Cracking Windows Passwords

¿Alguna vez te has preguntado cómo Windows almacena tus contraseñas y qué tan seguras son realmente?

En este artículo, exploraremos el modo de almacenamiento de contraseñas en Windows y cómo se pueden descifrar los hashes de estos password.

Proyecto con fines educativos donde desentrañaremos los secretos detrás de la seguridad de las contraseñas en Windows, desde los conceptos básicos hasta las técnicas más avanzadas. Descubre cómo funciona el proceso de hash de contraseñas, los diferentes algoritmos utilizados y las herramientas que los expertos en seguridad utilizan para poner a prueba la fortaleza de las mismas.

Obteniendo las claves de Windows en 3 pasos

¿Qué es un hash?

Las funciones hash son algoritmos que transforman una entrada de datos en una cadena de longitud fija, conocida como "hash". Diferentes algoritmos hash producen resultados distintos para la misma entrada. Por ejemplo:

- El hash MD5 de "Admin" es "21232f297a57a5a743894a0e4a801fc3".
- El hash SHA-256 de "Admin" es "8c6976e5b5410415bde908bd4dee15dfb167a9c873fc4bb8a81f6f2ab448a918".

Algunos algoritmos hash más antiguos, como MD5, han demostrado ser vulnerables y no se consideran seguros en la actualidad. Algoritmos más recientes, como SHA-3, ofrecen mayor seguridad.

En el contexto de los sistemas Windows, se han utilizado principalmente dos tipos de hash para almacenar las contraseñas de los usuarios: **LM** y **NTLM**.

LM (LAN Manager):

Este fue el primer algoritmo hash utilizado por los sistemas Windows, desde Windows 3.1 hasta Windows XP. A partir de Windows Vista, se dejó de usar LM debido a sus debilidades de seguridad, aunque aún puede ser activado por motivos de compatibilidad. Las características del algoritmo LM incluyen:

1. Rellenar la contraseña con ceros hasta alcanzar 14 caracteres.
2. Convertir todas las letras a mayúsculas.
3. Dividir la contraseña en dos partes de 7 caracteres cada una.
4. Aplicar el cifrado DES a cada una de las partes.

Debido a que el algoritmo divide la contraseña en dos partes, es más fácil para los atacantes descifrar contraseñas, ya que es más sencillo romper dos cadenas de 7 caracteres que una de 14. Además, si la contraseña original tiene menos de 8 caracteres, la segunda mitad será una cadena de ceros, lo que facilita aún más su vulnerabilidad.

Ejemplo:

Vamos a desglosar paso a paso cómo Windows aplicaba el algoritmo LM usando la contraseña **Admin1234** como ejemplo:

1. **Rellenar con ceros hasta 14 caracteres:** Si la contraseña tiene menos de 14 caracteres, el sistema la completa con ceros.
   - Contraseña original: **Admin1234**
   - Contraseña ajustada: **Admin123400000**
2. **Convertir todas las letras a mayúsculas:** LM no diferencia entre mayúsculas y minúsculas.
   - Resultado: **ADMIN123400000**
3. **Dividir la contraseña en dos partes de 7 caracteres cada una:** LM parte la contraseña ajustada en dos bloques de 7 caracteres.
   - Bloque 1: **ADMIN12**
   - Bloque 2: **3400000**

4. **Aplicar el cifrado DES a cada parte:** Cada bloque de 7 caracteres se cifra de forma independiente usando el algoritmo **DES (Data Encryption Standard)**.

Resultado final:

El algoritmo genera dos hashes independientes para los bloques **ADMIN12** y **3400000**, que luego se concatenan para formar el hash LM final.

Vulnerabilidad del algoritmo:

Este enfoque hace que las contraseñas sean vulnerables por varios motivos:

- **Fragmentación:** Si un atacante descifra uno de los bloques (por ejemplo, **3400000**, que es trivial), solo le queda descifrar el otro bloque.
- **Ceros predecibles:** Si la contraseña tiene menos de 8 caracteres, la segunda parte siempre será **0000000**, facilitando su descifrado.
- **Ignorar mayúsculas/minúsculas:** Esto reduce la complejidad del descifrado.

