Audit des Délégations Active Directory avec BloodHound et Adalanche 2026

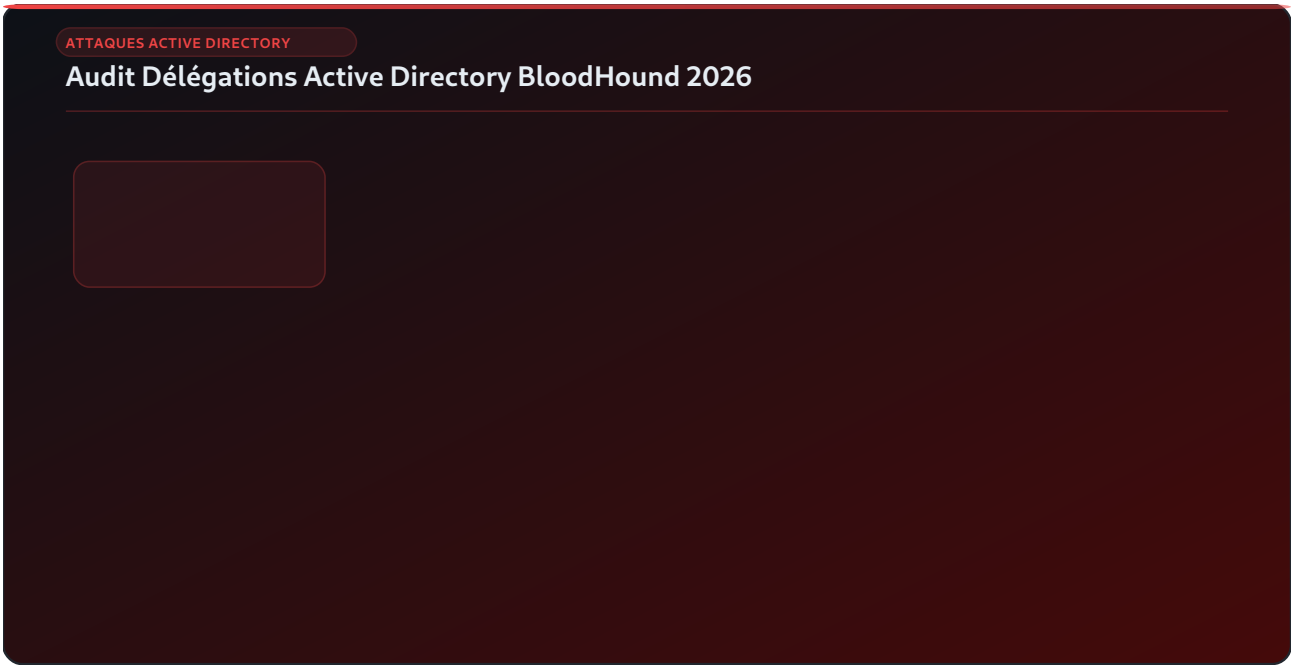
> Auditez les délégations Active Directory avec BloodHound CE et Adalanche : 16 requêtes Cypher, scripts PowerShell, remédiation et conformité NIS 2.

📋 EN BREF

- Les délégations Active Directory (Unconstrained, Constrained, RBCD) représentent l'un des vecteurs d'escalade de privilèges les plus exploités par les attaquants dans les environnements Windows.
- BloodHound Community Edition avec SharpHound permet de cartographier automatiquement tous les chemins d'attaque liés aux délégations via des requêtes Cypher sur Neo4j.
- Adalanche offre une alternative open source performante pour auditer les ACL et délégations AD sans dépendance à Neo4j.
- Des scripts PowerShell natifs permettent d'inventorier toutes les délégations (Unconstrained, Constrained, RBCD, ACL) et de générer un rapport CSV/HTML exploitable par le RSSI.
- La remédiation passe par le modèle de Tiering AD, les comptes Protected Users, et l'audit SACL sur les objets Tier 0 — obligatoire dans le cadre de la conformité NIS 2 en France.

L'audit des délégations Active Directory est devenu une priorité absolue pour les équipes sécurité des entreprises françaises, à mesure que les groupes de ransomware et les red teams exploitent systématiquement les mauvaises configurations de délégation pour réaliser des escalades de privilèges vers Domain Admin en quelques minutes. Paris, Lyon, Marseille : aucune organisation

disposant d'un Active Directory on-premises ou hybride n'est à l'abri d'une compromission par délégation mal configurée. En France, la transposition de la directive NIS 2 impose désormais aux opérateurs d'importance vitale et aux entités essentielles un audit annuel des droits d'accès privilégiés, ce qui inclut explicitement les délégations Kerberos et les ACL objets AD. Le RSSI doit pouvoir répondre à une question simple mais redoutable : qui peut se faire passer pour n'importe quel utilisateur dans mon domaine ? Cet article présente une méthodologie complète d'audit des délégations AD avec BloodHound Community Edition, Adalanche, et PowerShell natif — couvrant les délégations Unconstrained, Constrained (KCD), Resource-Based Constrained Delegation (RBCD) et les délégations ACL (GenericAll, WriteDACL, AddMember, ForceChangePassword) — avec les requêtes Cypher, les Event IDs de détection, et les scripts de remédiation prêts à l'emploi pour sécuriser durablement votre environnement Active Directory en 2026.



Les délégations Active Directory permettent à un service ou un compte d'agir au nom d'un utilisateur pour accéder à d'autres ressources du réseau. Ce mécanisme, indispensable pour les architectures multi-tiers (IIS vers SQL Server, par exemple), est aussi l'un des plus dangereux

lorsqu'il est mal configuré. Il existe trois grands types de délégation Kerberos, auxquels s'ajoutent les délégations ACL sur les objets AD.

