PARTIE 1:

ADMINISTRATION ET SUPERVISION DES SYSTEMES

DOSSIER PROJET : MASTER 2 RESEAUX et TELECOMMUNICATIONS

Présenté par:

Habib BALE

Thillo TALL

TABLE DES MATIÈRES

Administration et Supervision des Systèmes

RESUME

le concept d'**historique des événements** ou de **logging** désigne l'enregistrement séquentiel dans un fichier ou une base de données de tous les événements affectant un processus particulier (application, activité d'un réseau informatique…). Le **journal** (en anglais *log file* ou plus simplement *log*), désigne alors le fichier contenant ces enregistrements. Généralement datés et classés par ordre chronologique, ces derniers permettent d'analyser pas à pas l'activité interne du processus et ses interactions avec son environnement.

Un **journal** est donc la partie d'un système de fichiers journalisé qui trace les opérations d'écriture tant qu'elles ne sont pas terminées et cela en vue de garantir l'intégrité des données en cas d'arrêt brutal.

Être en mesure de choisir un système de journalisation. Savoir configurer et analyser les traces qui seront remontées. Un système qui ne garde aucune trace de l'activité est un système inutile pour l'administrateur. Ainsi, nous verrons pourquoi ce système est indispensable et ce qu'il permet de faire.

1. LE SYSTEME DE JOURNALISATION ET LES FICHIERS DE JOURNALISATION DU SYSTEME UBUNTU

1.1 LES SYSTEMES DE FICHIERS UNIX

En informatique, un système de fichiers (FS ou FileSystem en anglais) est une méthode d'organisation des données persistantes sur un médium durable (par exemple : disque dur, disquette, CDROM, clef USB ...).

UNIX gère les inodes (data structure) dans une table qui contient des informations telles que

- le propriétaire
- le groupe de fichiers
- les droits d'accès
- la date de modification
- le type de fichier

Sous UNIX et LINUX on trouve des partitions, ex:boot,var,tmp,home qui permettent d'organiser le systeme de fichiers. Au départ EXT2 sous UNIX, aujourd'hui EXT3 par défaut correspond a EXT2 plus le journal, qui a l'avantage de consigner tout ce que fait le système, et qui en cas de crash lui permet de démarrer plus rapidement sans erreurs. Les deux formats sont compatibles IL existe toute sorte de systèmes de fichiers

1.2 LES SPECIFICITES

Ext3 est le système le plus populaire sous Linux... et pourtant ce n'est pas le meilleur ! Si on l'a recommandé autrefois et s'il est aussi populaire aujourd'hui c'est en grande partie car il est compatible avec le format Ext2 qui était le standard avant et que le format Reiserfs (qui est plus récent) ne permet pas de transformer directement une partition, il faut donc la reformater. C'est contraignant si vous avez déjà des données mais si vous installez Ubuntu pour la première fois, choisissez ce format (à part si vous avez un ordinateur portable car ce format influe sur la consommation et donc sur votre batterie).

Quel que soit le format choisi pour votre Ubuntu, il présente les avantages d'être journalisé et de gérer les permissions sur les fichiers.

1.3 LA JOURNALISATION

Parlons maintenant de la journalisation. Un système de fichiers journalisé est plus fiable lorsqu'on entre dans le domaine du stockage des données. Il a été expliqué plus haut ce qui se produit réellement lorsqu'un fichier est enregistré sur un disque dur (une suite de 1 et de 0 est inscrite sur le disque) ; mais que se produit-il si l'écriture de la chaîne est interrompue avant son terme (ce qui se produit, par exemple, lors d'une coupure de courant) ? Votre fichier devient « corrompu », incomplet.

Un système de fichiers journalisé travaille de façon à prévenir une telle corruption : lors de la sauvegarde d'un fichier, au lieu d'écrire immédiatement sur le disque dur les données à l'endroit exact où elles devraient être enregistrées, le système de fichiers écrit les données dans une autre partie du disque dur et note les changements nécessaire dans un journal, et ensuite, en arrière-plan, il repasse chacune des entrées du journal et termine le travail commencé ; lorsque la tâche est accomplie, il raye la tâche de la liste.

Mais comment cela prévient-il la perte de données ? Prenons un exemple : disons que vous cliquez sur le bouton ENREGISTRER de votre logiciel de traitement de texte pour sauvegarder le fichier `foo.txt`. L'ordinateur écrit d'abord un « brouillon » de `foo.txt` dans une partie différente du disque dur et écrit le changement dans le journal du système de fichiers. Une fois cela effectué, l'ordinateur commence à retranscrire le fichier (LA SUITE *de* 1 ET *de* 0) à son endroit définitif sur le disque dur. Soudain, il survient une panne de courant ; alors la transcription du fichier est interrompue. Lorsque le courant revient, même si le « propre », la version finale de votre fichier est incomplète, vous possédez toujours votre brouillon dans le journal du système de fichiers ; l'ordinateur recommence donc la retranscription du fichier, écrasant les données corrompues.

Et si, par hasard, le courant était coupé lorsque l'ordinateur écrivait dans le journal, vous disposeriez toujours d'un brouillon précédemment écrit dans le journal pour récupérer votre travail.

« MAIS, direz-vous, IL EST BEAUCOUP PLUS AVANTAGEUX D'UTILISER UN *système de fichiers* JOURNALISE! C'EST BIEN PLUS SECURISE! POURQUOI UTILISERAIS-JE UN *système de fichiers* NON JOURNALISE? »L'utilisation d'un journal requiert des capacités de stockage importantes sur vos périphériques; ces systèmes de fichiers ne sont donc pas adaptés aux médias de faible capacité, telles les cartes mémoires (MEMORY STICKS) et les disquettes.

