AYI NEDJIMI Consultants

BloodHound : Cartographie des Chemins d'Attaque Active

Catégorie : Attaques Active Directory Lecture : 11 min Publié le : 08/03/2026 Auteur : Ayi NEDJIMI

Guide complet BloodHound CE : cartographie des chemins d'attaque Active Directory, collecteurs SharpHound/RustHound, requêtes Cypher, chemins.

2.1 Noeuds, arêtes et relations de contrôle

La théorie des graphes fournit le cadre mathématique parfait pour modéliser la sécurité d'Active Directory. Dans le modèle BloodHound, chaque objet AD est représenté comme un **noeud** (node) et chaque relation de contrôle ou de permission constitue une **arête** (edge) orientée entre deux noeuds. Guide complet BloodHound CE : cartographie des chemins d'attaque Active Directory, collecteurs SharpHound/RustHound, requêtes Cypher, chemins. Active Directory reste la cible privilégiée des attaquants en environnement Windows. Comprendre bloodhound cartographie chemins attaque ad est indispensable pour les équipes offensives comme défensives. Nous abordons notamment : 7. défenses : identifier et remédier les chemins d'attaque, 8. bloodhound dans le workflow pentest et 9. bloodhound vs pingcastle : complémentarité des approches. Les professionnels y trouveront des recommandations actionnables, des commandes prêtes à l'emploi et des stratégies de mise en œuvre adaptées aux environnements d'entreprise.

Les **types de noeuds** principaux dans le graphe BloodHound sont :

Type de noeud	Représente	Exemples d'attributs collectés
User	Compte utilisateur AD	SID, UPN, adminCount, servicePrincipalName, lastLogon
Group	Groupe de sécurité	SID, membres, memberOf, adminCount
Computer	Compte ordinateur	OS, sessions actives, LocalAdmin, services
Domain	Domaine AD	Functional level, trusts, policies
GPO	Group Policy Object	GUID, liens OU, permissions d'écriture
OU	Unité d'Organisation	Distinguished Name, GPO linkées, ACL
Container	Conteneur AD	Distinguished Name, objets enfants

Les **arêtes** (edges) représentent les relations de contrôle entre ces noeuds. BloodHound CE reconnaît plus de **30 types d'arêtes**, dont les plus critiques sont :

- **MemberOf** : appartenance à un groupe (transitif -- un membre d'un groupe membre de Domain Admins hérite des privilèges)
- **AdminTo** : droits d'administration locale sur un ordinateur

- **HasSession** : session active d'un utilisateur sur un ordinateur (permet le credential harvesting)
- **GenericAll** : contrôle total sur un objet (modification d'attributs, réinitialisation de mot de passe, etc.)
- **GenericWrite** : écriture d'attributs arbitraires (permet le Targeted Kerberoasting via modification de SPN)
- **WriteDacl** : modification de la DACL d'un objet (permet de s'octroyer n'importe quel droit)
- **WriteOwner** : modification du propriétaire d'un objet (le propriétaire peut modifier la DACL)
- **ForceChangePassword** : réinitialisation du mot de passe sans connaître l'actuel
- **DCSync** : droits de réplication (DS-Replication-Get-Changes + DS-Replication-Get-Changes-All) permettant l'extraction de tous les hashes
- **GPLink** : liaison d'une GPO à une OU (contrôle des paramètres appliqués aux objets de l'OU)

2.2 Le concept de chemin d'attaque (Attack Path)

Un **chemin d'attaque** est une séquence ordonnée de noeuds et d'arêtes qui permet à un attaquant de progresser d'un point de départ (typiquement un compte utilisateur compromis) vers un objectif (typiquement le groupe Domain Admins). L'algorithme de **plus court chemin** (Shortest Path, basé sur Dijkstra ou BFS) est fondamental dans BloodHound car il identifie la route la plus directe vers l'objectif.