Délégation Unconstrained (Non Contrainte)

La délégation Unconstrained est la forme la plus ancienne et la plus dangereuse. Un ordinateur ou service configuré avec `TrustedForDelegation=True` reçoit le TGT (Ticket Granting Ticket) complet de chaque utilisateur qui s'y authentifie. Un attaquant disposant d'un accès administrateur local sur cet ordinateur peut extraire tous les TGTs en mémoire via `Rubeus dump` ou `Mimikatz sekurlsa::tickets`, puis se faire passer pour n'importe lequel de ces utilisateurs — y compris les Domain Admins. Le vrai danger réside dans la combinaison avec les techniques de coercion d'authentification : Printer Bug (SpoolSample), PetitPotam, DFSCoerce forcent le contrôleur de domaine lui-même à s'authentifier vers la machine avec délégation Unconstrained, livrant un TGT de compte machine DC — soit une voie royale vers DCSync et la compromission totale du domaine.

Délégation Constrained (Kerberos Constrained Delegation — KCD)

La délégation Constrained, introduite avec Windows Server 2003, limite la délégation à une liste de services cibles définie dans l'attribut `msDS-AllowedToDelegateTo`. Le compte ne peut se faire passer pour un utilisateur que pour accéder aux services listés. Bien que plus sûre qu'Unconstrained, une KCD sur un compte avec des privilèges élevés (compte de service applicatif ayant accès à un DC) reste exploitable via des attaques de protocol transition (`S4U2Self` + `S4U2Proxy`).

Resource-Based Constrained Delegation (RBCD)

La RBCD, disponible depuis Windows Server 2012, inverse le contrôle : c'est la ressource cible qui définit qui peut déléguer vers elle, via l'attribut `msDS-AllowedToActOnBehalfOfOtherIdentity`. Si un attaquant dispose d'un droit GenericWrite ou WriteProperty sur un objet ordinateur, il peut écrire cet attribut pour permettre à n'importe quel compte qu'il contrôle de s'authentifier en tant

que Domain Admin auprès de cet ordinateur — avec seulement un compte ordinateur ou de service standard comme point de départ.

Délégations ACL Dangereuses sur les Objets AD

Au-delà des délégations Kerberos, les ACL (Access Control Lists) sur les objets Active Directory constituent un vecteur d'escalade discret mais redoutable :

Droit ACL	Cible dangereuse	Impact	Exploitation
`GenericAll`	Tout objet AD	Contrôle total	Reset MDP, modifier groupe, DCSync
`WriteDACL`	Domaine, OU, groupe	Modifier les ACL	S'accorder GenericAll, activer DCSync
`GenericWrite`	Utilisateur, ordinateur	Écrire attributs	SPN + Kerberoasting, RBCD, Shadow Creds
`AddMember`	Groupes privilégiés	Ajout membre	S'ajouter à Domain Admins
`ForceChangePassword`	Comptes admin	Reset MDP sans connaître l'actuel	Prendre le contrôle du compte cible
`AllExtendedRights`	Objet domaine, utilisateur	Droits étendus complets	DCSync (DS-Replication-Get-Changes)
`WriteProperty (msDS-KeyCredentialLink)`	Utilisateur, ordinateur	Shadow Credentials	Authentification sans MDP via certificat

Plusieurs délégations dangereuses existent par défaut dans Active Directory : les **Exchange Trusted Subsystem** disposent historiquement de droits WriteDACL sur le domaine (CVE exploitée dans PrivExchange), et de nombreuses GPO de déploiement logiciel créent des comptes de service avec des droits excessifs sur des OU entières. L'inventaire de l'existant est donc la première étape indispensable avant toute remédiation.

BloodHound CE : Collecte et Analyse des Chemins d'Attaque

BloodHound Community Edition (CE) est l'outil de référence pour la cartographie des chemins d'attaque dans Active Directory. Il repose sur Neo4j, une base de données en graphe, pour stocker les relations entre objets AD et permettre des requêtes de parcours de graphe très performantes. Depuis la version CE (successeur de BloodHound Legacy), l'interface est entièrement repensée avec un backend API REST.

> ⚡ **Retour terrain**
>
> J'utilise BloodHound systématiquement en début de mission AD. Pour un groupe de distribution retail, la cartographie a révélé un chemin de 4 sauts depuis un compte utilisateur standard vers Domain Admin — via un GPO modifiable par le compte, appliqué à une OU contenant un admin, avec une ACL WriteOwner sur un groupe privilégié. Aucun des quatre liens n'était visible sans la vue graphique. La correction a pris 2 heures une fois le chemin identifié.

BloodHound CE vs BloodHound Enterprise

BloodHound CE est gratuit, open source (Apache 2.0), auto-hébergé. BloodHound Enterprise (SpecterOps) ajoute une gestion multi-domaines, des analyses continues, et une intégration SIEM — pertinent pour les grandes entreprises françaises avec des forêts AD complexes. Pour l'audit ponctuel ou les équipes internes, CE suffit amplement.

Déploiement BloodHound CE avec Docker

```
 # Déploiement BloodHound CE
curl -L https://ghcr.io/bloodhoundad/bloodhound-ce/bloodhound-ce-docker-compose.yml -o docker-com
docker compose up -d
# Interface disponible sur http://localhost:8080
# Credentials initiaux: admin / (générés au premier démarrage — voir logs)
docker compose logs bloodhound | grep "Initial Password"
```

Collecte avec SharpHound

SharpHound est le collecteur officiel pour BloodHound. Il énumère tous les objets AD, les ACL, les sessions actives, et les relations de délégation.

```
 # Collecte complète — exécuter depuis un poste joint au domaine avec un compte standard
./SharpHound.exe -c All --zipfilename bloodhound.zip --outputdirectory C:\Temp\

# Collecte ciblée ACL uniquement (plus rapide, moins de bruit réseau)
./SharpHound.exe -c ACL --zipfilename bloodhound_acl.zip

# Collecte avec stealth (évite les scans de session)
./SharpHound.exe -c DCOnly --zipfilename bloodhound_dc.zip

# Depuis PowerShell (version PS)
Import-Module .\SharpHound.ps1
Invoke-BloodHound -CollectionMethod All -ZipFileName bloodhound.zip
```

Le fichier ZIP généré contient plusieurs fichiers JSON (computers.json, users.json, groups.json, domains.json, gpos.json, ous.json, containers.json). Importez-le dans BloodHound CE via l'interface web : Administration → File Ingest → Upload Files.