Ubuntu utilise le système de fichier Ext3 qui utilise un fichier journal :

- Celui-ci trace les dernières modifications à effectuer sur un système de fichiers.
- L'intérêt est de pouvoir plus facilement et plus rapidement récupérer les données en cas d'arrêt brutal du système d'exploitation (coupure d'alimentation par exemple…), alors que les partitions n'ont pas été correctement synchronisées et démontées.
- Sans un tel fichier journal, un outil de récupération de données après un arrêt brutal doit parcourir l'intégralité du système de fichier pour vérifier sa cohérence.
- Lorsque la taille du système de fichiers est importante, cela peut durer très longtemps (plusieurs heures) pour un résultat moins efficace (plus de perte de données).

Le système de comptabilité (accounting), le noyau et de nombreux démons génèrent des données et des informations qui se trouvent écrites dans des fichiers de journalisation (log files).

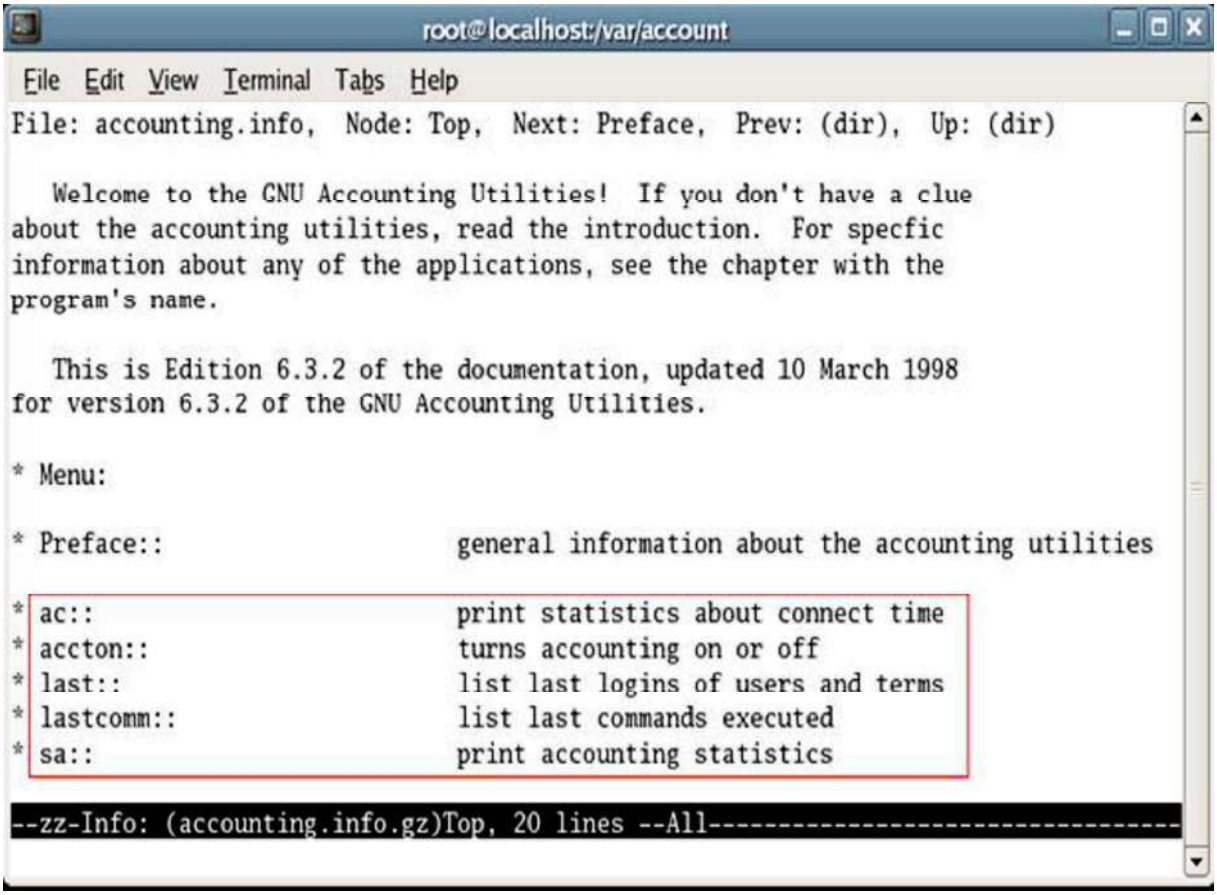
Les fichiers de journalisation permettent d'auditer le système et peuvent servir à :

- détecter les accès non autorisés au système (intrus)
- facturer les utilisateurs suivant les ressources utilisées (temps de connexion par exemple)
- diagnostiquer les problèmes de configuration.

La journalisation a un coût :

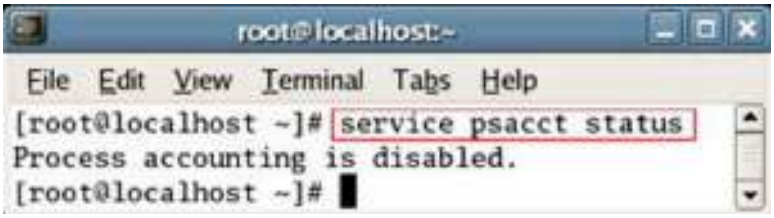
- en performance du système
- en espace disque
- en temps d'analyse et de traitement des données
- en maintenance

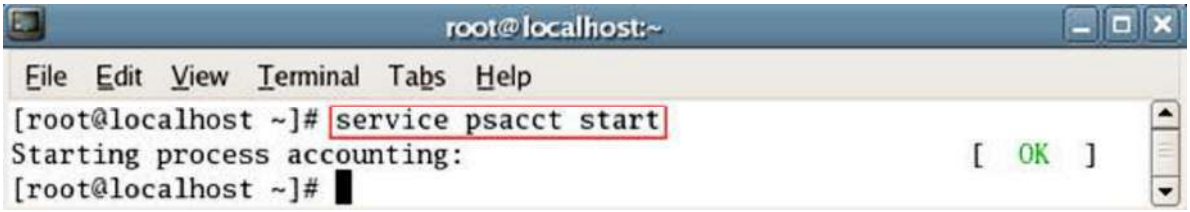
Par défaut, seuls les accès au système, aux imprimantes et les messages des démons sont journalisés. Les ressources utilisées par chaque processus (temps CPU, mémoire, etc.) ne sont comptabilisées que lorsque l'utilitaire accounting est utilisé (info accounting) pour plus de détails.