Esta metodología es una de las razones por las cuales LM fue descontinuado en las versiones modernas de Windows.

NTLM (NT LAN Manager):

NTLM es una versión mejorada de LM que ofrece mayores medidas de seguridad. Entre sus características destacan:

- Distingue entre mayúsculas y minúsculas.
- Es más simple y robusto que LM.
- Utiliza el algoritmo de cifrado MD4.

Aunque NTLM es más seguro que LM, se recomienda utilizar contraseñas de más de 8 caracteres que combinen mayúsculas, minúsculas, números y caracteres especiales para aumentar la seguridad.

Almacenamiento de hashes en Windows:

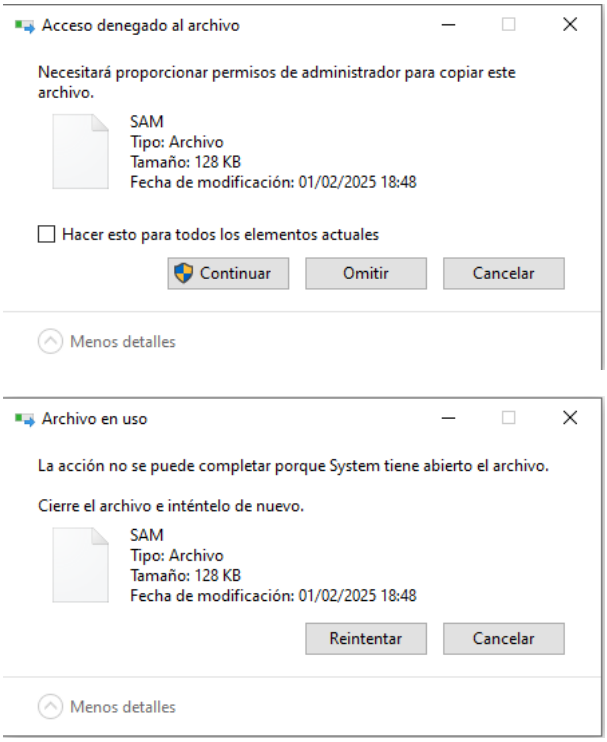
Windows almacena los hashes de las contraseñas en el archivo "**SAM**", ubicado en la ruta %systemdrive%\Windows\System32\config. Este archivo está cifrado, pero la clave para descifrarlo se encuentra en otro archivo llamado "**SYSTEM**" en la misma carpeta.

En sistemas Windows Server con Active Directory, los hashes se almacenan en el archivo "ntds.dit" en la ruta %systemdrive%\Windows\NTDS.

Es importante destacar que, aunque NTLM es más seguro que LM, ambos algoritmos tienen vulnerabilidades conocidas. Por ello, se recomienda utilizar versiones más recientes de Windows y considerar el uso de algoritmos de hash más seguros, como los de la familia SHA-2 o SHA-3, para proteger las contraseñas de los usuarios.

Paso 1 – Obtener los archivos SAM y SYSTEM

Para obtener los archivos **SAM** y **SYSTEM** de Windows, el sistema bloquea su acceso mientras está en funcionamiento, incluso con privilegios administrativos.



Sin embargo, hay varias técnicas conocidas que los atacantes o administradores de sistemas pueden utilizar para el acceso:

1. Utilizar una copia de sombra (Volume Shadow Copy)

Windows crea copias de seguridad de los archivos críticos del sistema mediante el servicio de **Volume Shadow Copy**. Se puede acceder a estos archivos aunque estén bloqueados.

Procedimiento:

- Abre una terminal de **PowerShell** con privilegios de administrador.
- Ejecuta el siguiente comando para listar las copias de sombra disponibles:

vssadmin list shadows

- Monta la copia de sombra en una ubicación accesible:

mklink /D C:\ShadowCopy \\?\GLOBALROOT\Device\HarddiskVolumeShadowCopy1

- Accede a la ubicación:

C:\ShadowCopy\Windows\System32\config

- Copia los archivos **SAM** y **SYSTEM**.