Interface BloodHound CE : Nœuds, Arêtes et Chemins d'Attaque

Dans le graphe BloodHound, chaque objet AD (utilisateur, ordinateur, groupe, GPO, OU, domaine) est un **nœud**. Les relations entre objets (MemberOf, AdminTo, HasSession, Owns, WriteDACL, GenericAll, AllowedToDelegate, AllowedToAct, etc.) sont des **arêtes**. La recherche de *shortest paths* (plus courts chemins) permet d'identifier en quelques secondes le chemin

d'attaque minimal entre un compte standard et Domain Admins. Pour les délégations, les arêtes clés sont : `Unconstrained`, `AllowedToDelegate`, `AllowedToAct`, `GenericAll`, `WriteDACL`, `GenericWrite`, `AddMember`, `ForceChangePassword`, `AllExtendedRights`, `WriteAccountRestrictions`.

Requêtes BloodHound Cypher pour les Délégations

BloodHound CE expose une interface de requêtes Cypher personnalisées (Explore → Custom Queries). Voici plus de 15 requêtes essentielles pour auditer les délégations et les chemins d'attaque associés.

1. Ordinateurs avec Délégation Unconstrained (hors DC)

```
MATCH (c:Computer {unconstraineddelegation: true})
WHERE NOT c.objectid ENDS WITH '-516' AND NOT c.objectid ENDS WITH '-502'
RETURN c.name, c.operatingsystem, c.enabled
ORDER BY c.name
```

2. Comptes Utilisateurs avec Délégation Unconstrained

```
MATCH (u:User {unconstraineddelegation: true})
WHERE u.enabled = true
RETURN u.name, u.displayname, u.lastlogon
ORDER BY u.lastlogon DESC
```

3. Comptes avec Délégation Constrained (KCD)

```
MATCH (n)
WHERE n.allowedtodelegate IS NOT NULL AND n.allowedtodelegate <> []
RETURN labels(n)[0] AS type, n.name, n.allowedtodelegate
ORDER BY type, n.name
```

4. RBCD : Arêtes WriteAccountRestrictions

```
MATCH p=(n)-[:WriteAccountRestrictions]->(c:Computer)
RETURN n.name AS source, labels(n)[0] AS source_type, c.name AS target_computer
ORDER BY c.name
```

5. RBCD : Chemins d'attaque complets via WriteAccountRestrictions

```
MATCH p=shortestPath((n)-[r:WriteAccountRestrictions|MemberOf*1..]->(c:Computer))
WHERE NOT n:Computer AND n.enabled = true
RETURN p LIMIT 25
```

6. GenericAll vers Domain Admins — Tous les Chemins

```
MATCH p=shortestPath((n)-[r:GenericAll|MemberOf*1..]->(g:Group))
WHERE g.objectid ENDS WITH '-512' AND n <> g AND n.enabled = true
RETURN p LIMIT 50
```

7. WriteDACL sur le Domaine ou les Groupes Privilégiés

```
MATCH p=(n)-[:WriteDACL]->(t)
WHERE t:Domain OR (t:Group AND (t.objectid ENDS WITH '-512' OR t.objectid ENDS WITH '-519' OR t.o
RETURN n.name AS source, labels(n)[0] AS source_type, t.name AS target, labels(t)[0] AS target_ty
ORDER BY target_type, target
```

8. AddMember sur les Groupes Privilégiés

```
 MATCH p=(n)-[:AddMember]->(g:Group)
WHERE g.objectid ENDS WITH '-512'
   OR g.objectid ENDS WITH '-518'
   OR g.objectid ENDS WITH '-519'
   OR g.name =~ '(?i).*domain admins.*'
   OR g.name =~ '(?i).*enterprise admins.*'
   OR g.name =~ '(?i).*schema admins.*'
RETURN n.name AS who_can_add, g.name AS privileged_group
ORDER BY g.name
```

9. ForceChangePassword sur des Comptes Admin

```
 MATCH p=(n)-[:ForceChangePassword]->(u:User)
WHERE u.admincount = true AND u.enabled = true
RETURN n.name AS attacker, u.name AS target_admin
ORDER BY target_admin
```

10. AllExtendedRights (DCSync capable) sur le Domaine

```
 MATCH p=(n)-[:AllExtendedRights]->(d:Domain)
WHERE NOT n.objectid ENDS WITH '-512'
  AND NOT n.objectid ENDS WITH '-516'
  AND NOT n.objectid ENDS WITH '-519'
RETURN n.name AS principal, labels(n)[0] AS type, d.name AS domain
ORDER BY type
```

11. Shadow Credentials : WriteProperty sur msDS-KeyCredentialLink

```
MATCH p=(n)-[:AddKeyCredentialLink]->(t)
WHERE t.enabled = true
RETURN n.name AS attacker, labels(n)[0] AS attacker_type,
       t.name AS target, labels(t)[0] AS target_type
ORDER BY target_type
```

12. Chemins depuis Domain Users vers Domain Admins

```
MATCH p=shortestPath(
  (g:Group {objectid: "S-1-5-21-XXXXXXXXX-XXXXXXXXX-XXXXXXXXX-513"})-[r*1..10]->(da:Group)
)
WHERE da.objectid ENDS WITH '-512'
RETURN p
// Remplacer le SID par le SID réel du groupe Domain Users de votre domaine
```

13. Plus Courts Chemins depuis un Compte Standard Spécifique

```
MATCH p=shortestPath(
  (u:User {name: "JDUPONT@ENTREPRISE.LOCAL"})-[r*1..]->(g:Group)
)
WHERE g.objectid ENDS WITH '-512'
RETURN p
```

