Gestion des logs et cron

Gestion des comptes

TD 8 – Jérémy Briffaut

1 Gestion des logs avec syslog-ng

1.1 Syslog-ng

Le daemon Syslog-ng (ng pour nouvelle génération) est compatible avec son ainé syslog, il lui ajoute une grande souplesse dans la gestion des logs. Il est ainsi possible de décrire la source du log qui peut être le daemon lui-même, un fichier, pipe, réseau. La destination des journaux est elle aussi variée : fichier, tcp, usertty, program. Le filtre sélectionnera les lignes de logs qui nous intéresseront et pour finir le log composé d'une source , d'une destination et d'un filtre assemblera le tout. C'est grace à ce dernier que nous pourrons affiner le filtrage de nos logs.

Dans le fichier /etc/syslog-ng/syslog-ng.conf sont déjà définis de nombreuses source, destinations et log :

```
source src { unix-stream("/dev/log"); internal(); pipe("/proc/kmsg"); };

destination messages { file("/var/log/messages"); };
destination console_all { file("/dev/tty12"); };

log { source(src); destination(messages); };
log { source(src); destination(console_all); };
```

question 1 : quelle est la configuration par défaut de gestion des logs ?

1.2 Ajout d'un filtre

Nous allons ajouter un filtre pour centraliser les traces d'authentification authorisées ou refusées pour le serveur ssh.

Nous commençons par ajouter une nouvelle destination :

```
destination d_ssh { file("/var/log/auth-sshd.log"); };
```

Le filtre ' **f_ssh_login_attempt** ' filtre les messages contenant '**failure**' ou '**opened**' générés par le daemon sshd :

```
filter f_ssh_login_attempt {
   program("sshd.*")
   and match("failure|opened");
};
```

La source que nous utiliserons est '**src**', elle représente les fichiers gérés par syslog-ng (voir sa définition dans le fichier de conf).

```
log {
 source(src);
 filter(f_ssh_login_attempt);
 destination(d_ssh);
};
```

Redémarrez ensuite votre service syslog-ng et testez cette configuration en testant avec deux sessions ssh (une réussie et une qui échoue)

Votre fichier de logs doit contenir les traces suivantes :

cat /var/log/auth-sshd.log

Oct 25 15:52:40 briffaut sshd(pam_unix)[5025]: authentication failure; logname= uid=0 euid=0 tty=ssh ruser= rhost=172.16.95.1 user=root

Oct 25 15:52:42 briffaut sshd[5020]: error: PAM: Authentication failure for root from 172.16.95.1

Oct 25 15:52:56 briffaut sshd(pam_unix)[5028]: session opened for user root by root(uid=0)

question 2 : modifier votre configuration pour générer deux fichiers de traces (1 pour les sessions authorisées, et 1 pour les sessions refusées)

question 3 : Etudiez le fichier suivant. Quelle est la différence avec votre configuration ? Dans quelles circonstances faut-il utiliser ce fichier ?

/usr/portage/app-admin/syslog-ng/files/syslog-ng.conf.gentoo.hardened

1.3 Rotation des logs

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.

Installez ce programme :

emerge logrorate

Etudiez les fichiers **/etc/logrotate.conf , /etc/logrotate.d/syslog-ng , /etc/logrotate.d/apache2**

question 4 : ajoutez une configuration pour vos logs ssh précédents

Forcer son exécution en lancant :

logrotate -f /etc/logrotate.conf

Vérifiez que votre configuration fonctionne.

2 Tâches planifiées : cron

Cron est un démon qui lance des tâches qui ont été planifiées à l'aide de la commande crontab. Pour cela, il se réveille à chaque minute pour vérifier s'il n'y a pas une tâche à lancer en regardant les crontabs des utilisateurs.

Vixie Cron est une implémentation complète de cron basée sur SysV cron. Chaque utilisateur possède sa propre crontab et peut y spécifier des variables d'environnement. À la différence des autres variantes de cron, il supporte SELinux et PAM. Il supporte moins d'architectures que Dcron mais plus que Fcron.

Voici les fonctionnalités de sys-process/vixie-cron :

- Supporte SELinux.
- Supporte le fichier /etc/security/limits.conf de PAM.
- Chaque utilisateur peut avoir son propre crontab. L'accès est contrôlé par les fichiers cron.allow et cron.deny.

2.1 Configuration par défaut

Etudiez les fichiers de configuration suivant :

- /etc/crontab
- /etc/cron.daily/logrotate
- /etc/cron.weekly/makewhatis

2.2 Ajout d'un cron personnalisé

Question 5 : ajoutez une tache qui doit s'exécuter toutes les minutes et qui affiche « coucou !! ».

