



## Effacer ses données

Le 28 juin 2008, à 3:24 par Ulhume...

Effacer ses données, c'est une nécessité lorsque l'on vend un ordinateur ou un disque dur tout n u. Cependant beaucoup croient que le simple fait de formater suffit à rendre illisible les données, grave erreur...

### Les données d'un disque ou d'une partition

Un formatage, ne détruit pas les informations que contiennent un disque. Cela ne fait que... formater le disque pour recevoir de nouvelle donnée. C'est un peu, en caricaturant, comme si vous pensiez rendre un livre illisible en arrachant la table de matière... Et ce n'est pas si caricatural que cela lorsqu'il s'agit de "formatage rapide" dont c'est exactement le fonctionnement. Pour s'en convaincre, un group d'étude s'était attaqué à ce problème en achetant 111 disques durs "vierges" d'occasion. Et ils y ont trouvé de tout jusqu'aux [preuves d'infidélité d'une femme mariée](#) <sup>[1]</sup>.

Il existe pourtant de nombreux outils pour effacer réellement. Une version basique consiste à simplement remplir le disque de 0 ou mieux, de nombre aléatoires. Sous GNU/Linux cela se fait très simplement en utilisant le concept `tout n'est que fichier`. Ainsi votre premier disque IDE, ou la première partition de ce disque, sont les fichiers `/dev/hda` et `/dev/hda1`. Du coup, il est très simple de remplir ce fichier de zéros grâce à la fameuse command `dd` :

```
|| dd [2] if= dev zero of= dev hda
```

La même chose est possible, en plus long, avec des nombres aléatoires :

```
|| dd [2] if= dev random of= dev hda
```

A ce stade, on peut décemment se dire que le disque est illisible. En réalité c'est très loin d'être le cas. Avec un appareillage approprié, il est parfaitement possible de lire l'ancienne donnée sous le zéro. Et juste un peu plus difficile sous le nombre aléatoire.

En effet, la surface magnétique garde mémoire de son passé par le simple fait qu'une écriture "au même endroit" n'est jamais positionnée pile-poil sur l'ancienne. Un décalage invisible pour un contrôleur de disque standard, mais parfaitement clair pour des professionnels.

La solution est alors d'écrire plusieurs fois de suites au même endroit de sorte à finir par rendre cette maudite donnée d'origine totalement illisible, allant jusqu'au plus violent, le `Gutmann Wipe` et ses 35 passages aléatoires dans les données ET dans l'ordre de passage.

Pour obtenir un tel résultat, `dd` ne suffit plus, il faut passer à des outils comme [Darik's Boot and Nuke](#) <sup>[3]</sup>. Et là c'est plusieurs heures qu'il va falloir passer sur ce disque...

Maintenant, il faut revenir sur terre. Mulder a sûrement intérêt à passer par DBAN pour vider son portable avec de le revendre sur eBay. Mais Mme Michu, elle, elle n'a pas forcément d'informations critique

susceptible d'intéresser la moindre agence gouvernementale. Le coût de récupération serait pour elle largement supérieur à ce que peuvent rapporter ses données. Et c'est finalement un élément clef de sécurité : le risque est fonction de l'intérêt qu'un tiers peut porter à ce que l'on cherche à protéger. Dans le cas du commun des mortels dont je suis, il faut juste que le commun des mortels à que je revend mon disque ne soit pas capable de lire ce qui s'y trouve, même en passant par des outils bas niveaux.

Du coup, un bon vieux remplissage de zéros est à ma connaissance largement suffisante pour protéger efficacement les données de toute actions qui n'inclue pas un tournevis pour extraire les plateaux du disque dur.

Cependant, même si `dd` répond donc bien à ce cahier des charges, un utilitaire plus simple d'usage existe dans toute distribution Linux de base : `shred`.

Le but de cet outil est d'écrire N fois des nombres aléatoires dans un fichier, et éventuellement de terminer par une couche de zéros. Par exemple un ultra paranoïaque ayant quelques heures à perdre, peut taper :

```
|| shred [4] -zvf -n 100 dev hda
```

Là, c'est 100 couches de nombres aléatoires suivie d'une couche de zéros (le `-z`) qui viennent recouvrir nos données. Quelqu'un de moins paranoïaque ou de plus réaliste utilisera seulement la couche de zéros :

```
|| shred [4] -zvf dev hda
```

## Effacer un fichier

Là, pour un fichier simple, ça se complique. En effet, les systèmes de fichiers modernes sont tous "journalisés" (EXT3, NTFS, Reiser, etc). Cela veut dire que les données d'un fichiers ne sont pas forcément stockées à l'endroit où se trouve le fichier... L'idée de base d'un journalisé est qu'une écriture n'est pas directement placée à sa place sur le disque, mais historisé dans un journal. C'est ce journal qui est rejoué au redémarrage lorsque votre système a planté ou que le chat s'est battu avec les câbles électriques.

Du coup, si vous utilisez par exemple `shred` sur un fichier, les données aléatoires ou les zéros ne peuvent garantir que les données réelles du fichier ont bien été remplacées. Vous pensez votre fichier détruit mais il reste visible avec un outil bas niveau de lecture des données du disque.

En somme, il n'est juste pas possible d'effacer sérieusement un fichier seul sur un système journalisé?



L'assertion précédente est à prendre au sens général. Pour certains journalisés comme EXT3, il semblerait que seules les métadonnées le soient vraiment, et donc qu'un `shred` fonctionnerait. Mais pour un paranoïaque, même léger, ce n'est pas très satisfaisant.

## Conclusion

`shred` permet simplement et rapidement d'obtenir un niveau suffisant de sécurité. Mais si vous avez de véritables données sensibles, par exemple une entreprise avec une machine ayant servi à des projets

critiques, l'utilisation de DBAN s'impose. Mais dans tous les cas, ces solutions sont à applique sur la partition, ou mieux, sur le disque complet.

**Liens:**

[1] <http://www.timesonline.co.uk/tol/news/uk/article515353.ece>

[2] <http://pwet.fr/man/linux/commandes/dd>

[3] <http://dban.sourceforge.net>

[4] <http://pwet.fr/man/linux/commandes/shred>