14. Tous les Principaux avec AdminTo + Délégation Unconstrained

```
MATCH (n)-[:AdminTo]->(c:Computer {unconstraineddelegation: true})
WHERE c.enabled = true
RETURN n.name AS who_is_admin, labels(n)[0] AS type, c.name AS unconstrained_computer
ORDER BY unconstrained_computer
```

15. Objets Owned avec Chemin vers Tier 0

```
MATCH p=shortestPath((n {owned: true})-[r*1..]->(t))
WHERE t:Domain OR (t:Group AND t.objectid ENDS WITH '-512')
RETURN p LIMIT 25
```

16. Comptes avec HasSIDHistory vers Groupes Privilégiés

```
MATCH p=(u:User)-[:HasSIDHistory]->(g:Group)
WHERE g.objectid ENDS WITH '-512'
   OR g.objectid ENDS WITH '-519'
RETURN u.name AS user_with_sidhistory, g.name AS privileged_group_in_history
```

Ces requêtes peuvent être sauvegardées dans BloodHound CE comme *Custom Queries* en les ajoutant dans le fichier de configuration, ou directement via l'interface Explore. Pour une utilisation en audit reproductible, il est recommandé de scripter l'export via l'API REST de BloodHound CE (`POST /api/v2/graphs/cypher`) et d'automatiser la génération d'un rapport.

Adalanche : Alternative Open Source à BloodHound

Adalanche est un outil open source développé en Go par Lasse Mikkel Reinhold, conçu spécifiquement pour l'audit des délégations et ACL Active Directory. Contrairement à BloodHound qui nécessite Neo4j et un collecteur séparé (SharpHound), Adalanche est un binaire unique qui collecte, stocke et visualise — particulièrement adapté aux environnements où l'installation de Neo4j est contrainte.

Adalanche vs BloodHound : Différences Clés

Critère	BloodHound CE	Adalanche
Backend	Neo4j (graph DB)	Fichiers JSON + moteur interne Go
Déploiement	Docker (multi-container)	Binaire unique
Collecte	SharpHound (Windows)	Adalanche (Linux/Windows, via LDAP)
Requêtes	Cypher (puissant, complexe)	Interface web simplifiée + requêtes prédéfinies
Focus principal	Chemins d'attaque généraux	ACL et délégations AD spécifiquement
Collecte Linux	Non (SharpHound = .NET)	Oui (via LDAP depuis Linux)
Licence	Apache 2.0	GPLv3

Installation et Collecte avec Adalanche

```
 # Téléchargement (Linux x64)
wget https://github.com/lkarlslund/Adalanche/releases/latest/download/adalanche-linux-x64
chmod +x adalanche-linux-x64

# Collecte depuis Linux via LDAP (compte de service AD suffisant)
./adalanche-linux-x64 collect activedirectory \
  --server dc01.entreprise.local \
  --domain entreprise.local \
  --username auditeur@entreprise.local \
  --password "MotDePasseAuditeur" \
  --outputpath ./adalanche-data/

# Lancement de l'interface web d'analyse
./adalanche-linux-x64 analyze \
  --datapath ./adalanche-data/ \
  --bind 127.0.0.1:8000

# Accès: http://127.0.0.1:8000
```

Requêtes Adalanche pour les Délégations ACL

L'interface web Adalanche propose des analyses prédéfinies accessibles depuis le menu "Analyze" :

Unconstrained delegation : liste immédiate des objets avec `TrustedForDelegation`

Who can DCSync : tous les principaux avec DS-Replication-Get-Changes + DS-Replication-Get-Changes-All

Paths to Domain Admin : visualisation interactive des chemins d'attaque

ACL anomalies : objets avec des ACL héritées anormales ou des entrées orphelines

Adalanche excelle particulièrement dans la détection des **ACL héritées anormales** — cas où des droits excessifs ont été accordés sur une OU parente et se propagent à des milliers d'objets enfants sans que l'administrateur en ait conscience. Cette visibilité sur la propagation des ACL est l'avantage différenciateur principal d'Adalanche par rapport à BloodHound pour les audits ACL purs.

Audit PowerShell des Délégations — Sans Outil Graphique

Pour les équipes sans accès à BloodHound ou dans des environnements restreints, PowerShell avec le module Active Directory permet d'inventorier toutes les délégations nativement. Voici un script complet d'inventaire qui produit un rapport CSV et HTML.

Commandes PowerShell Individuelles

```
 # Ordinateurs avec délégation Unconstrained
Get-ADComputer -Filter {TrustedForDelegation -eq $true} `
  -Properties TrustedForDelegation, OperatingSystem, LastLogonDate |
  Select-Object Name, OperatingSystem, LastLogonDate |
  Sort-Object LastLogonDate -Descending

# Utilisateurs avec délégation Unconstrained
Get-ADUser -Filter {TrustedForDelegation -eq $true} `
  -Properties TrustedForDelegation, LastLogonDate, Enabled |
  Where-Object {$_.Enabled -eq $true} |
  Select-Object SamAccountName, LastLogonDate, Enabled

# Comptes de service administrés (gMSA/MSA) avec délégation Unconstrained
Get-ADServiceAccount -Filter {TrustedForDelegation -eq $true} `
  -Properties TrustedForDelegation

# Délégation Constrained (KCD) — attribut msDS-AllowedToDelegateTo
Get-ADObject -Filter {msDS-AllowedToDelegateTo -like "*"} `
  -Properties msDS-AllowedToDelegateTo, ObjectClass, SamAccountName |
  Select-Object Name, ObjectClass, SamAccountName, msDS-AllowedToDelegateTo

# RBCD — attribut msDS-AllowedToActOnBehalfOfOtherIdentity
Get-ADComputer -Filter * -Properties msDS-AllowedToActOnBehalfOfOtherIdentity |
  Where-Object {$_."msDS-AllowedToActOnBehalfOfOtherIdentity" -ne $null} |
  Select-Object Name, @{N="RBCD_Trustees";E={
    (New-Object Security.AccessControl.RawSecurityDescriptor(
      $_."msDS-AllowedToActOnBehalfOfOtherIdentity", 0
    )).DiscretionaryAcl | ForEach-Object {
      (New-Object System.Security.Principal.SecurityIdentifier($_.SecurityIdentifier)).Translate(
        [System.Security.Principal.NTAccount]) 2>/dev/null
    }
  }}
```