Vérifiez que cette tâche fonctionne.

question 6 : créez une nouvelle tache qui doit s'exécuter tous les soirs et qui créé une archive de /etc dans /var/backup du type **etc-DATE.zip**

3 Gestion des utilisateurs Unix

3.1 Définition des utilisateurs (à traiter comme utilisateur)

1- Combien d'utilisateurs UNIX sont définis localement sur votre système ? Parmi ceux-ci, combien correspondent à des utilisateurs réels (humains ou humanoïdes) ?
2- Quel est le shell de l'utilisateur root ? Et celui de l'utilisateur halt ? A quoi sert ce dernier ?
3- Que fait la commande /sbin/nologin ? Quels comptes l'utilisent-elle, pourquoi ?

3.2 Création d'utilisateurs et de groupes (à traiter comme administrateur (root))

Gestion des utilisateurs
La création d'un utilisateur UNIX requiert au moins les étapes suivantes : 1. ajouter les informations dans les fichiers /etc/passwd et /etc/shadow, ou dans l'annuaire (NIS, LDAP ou autre) utilisé. 2. créer le répertoire de connexion de l'utilisateur, et y placer les fichiers de configuration minimaux ; 3. configurer si nécessaire le système de messagerie électronique (e-mail). La création d'un utilisateur local (défini simplement sur votre système) est facilité par la commande useradd. La création d'un groupe est similaire (mais sans création de répertoire), via la commande groupadd. Le mot de passe d'un utilisateur est changé par la commande passwd. L'administrateur (root) peut changer le mot de passe d'un utilisateur quelconque en indiquant passwd login. Pour ajouter un utilisateur à un groupe, on édite le fichier /etc/group. Le shell d'un utilisateur est changé par la commande chsh. useradd création utilisateur local groupadd création groupe local passwd modification mot de passe chsh modification shell de login

1- Lire la documentation de la commande *useradd*, puis créer quelques utilisateurs, dont un avec votre nom et prénom.
Immédiatement après création, quel est le mot de passe de l'utilisateur ? Pourquoi ?

2- Dans quels groupes sont vos utilisateurs ?

3- Créer un groupe *tpgtr* réunissant deux de vos utilisateurs.

3.3 Droits (commandes `chown`, `chgrp`, `chmod`)

1- Changer (en tant qu'etudiant) les droits sur le compte "etudiant" afin que les autres utilisateurs ne puisse pas y accéder.

2- Créer un répertoire dans /*tmp* qui ne soit accessible (rx) que par les membres du groupe *etudiant*, puis y créer (toujours en tant qu'etudiant) un fichier toto qui soit lisible et modifiable par les utilisateurs du groupe `etudiant`, mais pas par les autres.
Tester (ajouter un autre utilisateur au groupe *etudiant*).

Les utilisateurs du groupe *etudiant* peuvent ils supprimer le fichier *toto* ? Pourquoi ?

3.4 Droits d'accès

Exercice à traiter comme utilisateur (etudiant), non root !
1- Essayer (dans un shell etudiant) de supprimer ou de modifier le fichier */var/log/messages*.
Que se passe-t-il ? Expliquer la situation à l'aide de la commande *ls -l*

2- A l'aide de la commande id, vérifier votre identité et le(s) groupe(s) auquel vous appartenez.

3- Créer un petit fichier texte (de contenu quelconque), qui soit lisible par tout le monde, mais pas modifiable (même pas par vous).

4- Créer un répertoire nommé *secret*, dont le contenu soit visible uniquement par vous même. Les fichiers placés dans ce répertoire sont ils lisibles par d'autres membres de votre groupe ?

5- Créer un répertoire nommé *connaisseurs* tel que les autres utilisateurs ne puissent pas lister son contenu mais puissent lire les fichiers qui y sont placés. On obtiendra :

```
$ ls connaisseurs
ls : connaisseurs: Permission denied
$ cat connaisseurs/toto
<...le contenu du fichier toto (s'il existe)...>
```

6- Chercher dans le répertoire */usr/bin* trois exemples de commandes ayant la permission *SUID*. De quelle genre de commande s'agit il ?

3.5 Limites et restrictions

Modifier le fichier */etc/security/limits.conf* afin de limiter les utilisateurs de votre système à *:*

- *3 connexions maximum*
- *20 processus*
- *1 minutes de temps CPU*

Tester ensuite ces paramètres avec un compte utilisateur.
N'autoriser que les connections sur le terminal 1 à l'aide du fichier /etc/securetty.
N'autoriser que le *bash* dans */etc/shells*.

Etudier le fichier */etc/login.defs*. Que permet de faire ce fichier?