

La Comptabilité UNIX System V

Acctcom

(Groupe GLACE)

Hervé Schauer

schauer@enst.fr

Introduction

Acctcom est le système de comptabilité (*accounting*) d'Unix System V. Il a été conçu pour permettre une facturation du temps CPU utilisé par chaque utilisateur d'un système Unix.

Le but ici, est d'utiliser le mécanisme d'*accounting* d'Unix comme système d'audit permettant une détection de piratages ou d'actions néfastes envers le système.

Principe de base

Unix enregistre toutes les extinctions de processus lancés sur le système. Ainsi lorsqu'à 15h12 l'utilisateur *user1* tape la commande `who`:

```
$ date
Fri May 26 15:12:35 HFE 1989
$ who am i
user1  tty12  May 26 14:50
```

Unix enregistre que *user1* connecté du terminal `tty12` a utilisé la commande `who` à 15h12. L'administrateur de la machine (*root*), pourra voir que *user1* a utilisé `who` à 15h12 ce jour la.

Mise en oeuvre

L'*accounting* est un outil à disposition de l'administrateur et de lui seul, il faut donc être sous *root* pour le mettre en oeuvre (*su* conseillé). L'*accounting* est mis en route par la commande `/etc/accton` suivie du nom de fichier dans lequel l'*accounting* sera enregistré, normalement `/usr/adm/pacct`:

```
# accton /usr/adm/pacct
```

Cette commande est utilisée au boot du système, par exemple en étant dans `/etc/rc`, ou éventuellement manuellement par l'administrateur. Un audit de tous les processus ayant été lancés est réalisé à leur mort dans le fichier indiqué d'*accounting* (généralement `/usr/adm/pacct`). Ce travail est réalisé dans le kernel (noyau) d'Unix. Le programme `/etc/accton` configure le noyau avec l'appel système `acct()`. Le lancement de l'*accounting* ne génère pas de processus `/etc/accton` visible par `/bin/ps`. On peut mettre dans `/etc/rc` (ou suivant le système `/etc/rc2` ou `/etc/rc.d/6/kinit` ou `/etc/rc2.d/S09acct`) les lignes suivantes afin de mettre en route l'*accounting* au boot automatiquement :

```
mv /usr/adm/pacct /usr/adm/opacct
> /usr/adm/pacct; chmod 644 /usr/adm/pacct
[ -x /etc/accton ] && /etc/accton /usr/adm/pacct
```

L'*accounting* n'est normalement utilisé qu'en mode *multi-user* (multi-utilisateurs), et initialisé juste avant le lancement des *demons* comme `/usr/lib/lpsched` pour l'imprimante.

Dans certains cas, Unix crée automatiquement des fichiers numérotés `/usr/adm/pacctNN` quand le fichier d'accounting dépasse une certaine taille. `/usr/adm/pacct` représente le fichier en cours de remplissage. Il est fondamental d'effacer régulièrement les fichiers d'accounting qui grossissent, afin que l'accounting ne provoque pas une saturation de l'espace disque disponible. Suivant l'utilisation du système et le nombre d'utilisateurs, le fichier d'accounting peut représenter plusieurs centaines de Kilo-octets par jour. Chaque enregistrement du fichier `/usr/adm/pacct` fait 32 octets. Un processus exécuté régulièrement grâce au mécanisme du *cron* qui va très régulièrement effacer les fichiers d'accounting est la meilleure solution. Ne pas indiquer des dates de lancement du *cron* en dehors des heures de fonctionnement du système, en particulier faire attention si la machine est arrêtée la nuit ou le week-end. L'accounting peut être stoppé en utilisant `/etc/accton` sans arguments, `/etc/acctoff` n'existe pas.

Utilisation

L'accounting ainsi lancé est exploitable par l'administrateur (*root*) avec la commande `/usr/bin/acctcom`, qui va lire le fichier d'accounting et produire un état de sortie sous forme ASCII. La forme du fichier d'accounting (`/usr/adm/pacct`) est décrite dans le fichier d'en-tête `<sys/acct.h>`.