Script Complet d'Inventaire des Délégations (CSV + HTML)

```
<#
.SYNOPSIS
    Inventaire complet des délégations Active Directory
    Produit un rapport CSV et HTML
.NOTES
    Requiert: Module ActiveDirectory, droits lecture AD
#>
param(
    [string]$OutputPath = "C:\Audit\Delegations_$(Get-Date -Format 'yyyyMMdd')"
)

Import-Module ActiveDirectory -ErrorAction Stop
$null = New-Item -ItemType Directory -Path $OutputPath -Force
$results = [System.Collections.Generic.List[PSObject]]::new()

Write-Host "[*] Collecte des délégations Unconstrained (ordinateurs)..." -ForegroundColor Cyan
Get-ADComputer -Filter {TrustedForDelegation -eq $true} `
  -Properties TrustedForDelegation, OperatingSystem, LastLogonDate, Enabled, Description |
  ForEach-Object {
    $results.Add([PSCustomObject]@{
        Type             = "Unconstrained"
        ObjectClass      = "Computer"
        Name             = $_.Name
        Enabled          = $_.Enabled
        LastLogon        = $_.LastLogonDate
        DelegationType   = "TrustedForDelegation"
        DelegationTarget = "ALL SERVICES (UNCONSTRAINED)"
        OS               = $_.OperatingSystem
        Description      = $_.Description
        Risk             = "CRITIQUE"
    })
}

Write-Host "[*] Collecte des délégations Unconstrained (utilisateurs)..." -ForegroundColor Cyan
Get-ADUser -Filter {TrustedForDelegation -eq $true} `
  -Properties TrustedForDelegation, LastLogonDate, Enabled, Description |
  ForEach-Object {
    $results.Add([PSCustomObject]@{
        Type             = "Unconstrained"
        ObjectClass      = "User"
        Name             = $_.SamAccountName
        Enabled          = $_.Enabled
        LastLogon        = $_.LastLogonDate
```

```
        DelegationType   = "TrustedForDelegation"
        DelegationTarget = "ALL SERVICES (UNCONSTRAINED)"
        OS               = ""
        Description      = $_.Description
        Risk             = "CRITIQUE"
    })
}

Write-Host "[*] Collecte des délégations Constrained (KCD)..." -ForegroundColor Cyan
Get-ADObject -Filter {msDS-AllowedToDelegateTo -like "*"} `
  -Properties msDS-AllowedToDelegateTo, ObjectClass, SamAccountName, Enabled, LastLogonDate |
  ForEach-Object {
    $targets = ($_.("msDS-AllowedToDelegateTo") -join " | ")
    $results.Add([PSCustomObject]@{
        Type             = "Constrained (KCD)"
        ObjectClass      = $_.ObjectClass
        Name             = if ($_.SamAccountName) { $_.SamAccountName } else { $_.Name }
        Enabled          = $_.Enabled
        LastLogon        = $_.LastLogonDate
        DelegationType   = "msDS-AllowedToDelegateTo"
        DelegationTarget = $targets
        OS               = ""
        Description      = ""
        Risk             = "ELEVE"
    })
}

Write-Host "[*] Collecte des délégations RBCD..." -ForegroundColor Cyan
Get-ADComputer -Filter * `
  -Properties msDS-AllowedToActOnBehalfOfOtherIdentity, Description, Enabled |
  Where-Object {$_."msDS-AllowedToActOnBehalfOfOtherIdentity" -ne $null} |
  ForEach-Object {
    $sd = New-Object Security.AccessControl.RawSecurityDescriptor(
        $_."msDS-AllowedToActOnBehalfOfOtherIdentity", 0)
    $trustees = $sd.DiscretionaryAcl | ForEach-Object {
        try {
            (New-Object System.Security.Principal.SecurityIdentifier($_.SecurityIdentifier)
            ).Translate([System.Security.Principal.NTAccount]).Value
        } catch { $_.SecurityIdentifier.ToString() }
    }
    $results.Add([PSCustomObject]@{
        Type             = "RBCD"
        ObjectClass      = "Computer"
        Name             = $_.Name
```

```
        Enabled          = $_.Enabled
        LastLogon        = ""
        DelegationType   = "msDS-AllowedToActOnBehalfOfOtherIdentity"
        DelegationTarget = ($trustees -join " | ")
        OS               = ""
        Description      = $_.Description
        Risk             = "ELEVE"
    })
}

$csvPath = "$OutputPath\delegations_inventaire.csv"
$results | Export-Csv -Path $csvPath -NoTypeInformation -Encoding UTF8 -Delimiter ";"
Write-Host "[+] CSV exporté: $csvPath" -ForegroundColor Green

$critiques = @($results | Where-Object {$_.Risk -eq "CRITIQUE"}).Count
$eleves    = @($results | Where-Object {$_.Risk -eq "ELEVE"}).Count
$htmlPath  = "$OutputPath\rapport_delegations.html"
$html = @"
<!DOCTYPE html><html lang='fr'><head><meta charset='UTF-8'>
<title>Rapport Délégations AD - $(Get-Date -Format 'yyyy-MM-dd')</title>
</head><body>
<h1>Rapport d'Audit des Délégations Active Directory</h1>
<p>Total: $($results.Count) — CRITIQUES: $critiques — ELEVES: $eleves</p>
"@
$html | Out-File -FilePath $htmlPath -Encoding UTF8
Write-Host "[+] Terminé. $($results.Count) délégations inventoriées." -ForegroundColor Green
$results | Group-Object Risk | ForEach-Object {
    Write-Host "    $($_.Name): $($_.Count)" -ForegroundColor Yellow
}
```

Délégation Unconstrained : Détection et Remédiation

La délégation Unconstrained est particulièrement dangereuse en combinaison avec les techniques de **coercion d'authentification** qui forcent un contrôleur de domaine à s'authentifier vers une machine contrôlée par l'attaquant.