2. Herramientas especializadas de copia en caliente

Hay herramientas que permiten acceder a archivos bloqueados por el sistema.

Ejemplos:

- **RawCopy:** Permite copiar archivos bloqueados en Windows.

RawCopy.exe \\.\C: C:\Destination SYSTEM SAM

- **NTFSUtil:** Permite acceder y extraer archivos protegidos.

3. Explotación de vulnerabilidades

Si existen vulnerabilidades en el sistema operativo o configuraciones incorrectas, podrían permitir el acceso directo a los archivos protegidos.

Ejemplo:

- **Exploits de privilegios elevados:** Han existido fallos en versiones anteriores de Windows que permitían obtener los hashes de contraseñas

4. Acceso remoto (Active Directory)

En servidores Windows con Active Directory, los archivos de credenciales se almacenan en el archivo **ntds.dit**.

Procedimiento:

- Utilizar herramientas como **ntdsutil.exe** o herramientas de post-explotación (como **Mimikatz**) para extraer los hashes.

5. Uso de reg save

En Windows, es posible extraer una copia del archivo **SAM** y **SYSTEM** desde el Registro.

Procedimiento:

1. Abre CMD con privilegios de administrador.
2. Ejecuta:

reg save HKLM\SAM C:\sam_backup

reg save HKLM\SYSTEM C:\system_backup

6. Modo de arranque seguro o live USB

Inicia el sistema en un entorno externo para eludir las restricciones del sistema de archivos.

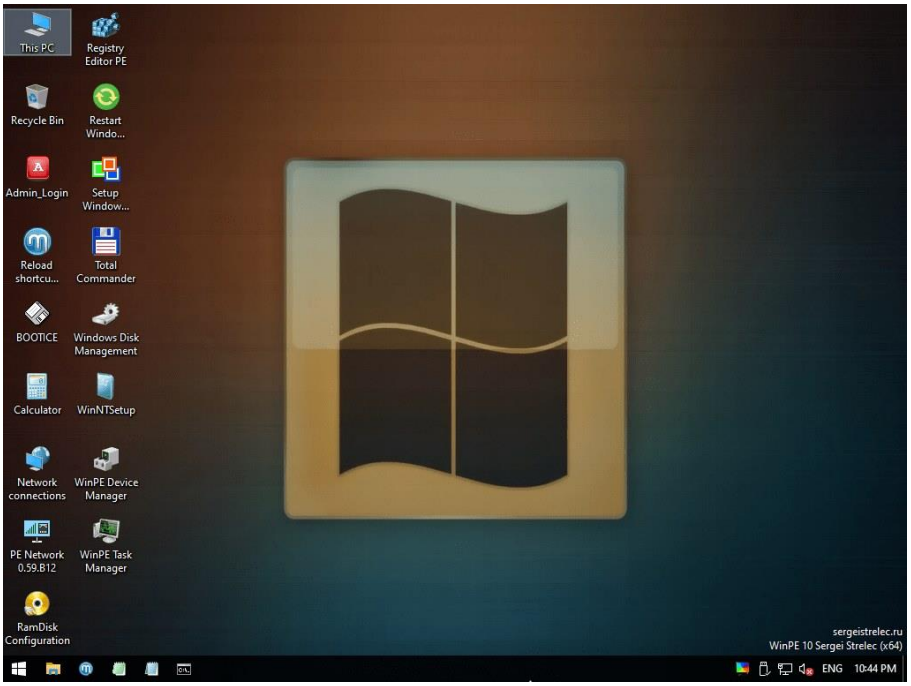
Procedimiento:

- Arranca desde un **USB booteable de Linux** (como Ubuntu) o una herramienta de rescate (Hirens Boot CD o *Sergei Strelec's WinPE*).
- Monta la partición donde está instalado Windows.