Prenons un exemple concret. Un attaquant compromet le compte `jdupont` (utilisateur standard du département comptabilité). BloodHound révèle la chaîne suivante :

```
# Chemin d'attaque composite - Exemple réel anonymisé
jdupont (User)
  --[MemberOf]--> GRP-Comptabilite (Group)
    --[GenericWrite]--> svc-backup (User)
      # Targeted Kerberoasting : modification du SPN
      --[MemberOf]--> GRP-Backup-Operators (Group)
        --[AdminTo]--> SRV-DC02 (Computer)
          # Accès admin local au DC secondaire
          --[HasSession]--> admin-tier0 (User)
            # Credential harvesting via Mimikatz
            --[MemberOf]--> Domain Admins (Group)
              # Compromission totale du domaine
```

Une compromission d'un seul poste de travail pourrait-elle mener à votre contrôleur de domaine ?

Ce chemin exploite cinq relations distinctes, chacune apparemment bénigne quand examinée isolément. Le groupe Comptabilité ayant GenericWrite sur un compte de service ? Probablement une délégation administrative mal configurée. Le compte de service dans Backup Operators ? Une décision d'administration système. Backup Operators ayant AdminTo sur un DC ? Un comportement attendu. Un administrateur Tier 0 ayant une session sur ce DC ? Absolument normal. Mais la **combinaison** de ces relations crée un chemin direct de la comptabilité au Domain Admin en quatre étapes.

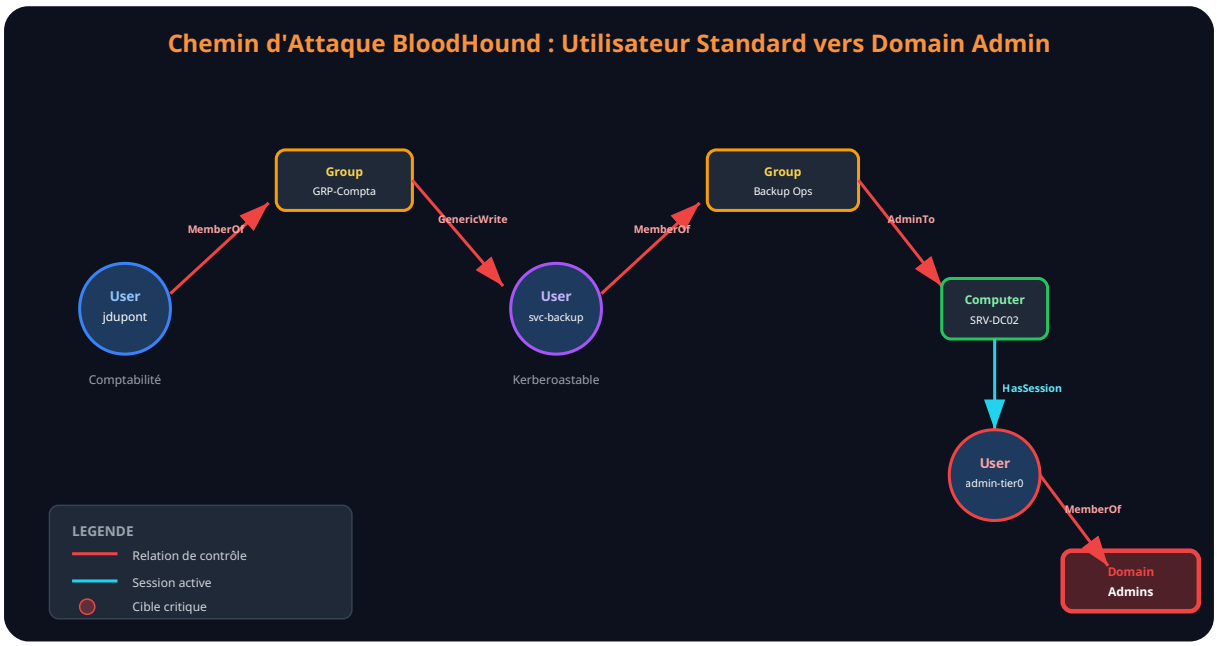

Figure 1 -- Chemin d'attaque composite : d'un utilisateur Comptabilité au Domain Admin en 6 étapes

2.3 Métriques de graphe pour prioriser la remédiation

La théorie des graphes offre des métriques puissantes pour quantifier le risque et prioriser les actions correctives :