La comptabilité des ressources utilisées par les processus engendre d'énormes quantités de données et s'avère souvent inutile. Il est recommandé de ne l'utiliser que si c'est vraiment nécessaire.

```
# dpkg –l | grep psacct
```





1.4 FICHIERS DE JOURNALISATION

Les fichiers de journalisation peuvent être stockés dépendamment des familles (BSD et SYS V) et des constructeurs dans :

- /var/log (RedHat)
- /var/adm
- /usr/adm
- /var/run
- /etc

Les fichiers de journalisation ont tendance à croître et la plupart doivent être régulièrement purgés. On le faire de deux façons:

- Les remettre à zéro régulièrement. On peut simplement les effacer en les ayant aux préalables sauvegardés.
- Faire des rotations sur plusieurs jours. On peut par exemple conserver un journal sur 3 jours. Le script pour la rotation d'un fichier journal serait le suivant :

```
#!/bin/sh
rm     -f      journal.3
mv    –f       journal.2     journal.3
mv    –f       journal.1     journal.2
mv    –f       journal       journal.1
cat       /dev/null  >    journal
chmod        644          journal
```

- Certains fichiers ne doivent pas être purgés, tandis que pour d'autres certaines précautions doivent être prises.

1.5 TYPE DE JOURNALISATION

Il y a 2 types de journalisation :

- La journalisation faite directement par les démons.
- La journalisation faite par l'intermédiaire du démon syslogd

.

Journalisation via les démons

La journalisation faite directement par les démons concerne :

- les ressources utilisées par les processus.
- les accès au système.

Journalisation des ressources

Les ressources utilisées par les processus sont journalisées dans /var/account/pacct.

Journalisation des accès

Les accès au système sont conservés dans trois fichiers :

- utmp
- lastlog
- wtmp

1) utmp
Fichier binaire (situé dans /var/run) qui contient des informations sur les utilisateurs ayant une session active. Les informations contenues dans ce fichier sont affichables à l'aide de la commande who

```
# who -H
USER   LINE     LOGIN- TIME   FROM
root     tty1     May 26 10:10
habib    tty2     May 27 08:15
root     tty3     May 27 08:16
```

2) lastlog
Fichier binaire (situé dans /var/log) qui contient les informations sur la dernière session de chacun des utilisateurs du système. On peut accéder à l'information contenue dans ce fichier à l'aide des commandes lastlog et finger.

```
# lastlog
Username         Port     From     Latest
Root             tty3              Sun May 27 08:16:46 -0400 2001
bin                               **Never logged in**
daemon                            **Never logged in**
adm                               **Never logged in**
lp                                **Never logged in**
sync                              **Never logged in**
shutdown                          **Never logged in**
radvd                              **Never logged in**
halt                              **Never logged in**
mail                              **Never logged in**
news                              **Never logged in**
uucp                               **Never logged in**
operator                           **Never logged in**
games                             **Never logged in**
gopher                             **Never logged in**
ftp                                **Never logged in**
nobody                             **Never logged in**
vcsa                               **Never logged in**
mailnull                          **Never logged in**
rpm                                **Never logged in**
ntp                                **Never logged in**
rpc                                **Never logged in**
xfs                               **Never logged in**
gdm                                **Never logged in**
rpcuser                            **Never logged in**
nfsnobody                         **Never logged in**
nscd                              **Never logged in**
ident                              **Never logged in**
habib            tty2             Sun May 27 08:15:59 -0400 2001
apache                            **Never logged in**
usager2          pts/0 lune       Wed Apr 18 16:42:33 -0400 2001
u44              tty3              Wen may 1810:21:07 -0400 2001
```

```
# finger root
Login: root                                Name: root
Directory: /root                           Shell: /bin/bash
On since        Thu     May 26 10:10 (EDT) on tty1
On since        Fri     May 27 08:16 (EDT) on tty3 9 minutes
55 seconds idle
Mail   last read Sun    Jan 22 04:02 2005 (EST)
No Plan.
```

REMARQUE
Les fichiers /var/run/utmp et /var/log/lastlog ne doivent pas être purgés. Leur taille reste toujours faible.