Nous allons étudier la commande *acctcom* de fabrication d'états de sortie des fichiers d'accounting.

Syntaxe : **acctcom** **[[options][file]] ...**

Description : *acctcom* lit le fichier "file" et écrit sur la sortie standard les enregistrements sélectionnés. Si aucun fichier n'est spécifié, *acctcom* lit l'entrée standard. Si l'entrée standard est associée à un terminal ou à `/dev/null` (utilisation de "&" sous shell par exemple), alors *acctcom* lit `/usr/adm/pacct`. C'est le cas par défaut le plus courant. Si plusieurs fichiers sont indiqués, ils sont lus dans l'ordre dans lequel ils ont été mis en argument. Chaque ligne de l'état de sortie représente un processus;

acctcom

COMMAND			START	END	REAL	CPU	MEAN
NAME	USER	TTYNAME	TIME	TIME	(SECS)	(SECS)	SIZE(K)
#accton	root	?	14:30:10	14:30:10	0.05	0.05	30.80
sh	adm	?	14:30:08	14:30:09	1.95	0.49	26.29
sh	root	console	14:30:10	14:30:10	0.06	0.04	41.50
ncscstat	adm	?	14:30:09	14:30:11	2.31	0.24	40.25
#sh	adm	?	14:30:01	14:30:11	10.82	0.11	25.82
cp	adm	?	14:30:11	14:30:17	6.83	0.91	26.51
acctwtmp	adm	?	14:30:18	14:30:18	0.28	0.08	28.25
cp	adm	?	14:30:18	14:30:18	0.25	0.12	28.50
chgrp	adm	?	14:30:20	14:30:20	0.06	0.06	33.33
chown	adm	?	14:30:20	14:30:20	0.07	0.07	30.57
sh	adm	?	14:30:19	14:30:19	0.95	0.32	23.75
sleep	root	console	14:30:10	14:30:24	14.69	0.06	29.00
sh	root	console	14:30:25	14:30:25	0.12	0.02	51.00
ls	root	console	14:30:25	14:30:25	0.78	0.18	29.67
tail	root	console	14:30:25	14:30:25	0.83	0.10	29.00

```

sh      root      console 14:30:26 14:30:26 0.02 0.02 47.00
sed     root      console 14:30:26 14:30:26 0.61 0.15 44.00
cat     adm       ?       14:30:28 14:30:28 0.26 0.07 32.86
touch  root      console 14:30:28 14:30:28 0.33 0.12 26.67
cp      adm       ?       14:30:29 14:30:29 0.09 0.06 28.67
cp      adm       ?       14:30:30 14:30:30 0.14 0.06 28.67
chmod   adm       ?       14:30:30 14:30:30 0.04 0.04 30.00
.
.
.

```

Les informations suivantes sont donc obtenues sur chaque processus :

- Nom de la Commande sans le PATH absolu et sans les arguments. Le nom de la commande est précédé d'un dièse "#" si elle a été exécutée avec les droits de *root* (Première et 5ème lignes).
- Nom de l'utilisateur (si l'utilisateur *user1* fait *su guest* avec succès, c'est *guest* qui apparaîtra et pas *user1*).
- TTY duquel le processus a été exécuté. Si un processus n'est pas associé avec un terminal connu, un "?" est affiché. C'est toujours le cas des programmes lancés par *cron*.
- date de lancement du processus.
- date de mort du processus.
- durée: *real time* et *cpu time* en secondes.
- taille du processus en Kilo-octets.

optionnellement :

- un drapeau indiquant un *fork* (duplication de processus) suivi d'un *exec* (drapeau = 0) ou sans *exec* (drapeau = 1)
- l'état d'*exit* (sortie) du processus.