- **Shortest Path Count** : nombre de chemins les plus courts vers Domain Admin depuis chaque noeud. Un noeud avec 50 shortest paths est prioritaire par rapport à un noeud avec 2.
- **Betweenness Centrality** : mesure la fréquence à laquelle un noeud apparaît sur les chemins les plus courts entre d'autres paires de noeuds. Un noeud à haute centralité est un **point de passage critique** -- le supprimer élimine de nombreux chemins d'attaque simultanément.
- **Degree Centrality** : nombre d'arêtes entrantes et sortantes d'un noeud. Un compte avec un degree élevé a de nombreuses relations de contrôle -- c'est une cible de choix et un point de remédiation prioritaire.
- **Tier Reachability** : pourcentage d'objets Tier 0 (Domain Admins, Domain Controllers, KRBTGT, etc.) atteignables depuis un noeud donné. Métrique directe du risque de compromission totale.

Notre avis d'expert
Kerberos, conçu il y a des décennies, porte en lui des faiblesses architecturales que les attaquants exploitent quotidiennement. Le passage à une authentification moderne basée sur des certificats et FIDO2 n'est plus optionnel — c'est une question de survie numérique.

AzureHound étend la couverture de BloodHound à **Microsoft Entra ID** (Azure AD) et aux ressources Azure. Il collecte les relations entre les objets cloud : utilisateurs Entra ID, groupes, applications, service principals, rôles RBAC Azure, abonnements, resource groups, et Key Vaults. Pour une compréhension approfondie de la sécurité Entra ID, consultez notre article sur la sécurisation d'Entra ID avec Conditional Access.

```
# Installation AzureHound
# Télécharger depuis https://github.com/BloodHoundAD/AzureHound/releases

# Collecte avec authentification interactive
azurehound -t tenant-id list --tenant corp.onmicrosoft.com -o azure_data.json

# Collecte avec Service Principal
azurehound list -t tenant-id -a app-id -s 'client-secret' -o azure_data.json

# Collecte complète (toutes les ressources)
azurehound list --tenant corp.onmicrosoft.com -o azure_full.json
```

4.4 Ingestion des données dans BloodHound CE

L'ingestion des fichiers collectés dans BloodHound CE se fait via l'interface web ou l'API REST :

```
# Ingestion via l'interface web :
# 1. Connectez-vous à http://localhost:8080
# 2. Cliquez sur l'icône "Upload" (flèche vers le haut)
# 3. Sélectionnez le fichier ZIP généré par SharpHound/RustHound
# 4. L'ingestion commence automatiquement

# Ingestion via API REST (pour automatisation)
curl -X POST "http://localhost:8080/api/v2/file-upload" \
  -H "Authorization: Bearer YOUR_TOKEN" \
  -F "file=@audit_domaine_2026.zip"

# Vérification de l'ingestion
curl -s "http://localhost:8080/api/v2/bloodhound-stats" \
  -H "Authorization: Bearer YOUR_TOKEN" | jq .
# Résultat : nombre de noeuds, arêtes, sessions importées
```

```
// Utilisateurs non Tier-0 ayant des sessions sur les DCs
MATCH (c:Computer)-[:MemberOf*1..]->(g:Group {name: "DOMAIN CONTROLLERS@CORP.LOCAL"})
MATCH (u:User)-[:HasSession]->(c)
WHERE NOT (u)-[:MemberOf*1..]->(:Group {name: "DOMAIN ADMINS@CORP.LOCAL"})
RETURN u.name, c.name
ORDER BY c.name
```

Chaînes AdminTo

```
// Chaînes de déplacement latéral via AdminTo
MATCH p=(u:User {enabled:true})-[:AdminTo*1..5]->(c:Computer)
WHERE (c)-[:MemberOf*1..]->(:Group {name: "DOMAIN CONTROLLERS@CORP.LOCAL"})
RETURN p
ORDER BY length(p) ASC
LIMIT 10
```

