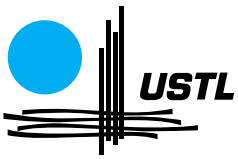

Licence d'informatique
Module de C/Unix

Projet Unix

`intègre` — Détection de la corruption de fichiers

Philippe MARQUET

Octobre 1996

La commande `integre` proposée ici est inspirée de la commande `tripwire` développée par Gene H. KIM et Eugene H. SPAFFORD, COAST Laboratory, Purdue University, West Lafayette, IN (`ftp://coast.cs.purdue.edu/pub/COAST/Tripwire/`).

Détection de la corruption de fichiers

Nous allons écrire une commande Unix `integre`[1] qui est un outil d'aide à la détection de modifications non-autorisées de fichiers. Cette commande est utilisée pour reporter tout changement dans un ensemble défini de fichiers. Une utilisation typique d'`integre` est la notification à l'administrateur système de la corruption de fichiers afin que des mesures puissent être prises au plus vite. De plus, dans le cas d'une certitude acquise de corruption d'un système de fichiers, la connaissance de l'intégrité de fichiers permet des gains de temps dans la ré-installation du système de fichiers.

`Integre` est donc un utilitaire qui compare un ensemble de fichiers avec des informations précédemment sauvées dans une base de données. Les créations et suppressions de fichiers ainsi que les changements par rapport aux données préalablement enregistrées sont reportés par `integre`.

Le principe de fonctionnement de `integre` est le suivant. Les attributs de chacun des fichiers à surveiller sont comparés entre les valeurs précédemment rangés et les valeurs actuelles. Ces attributs sont par exemple la taille, le propriétaire, l'i-nœud et la date du fichier. Cependant, il est possible de modifier le contenu d'un fichier en laissant ces différentes informations inchangées. `Integre` calcule donc pour chaque fichier une signature numérique. Une signature numérique est la valeur retournée par une fonction prenant en entrée la valeur d'un fichier. Si le contenu du fichier change, sa signature doit aussi changer. La fiabilité d'`integre` repose sur cette fonction de signature. Il ne doit pas être ¡¡ facile ¿¿ de générer une même signature à partir d'un fichier différent. En particulier les fonctions difficiles à inverser et qui génèrent des valeurs sur un grand nombre de bits sont de bonnes candidates. Il apparaît aussi évident que cette fonction signature doit rester simple à calculer pour que les performances de la commande `integre` restent raisonnables.

Fonction signature

On utilisera dans un premier temps un algorithme dit CRC (Cyclic Redundancy Code) dont le principe est le suivant. Le fichier est vu comme un énorme nombre binaire qui est divisé par un autre nombre binaire fixe. Le reste de cette division est la signature du fichier. Le choix du diviseur donne la taille de la signature ; on parlera de CRC16 pour des signatures sur 16 bits, de CRC32 pour des signatures sur 32 bits, etc.

Quelques principes arithmétiques permettent de calculer le reste de cette division à l'aide d'opérations bits à bits élémentaires. Les exécutables d'un programme produisant le CRC16

[1] Bon courage aux puristes qui préféreront nommer leur commande `intègre`.

Id : integre.tex,v 1.5 2000/12/03 10 :47 :49 marquet Exp

d'un fichier dont le nom est passé en paramètre sont disponibles dans le répertoire `~marquet/c-unix/integre/`.

(La page `http://www.ross.net/crc` est un point de départ intéressant pour plus de détails sur les CRC. En particulier, on consultera le document ¡¡ A Painless Guide to CRC Error Detection Algorithms ¿¿ accessible par exemple à partir des URLs `http://www.repairfaq.org/filipg/LINK/F_crc_v3.html` (version HTML), ou `ftp://ftp.rocksoft.com/clients/rocksoft/papers/crc_v3.txt` (version texte).)

Trois modes de fonctionnement

`Integre` fonctionne dans différents modes :

Initialisation (option `-init`). Crée une base de données initiale qui sera utilisée par les appels suivants à la commande.

Mise à jour (option `-maj` *`pathname...`*). Cette option met à jour la base de données pour les entrées dans le système de fichiers spécifiées. Si une entrée est un fichier, seul ce fichier est mis à jour. Pour une entrée de type répertoire, ce répertoire et toutes ses entrées filles (sous-répertoires et fichiers) sont mises à jour.

Test d'intégrité (option `-test`, valeur par défaut). `Integre` reporte les changements entre l'état actuel du système de fichiers et celui sauvegardé dans la base de données.

Le fonctionnement de ces différents modes est détaillé ci-après.

Syntaxe de la commande `integre`

La syntaxe de la commande `integre` est la suivante :

`integre` [`-init`|`-maj` *pathname...*|`-test`] [`-d` *dbasefile*] [`-c` *configfile*] [`-q`|`-v`]

`-d` *`dbasefile`* lit la base de données depuis le fichier précisé. La valeur par défaut est `integre.db`.

`-c` *`configfile`* lit le fichier précisé comme fichier de configuration. La valeur par défaut est `integre.cfg`.

`-q` Mode quiet. Seules les informations absolument nécessaires sont affichées.

`-v` Mode verbose. Toutes les informations disponibles sont affichées ; par exemple : état d'avancement de l'algorithme, nom du fichier en cours de traitement...

Attributs

Les attributs associés à un fichier sont les suivants. Les lettres minuscules qui suivent chacun des attributs seront utilisées pour les référencer dans la suite.