Techniques de Coercion d'Authentification

Printer Bug (SpoolSample) : exploite le service Print Spooler pour forcer l'authentification NTLM/Kerberos d'un DC

PetitPotam : exploite MS-EFSRPC pour forcer l'authentification d'un DC ou d'une CA ADCS

DFSCoerce : exploite MS-DFSNM pour coercer l'authentification via DFS

Coercer (outil) : automatise la tentative de toutes les méthodes de coercion connues

Si un DC s'authentifie vers une machine avec délégation Unconstrained, son TGT est mis en cache sur cette machine. L'attaquant récupère ce TGT via `Rubeus.exe monitor /interval:5` (écoute continue) ou `Rubeus.exe dump /service:krbtgt`, puis l'utilise pour un Pass-the-Ticket et un DCSync.

Détection via les Événements Windows

Les délégations Unconstrained génèrent des événements Kerberos spécifiques sur les contrôleurs de domaine :

Event 4769 (Kerberos Service Ticket Request) : ticket demandé avec le flag `forwardable` (bit 30) ou `forwarded` (bit 29) dans le champ Ticket Options — indique une délégation Unconstrained active

Event 4768 (Kerberos AS Request) : authentification initiale — surveiller les requêtes depuis des comptes machine DC vers des serveurs non-DC

Requête KQL (Azure Sentinel / Microsoft Defender) pour détecter les TGTs délégués :

```
SecurityEvent
| where EventID == 4769
| where TicketOptions has "0x60810010"
| where ServiceName !endswith "$"
| summarize count() by Account, IpAddress, ServiceName, bin(TimeGenerated, 1h)
| where count_ > 10
| order by count_ desc
```

Remédiation des Délégations Unconstrained

```
 # Script de nettoyage des délégations Unconstrained (hors DCs)
# ATTENTION: tester en environnement de recette avant production

$dcList = (Get-ADDomainController -Filter *).Name
Get-ADComputer -Filter {TrustedForDelegation -eq $true} -Properties TrustedForDelegation |
  Where-Object {$_.Name -notin $dcList} |
  ForEach-Object {
    Write-Host "Désactivation Unconstrained sur: $($_.Name)" -ForegroundColor Yellow
    # Set-ADComputer $_ -TrustedForDelegation $false
  }

Get-ADUser -Filter {TrustedForDelegation -eq $true} -Properties TrustedForDelegation |
  ForEach-Object {
    Write-Host "Désactivation Unconstrained sur user: $($_.SamAccountName)" -ForegroundColor Yell
    # Set-ADAccountControl $_ -TrustedForDelegation $false
  }

# Ajouter les comptes sensibles à Protected Users
$privilegedAccounts = Get-ADGroupMember "Domain Admins" -Recursive |
  Where-Object {$_.objectClass -eq "user"}
$privilegedAccounts | ForEach-Object {
    Add-ADGroupMember -Identity "Protected Users" -Members $_
    Write-Host "Ajouté à Protected Users: $($_.SamAccountName)" -ForegroundColor Green
}
```

Resource-Based Constrained Delegation (RBCD) : Attaque et Défense

L'attaque RBCD est l'une des techniques d'escalade de privilèges les plus élégantes dans les environnements Active Directory modernes. Elle nécessite deux conditions : un droit d'écriture (`GenericWrite` , `WriteProperty` , ou `GenericAll`) sur un objet ordinateur cible, et un compte contrôlé avec un SPN (compte de service ou compte machine).

Déroulement de l'Attaque RBCD

```
 # Étape 1: Créer un compte machine (si MachineAccountQuota > 0)
# New-MachineAccount -MachineAccount "FakeComputer" -Password (ConvertTo-SecureString "Pass@123"

# Étape 2: Écrire msDS-AllowedToActOnBehalfOfOtherIdentity sur la cible
# (avec un droit GenericWrite sur SERVEUR-CIBLE$)
$SD = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList "O:BAD:(A;;CCDCLCSWRP
$SDBytes = New-Object Byte[] ($SD.BinaryLength)
$SD.GetBinaryForm($SDBytes, 0)
Get-ADComputer SERVEUR-CIBLE | Set-ADComputer -Replace @{"msDS-AllowedToActOnBehalfOfOtherIdentit

# Étape 3: Obtenir un TGS en tant que Domain Admin via S4U2Self + S4U2Proxy
# Rubeus.exe s4u /user:FakeComputer$ /rc4:<hash> /impersonateuser:Administrateur /msdsspn:cifs/SE
```

Détection RBCD

Event 4742 (Computer Account Changed) : modification du compte ordinateur cible — surveiller l'attribut `msDS-AllowedToActOnBehalfOfOtherIdentity` dans les attributs modifiés

Event 5136 (Directory Service Object Modified) : filtrer sur l'attribut `msDS-AllowedToActOnBehalfOfOtherIdentity` avec SubjectUserName différent d'un compte légitime

Surveiller la création de comptes machine (**Event 4741**) par des utilisateurs non-administrateurs

Requête KQL pour détecter les modifications RBCD suspectes :

```
SecurityEvent
| where EventID == 5136
| where EventData has "msDS-AllowedToActOnBehalfOfOtherIdentity"
| where SubjectUserName !in~ ("MSOL_XXXXXXXX", "AAD_XXXXXXXX")
| project TimeGenerated, SubjectUserName, Computer
| order by TimeGenerated desc
```

Défense contre RBCD

Protected Users : les comptes membres du groupe Protected Users ne peuvent pas être usurpés via RBCD (Kerberos S4U2Self bloqué)

Réduire ms-DS-MachineAccountQuota à 0 : empêche les utilisateurs standards de créer des comptes machine exploitables en RBCD

SACL sur les objets Tier 0 : activer l'audit de modification sur tous les DCs et serveurs critiques pour détecter Event 5136

Privileged Access Workstations (PAW) : les PAW isolent les comptes Tier 0 des segments réseaux à risque où RBCD est exploitable

---

Remédiation : Nettoyage et Sécurisation des Délégations

La remédiation des délégations excessives suit le **principe du moindre privilège** et s'inscrit dans le modèle de Tiering Active Directory. L'objectif est de documenter les délégations légitimes et de supprimer toutes les autres.