5.3 Requêtes personnalisées pour scénarios spécifiques

Au-delà des requêtes standard, voici des requêtes avancées que nous utilisons régulièrement lors de nos audits AD :

```
// Comptes de service avec mot de passe ancien (> 365 jours)
MATCH (u:User {hasspn: true, enabled: true})
WHERE u.pwdlastset < (datetime().epochSeconds - (365 * 86400))
RETURN u.name,
  datetime({epochSeconds: toInteger(u.pwdlastset)}) as LastPwdChange,
  u.serviceprincipalnames
ORDER BY u.pwdlastset ASC

// Groupes avec permissions dangereuses sur le conteneur AdminSDHolder
MATCH (n)-[r:GenericAll|WriteDacl|WriteOwner|GenericWrite]->(c:Container)
WHERE c.distinguishedname CONTAINS "AdminSDHolder"
  AND NOT n.name CONTAINS "DOMAIN ADMINS"
RETURN n.name, type(r), c.distinguishedname

// Comptes avec le flag "Do not require Kerberos preauthentication" (AS-REP roasting)
MATCH (u:User {dontreqpreauth: true, enabled: true})
RETURN u.name, u.description, u.pwdlastset
ORDER BY u.name
```

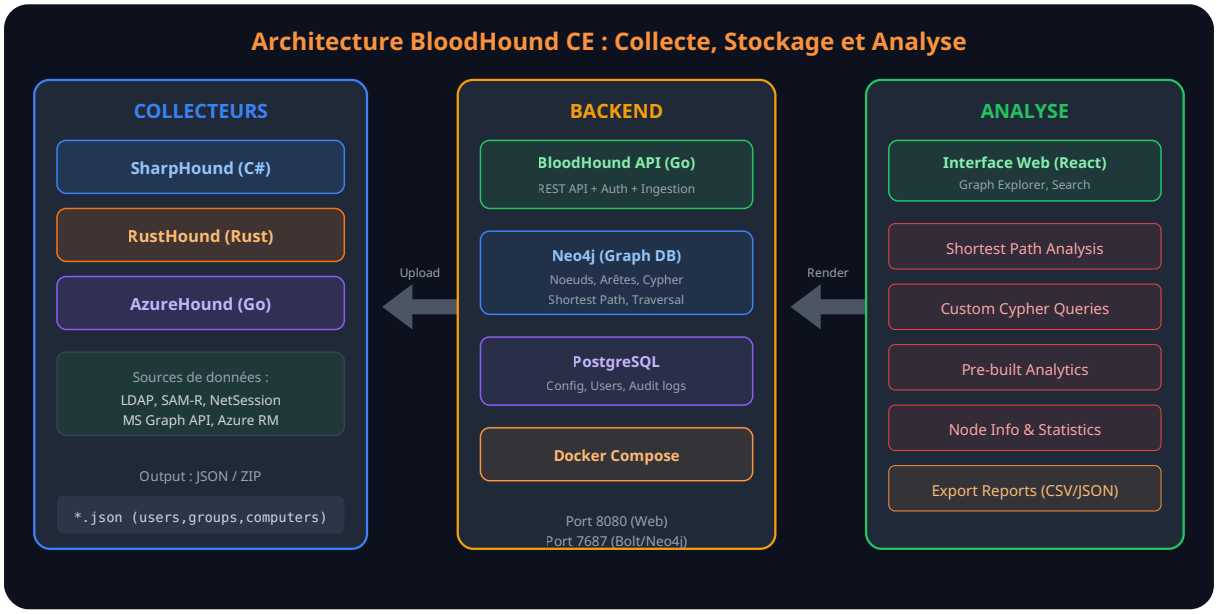

Figure 2 -- Architecture BloodHound CE : des collecteurs à l'interface d'analyse

BloodHound représente ces droits via l'arête **DCSync**. La requête suivante identifie tous les objets non légitimes disposant de ces droits :

```
// Objets non légitimes avec droits DCSync
MATCH (n)-[:DCSync|GetChangesAll*1..]->(d:Domain)
WHERE NOT n.name STARTS WITH "DOMAIN CONTROLLERS"
  AND NOT n.name STARTS WITH "DOMAIN ADMINS"
  AND NOT n.name STARTS WITH "ENTERPRISE ADMINS"
  AND NOT n.name = "ADMINISTRATOR@" + d.name
RETURN DISTINCT n.name, labels(n), d.name as Domain
ORDER BY n.name
```

6.4 Abus de délégation OU

La **délégation d'administration sur les OU** est un mécanisme légitime qui permet de confier la gestion d'un sous-ensemble d'objets à des équipes spécifiques (helpdesk, équipes régionales, etc.). Cependant, une délégation mal configurée peut créer des chemins d'attaque directs. Les scénarios les plus dangereux identifiés par BloodHound :