3) wtmp

Fichier binaire (situé dans /var/log) qui contient un historique de toutes les sessions effectuées sur le système. Les informations contenues dans ce fichier sont affichables à l'aide des commandes last et ac.

```
# last
root     tty3     Sun May 27 08:16   still logged in
habib    tty2     Sun May 27 08:15   still logged in
root     tty1     Thu May 26 10:10   still logged in
root     tty1     Tue May 18 16:11 -  down (00:00)
root     tty1     Tue May 18 15:44 -  16:11 (00:27)
root     tty4     Wen May 18 10:25 -  down (01:14)
u44      tty3     Wen May 18 10:21 - down (01:18)
u44      tty3     Wen May 18 10:12 - 10:21 (00:08)
habib    tty2     Wen May 18 10:09 - 11:11 (01:01)
habib    tty2     Wen May 18 10:03 -  10:09 (00:06)
habib    tty2     Wen May 18 09:59 -  10:03 (00:04)
root     tty1     Wen May 18 09:57 - down (01:42)
reboot   system boot Wen May 18 09:53        (01:46)
root     tty1     Mon May 9 13:26 -  down (3+03:06)
root     pts/0 :0 Mon Mar 9 18:53 -  10:10 (15:17)
habib    pts/0 lune Sun Mar 8 22:03 -  22:04 (00:00)
21:22 - down (1+14:15)

wtmp begins Tue Jun 7 09:30:57 2011
```

2. FICHIERS DE JOURNALISATION DES SOUS-SYSTEMES

2.1 PRINCIPES DE FONCTIONNEMENT :

Afin de simplifier la programmation de la journalisation et de laisser un meilleur contrôle à l'administrateur, de nombreux sous-systèmes ne font pas la journalisation eux-mêmes mais effectuent des appels système au démon syslog (openlog(), syslog(), closelog()).
Les évènements qu'un programme envoie au démon syslog sont identifiés par 3 éléments :

- Le sous-système (**facility**)
- Le niveau (**level**)
- L'action à réaliser

1) Le sous-système permet d'identifier quelle partie du système d'exploitation envoie le message. Les valeurs possibles ont : **auth, authpriv, cron, daemon, kern, lpr, mail, news, syslog, user, uucp et local0 à local7.**

2.2 LES « FACILITIES »

– **user** : messages des processus utilisateurs

– **kern** : messages du noyau (et du firewall!)

– **daemon** : messages des services

– **mail** : messages des services de mail (pop/smtp)

– **news** : messages des services de news

– **auth** : messages d'authentification et de login

– **cron** : messages des planificateurs de tâches

– **local0-7** : des facilities en libre services

– **lpr / uucp :** imprimantes

2) Le niveau permet d'identifier la sévérité de l'évènement. Les valeurs possibles sont (du plus important au moins important) : emerg, alert, crit, err, warning, notice, info et debug.
A chaque facilité est associé un niveau de trace, du plus grave au moins important:

2.3 NIVEAUX / PRIORITÉS

Niveaux possibles de sévérité/criticité des erreurs (« level ») sont dans l'ordre :

– **emerg(ency):** Erreurs diffusées à tout le monde. Le système ne sera plus en état de fonctionner

– **alert:** Erreurs qui doivent être corrigées immédiatement.

– **crit(ical):** Erreurs critiques à corriger le plus tôt possibles (erreurs de périphériques)

– **err(or):** Erreurs de niveau « standard »

– **warning:** Erreurs légères

– **notice, info, debug:** Informations avec plus ou moins de détails ou d'importance

3) Après avoir défini une facilité et ses niveaux, il faut y ajouter une action :

a. Ecrire les traces dans **fichier** ***(*** par exemple **/var/log/messages*)***

b. Rediriger les traces du vers la **`machine`** (**@machine**)

c. Envoyer les messages vers l'écran de l'utilisateur s

3. JOURNALISATION DES ACCÈS

1) Les commandes ou démons qui demandent des logins et des mots de passe (su, login, getty, etc) effectuent leur journalisation par syslogd au niveau du sous-système **auth**.

2) Pour ce sous-système, il peut être intéressant de journaliser les messages de sévérité supérieure ou égale à *notice* dans un fichier séparé (/var/log/authlog par exemple) en utilisant dans /etc/syslog.conf une entrée de la forme : **auth.notice /var/log/authlog**

3) Ce fichier contiendra :

- Les **logins** comme root (sévérité **notice**)
- Les utilisations de la commande **su** (réussite avec sévérité **notice** et échec avec sévérité **crit**)
- Les échecs lors de 5 tentatives successives infructueuses (avec sévérité **crit**)

4. SYSLOGD : CONFIGURATION DU DEMON

Configuration de /etc/syslog.conf

Le fichier « **/etc/syslog.conf »** est le principal fichier de configuration du démon « **syslogd** ».

Syntaxe générale

Chaque ligne de ce fichier indique le type du message (appelé Service ou Facility en anglais), le niveau de gravité (appelé Priorité) et sa Destination (fichier, terminal,...).

Par exemple, la ligne suivante permet d'envoyer les messages d'information des programmes de messagerie dans le fichier « mail.info » :

mail.info /var/log/mail.info

Chaque ligne est de la forme :

Service.Priorité Destination

• Service correspond au type de programme (Démon, Noyau,...) et doit être l'un de ces mots-clés : auth, authpriv, cron, daemon, kern, lpr, mail, mark, news, security (identique à auth), syslog, user, uucp et local0 à local7.

• Priorité représente le niveau de gravité du message et doit être l'un de ces mots clés : debug, info, notice, warning, warn (identique à warning), err, error (identique à err), crit, alert, emerg, panic (identique à emerg). ATTENTION : En indiquant un niveau, syslog enverra les messages de ce niveau et tous les messages des niveaux plus importants. Donc en mettant « debug », syslog enverra tous les messages (de debug à panic).

• Destination représente la destination du message est peut être un chemin vers un fichier texte (ex : /var/log/mail.info) ou le nom d'une console pour envoyer les messages à l'écran (ex : /dev/tty8) ou un autre serveur (ex : @MonAutreServeur).