Modèle de Tiering et Délégations

Dans le modèle de Tiering Microsoft (Tier 0 / Tier 1 / Tier 2), les délégations ne doivent jamais traverser les frontières de tier :

Tier 0 (DCs, CA, ADFS, AAD Connect) : aucune délégation vers ou depuis des ressources Tier 1/2. Les comptes Tier 0 doivent être dans Protected Users.

Tier 1 (serveurs membres) : délégations Constrained uniquement, vers des services Tier 1 spécifiques. Jamais vers un DC.

Tier 2 (postes de travail) : pas de délégation Kerberos. Jamais vers Tier 0 ou Tier 1.

Script PowerShell de Remédiation des Délégations Excessives

```
 <#
.SYNOPSIS
    Remédiation des délégations excessives AD
    Mode simulation par défaut. Passer -Apply pour appliquer.
#>
param(
    [switch]$Apply,
    [string]$LogPath = "C:\Audit\remediation_$(Get-Date -Format 'yyyyMMdd_HHmm').log"
)

function Write-Log { param($Message, $Level="INFO")
    "$(Get-Date -Format 'yyyy-MM-dd HH:mm:ss') [$Level] $Message" | Tee-Object -FilePath $LogPath

Import-Module ActiveDirectory
$dcList = (Get-ADDomainController -Filter *).Name
Write-Log "=== Début remédiation délégations AD ==="
Write-Log "Mode: $(if ($Apply) {'APPLIQUER'} else {'SIMULATION'})"

# 1. Désactiver Unconstrained sur ordinateurs non-DC
Get-ADComputer -Filter {TrustedForDelegation -eq $true} -Properties TrustedForDelegation |
  Where-Object {$_.Name -notin $dcList} | ForEach-Object {
    Write-Log "UNCONSTRAINED->DÉSACTIVER: $($_.DistinguishedName)" "WARN"
    if ($Apply) { Set-ADAccountControl $_ -TrustedForDelegation $false
                  Write-Log "APPLIQUÉ: Unconstrained désactivé sur $($_.Name)" "OK" }
}

# 2. Désactiver Unconstrained sur utilisateurs
Get-ADUser -Filter {TrustedForDelegation -eq $true} -Properties TrustedForDelegation | ForEach-Ob
    Write-Log "UNCONSTRAINED_USER->DÉSACTIVER: $($_.SamAccountName)" "WARN"
    if ($Apply) { Set-ADAccountControl $_ -TrustedForDelegation $false
                  Write-Log "APPLIQUÉ: user $($_.SamAccountName)" "OK" }
}

# 3. Supprimer RBCD non documentés
$RBCDWhitelist = @("SERVEUR-LEGITIME-01", "SERVEUR-LEGITIME-02")
Get-ADComputer -Filter * -Properties msDS-AllowedToActOnBehalfOfOtherIdentity |
  Where-Object { $_."msDS-AllowedToActOnBehalfOfOtherIdentity" -ne $null -and $_.Name -notin $RBC
  ForEach-Object {
    Write-Log "RBCD_NON_DOCUMENTÉ->SUPPRIMER: $($_.Name)" "WARN"
    if ($Apply) { Set-ADComputer $_ -Clear "msDS-AllowedToActOnBehalfOfOtherIdentity"
                  Write-Log "APPLIQUÉ: RBCD supprimé sur $($_.Name)" "OK" }
}
```

```
# 4. Réduire ms-DS-MachineAccountQuota à 0
$domain = Get-ADDomain
$quota = (Get-ADObject $domain.DistinguishedName -Properties "ms-DS-MachineAccountQuota")."ms-DS-
Write-Log "ms-DS-MachineAccountQuota actuel: $quota"
if ($quota -gt 0 -and $Apply) {
    Set-ADDomain -Identity $domain -Replace @{"ms-DS-MachineAccountQuota"=0}
    Write-Log "APPLIQUÉ: MachineAccountQuota = 0" "OK"
}
Write-Log "=== Fin remédiation. Log: $LogPath ==="
```

Documentation des Délégations Légitimes

Chaque délégation maintenue doit être documentée dans un registre (fichier CSV ou CMDB) incluant : le nom du compte ou de l'objet, le type de délégation, la justification métier, le service cible, la date de création, le propriétaire applicatif, et la date de prochaine révision (annuelle). Ce registre constitue la baseline utilisée lors des audits suivants pour identifier les délégations non documentées.

Conformité NIS 2 : Les Délégations AD dans le Cadre Réglementaire Français

La directive NIS 2, transposée en droit français, impose aux entités essentielles et importantes des exigences explicites sur la gestion des accès à privilèges. L'article 21§2.i de la directive NIS 2 couvre la *gestion des actifs, y compris les contrôles d'accès et la gestion des identités*. En pratique, l'ANSSI traduit cela par plusieurs obligations directement liées aux délégations Active Directory.