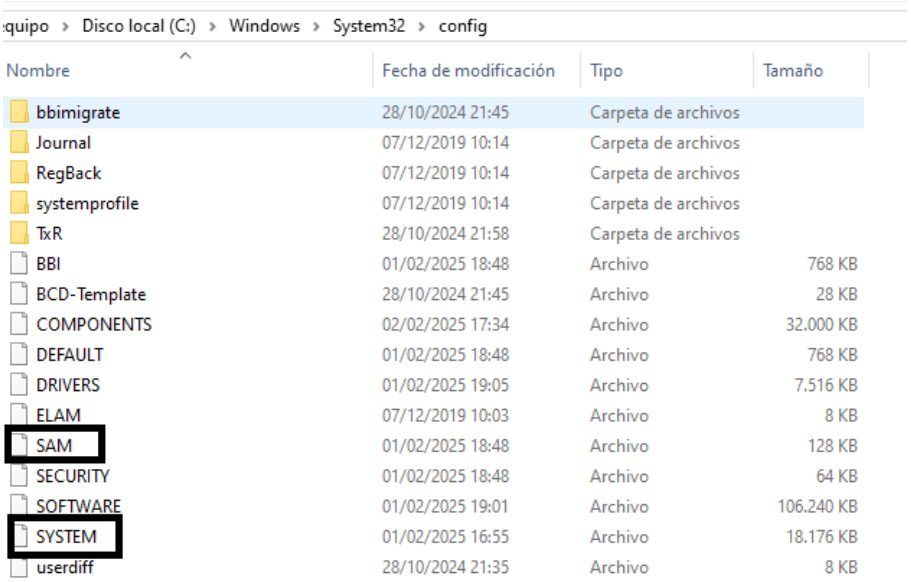
- Accede a la ruta **Windows\System32\config** y copia los archivos **SAM** y **SYSTEM**.

De todas estas opciones, la única que no implica partir de una sesión abierta es esta última y por tanto la que he elegido para este manual.

Hay muchas opciones para iniciar desde un USB booteable y acceder al contenido del disco. Yo he optado por Sergei Strelec's WinPE, simplemente porque es el primero que tenía a mano.



Solamente tienes que entrar a la ruta desde el Live USB y copiar ambos archivos:



Paso 2 – Hacer un dump de los hashes

Una vez que tienes los archivos **SAM** y **SYSTEM**, no puedes leer directamente las contraseñas de los usuarios en texto plano. Estos archivos contienen los **hashes de las contraseñas** (no las contraseñas mismas) en formatos cifrados y protegidos. Necesitamos **hacer un dump** para obtener los hashes:

Razones para hacer el dump

1. **Extraer los datos cifrados:** El archivo **SAM** por sí solo no permite acceder a la información relevante de las contraseñas, ya que los hashes están cifrados. El archivo **SYSTEM** contiene la clave de cifrado necesaria para descifrar el contenido del **SAM**.

2. **Combinar SAM y SYSTEM para descifrar los hashes:**

   - **SAM:** Contiene los hashes de las contraseñas.

   - **SYSTEM:** Contiene las claves de cifrado usadas por Windows (bootkey) para proteger el contenido del SAM.

   Un dump combina ambos archivos y utiliza la clave del SYSTEM para descifrar los hashes.

3. **Formato amigable para cracking:** Una vez descifrados los hashes, el dump los organiza en formatos compatibles con herramientas de cracking como **John the Ripper** o **Hashcat**.

Herramientas para hacer el dump

Algunas herramientas en linux para extraer los hashes podrían ser:

Samdump2

Utilizado en combinación con **samdump2** y **bkhive**.

