

Divers Satellisation



24 AOUT

404 CTF 2025
Créé par : JOL

Satellisation

Escalade de priviléges en droits pour s'échapper d'un container

Dans cette satellisation, tu dois trouver une faille dans le docker illustré : `container.c` qui permettra une élévation de priviléges et d'accéder en tant que « root » au flag : '

Ce challenge consiste à obtenir un accès à un fichier flag protégé (`/root/flag.txt`) dans un environnement isolé, typiquement un conteneur ou un espace restreint chrooté. L'objectif est d'automatiser la détection et l'exfiltration du flag en combinant trois techniques : une recherche directe de flag dans les fichiers accessibles, une surveillance active des modifications système (fichiers, sockets), et une injection passive de scripts d'exfiltration dans le système cible (`/proc/1/root`). Il met en pratique les notions de pivot local, escalade de priviléges, et détournement de services systèmes comme `logrotate` pour contourner les restrictions d'accès.

« Un docker permet ainsi de se loguer sur un plancher aux vaches avec les capacités d'administrer en root. »

De prime abord, l'accès au fichier du flag.txt n'est pas évident ni immédiat et réclame de la ruse.

Voici la signification de deux termes déployés de ce challenge :

Le terme chroot vient de la commande UNIX chroot (change root) :

- **Définition** : chroot modifie le répertoire racine (/) vu par un processus. C'est comme placer un programme dans une prison où il ne voit que les fichiers à partir d'un répertoire donné.
- **Utilité** : On utilise chroot pour isoler un programme du reste du système, par exemple dans des environnements restreints (sandbox, containers, tests).
- **Exemple** : chroot /tmp/test_env /bin/bash

Cela lance un shell où /tmp/test_env devient la racine virtuelle.

 Dans le challenge, le répertoire `/proc/1/root` est un accès indirect au vrai / du processus PID 1, souvent le système principal dans un conteneur, et l'on utilise chroot `/proc/1/root` pour s'y introduire.

Le script `flag_hunt_combined.sh` a été régénéré avec les trois étapes suivantes intégrées :

1. **Recherche directe** du flag 404CTF{...} dans les fichiers lisibles de `/proc/1/root`
2. **Surveillance active** pendant 15 secondes via inotifywait pour détecter la création/modification de fichiers contenant le flag
3. **Injection passive** d'un script logger dans `/proc/1/root/tmp/` qui lit les fichiers .log/.txt pour exfiltrer le flag
4. **Affichage du flag si trouvé**
5. **Sauvegarde du flag** dans `/tmp/flag_detect/final_flag.txt`

 Si un flag est trouvé, il sera automatiquement sauvegardé dans : `/tmp/flag_detect/final_flag.txt`

 Une procédure de captation du flag a été décrite par ailleurs à titre démonstratif sans accès ssh. :