Obligations NIS 2 Applicables aux Délégations AD

Inventaire annuel des délégations : toute entité soumise à NIS 2 doit réaliser au moins un audit annuel des droits de délégation AD et en conserver la trace documentaire

Principe du moindre privilège : les délégations doivent être strictement limitées aux besoins opérationnels démontrés — les délégations Unconstrained sont de fait incompatibles avec NIS 2 sauf justification technique exceptionnelle

Détection des modifications non autorisées : l'activation de l'audit SACL (Events 5136, 4742) sur les objets Tier 0 est requise pour détecter les modifications de délégation non autorisées

Plan de remédiation documenté : les délégations excessives identifiées doivent faire l'objet d'un plan de remédiation daté et suivi

Pour les entreprises françaises à Paris et en régions ayant désigné un RSSI, l'audit des délégations AD s'inscrit naturellement dans le plan d'audit annuel SSI. Les outils présentés (BloodHound CE, Adalanche, scripts PowerShell) sont entièrement compatibles avec une utilisation on-premises sans transfert de données hors du SI — ce qui est cohérent avec les exigences de souveraineté des données imposées par le RGPD et les référentiels ANSSI. L'Authentication Policy Silos de Windows Server 2025 constitue un mécanisme complémentaire pour isoler les comptes Tier 0 et réduire la surface d'attaque des délégations. Pour une surveillance opérationnelle continue, la mise en place d'honeypots Active Directory (Honey Users, Honey SPNs) permet de détecter en temps réel les tentatives d'exploitation des délégations, tandis que la surveillance des événements Windows sur les contrôleurs de domaine (Events 4769, 4768, 5136, 4742) garantit la traçabilité requise par NIS 2.

Questions Fréquentes sur l'Audit des Délégations AD

Quelle est la différence entre délégation Unconstrained et RBCD en termes de risque ?
La délégation Unconstrained est la plus dangereuse : elle permet à un attaquant ayant compromis le serveur de récupérer les TGTs de *tous* les utilisateurs qui s'y authentifient, y compris les Domain Admins. La RBCD est plus ciblée — elle nécessite de contrôler un compte avec SPN et d'avoir des droits d'écriture sur la cible — mais elle est plus difficile à détecter car elle modifie un attribut ordinateur (msDS-AllowedToActOnBehalfOfOtherIdentity) plutôt que d'activer un flag global. En 2026, la RBCD est le vecteur d'escalade le plus fréquemment exploité dans les intrusions AD car de nombreux administrateurs ignorent encore la signification de cet attribut.
Peut-on auditer les délégations AD sans BloodHound, depuis un poste Linux ?

Oui, avec Adalanche (binaire Go multiplateforme) ou avec des requêtes LDAP directes (`ldapsearch`). Adalanche collecte toutes les délégations via LDAP depuis Linux avec un simple compte de lecture AD. Pour les délégations ACL complexes, Adalanche est même plus exhaustif que certaines configurations SharpHound car il parse directement les ACL LDAP. Les requêtes LDAP natives restent possibles pour un audit ponctuel : `ldapsearch -H ldap://dc01 -b "DC=domaine,DC=local" "(TrustedForDelegation=TRUE)" dn` .

Comment détecter une exploitation des délégations Unconstrained en temps réel ?

La détection repose sur l'Event 4769 (Ticket Service Kerberos) avec les flags `forwardable` et `forwarded` sur les contrôleurs de domaine, combiné à la corrélation avec les accès réseau vers les machines configurées en Unconstrained. Un compte DC qui s'authentifie soudainement vers un serveur membre via Kerberos (Event 4769 avec ServiceName = compte machine du serveur) indique une probable coercion d'authentification. Les solutions SIEM disposent de règles de détection prêtes à l'emploi pour ce pattern.

Les comptes Protected Users sont-ils immunisés contre toutes les attaques de délégation ?

Les comptes membres du groupe **Protected Users** bénéficient de plusieurs protections Kerberos : leurs tickets ne peuvent pas être délégués (flag *not delegated* forcé), ils ne peuvent pas utiliser NTLM, DES ou RC4, et leur TGT a une durée de vie réduite à 4 heures. Cela les immunise contre les attaques de délégation Unconstrained et Constrained. Cependant, Protected Users ne protège pas contre toutes les attaques ACL (GenericAll, WriteDACL sur l'objet utilisateur restent exploitables) ni contre les attaques de Shadow Credentials.

Quelle est la fréquence recommandée pour un audit complet des délégations AD ?

L'ANSSI et les référentiels NIS 2 recommandent un audit complet annuel, mais la réalité opérationnelle impose une surveillance plus fréquente : audit complet annuel (BloodHound CE ou Adalanche + rapport managérial), audit partiel trimestriel (script PowerShell d'inventaire), et détection continue via SIEM (Events 5136, 4742, 4769) pour les modifications de délégation en temps réel. Après tout changement d'infrastructure majeur (migration, fusion, intégration d'un nouveau domaine), un audit ciblé est indispensable.

À RETENIR

La délégation Unconstrained est le risque le plus élevé : tout serveur avec `TrustedForDelegation=True` est une cible privilégiée pour la coercion d'authentification (Printer Bug, PetitPotam) vers un accès Domain Admin complet.

BloodHound CE avec SharpHound cartographie l'ensemble des chemins d'attaque via délégations en quelques minutes — les requêtes Cypher `writeAccountRestrictions`, `AllowedToDelegate` et `GenericAll vers DA` doivent être vérifiées à chaque audit.

Adalanche offre un audit ACL plus granulaire depuis Linux sans Neo4j — idéal pour les équipes souhaitant un outil léger et souverain pour l'inventaire des délégations AD.

La remédiation repose sur trois piliers : désactiver toute délégation Unconstrained hors DC, ajouter tous les comptes Tier 0 dans Protected Users, et réduire `ms-DS-MachineAccountQuota` à zéro pour bloquer la création non autorisée de comptes machine exploitables en RBCD.

NIS 2 impose un audit annuel documenté des délégations AD : le script PowerShell d'inventaire CSV/HTML présenté dans cet article constitue une base directement utilisable pour répondre aux exigences de l'article 21§2.i de la directive et aux contrôles ANSSI.

Références et documentation

Microsoft Learn — Sécurité Windows Server

ANSSI — Guide de sécurisation Active Directory

MITRE ATT&CK — Techniques Active Directory