- **GenericAll sur une OU contenant des comptes Tier 0** : permet la réinitialisation de mot de passe des administrateurs, la modification de leurs attributs, voire la suppression de leur compte.
- **WriteDacl sur une OU de servers** : permet de modifier les ACL des serveurs contenus, s'octroyant des droits d'administration locale.
- **WriteOwner sur des OU sensibles** : le propriétaire d'une OU peut modifier sa DACL, s'octroyant ensuite n'importe quel droit sur les objets enfants.
- **GPLink abuse** : si un utilisateur peut lier des GPO à une OU, il peut déployer des scripts malveillants sur tous les ordinateurs de cette OU.

6.5 AS-REP Roasting et Shadow Credentials

Au-delà du Kerberoasting et du DCSync, BloodHound identifie d'autres chemins d'attaque avancés :

AS-REP Roasting cible les comptes dont le flag `DONT_REQ_PREAUTH` est activé. Ces comptes répondent aux requêtes AS-REQ sans pré-authentification, permettant de récupérer un blob chiffré avec le hash du mot de passe, craquable offline. BloodHound marque ces comptes avec la propriété `dontreqpreauth: true`.

Shadow Credentials exploite l'attribut `msDS-KeyCredentialLink`. Si un attaquant possède les droits d'écriture sur cet attribut (via GenericWrite ou GenericAll), il peut ajouter sa propre clé publique et s'authentifier en tant que cette identité via PKINIT. BloodHound CE identifie ces permissions comme des arêtes `AddKeyCredentialLink`.

7. Défenses : identifier et remédier les chemins d'attaque

7.1 Le modèle de Tiering (PAW / ESAE)

Le **modèle de tiering** (ou Enhanced Security Admin Environment -- ESAE) est la stratégie défensive la plus efficace pour éliminer les chemins d'attaque. Le principe est de segmenter l'infrastructure en trois niveaux :

Tier	Périmètre	Exemples d'objets	Règle fondamentale
Tier 0	Contrôle du domaine	Domain Controllers, KRBTGT, DA, EA, Schema Admins	Aucune relation de contrôle depuis Tier 1 ou 2
Tier 1	Serveurs et applications	Serveurs membres, comptes de service, admins serveur	Aucune relation de contrôle depuis Tier 2
Tier 2	Postes de travail et utilisateurs	Workstations, utilisateurs standard, helpdesk	Point d'entrée initial des attaquants

BloodHound est l'outil idéal pour **valider l'implémentation du tiering**. La requête suivante identifie les violations de tiering (relations cross-tier) :

```
// Violations de tiering : chemins de Tier 2 vers Tier 0
MATCH p=shortestPath(
  (source)-[*1..]->(target:Group {name: "DOMAIN ADMINS@CORP.LOCAL"})
)
WHERE NOT (source)-[:MemberOf*1..]->(:Group {admincount: true})
  AND labels(source) IN [["User"], ["Computer"]]
  AND source.enabled = true
RETURN source.name, labels(source), length(p) as PathLength
ORDER BY PathLength ASC
LIMIT 50
```

7.2 Stratégie de remédiation par priorité

La remédiation des chemins d'attaque doit suivre une approche méthodique basée sur l'**impact maximum avec l'effort minimum**. Nous recommandons la stratégie suivante :

Phase 1 -- Quick Wins (semaine 1-2) :

- Supprimer les **droits DCSync non légitimes** (impact immédiat, effort minimal)
- Désactiver les comptes avec **DONT_REQ_PREAUTH** ou retirer le flag
- Supprimer les **SPN orphelins** sur les comptes utilisateurs
- Retirer les **sessions actives non autorisées** sur les Domain Controllers

Phase 2 -- Remédiation structurelle (semaine 3-8) :

- Migrer les comptes de service vers des **gMSA**
- Nettoyer les **ACL abusives** sur les OU (GenericAll, WriteDacl, WriteOwner)
- Implémenter **LAPS** sur tous les postes et serveurs
- Revoir les **appartenances aux groupes** à privilèges (AdminCount audit)

Phase 3 -- Architecture (mois 3-6) :

