Documentation : LD_PRELOAD

1- Qu'est-ce que LD_PRELOAD ?

LD_PRELOAD est une variable d'environnement sur Linux qui permet de charger une bibliothèque partagée (.so) avant toutes les autres lors de l'exécution d'un programme.

Ce mécanisme est utilisé principalement pour :

- Intercepter et rediriger des appels systèmes (open(), read(), write(), readdir()...).
- Modifier le comportement des programmes sans modifier leur code source.
- Créer des rootkits userland pour masquer des fichiers, des connexions, ou des processus.

2-Comment fonctionne LD_PRELOAD ?

Lorsqu'un programme est exécuté, il charge dynamiquement ses bibliothèques nécessaires (libc.so, libpthread.so, etc.). Avant d'accéder aux bibliothèques système classiques, Linux **charge d'abord toute bibliothèque définie dans LD_PRELOAD**.

3- Fonctionnalités de LD_PRELOAD utilisées dans ce projet

Dans ce projet, plusieurs techniques ont été implémentées avec **LD_PRELOAD** :

1. **Masquage des connexions réseau (hide_connections.so)**
2. **Masquage des fichiers (hide_files.so)**
3. **Masquage des logs (hide_logs.so)**
4. **Interception des identifiants SSH via PAM (pam_stealer.so)**

4-Masquage des Connexions (hide_connections.so)

Objectif :

Cacher les connexions ouvertes par le malware, en supprimant certains ports de la sortie de netstat et ss.

Technique utilisée :

- Intercepter read(), qui est utilisé pour lire /proc/net/tcp et /proc/net/udp.

- Filtrer la sortie pour masquer les ports liés au malware.

Fonctionnement :

- Un programme comme netstat ou ss ouvre /proc/net/tcp et /proc/net/udp pour afficher les connexions réseau.
- Le code redéfinit read() pour supprimer toute ligne contenant les ports spécifiques.

5- Masquage des Fichiers (hide_files.so)

Objectif :

Empêcher la détection du malware en cachant ses fichiers.

Technique utilisée :

- **Intercepter readdir()**, qui est utilisé pour lister les fichiers d'un répertoire (ls, find, stat).
- **Filtrer les fichiers sensibles** avant qu'ils ne soient retournés par readdir().

Fonctionnement :

- Lorsqu'un utilisateur tape ls ou find, le système appelle readdir().
- On modifie la sortie de readdir() pour cacher les fichiers contenant "malware", "port_knock", "hide_files".. ect

6-Masquage des Logs (hide_logs.so)

Objectif :

Éviter que les actions du malware soient enregistrées dans les logs.

Technique utilisée :

- **Intercepter write()** et bloquer les logs liés à SSH (Accepted password, sshd).
- **Effacer les logs système (journalctl, /var/log/auth.log)**.

Fonctionnement :

- Lorsqu'une application veut écrire dans un fichier log (auth.log), le système appelle write().
- On intercepte write() et on bloque les logs contenant "sshd", "pam_unix", etc.

7- Interception des Identifiants SSH via PAM

Qu'est-ce que PAM ?

PAM (Pluggable Authentication Modules) est le mécanisme utilisé sur Linux pour gérer l'authentification des utilisateurs.

Exploitation de PAM pour intercepter les mots de passe SSH

L'objectif est d'intercepter les **mots de passe en clair** lorsqu'un utilisateur se connecte via SSH.

Technique utilisée :

- **Créer un module PAM malveillant (pam_stealer.so)**.
- **Remplacer le module d'authentification par défaut (/etc/pam.d/sshd)**.
- **Capturer et envoyer les credentials** à un serveur distant.

Fonctionnement :

- Lorsqu'un utilisateur se connecte via SSH, le module PAM pam_unix.so est utilisé pour vérifier son mot de passe.
- On crée un module personnalisé qui capture les identifiants et les envoie à un serveur distant.

8- Port Knocking (Activation Cachée du Malware)

Objectif :

Démarrer un serveur SSH caché **uniquement si la bonne séquence de ports est frappée**.

Fonctionnement :

1. Un serveur écoute sur **trois ports UDP**.
2. L'attaquant doit envoyer des paquets dans le bon ordre.
3. Lorsque la séquence est correcte, sshd démarre en mode furtif.