Remarque : Chaque programme détermine les « Services » et les « Priorités » qu'il utilise et il est rarement possible de les modifier. Par exemple, les programmes Postfix et Fetchmail utilisent la Service « mail »

Le signe « ; » pour séparer plusieurs « Services »

Il est possible de mettre plusieurs services pour une même destination en utilisant le « ; » :

Service1.Priorité1; Service2.Priorité2 Destination

Le signe « , » pour plusieurs « Services » avec une même « Priorité »

Il est possible d'indiquer plusieurs « Services » pour le même niveau de « Priorité » avec le signe « , ». La ligne suivante envoie dans le fichier « mail_news » les messages en provenance des « Services » « mail » et « new » dont la « Priorité » est supérieure ou égale à « info ». :

mail,news.info /var/log/mail_news

Le signe « = » pour ne traiter que la « Priorité » indiquée

Il est possible de n'envoyer que les messages d'une seule « Priorité » (sans les « Priorités » de niveaux supérieures) avec le singe « = ». Exemple :

mail.=err /var/log/mail.err

Le signe « * » pour traiter toutes les « Services »

La commande suivante permet d'envoyer les messages d'erreurs de toutes les « Services » dans le fichier « error »

***.error /var/log/error**

Le signe « * » pour traiter toutes les « Priorités »

Le signe « * » permet d'indiquer que l'on souhaite toutes les « Priorités » (Le résultat est le même qu'en mettant « debug », mais c'est plus lisible). Exemple :

mail.* /var/log/mail

Le signe « * » pour envoyer des messages sur toutes les consoles ouvertes

La ligne suivante, permet d'envoyer tous les messages d'erreurs sur toutes les consoles ouvertes grâce au signe « * » placé en destination à la fin de la ligne :

***.alert ***

Le signe « ! » pour exclure les niveaux supérieurs ou égaux à la « Priorité » indiquée

Le signe « ! » permet d'indiquer que l'on souhaite exclure le niveaux indiqué et tous les niveaux supérieurs à celui indiqué. La ligne suivante, permet d'envoyer dans le fichier « mail » tous les messages sauf ceux supérieurs ou égaux à la « Priorité » « warn » :

mail.*;mail.!warn /var/log/mail

La ligne suivante permet d'envoyer dans le fichier « mail » tous les messages supérieurs ou égaux à la « Priorité » « notice » et inférieure à la « Priorité » « crit » :

mail.notice;mail.!crit /var/log/mail

Le signe « - » pour améliorer les performances en écriture

Le signe « - » est utilisé devant les chemins de fichiers les moins critiques pour améliorer les performances en écriture au risque de perdre des données en cas de crash du système (pas de synchronisation des fichiers). Cette ligne enregistre tous les messages de la « Service » « mail » dans le fichier « mail » :

mail.* -/var/log/mail

Le signe « \ » pour écrire une instruction sur plusieurs lignes

La commande suivante écrite sur deux lignes grâce au signe « \ », permet d'envoyer tous les messages de mails ou de news dans le fichier « mail_news » :

mail.*;

news.* /var/log/mail_news

Envoyer les logs dans une console

La ligne suivante permet d'envoyer tous les logs dans la console « tty8 » (CTRL+ALT+F8) :

***.* /dev/tty8**

Les lignes suivantes permettent d'envoyer les logs des mails dans le fichier « mail » et en même temps sur la console « tty8 »

mail.* /var/log/mail

mail.* /dev/tty8

Le signe @ pour envoyer les logs sur un autre serveur

La ligne suivante permet d'envoyer tous les logs sur un autre serveur nommé « pgdebian » :

***.* @pgdebian**

Redémarrer le démon

A chaque modification du fichier « /etc/syslog.conf », il faut redémarrer le démon :

/etc/init.d/sysklogd restart

Lors du démarrage, le démon créera les nouveaux fichiers de logs, si ceux-ci n'existent pas.

Mettre syslogd à l'écoute du réseau

Pour pouvoir centraliser les logs en provenance d'autres serveurs ou routeurs, il faut mettre le démon « syslogd » à l'écoute du réseau.

Pour cela, il faut ajouter le paramètre « -r » sur la commande de démarrage du démon.

Sur Debian testing, il faut modifier le fichier « /etc/default/syslogd » et renseigner la variable « SYSLOGD » comme indiquée ci-dessous :

SYSLOGD="-r"

Il faut aussi configurer les routeurs ou les autres serveurs pour que ceux-ci exportent leurs logs vers le serveur principal.

Exporter des logs d'un autre poste sous Linux

Pour exporter tous les logs d'un poste vers un serveur appelé par exemple « ubuntu », il suffit d'ajouter la ligne suivante dans le fichier « /etc/syslog.conf » de ce poste :

***.* @ubuntu**

La ligne suivante permet de n'envoyer que les messages d'erreurs :

***.notice{{ }}@ubuntu**

5. FICHIERS DE LOGS

5.1 LES FICHIERS DE LOGS IMPORTANTS SOUS LINUX SONT :

– /var/log/messages : messages du système

– /var/log /syslog : fichier de log important !

– /var/log /auth.log : tout ce qui est authentification

– /var/log /boot.log : log du démarrage du système

– /var/log /kernel/* : tous les messages du noyau

– /var/log /daemons/* : tous les messages des services

– /var/log/secure : logins, ssh, su, ftp

– /var/log/xferlog : les transferts ftp

– /var/log/dmesg : le début de la séquence de boot

– /var/log/debug : Debugging log messages

– /var/log/dmesg : Linux kernel ring buffer log

– /var/log/dpkg.log : All binary package log includes package installation and other information

– /var/log/faillog : User failed login log file

– /var/log/lpr.log : Printer log file

– /var/log/mail.* : All mail server message log files

– /var/log/mysql.* : MySQL server log file

– /var/log/user.log : All userlevel logs

– /var/log/xorg.0.log : X.org log file

– /var/log/apache2/* : Apache web server log files directory

– /var/log/lighttpd/* : Lighttpd web server log files directory

– /var/log/fsck/* : fsck command log

– /var/log/apport.log : Application crash report / log file– /var/log/httpd/

• access_log : requetes http

• error_log : erreurs dans les requêtes

– ….