- Implémenter le **modèle de tiering complet**
- Déployer des **Privileged Access Workstations** (PAW) pour les administrateurs Tier 0
- Configurer les **Authentication Policies et Silos** pour isoler les comptes Tier 0
- Mettre en place le **monitoring continu** avec des collectes BloodHound régulières

7.3 Monitoring continu avec BloodHound

La remédiation n'est pas un événement ponctuel mais un processus continu. Nous recommandons d'intégrer BloodHound dans le cycle de monitoring sécurité :

- **Collecte hebdomadaire** : exécuter SharpHound/RustHound une fois par semaine (planification via tâche planifiée ou cron)
- **Dashboard de métriques** : suivre l'évolution du nombre de chemins vers DA, du nombre de Kerberoastable users, et des violations de tiering semaine après semaine
- **Alertes sur régression** : détecter l'apparition de nouveaux chemins d'attaque (nouveau GenericAll accordé, nouveau SPN ajouté, etc.)
- **Intégration SIEM** : corréler les données BloodHound avec les logs SOC/SIEM pour une détection proactive

8. BloodHound dans le workflow pentest

8.1 Intégration dans la méthodologie d'audit

BloodHound s'intègre à chaque phase d'un pentest Active Directory. Voici notre workflow optimisé :

Phase de reconnaissance (jour 1) : Dès l'obtention d'un accès authentifié au domaine, lancer une collecte SharpHound `-c All`. Analyser les résultats pour identifier les chemins prioritaires et planifier la stratégie d'exploitation.

Phase d'exploitation (jours 2-4) : Suivre les chemins identifiés par BloodHound. À chaque nouveau compte compromis, relancer une collecte de sessions (`-c Session`) pour mettre à jour le graphe avec les nouvelles sessions disponibles. Le graphe s'enrichit à chaque pivot.

Phase de post-exploitation (jours 4-5) : Utiliser BloodHound pour documenter les chemins exploités, identifier les chemins alternatifs, et préparer les recommandations de remédiation.

8.2 Combinaison avec d'autres outils

BloodHound est encore plus puissant quand combiné avec d'autres outils de l'écosystème pentest AD :

Outil	Fonction	Synergie avec BloodHound
Impacket	Suite d'outils réseau Python	`secretsdump.py` pour DCSync, `GetNPUsers.py` pour AS-REP roasting
Rubeus	Manipulation Kerberos	Kerberoasting, delegation abuse sur les comptes identifiés par BH
Mimikatz	Credential harvesting	Extraction credentials sur les machines identifiées (HasSession)
Certipy	Attaques AD CS	ESC1-ESC8 sur les templates identifiés dans le graphe
CrackMapExec	Exécution latérale	Validation des chemins AdminTo identifiés par BloodHound
Hashcat	Cracking de hashes	Cracking des tickets Kerberos des comptes Kerberoastable

9. BloodHound vs PingCastle : complémentarité des approches

9.1 Philosophies différentes

BloodHound et **PingCastle** sont souvent comparés, mais ils répondent à des besoins différents et sont en réalité complémentaires :

Critère	BloodHound CE	PingCastle
Approche	Graphe de relations	Score de risque global
Focus	Chemins d'attaque exploitables	Conformité et bonnes pratiques
Audience principale	Pentesters, Red Team	Auditeurs, RSSI, Blue Team
Output	Graphe interactif + requêtes Cypher	Rapport HTML avec score (0-100)
Données collectées	Relations, sessions, ACL	Configuration, GPO, trusts, anomalies
Installation	Docker (Neo4j + PostgreSQL)	Exécutable standalone (.exe)
Licence	Apache 2.0 (Community Edition)	Gratuit (limitations) / Commercial
Force principale	Visualisation des chemins composites	Vue d'ensemble rapide du niveau de sécurité

9.2 Workflow combiné recommandé

Pour un audit AD complet, nous utilisons les deux outils de manière séquentielle :

1. **PingCastle en premier** : obtenir le score global, identifier les catégories de risque (Privileged Accounts, Trusts, Anomalies, Stale Objects). Le rapport PingCastle fournit un contexte rapide sur la maturité sécurité de l'AD.
2. **BloodHound ensuite** : approfondir les risques identifiés par PingCastle. Par exemple, si PingCastle signale des "Privileged Accounts with old passwords", BloodHound révèle si ces comptes ont des chemins vers des objets critiques.