Quelques options utiles (voir votre manuel de référence pour toutes les options, *acctcom* a couramment plus d'une vingtaine d'options) :

- b Chaque fichier est lu par défaut dans l'ordre chronologique par date de mort des processus, "-b" lit à l'envers en commençant par la dernière commande enregistrée.
- f Affiche le drapeau *fork/exec* et le *system exit status*.
- l *tty* Sélectionne les processus attachés à la ligne */dev/tty*.
- u *user* Sélectionne les processus de l'utilisateur *user*, *user* pouvant être un *uid* (no d'utilisateur dans */etc/passwd*) ou un nom de login. Un dièse "#" désigne les processus exécutés avec les accès de *root* (exécuté par *root* ou *suid root*), et un "?" désigne les processus associés à des *uid* inconnus.
- g *group* Sélectionne les processus du groupe *group*, *group* pouvant être un *gid* (no de groupe) ou un nom de groupe.
- s *time* Sélectionne les processus à partir de *time*. *time* est de la forme, *hr* ou *hr:min* ou *hr:min:sec*.

- e *time* Sélectionne les processus antérieurs à *time*. Avec la même heure dans les options -s et -e, on sélectionne les processus qui ont existé à une date précise.
- d *mm/dd* Précédant une option -s ou -e, -d permet d'indiquer un mois et un jour, permettant de travailler sur des fichiers vieux de plus de 24 heures.
- n *pattern* Sélectionne les processus "matchants" avec *pattern*, *pattern* étant une expression régulière comme dans `/bin/ed`, excepté que + signifie une ou plusieurs occurrences.

L'utilisation de plusieurs options a l'effet d'un "et logique" entre celles-ci. Avec la sortie précédente on aura :

```
# acctcom -l console -u root -e 14:30:20
COMMAND          START      END      REAL    CPU    MEAN
NAME      USER      TTYNAME  TIME      TIME    (SECS) (SECS) SIZE(K)
sh         root      console  14:30:10  14:30:10  0.06   0.04   41.50
sleep     root      console  14:30:10  14:30:24  14.69  0.06   29.00
```

Acctcom ne reporte que les processus qui se sont terminés. Il faut utiliser `/bin/ps` pour voir les processus en cours. Une méthode pour ne pas faire apparaître un processus dans l'accounting est de ne jamais le laisser mourir, il sera alors visible par `/bin/ps`.

Exploitation pour de la surveillance

Nous allons exploiter l'accounting pour auditer le système. Il faut prendre l'habitude d'utiliser *acctcom* et de regarder toutes les commandes qui sont réalisées par les utilisateurs. Le mécanisme de *cron* provoque régulièrement l'exécution du shellscrip d'effacement des fichiers d'accounting. Avant que les fichiers soient détruits, il faut que l'administrateur n'hésite pas à s'envoyer par mail les informations qu'il juge utiles pour la surveillance de son site. Par exemple, il faut utiliser *acctcom* avec l'option `-l` pour sélectionner les lignes de connexions externes. L'administrateur doit connaître les utilisateurs de son système, dans la mesure du possible, c'est à dire suivant le nombre d'utilisateurs personnellement ou par type d'individus. Ceci permet de connaître les habitudes de ceux-ci, et de détecter une utilisation d'un accès au système par un intru.

La lecture de l'accounting peut permettre de détecter 3 types de problèmes :

- la tentative de pénétration du système par une personne ne possédant pas de compte
- l'utilisation d'un compte par une personne n'étant pas le propriétaire
- le changement de droits qu'un utilisateur aurait réussi à obtenir en particulier devenir *root*

Le premier cas se détecte par un nombre excessifs de `login` successifs ou une suite de `login/getty` successifs. Un tel accounting indique une tentative de pénétration du système par quelqu'un qui essaye une liste de couples *login/passwd*.

Détection de piratage de compte

Il est possible de manière empirique de soupçonner un piratage de compte utilisateur simplement par l'utilisation de ce compte lue dans l'accounting.