ATTENTION : La structure du /var/log varie suivant les systèmes et les administrateurs !!!!

5.2 VISUALISER LES FICHIERS DE LOGS SUR UN TERMINAL

Utiliser les commandes : tail, more, less and grep.

tail -f /var/log/apport.log

more /var/log/xorg.0.log

cat /var/log/mysql.err

less /var/log/messages

grep -i fail /var/log/boot

La commande suivante permet de voir les 10 dernières lignes du fichier « syslog »

 # tail /var/log/syslog

La commande suivante, permet de voir les 200 dernières lignes :

 # tail -200 /var/log/syslog

La ligne suivante permet de voir les 10 dernières lignes avec une actualisation en temps réel :

 # tail -f /var/log/syslog

Cette commande retourne les lignes du fichiers « syslog » contenant la chaîne de caractères « LeMotif » :

 # cat /var/log/syslog | grep -i LeMotif

En combinant les deux commandes précédentes, il est possible par exemple d'avoir en temps réel, les logs de fetchmail :

 # tail -f /var/log/syslog | grep fetchmail

Pour avoir uniquement les logs du noyau Linux au démarrage du poste :

 # dmesg

6. LA ROTATION DES LOGS AVEC LOGROTATE

logrotate permet :

- d'éviter l'explosion des logs et la saturation des disques
- de sauvegarder et de compresser les logs
- de définir un délai de conservation des informations
- d'effacer automatiquement les données trop anciennes

6.1 DESCRIPTION

logrotate est conçu pour faciliter l'administration des systèmes qui génèrent un grand nombre de journaux. Il automatise la permutation, la compression, la suppression, et l'envoi des journaux. Chaque journal peut être traité quotidiennement, hebdomadairement, mensuellement, ou quand il devient trop volumineux.

Normalement, logrotate est lancé comme un travail quotidien de cron. Il ne modifie pas un journal plusieurs fois dans la même journée à moins que le critère ne soit basé sur la taille du journal et que logrotate ne soit lancé plusieurs fois chaque jour, ou à moins que l'option **-f** ou **-**force ne soit utilisée.

Un nombre variable de fichiers de configuration peut être donné en ligne de commande. Les derniers fichiers de configuration peuvent outrepasser les options données dans les précédents, ainsi l'ordre dans lequel les fichiers de configuration de logrotate sont listés est important. Normalement, il faudrait utiliser un unique fichier de configuration qui inclut tous les autres fichiers de configuration nécessaires. Regardez ci-dessous pour plus d'informations sur la façon d'utiliser la directive *include* dans ce but. Si un répertoire est donné en ligne de commande, chaque fichier de ce répertoire est utilisé comme fichier de configuration.

6.2 OPTIONS

-d

Passe en mode débogage et implique **-v**. En mode débogage, aucun changement ne sera fait sur les journaux ou sur le fichier d'état de logrotate.

-f, --force

Indique à logrotate de forcer la permutation, même si il n'estime pas que c'est nécessaire. Quelques fois, cela est utile après ajout de nouvelles entrées à logrotate, ou si d'anciens journaux on été supprimés à la main, les nouveaux fichier seront créés, et la journalisation continuera correctement.

-s, --state <fichier d'état>

Indique à logrotate d'utiliser un autre fichier d'état. Cela est utile si logrotate est lancé par différents utilisateurs pour des groupes de journaux distincts. Le fichier d'état par défaut est **/var/lib/logrotate.status**.

--usage

Affiche un court résumé du manuel, le numéro de version, et les informations légales.

6.3 FICHIER DE CONFIGURATION

logrotate lit tout à propos des journaux qu'il est censé traiter à partir de la série de fichiers de configuration spécifiée en ligne de commande. Chaque fichier de configuration peut définir des options globales (les options locales outrepassent les globales, et les dernières définitions outrepassent les précédentes) et précise le journal à permuter. Un fichier de configuration simple ressemble à :

```
# exemple de fichier de configuration de logrotate
errors sysadmin@my.org
compress

/var/log/messages {
    rotate 5
    weekly
    postrotate
                                /sbin/killall -HUP syslogd
    endscript
}

"/var/log/httpd/access.log" {
    rotate 5
    mail www@my.org
    errors www@my.org
    size=100k
    postrotate
                                /sbin/killall -HUP httpd
    endscript
}

/var/log/news/* {
    monthly
    rotate 2
    missingok
    errors newsadmin@my.org
    postrotate
                                kill -HUP `cat /var/run/inn.pid`
    endscript
    nocompress
}
```

Les premières lignes définissent les options globales ; chaque erreur apparaissant durant le traitement des journaux est envoyée à sysadmin@my.org et les fichiers sont compressés après être permutés. Remarquez que les commentaires peuvent apparaître n'importe où dans le fichier de configuration à partir du moment où le premier caractère non blanc de la ligne est un #.

La section suivante du fichier de configuration décrit comment traiter le journal */var/log/messages*. Le journal passera par cinq rotations hebdomadaires avant d'être supprimé. Une fois que le journal aura été permuté (mais avant que l'ancienne version du journal ne soit compressée), la commande */sbin/killall -HUP syslogd* sera exécutée.

La section suivante décrit les paramètres pour */var/log/httpd/access.log*. Il est permuté dès que sa taille dépasse 100 ko, et les anciens journaux sont envoyés (sans compression) à www@my.org après être passés par 5 rotations, au lieu d'être supprimés. De la même

manière, toutes les erreurs apparaissant durant le traitement du journal sont envoyées à www@my.org (outrepassant la directive globale *errors*). Remarquez que les doubles guillemets autour du nom de fichier au début de cette section permettent à logrotate de permuter des journaux dont le nom comprend des espaces.