3. **Corrélation des résultats** : combiner les recommandations des deux outils pour prioriser la remédiation. Un problème signalé par les deux outils est plus critique qu'un problème identifié par un seul.

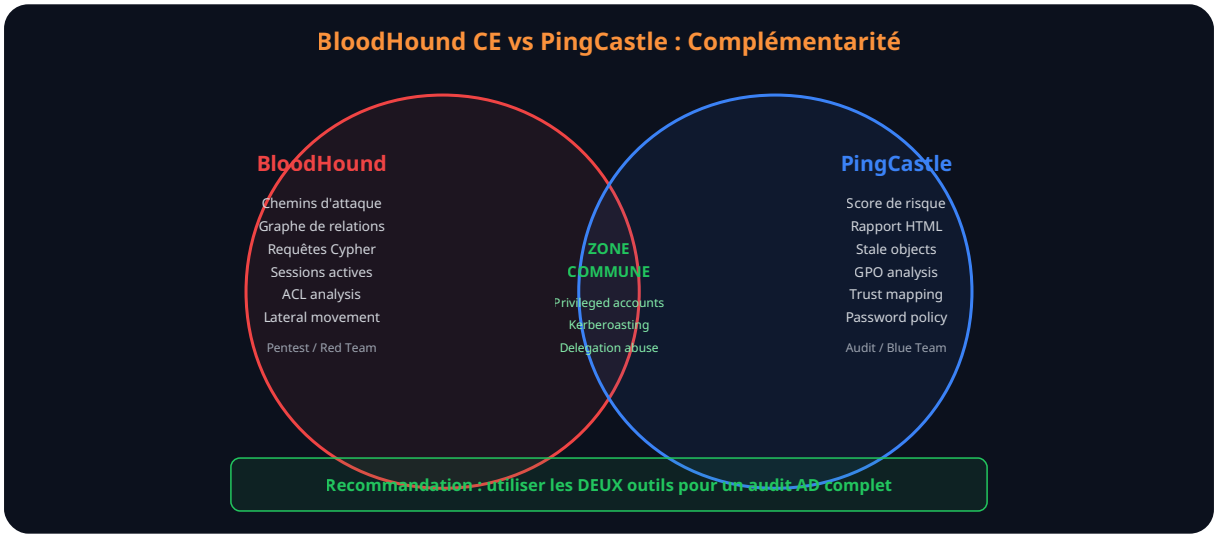

Figure 3 -- BloodHound et PingCastle : deux approches complémentaires pour l'audit AD

10. Checklist d'audit Active Directory avec BloodHound

Cette checklist synthétise les points de contrôle essentiels à vérifier lors d'un audit AD avec BloodHound. Chaque point correspond à une requête Cypher ou une analyse spécifique dans l'interface :

Contrôles Tier 0 (Critiques)

- Nombre de membres (directs et transitifs) de **Domain Admins**, **Enterprise Admins**, **Schema Admins**
- Comptes avec droits **DCSync** non légitimes
- Sessions de comptes non Tier-0 sur les **Domain Controllers**
- Comptes dans **AdminSDHolder** sans justification
- Ancienneté du mot de passe **KRBTGT** (doit être changé tous les 180 jours)

Contrôles Kerberos

- Nombre de comptes **Kerberoastable** (hasspn=true)
- Comptes Kerberoastable avec **chemins vers DA**
- Ancienneté des mots de passe des comptes de service (> 365 jours = risque)
- Comptes avec **DONT_REQ_PREAUTH** (AS-REP roasting)
- Comptes avec **delegation non contrainte** (unconstrained delegation)

Contrôles ACL et délégation

- Objets non admin avec **GenericAll/WriteDacl/WriteOwner** sur des objets Tier 0
- Délégations **OU abusives** (cross-tier)
- Permissions **AddKeyCredentialLink** sur des comptes privilégiés (Shadow Credentials)
- Groupes avec **AdminTo** sur plus de 50 machines (excessive local admin)