bkhive SYSTEM bootkey

samdump2 SAM bootkey > hashes.txt

impacket-secretsdump

Con los archivos `SYSTEM` y `SAM`, yo he usado esta herramienta que suele ser más efectiva para extraer hashes:

impacket-secretsdump -sam SAM -system SYSTEM LOCAL > dump.txt

Contenido obtenido del dump

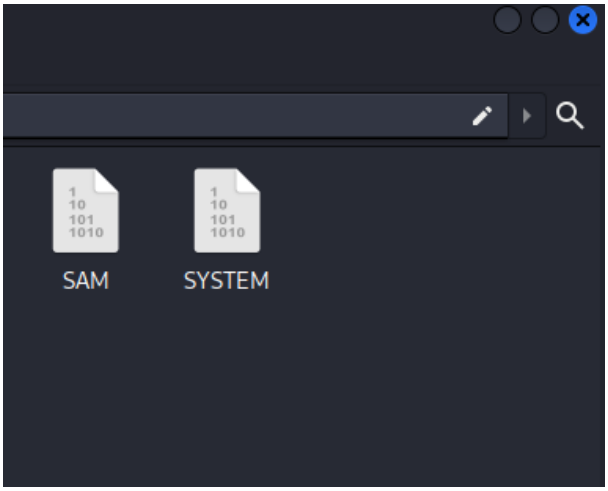
Después de hacer el dump, se obtienen los hashes de las contraseñas, que lucen algo así:

Administrator:500:0DCCCE48C9F00F11AAD3B435B51404EE:31D6CFE0D16AE931B73C59D7E0C089C0:::

Explicación:

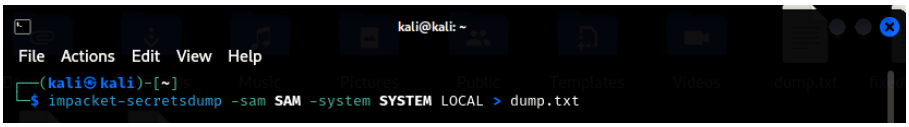
- **Nombre de usuario:** Administrator
- **Identificador RID:** 500
- **Hash LM:** 0DCCCE48C9F00F11AAD3B435B51404EE (si existe)
- **Hash NTLM:** 31D6CFE0D16AE931B73C59D7E0C089C0

Tenemos los dos archivos en nuestra máquina Kali:

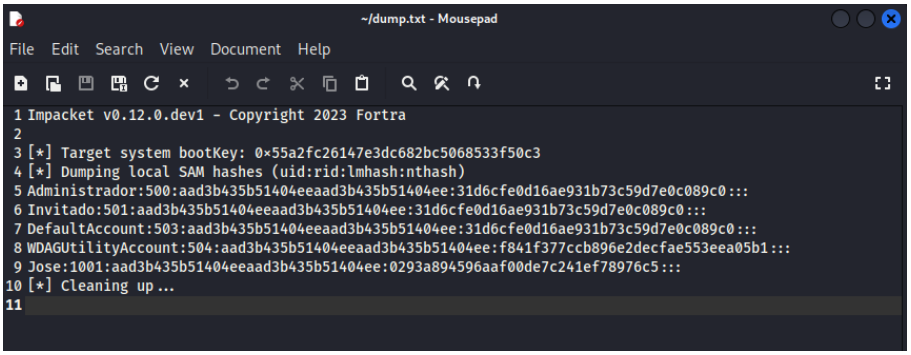


Y abrimos una terminal en el mismo directorio y ejecutamos el comando:

impacket-secretsdump -sam SAM -system SYSTEM LOCAL > dump.txt



El archivo dump.txt obtenido tiene el siguiente aspecto:



Voy a desglosar y analizar el resultado del dump línea por línea para explicarte los componentes y por qué algunas cadenas son iguales:

Formato de un dump de hashes SAM

Usuario:RID:LMHash:NTLMHash:::

Explicación:

- **Usuario:** Nombre de la cuenta en Windows.
- **RID (Relative Identifier):** Identificador único relativo de la cuenta.
- **LM Hash:** Hash del algoritmo LAN Manager (LM) para la contraseña.
- **NTLM Hash:** Hash del algoritmo NTLM para la contraseña.
- **Campos vacíos (:::)** Separadores sin contenido relevante.

Análisis de nuestro dump:

Administrador:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::

Invitado:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::

DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::

WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:f841f377ccb896e2decfae553eea05b1:::

Jose:1001:aad3b435b51404eeaad3b435b51404ee:0293a894596aaf00de7c241ef78976c5:::

1. Hash LM (LAN Manager)

El valor aad3b435b51404eeaad3b435b51404ee aparece repetidamente para el campo LM Hash.