La dernière section décrit les paramètres pour tous les fichiers dans */var/log/news*. Chaque fichier est permuté sur une base mensuelle, et les erreurs sont envoyées à newsadmin@my.org. Ceci est considéré comme une unique directive, et si des erreurs se produisent pour plus d'un fichier elle sont envoyées dans un unique message. Dans ce cas, les journaux ne sont pas compressés.

Vous trouverez ici plus d'informations sur les directives que vous pouvez inclure dans un fichier de configuration de logrotate :

compress

Compresse les anciens journaux avec **gzip**. Voir aussi **nocompress**.

copytruncate

Tronque le journal original en place après en avoir créé une copie, au lieu de déplacer l'ancien journal et optionnellement d'en créer un. Cela peut être utilisé quand il n'est pas possible de demander à un programme de fermer son journal, et par conséquent que l'écriture (l'ajout) puisse continuer sur le précédent journal. Remarquez qu'il existe une très courte période entre la copie du fichier et sa troncature, aussi des données journalisées peuvent être perdues. Quand cette option est utilisée, l'option **create** est sans effet, attendu que le journal reste en place.

create *mode propriétaire groupe*

Immédiatement après la permutation (avant que le script **postrotate** ne soit exécuté), un journal est créé (du même nom que le journal juste permuté). *mode* précise les permissions du journal en octal (de la même façon que pour **chmod(2)**), *propriétaire* précise le nom de l'utilisateur propriétaire du journal, et *group* précise le groupe auquel appartient le journal. Chacun des attributs du journal peut être omis, auquel cas ces attributs omis pour le nouveau journal prendront les même valeurs que pour le journal original. Cette option peut être désactivée en utilisant l'option **nocreate**.

daily

Les journaux sont permutés chaque jour.

delaycompress

Reporte la compression du journal précédent au prochain cycle de permutation. Ceci n'a un effet qu'utilisé en combinaison avec l'option **compress**. Elle peut être utilisée quand il n'est pas possible de demander à un programme de fermer son journal et qu'il puisse par conséquent continuer à écrire pour un moment dans le journal précédent.

errors *adresse*

Toutes les erreurs se produisant durant le traitement des journaux sont envoyées à l'adresse donnée.

extension *ext*

Les journaux se voient attribuer le suffixe *ext* après permutation. Si la compression est utilisée, le suffixe de compression (normalement **.gz**) apparaît après *ext*.

ifempty

Permute le journal même s'il est vide, outrepassant l'option **notifempty** (ifempty est l'option par défaut).

include *fichier_ou_répertoire*

Lit le fichier donné en argument comme s'il était inclus littéralement à l'endroit où la directive **include** apparaît. Si un répertoire est donné, la plupart des fichiers de ce répertoire sont lus avant de continuer le traitement. Les seuls fichiers ignorés sont les fichiers non réguliers (tels que les répertoires et les tubes nommés) et les fichiers dont les noms se terminent par une des extensions taboues, comme précisé par la directive **tabooext**. La directive **include** ne devrait pas apparaître dans la définition d'un fichier journal.

mail *adresse*

Quand un journal est permuté, il est envoyé à *adresse*. Si aucun mail ne devrait être généré pour un journal particulier, la directive **nomail** peut être utilisée.

mailfirst

Lors de l'utilisation de la commande **mail**, envoie le journal juste permuté, au lieu d'envoyer le journal sur le point d'expirer.

maillast

Lors de l'utilisation de la commande **mail**, envoie le journal sur le point d'expirer, au lieu d'envoyer le journal juste permuté (ceci est l'option par défaut).

missingok

Si le journal est manquant, continue avec le suivant sans produire de message d'erreur. Voir aussi **nomissingok**.

monthly

Les journaux sont permutés la première fois que **logrotate** est lancé dans le mois (c'est normalement le premier jour du mois).

nocompress

Les anciennes versions des journaux ne sont pas compressées avec **gzip**. Voir aussi **compress**.

nocopytruncate

Ne tronque pas le journal original en place après avoir créé une copie (ceci outrepasse l'option **copytruncate**).

nocreate

Les nouveaux journaux ne sont pas créés (ceci outrepasse l'option **create**).

nodelaycompress

Ne reporte pas la compression du journal précédent au prochain cycle de permutation (ceci outrepasse l'option **delaycompress**).

nomail

N'envoie aucun ancien journal.

nomissingok

Si un journal n'existe pas, produit une erreur (ceci est la valeur par défaut).

noolddir

Les journaux sont permutés dans le répertoire où ils résident normalement (ceci outrepasse l'option **olddir**).

notifempty

Ne permute pas le journal s'il est vide (ceci outrepasse l'option **ifempty**).

olddir *répertoire*

Les journaux sont déplacés dans *répertoire* pour permutation. Le *répertoire* doit être sur le même périphérique physique que le journal en cours de permutation. Quand cette option est utilisée, toutes les anciennes versions des journaux terminent dans *répertoire*. Cette option peut être outrepassée par l'option **noolddir**.

postrotate/endscript

Les lignes entre *postrotate* et *endscript* (chacun devant apparaître sur une ligne isolée) sont exécutées après permutation du journal. Ces directives doivent apparaîtrent dans la définition d'un journal. Voir aussi **prerotate**.

prerotate/endscript

Les lignes entre *prerotate* et *endscript* (chacun devant apparaître sur une ligne isolée) sont exécutées avant permutation du journal. Ces directives doivent apparaîtrent dans la définition d'un journal. Voir aussi **postrotate**.

rotate *nombre*

Les journaux sont permutés <nombre> fois avant d'être supprimés ou envoyés à l'adresse précisée dans une directive **mail**. Si *nombre* est 0, les anciennes versions sont supprimées au lieu d'être permutés.