Contrôles de sessions et déplacement latéral

- Sessions d'administration sur des **workstations Tier 2** (violation de tiering)
- Ordinateurs sans **LAPS** activé
- Chaînes **AdminTo** traversant les tiers
- Comptes avec sessions **RDP** sur plus de 10 machines simultanément

Contrôles Azure/Entra ID (si AzureHound)

- Comptes on-premises **synchronisés** avec droits Global Admin dans Entra ID
- **Applications** avec permissions excessives (Application.ReadWrite.All, etc.)
- Chemins d'attaque **hybrid** (on-prem vers cloud ou cloud vers on-prem)

Pour approfondir ce sujet, consultez notre outil open-source ad-security-audit qui facilite l'audit de sécurité complet d'Active Directory.

Questions frequentes

Comment mettre en place BloodHound dans un environnement de production ?

La mise en place de BloodHound en production necessite une planification rigoureuse, incluant l'evaluation des prerequis techniques, la definition d'une architecture cible, des tests de validation approfondis et un plan de deploiement progressif avec des points de controle a chaque etape.

Pourquoi BloodHound est-il essentiel pour la securite des systemes d'information ?

BloodHound constitue un element fondamental de la securite des systemes d'information car il permet de reduire significativement la surface d'attaque, d'ameliorer la detection des menaces et de renforcer la posture globale de securite de l'organisation face aux cybermenaces actuelles.

Comment détecter rapidement une attaque de type BloodHound : Cartographie des Chemins d'Attaque Active ?

Surveillez les événements Windows 4662, 4624 type 3 et 4672 via votre SIEM. Corrélez-les avec des connexions inhabituelles vers les contrôleurs de domaine en dehors des heures de travail.

Sources et références : MITRE ATT&CK Privilege Escalation · ADSecurity.org

Points clés à retenir

- 7. Défenses : identifier et remédier les chemins d'attaque
- 8. BloodHound dans le workflow pentest
- 9. BloodHound vs PingCastle : complémentarité des approches
- 10. Checklist d'audit Active Directory avec BloodHound
- Questions frequentes
- 11. Conclusion : BloodHound comme pilier de la sécurité AD

11. Conclusion : BloodHound comme pilier de la sécurité AD

BloodHound a fondamentalement changé la manière dont nous appréhendons la sécurité d'Active Directory. Avant son apparition, identifier les chemins d'attaque composites nécessitait des semaines d'analyse manuelle des ACL, des appartenances aux groupes et des configurations de délégation. BloodHound réduit ce travail à quelques minutes de collecte et quelques requêtes Cypher bien formulées.

Pour les **défenseurs**, BloodHound est devenu un outil indispensable de validation de la posture de sécurité. Chaque nouvelle modification d'ACL, chaque ajout de compte de service, chaque nouvelle délégation d'OU devrait être validée par une analyse BloodHound pour vérifier qu'elle ne crée pas de nouveau chemin d'attaque. L'intégration dans un pipeline CI/CD de sécurité (collecte automatisée, comparaison avec la baseline, alertes sur régression) représente l'état de l'art en matière de protection AD.

Pour les **auditeurs et pentesters**, BloodHound est l'outil qui transforme un audit AD d'une collection de points techniques en une démonstration visuelle et convaincante des risques. Un chemin d'attaque visualisé dans BloodHound est infiniment plus parlant qu'une liste de recommandations techniques dans un rapport PDF.

La prochaine frontière est l'**analyse continue et automatisée**. Avec BloodHound CE et son API REST, les organisations peuvent désormais intégrer l'analyse des chemins d'attaque dans leur SOC, avec des collectes quotidiennes et des alertes en temps réel sur l'apparition de nouveaux chemins critiques. Combiné avec les données d'analyse NTDS.dit et les recommandations de notre guide de sécurisation AD, BloodHound constitue le troisième pilier d'une stratégie de défense AD mature et proactive.

En résumé : BloodHound transforme la complexité d'Active Directory en intelligence actionnable. Utilisez-le régulièrement -- pas seulement lors des audits annuels, mais comme un outil de monitoring continu de votre posture de sécurité AD.

Ayi NEDJIMI Consultants — Expert cybersécurité offensive & intelligence artificielle

ayinedjimi-consultants.fr · ayi@ayinedjimi-consultants.fr