- permission du fichier (`p`) ;
- i-nœud du fichier (`i`) ;
- utilisateur propriétaire du fichier (`u`) ;
- groupe propriétaire du fichier (`g`) ;
- taille du fichier (`s`) ;
- date de dernier accès (`a`) ;
- date de dernière modification (`m`) ;
- signature du fichier (par exemple son CRC16 (`6`)).

Fichier de configuration

Le fichier de configuration (par défaut `integre.cfg`) contient la liste des entrées dans le système de fichiers qui seront gérées par `integre`. Une entrée dans ce fichier de configuration est une ligne au format suivant :

$$+|- \textit{pathname} \quad [\textit{flag}] \quad [\# \textit{comment}]$$

ou

$$\# \textit{comment}$$

+pathname est le nom absolu d'une entrée dans le système de fichiers. Cette entrée est ajoutée dans la liste des fichiers devant être gérés par `integre`. Dans le cas d'un répertoire, l'ensemble des fichiers qu'il contient doit être géré par `integre`, et ce récursivement. Cependant, `integre` ne traverse jamais les liens symboliques.

-pathname est le nom d'une entrée qui est exclue de la liste des fichiers devant être gérés. Si cette entrée est un répertoire, ce répertoire et toute sa descendance son exclus.

flag indique les attributs devant être enregistrés dans la base et dont la modification doit être reportée. Les valeurs possibles de *flag* sont :

- **R** (read-only, valeur par défaut) Seul l'attribut `a` peut être modifié ;
- **L** (log file) Seul le contenu du fichier peut être modifié. En conséquence les attributs `sam6` peuvent varier ;
- **A** (all) Aucun des attributs ne peut être modifié.
- **N** (none) Tous les attributs peuvent être modifiés. Cependant, dans le cas d'un répertoire, les destructions et ajouts de fichiers seront reportés.
- **G** (growing file) Le contenu du fichier peut être modifié, mais la taille du fichier ne peut décroître. Les attributs `am6` peuvent varier ; l'attribut `s` doit croître.

comment est un commentaire ; il est ignoré.

Mode initialisation

Ce mode crée une base de données à partir du fichier de configuration donné en entrée. Cette base de données est rangée dans le fichier `integre.db`. Pour un fonctionnement sûr d'`integre`, il est nécessaire de ranger ensuite ce fichier sur un support à lecture seule. Ce transfert doit se faire manuellement par l'utilisateur.

Mode mise à jour

Ce mode met à jour la base de données `integre.db` pour les entrées spécifiées sur la ligne de commande. L'ancienne base de données est renommée avec une extension `.old`. Ces entrées de la ligne de commande doivent être identifiées dans le fichier de configuration. Ce mode autorise la mise à jour d'une base de données d'une manière contrôlée et sans avoir à reconstruire toute la base.

Mode test

Ce mode autorise la vérification de l'intégrité d'une configuration. `Integre` lit le fichier de configuration et génère une nouvelle base de données reflétant les fichiers actuels. Ensuite `integre` compare cette base de données et la base de données préexistante et reporte la liste des fichiers ajoutés ou supprimés ainsi que celle des fichiers modifiés.

En plus de la liste des fichiers, le fichier de configuration contient aussi la liste des attributs qui peuvent être changés sans qu'ils aient à être reportés. Chaque fichier qui diffère des informations préalablement sauvegardées dans la base de données est considéré comme modifié. Cependant seuls les attributs listés dans le fichier de configuration sont affichés ; les valeurs attendues et observées sont présentées.

Exemple d'utilisation

Nous suivons ici plusieurs étapes d'utilisation de `integre` :

– Création d'un fichier de configuration

```
+/etc            R  # /etc ne peut-être que lu
+/etc/passwd     L  # /etc/passwd peut être modifié (par exemple lors d'un
                    # changement de mot de passe
-/etc/X11           # Le repertoire /etc/X11 ne doit pas être géré
+/etc/X11/xinit  N  # Le contenu du repertoire /etc/X11/xinit n'est pas géré,
                    # cependant les destructions et créations de fichiers son
                    # reportées
```

– Création initiale de la base

```
$ integre -init -v
  @@ Lecture du fichier de configuration integre.cfg
  @@@@ Traitement de /etc/X11/xinit
  @@@@ Traitement de /etc/passwd
  @@@@ Traitement de /etc
  @@ Création de la base de données integre.db
  @@ Fin
```

– On désire modifier le fichier `/etc/group` qui est géré par `integre` en ¡¡ read-only ¿¿.

1. On vérifie l'intégrité de tous les fichiers gérés, en particulier pour vérifier celle de `/etc/group` :

   ```
   $ integre -q -test
   ```

2. On modifie le fichier `/etc/group`.

3. On vérifie à nouveau l'intégrité des fichiers :

   ```
   $ integre -q -test
     @@ Fichier modifié /etc/group
     @@@@ -rw-r--r-- 1 65 /etc/group
     @@@@ valeur     attendue             observée
     @@@@ date   m   Sep 12 10:45         Sep 26 10:17
     @@@@ taille s   290                  298
     @@@@ sign   6   7F98                 6FD7
   ```

4. On met à jour la base de données pour `/etc/group`

   ```
   $ integre -v -maj /etc/group
     @@ Sauvegarde de la base de données dans integre.db.old
     @@ Lecture du fichier de configuration integre.cfg
     @@ Traitement du fichier /etc/group
     @@ Ecriture de la base de données integre.db
     @@ Fin
   ```

5. Nouvelle vérification

   ```
   $ integre -q -test
   ```