size *taille*

Les journaux sont permutés quand leur taille dépasse *taille* octets. Si *taille* est suivi par **M**, la taille est supposée être en mégaoctets. Si **k** est utilisé, la taille est en kilooctets. Ainsi **size 100**, *size 100k*, et *size 100M* sont tous valides.

tabooext [+] *liste*

Change la liste courante des extensions taboues (voir la directive **include** pour les informations sur les expressions taboues). Si un + précède la liste d'extensions, la liste courante des expressions taboues est augmentée, sinon elle est remplacée. Au démarrage, la liste des extensions taboues contient .rpmorig, .rpmsave, .v et ~.

weekly

Les journaux sont permutés si le jour courant est plus petit que le jour de la dernière permutation ou si plus d'une semaine s'est écoulée depuis la dernière permutation. Ceci est normalement identique à permuter les journaux le premier jour de chaque semaine, mais fonctionne mieux si *logrotate* n'est pas lancé chaque nuit.

7. MONITORING DES FICHIERS DE LOGS AVEC LOGWATCH

Logwatch :

- Analyse automatiquement les traces
- Repère les messages importants
- Affiche ou envoie par mail un rapport
- Utilise un script (perl) pour chaque service reconnu dans /etc/log.d/scripts/
- Typiquement lancé tous les jours par cron

7.1 DESCRIPTION

Logwatch est un analyseur de journaux système (appelés logs) pour Unix. Ce dernier est entièrement personnalisable à partir du fichier de configuration. Il permet de recevoir quotidiennement un mail récapitulatif de l'utilisation d'un serveur en analysant les logs situés dans /var/log (paquets installés, désinstallés, ou mis à jour ; mails envoyés ; utilisation des disques durs ; attaques potentielles reçues par le serveur etc.).

7.2 INSTALLATION

Note : Pour pouvoir utiliser Logwatch, une messagerie est nécessaire. Dans notre cas nous utiliserons Postfix.
Attention : Si vous utilisez deux machines identiques sous VMware, il faut changer le nom d'hôte de
l'une des deux sous peine de non réception du mail de la part de Logwatch.
Faites *Gedit /etc/hostname* ; indiquer le *nouveau_nom_hote* puis redémarrer la machine.
Afin de pouvoir procéder à l'installation de cet outil dans la salle C107, nous devons définir la variable
d'environnement http_proxy.

Ouvrez un terminal et tapez la commande suivante :

```
#export http_proxy=http://192.168.196.246:3128
```

Sur la majorité des distributions, la commande suivante fonctionne et permet d'installer « Logwatch » directement à partir d'Internet :

```
#sudo apt-get install logwatch
```

7.3 CONFIGURATION

Maintenant que l'installation est terminé, nous pouvons passé à sa configuration. Le nom du fichier permettant ceci se nomme « logwatch.conf » :

gedit /usr/share/logwatch/default.conf/logwatch.conf

Dans ce fichier, nous allons modifier les champs suivants :
Ligne n°34 : Nous indiquons l'adresse mail vers laquelle nous souhaitons recevoir les rapports de logs :

remplacer **MailTo = root** *par* **MailTo = mon@adresse.com**

Ligne n°65 : Indication du niveau de détail (Low, Medium, High ou un chiffre entre 1 et 10 du moins détaillé au plus détaillé) :

Detail = High

Ligne n° 79 : Indication des services que l'on souhaite voir dans le rapport :

nous laissons **Service =All** pour tout autoriser puis, s'il y a des services que vous ne désirez pas voir apparaître dans votre future analyse de logs, il suffit de les soustraire de cette manière :
Service = "-zz-network"
Service = "-zz-eximstats"
Service = "-iptables"

Notes : En procédant ainsi, les journaux systèmes seront plus simple à analyser ; dû à une meileure clarté. La liste des services surveillés par Logwatch se situent dans le répertoire suivant :
/usr/share/logwatch/default.conf/services

Ligne n° 109 : Indication du type de messagerie utilisé. Par défaut, Logwatch utilise Sendmail. En utilisant pas Postfix, nous pouvons laisser la ligne telle quelle :

mailer = "sendmail -t"

Ligne n° 48 : Nous désirons recevoir les informations uniquement par mail et non sur le terminal. Pour cela, nous allons changer la valeur suivante au champ Print :

Print = No

Nous devons également ajouter une ligne au fichier de configuration de Postfix :

gedit /etc/postfix/main.cf

Ajouter l'origine sous la forme suivante :

myorigin=logwatch.fr

7.4 UTILISATION

Paramètres de logwatch:

--detail level : niveau de détail (ou high, low, …)
--service : un de ceux du répertoire /etc/log.d/scripts (ou 'All')
--logfile : un des fichiers de logs (ou 'All')
--archives : explorer aussi les archives des logs (cf logrotate)
--range : période de temps (p. ex. : yesterday, all)
--print, --mail : affichage sur stdout, ou mail

Ex : #logwatch --service sshd --range all --detail high --print --archives

La configuration étant effectuée, nous pouvons taper la commande suivante pour vérifier le bon fonctionnement :

```
#logwatch --range=Today --print
```

Normalement, l'analyse des journaux systèmes devrait s'afficher sur le terminal. Nous pouvons maintenant utiliser la commande suivante afin de recevoir un mail :

```
# logwatch --range=Today
```

Vous devriez avoir reçu un mail.
Le script de lancement (00logwatch) est dans /etc/cron.daily
Désormais, vous recevrez chaque jour ce type de mail. Logwatch est un outil très pratique pour les administrateurs réseaux et systèmes car il procure un gain de temps relativement important.