Ripper Documentación Kali Linux

https://www.openwall.com/john/

https://github.com/openwall/john

https://www.sodapdf.com/password-protect-pdf

https://www.hackingarticles.in/beginner-guide-john-the-ripper-part-1/

https://www.hackingarticles.in/beginners-guide-for-john-the-ripper-part-2/

URL

https://www.openwall.com/john/

Installation

```
sudo apt-get install john
```

Pot Location

```
/root/.john/john.pot
~/.john/john.pot
```

Benchmark

```
john --test
```

Create Session

```
john hash.txt --session=session-name
```

Restore Session

```
john --restore=session-name
```

List Hash Formats

```
john --list=formats
```

List Rules

```
john --list=rules
```

View Status

```
john --status
```

Unshadow

```
unshadow passwd.txt shadow.txt > unshadowed.txt
```

Create Wordlist

```
john --wordlist=list.txt --stdout --external:[filter] > output.txt
```

Zip To John

```
zip2john file.zip > ziphash.txt
```

RAR To John

```
rar2john file.zip > rarhash.txt
```

Default Attack

```
john hash.txt
```

Incremental Attack

```
john --incremental hash.txt
```

Mask Attack

```
john -format=hashtype hash.txt -mask?a?a?a?a?a?a
```

Crack MD5

```
john --format=raw-md5 --wordlist=rockyou.txt hash1.txt
```

Crack SHA1

```
john --format=raw-sha1 --wordlist=rockyou.txt hash2.txt
```

Crack SHA256

```
john --format=raw-sha256 --wordlist=rockyou.txt hash3.txt
```

Crack Whirlpool

```
john --format=whirlpool --wordlist=rockyou.txt hash4.txt
```

Crack MD4

```
john --format=raw-md4 --wordlist=rockyou.txt hash5.txt
```

John Masks

```
?1 = lowercase          26 characters   abcdefghijklmnopqrstuvqxyz
?u = uppercase          26 characters   ABCDEFGHIJKLMNOPQRSTUVWXYZ
?d = digits             10 characters   0123456789
?s = special            33 characters   <>!"£$%^&*()_+{}[]'#~\/|?
?a = all                95 characters   upper,lower,digits,special
?h = hex                hex characters  0x00, 0xff
?A = all valid
?H = all except null
?L = non ASCII lower-case
?U = non ASCII upper-case
?D = non ASCII digits
?S = non ASCII specials
?w = Hybrid
```

John the Ripper Pentester Guide