Pour surveiller un compte utilisateur particulier, l'utilisation de l'option `-u` n'est pas idéale car elle ne permet pas de voir les éventuels changements d'*uid* de l'utilisateur au cours des sessions. Il vaut mieux utiliser l'option `-l` en sélectionnant les *tty* de connexion de l'utilisateur. L'option `-l`, ou l'utilisation de la commande `grep` sur la sortie ascii de *acctcom*, permettent de découper les informations d'accounting. La sortie fabriquée ne mélange pas tous les processus de toutes les sessions et du système, et sélectionne ce que l'on cherche.

```
# acctcom -b -l tty02 -l tty03 | fgrep -v user1 | fgrep rlogin
```

COMMAND			START	END	REAL	CPU	MEAN
NAME	USER	TTYNAME	TIME	TIME	(SECS)	(SECS)	SIZE(K)
rlogin	user2	tty03	03:16:14	03:19:22	188.07	0.29	49.00
rlogin	user2	tty03	03:22:24	03:23:22	58.18	0.12	47.00

Recherche des utilisations de `rlogin`, en commençant par la plus récente, sur les lignes de connexion par modem `tty02` et `tty03`, sans regarder les `rlogin` de l'utilisateur `user1`.

Voici des éléments caractéristiques des actions de quelqu'un qui n'est pas sur un accès autorisé, actions qui permettent de découvrir un piratage par rapport à une utilisation normale de compte :

- Appels réguliers de `who` ou d'un équivalent toutes les 5 à 10 commandes.
- Utilisation abusive de toutes les commandes donnant des informations sur le système et ses utilisateurs, et éventuellement le réseau environnant :
 - `ps`, `pstat`, `who`, `w`,
 - `finger`, `whodo`, `rwho`, `rusers`, `ruptime`
 - `df`, `dfspace`, `ipcs`, `uname`, `hostname`, `netstat`, `arp` ...
- Appels de plusieurs `cat` successifs ayant chacun un temps d'exécution important, égal au temps de transfert des fichiers sensibles à la vitesse de la ligne. Le pirate cherche à transférer des fichiers comme `/etc/passwd`, `/etc/hosts`,... sur son propre système.
- Utilisation de plusieurs `find` peu après la connexion. `find` permet de faire la liste de tous les fichiers ayant des bits `set-uid` ou `set-gid`, de rechercher les sources (`*.c`, `*.h`, ...) présents sur le système ...
- Longue suite de `ls` et `more` ou `pg` (entrecoupés de `who`), pour explorer le système.
- Utilisation des outils de communications et de transferts entre systèmes distants: `cu`, `tip`, `kermit`, `r-commandes` (Internet)...
- Connexions ou tentative de connexions sur d'autres systèmes à partir du système local : `rlogin`, `telnet`, `rsh`,... répétés.
- Utilisations importantes de `ln`.
- Utilisation de `touch` sans utiliser `make`.
- Utilisation d'un logiciel qui n'est pas celui utilisé habituellement par l'utilisateur. Par exemple, utilisation comme éditeur de `vi` pour un habitué d'`emacs`, utilisation comme pageur de `more` pour un habitué de `pg`.
- Utilisation régulière de 2 éditeurs différents sur le même compte, il est rare d'utiliser à la fois `vi` et `emacs` par exemple.
- L'exécution de commandes anormales pour certains utilisateurs : compilateur pour un non-informaticien, `ipcs` et utilisation de débogueurs (`adb`, `sdb`,...) pour des programmeurs débutants,
 - `vnews` pour quelqu'un qui n'a pas encore découvert `mail`...

Cet ensemble de comportements permet d'isoler un piratage. Il est évident que d'autres mécanismes non décrits ici, tels que l'étude des heures de connexions ou des factures Transpac et France Telecom, permettent aussi de détecter des problèmes. Il faut recouper les informations, se faire ses propres shellscripts de contrôle adaptés à son environnement, et analyser régulièrement les résultats.