¿Por qué se repite?

- Este valor específico aad3b435b51404eeaad3b435b51404ee es una cadena especial que indica que **no hay un hash LM** almacenado.
- Windows dejó de usar LM por sus debilidades de seguridad. Si el sistema no genera LM hashes, coloca esta cadena de "relleno" en su lugar.

2. Hash NTLM

El campo NTLM muestra diferentes valores, lo que indica el hash de la contraseña correspondiente para cada cuenta.

- 31d6cfe0d16ae931b73c59d7e0c089c0 es un hash especial que significa **contraseña en blanco**. Esto se ve en las cuentas:
    - Administrador
    - Invitado
    - DefaultAccount

- f841f377ccb896e2decfae553eea05b1 y 0293a894596aaf00de7c241ef78976c5 son hashes NTLM válidos, lo que indica que las cuentas **WDAGUtilityAccount** y **Jose** tienen contraseñas configuradas.

Resumen del dump

Usuario	RID	LM Hash	NTLM Hash	Contraseña
Administrador	500	aad3b435b51404ee...	31d6cfe0d16ae931...	Vacía
Invitado	501	aad3b435b51404ee...	31d6cfe0d16ae931...	Vacía
DefaultAccount	503	aad3b435b51404ee...	31d6cfe0d16ae931...	Vacía
WDAGUtilityAccount	504	aad3b435b51404ee...	f841f377ccb896e2...	Configurada
Jose	1001	aad3b435b51404ee...	0293a894596aaf00...	Configurada

Conclusión

- La cadena aad3b435b51404eeaad3b435b51404ee en el campo LM indica que no existe un hash LM, probablemente porque el sistema está configurado para deshabilitar este hash obsoleto.
- La cadena 31d6cfe0d16ae931b73c59d7e0c089c0 en el campo NTLM indica que la cuenta no tiene contraseña.
- Los hashes NTLM válidos como f841f377ccb896e2... y 0293a894596aaf00... muestran cuentas con contraseñas configuradas. Estos valores pueden ser usados para ataques de fuerza bruta o "Pass-the-Hash."

Paso 3 – Descifrar la contraseña que hay detrás de los hashes

Para descifrar las contraseñas a partir de los hashes extraídos en Linux, existen varias técnicas y herramientas de ataque.

1. Ataque de fuerza bruta

Consiste en probar todas las combinaciones posibles de caracteres hasta encontrar la contraseña.

Herramientas populares:

- **Hashcat:** Ejemplo de comando:

hashcat -m 1000 hashes.txt wordlist.txt

El parámetro -m 1000 indica el tipo de hash **NTLM**.

- **John the Ripper:** Comando básico:

john --format=NT hashes.txt --wordlist=wordlist.txt

Ventajas: Encuentra cualquier contraseña si se tiene suficiente tiempo.

Desventajas: Lento para contraseñas largas y complejas.

2. Ataque de diccionario

Se utilizan listas predefinidas de contraseñas comunes (diccionarios) para comparar con los hashes.

Ejemplo con Hashcat:

hashcat -m 1000 -a 0 hashes.txt /usr/share/wordlists/rockyou.txt

- -a 0: Modo de ataque por diccionario.
- /usr/share/wordlists/rockyou.txt: Archivo de diccionario popular en Linux.

Ejemplo con John the Ripper:

john --format=nt --wordlist=/usr/share/wordlists/rockyou.txt hashes.txt

Ventajas: Más rápido que el ataque de fuerza bruta.

Desventajas: Depende de la calidad del diccionario.

3. Ataque híbrido (diccionario + mutaciones)

Combina un diccionario con variaciones (por ejemplo, añadiendo números o símbolos al final).

Ejemplo con Hashcat:

hashcat -m 1000 -a 6 hashes.txt /usr/share/wordlists/rockyou.txt ?d?d?d

- -a 6: Diccionario + sufijo numérico de 3 dígitos (?d?d?d).