Détection de changement de privilèges

Enfin l'étude du résultat d'*acctcom* permet de détecter un passage *root*, et plus généralement l'utilisation d'autres *uid* et *gid* que ceux auxquels l'utilisateur est normalement autorisé.

La méthode normale pour acquérir des privilèges supérieurs au siens, et en particulier de *root*, est d'utiliser `/bin/su` pour changer d'*uid*, et `/bin/newgrp` pour changer de *gid*. Le fichier `/usr/adm/sulog` enregistre les tentatives échouées et réussies de changement d'*uid* par la commande `/bin/su`. `/usr/adm/sulog` est aisément destructible ou modifiable si on a obtenu les privilèges de *root*, et il n'audite pas les utilisations autres que `/bin/su` pour acquérir un nouvel *uid*. Lorsqu'un utilisateur réussit à devenir *root* illégalement, c'est en utilisant une erreur d'accès ou une erreur d'un programme existant sur le système. L'état de sortie de l'accounting par ligne *tty* permet de voir les changements d'*uid* illicites d'un utilisateur. Le champ "USER" indique de manière évidente le changement d'*uid*. S'il n'est pas précédé d'une utilisation de `su`, ou d'une commande munie d'un bit *setuid*, il y a piratage.

L'analyse des lignes de l'accounting qui précèdent le changement permettent de voir quel a été la méthode de passage *root* ou tout autre *uid*. Il faut corriger le problème ainsi détecté, ou faire appel à un spécialiste si cela n'est pas évident. L'analyse des lignes suivantes permet de voir les actions que l'utilisateur ayant acquis des droits importants illégalement a réalisés. En effet, son but est de modifier le système de manière à conserver pour toujours les privilèges acquis. Il faut remettre en place tout ce qu'il a pu modifier. C'est une phase très complexe, la meilleure solution pouvant être de tout réinstaller à partir des cartouches (ou disquettes) originales du système. Mais bien faire attention aux applications et fichiers récupérés à partir de sauvegarde, ils ont pu avoir été modifiés. Grâce à l'accounting, l'administrateur sait quel utilisateur a piraté, quand et comment.

La difficulté est plus grande si le pirate qui a acquis les privilèges de *root* a détruit les fichiers d'accounting afin de ne pas être détecté. Il est alors impossible de voir ce qui a pu se passer et la méthode utilisée. Mais il est impossible de détruire, arrêter ou modifier l'accounting sans que cela soit visible par l'administrateur. La dernière commande utilisée pour la modification de l'accounting ou sa remise en route est nécessairement elle-même audité, ainsi éventuellement que le shell qui était utilisé en dessous (car enregistré à sa mort). Les fichiers d'accounting étant d'une structure spécifique, il est nécessaire de faire un programme C pour les modifier plus subtilement qu'avec un simple `rm` qui laisse apparaître en première ligne de l'état de sortie *rm* utilisé par *root*. Enfin si l'accounting a été stoppé il y a évidemment piratage. Ainsi soit l'accounting est arrêté, soit il reste toujours au moins un processus aux accès *root* qui est bizarre tout seul en tête de l'accounting. Donc une analyse rigoureuse de ses états de sortie d'accounting permet à l'administrateur de détecter un changement illégal d'*uid* au jour le jour avant que les problèmes empirent.

Conclusion

Malgré les possibilités de détections vues ici, l'accounting d'Unix System V reste un outil limité qui n'a pas été conçu pour de l'audit. Ne sont pas audités les objets sur lesquels les processus s'appliquent, ni les arguments et options, ni le PATH complet, les noms des commandes enregistrées sont limités à 8 caractères, et l'accounting ne dit rien pour les programmes créés par les utilisateurs.

Dans certains cas l'accounting peut rendre de grands services faute de mieux, éviter des problèmes graves et permettre une surveillance du système qui peut sembler rébarbative et couteuse, mais très efficace pour un site sensibilisé à la sécurité et connecté sur l'extérieur.

Hervé Schauer