Ejemplo con John the Ripper:

john --format=nt --wordlist=wordlist.txt --rules hashes.txt

Ventajas: Aumenta la probabilidad de éxito con contraseñas ligeramente modificadas.

Desventajas: Mayor tiempo de procesamiento.

4. Ataque con "Rainbow Tables"

Se utilizan tablas precomputadas de hashes para buscar coincidencias rápidamente.

Herramientas populares:

- **rtgen:** Para generar las tablas.
- **rcrack:** Para hacer el crack de hashes.

Ejemplo de uso:

rcrack rainbow_table_directory -f hashes.txt

Ventajas: Muy rápido si se tienen las tablas adecuadas.

Desventajas: Las tablas ocupan mucho espacio en disco y deben estar precomputadas.

5. Pass-the-Hash (sin descifrar la contraseña)

En lugar de descifrar la contraseña, se usa el hash directamente para autenticarse en el sistema.

Herramientas populares:

- **Impacket:** Ejecutar un ataque Pass-the-Hash:

psexec.py administrator@target_ip -hashes aad3b435b51404eeaad3b435b51404ee:0293a894596aaf00de7c241ef78976c5

- **Evil-WinRM:** Herramienta para acceso remoto en Windows usando hashes.

Ventajas: No requiere descifrar el hash.

Desventajas: Requiere acceso a la red y permisos.

6. Ataques personalizados (Reglas avanzadas en Hashcat y John)

Puedes definir reglas complejas en **John the Ripper** o **Hashcat** para probar combinaciones específicas.

Ejemplo en Hashcat:

hashcat -m 1000 -a 0 hashes.txt --rules=rules/best64.rule wordlist.txt

Consejos para descifrar hashes NTLM

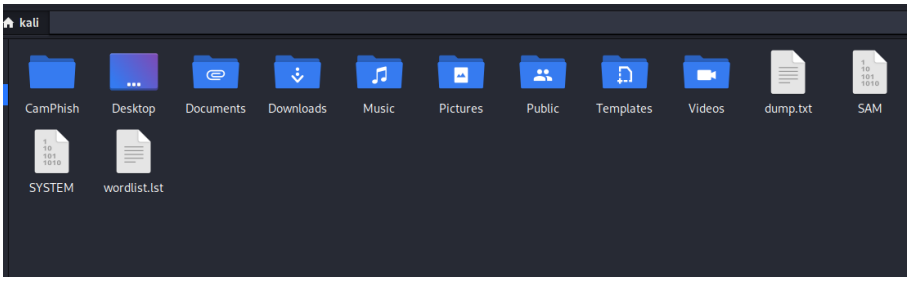
- **Identificar el tipo de hash:** Siempre verifica si el hash es LM o NTLM.

file hashes.txt

- **Revisar diccionarios populares:** Usa **rockyou.txt** o listas personalizadas.
- **Optimizar configuraciones:** Habilita el uso de GPU para acelerar los ataques con Hashcat.

De todas estas opciones voy a utilizar John the Ripper y un diccionario (wordlist.lst) para descifrar la contraseña.

Abrimos terminal en el mismo directorio donde tenemos el archivo dump.txt y el diccionario wordlist.lst:

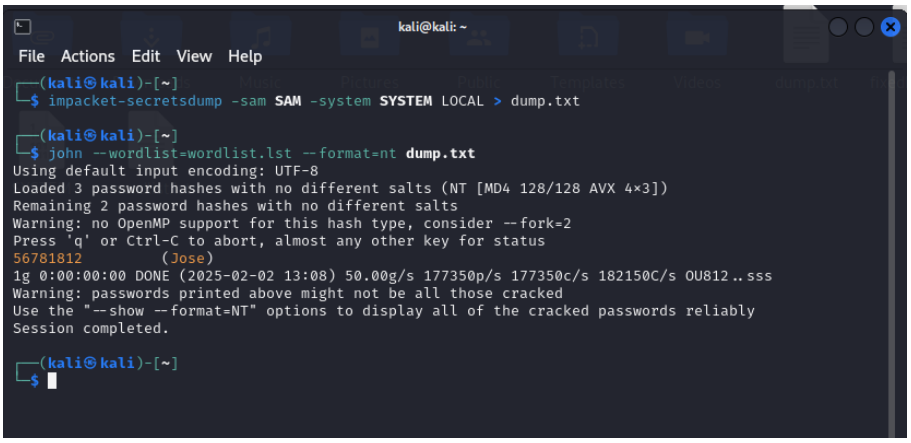


Y ejecutamos el comando:

john --wordlist=wordlist.lst --format=nt dump.txt

El formato que nosotros tenemos corresponde al dump típico de NTLM extraído de SAM en Windows. John entiende este formato de manera nativa, pero es mejor asegurarse indicando format=nt

Tienes una lista de formatos admitidos aquí.



Una vez terminado el proceso nos mostrará en rojo la contraseña para la cuenta de usuario. En mi caso creé una única cuenta en este equipo para el ejercicio y que era: **Jose** - **56781812**

El proceso de descifrar la contraseña por ataque de diccionario implica tener un buen diccionario (aquí tienes varios) y tener tiempo. Aunque estos diccionarios están formados por millones de las contraseñas más utilizadas, es posible que la que buscas no se encuentre en el diccionario; en ese caso habrás malgastado mucho tiempo.

Conclusión

Una contraseña robusta es aquella que es difícil de adivinar o descifrar por personas no autorizadas, protegiendo así tus cuentas y datos personales de posibles ataques. Para que una contraseña sea considerada robusta, debe cumplir con ciertos criterios y características. A continuación, te detallo los elementos clave que debes tener en cuenta al crear tus contraseñas:

Longitud adecuada

La longitud de una contraseña es fundamental. Se recomienda utilizar contraseñas de al menos 12 caracteres, aunque lo ideal sería que tengan 16 o más. Cuanto más larga sea la contraseña, más combinaciones posibles habrá, lo que dificultará su descifrado.

Combinación de caracteres

Una contraseña robusta debe combinar diferentes tipos de caracteres:

- **Mayúsculas y minúsculas**: Utiliza letras mayúsculas y minúsculas de forma aleatoria.
- **Números**: Incluye números en diferentes posiciones de la contraseña.
- **Símbolos**: Incorpora símbolos como !, @, #, $, %, &, entre otros.

La combinación de estos caracteres hará que tu contraseña sea más compleja y difícil de adivinar.

Evitar información personal

No incluyas información personal en tu contraseña, como nombres, fechas de nacimiento, números de teléfono o direcciones. Esta información es fácil de obtener y puede ser utilizada para adivinar tu contraseña.

Sin palabras comunes

Evita utilizar palabras que se encuentren en el diccionario o que sean de uso común. Estos programas prueban combinaciones de palabras comunes para descifrar contraseñas, por lo que es importante evitar este tipo de palabras.

Aleatoriedad

Una contraseña robusta debe ser aleatoria, es decir, no debe seguir patrones lógicos o secuencias predecibles. No utilices patrones de teclado (como qwerty o asdf) ni secuencias numéricas (como 123456).

No reutilizar contraseñas

Es fundamental utilizar contraseñas diferentes para cada una de tus cuentas. Si utilizas la misma contraseña para varias cuentas y una de ellas es comprometida, todas tus cuentas estarán en riesgo.

Actualización periódica

Se recomienda cambiar las contraseñas de forma periódica, por ejemplo, cada 3 o 6 meses. Esto reduce el riesgo de que una contraseña antigua sea utilizada para acceder a tus cuentas.

Siguiendo estos consejos, podrás crear contraseñas robustas que protejan tus cuentas y datos personales de posibles ataques. Recuerda que la seguridad de tus contraseñas es fundamental para proteger tu privacidad en el mundo digital.

Es importante tener en cuenta que estas herramientas que hemos mostrado pueden ser utilizadas tanto por usuarios legítimos como por personas malintencionadas. Por lo tanto, es fundamental utilizar estas herramientas de manera ética y responsable, y tomar medidas de seguridad adecuadas para protegernos de posibles ataques.