CHAPITRE 9 :
Planification des tâches
Programmation des tâches périodiques
Automatisation des tâches

Mohammed SABER

Département Électronique, Informatique et Télécommunications
École Nationale des Sciences Appliquées "ENSA"
Université Mohammed Premier OUJDA

Année Universitaire : 2018-2019

Plan de chapitre

1. Introduction
2. Planification par CRON
3. CRON utilisateur
4. Contrôle des accès à CRON
5. CRON système

Plan de chapitre

1. Introduction
2. Planification par CRON
3. CRON utilisateur
4. Contrôle des accès à CRON
5. CRON système

Introduction

- Sous Linux, un grand nombre de tâches doivent être réalisées périodiquement pour assurer la cohérence et les performances du système.
- Parmi ces tâches (certains tâches ponctuelles à des horaires précis pour éviter les interruptions de service ou répondre à des contraintes de fonctionnement de l'entreprise) répétitives, on peut citer par exemple :
  - Vérification des systèmes de fichiers,
  - Sauvegarde des données utilisateur,
  - Analyse des fichiers journaux du système,
  - Nettoyage de «`/tmp`»,
  - etc...
- L'administrateur doit lancer périodiquement des actions.
- Il existe un système automatisant ces lancements : **CRON**

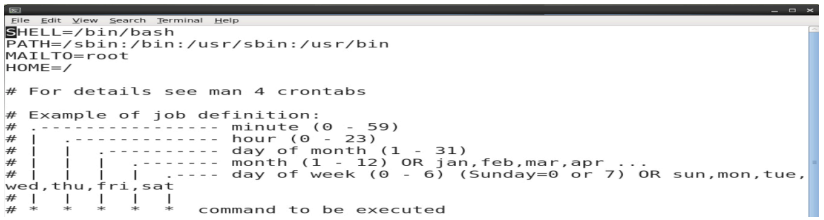
- Un démon «**crond**» de CRON est lancé par les scripts de démarrage du boot.

Plan de chapitre

1 Introduction
2 Planification par CRON
3 CRON utilisateur
4 Contrôle des accès à CRON
5 CRON système

Planification par CRON

- **CRON** est outil permettant d'automatiser l'exécution des tâches répétitives.
- Il se présente sous la forme d'un processus crond qui utilise un ensemble de fichiers consignant les travaux à effectuer avec la période d'exécution associée : les fichiers «**contrab**» ou «**cron table**» utilisateur.
- **CRON** (nom de la fonctionnalité en fait) est composé de 2 programmes, **crond** et **crontab**.
  - crond : service de traitement périodique (chaque minute) des **tables cron** (**crontabs**), lancé au démarrage du système.
  - **crontab** : utilitaire d'édition et de visualisation des **tables cron**, chaque **utilisateur** + **root** peut gérer ses propres **tables cron**, les **crontab** sont localiés dans `/var/spool/cron/`.



Planification par CRON
CRON table "crontab" utilisateur

- Chaque utilisateur possède son propre fichier **crontab** dans le répertoire `/var/spool/cron/`.
- Il est possible de consulter la table **CRON "crotab"** des utilisateurs avec Syntaxe :

root@hostname : # `crontab [-u user] -l|-r|-e`

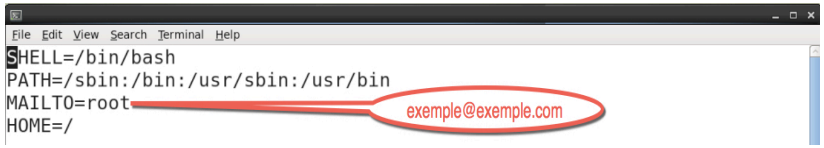
  - **-u user** : édition du fichier **crontab** d'un autre utilisateur (par root uniquement).
  - **-l** : lister le **crontab** actif de l'utilisateur.
  - **-r** : supprimer le **crontab** de l'utilisateur.
  - **-e** : édition du fichier **crontab** associé à l'utilisateur qui lance la commande.
- Le fichier **crontab** utilisateur est un fichier texte composé de deux types de lignes :
  - Les lignes d'initialisation de variables d'environnement.
  - Les lignes de commandes pour les démons CRON.

Plan de chapitre

1 Introduction
2 Planification par CRON
3 CRON utilisateur
4 Contrôle des accès à CRON
5 CRON système

CRON table "crontab" utilisateur

- Les lignes d'initialisation de variables d'environnement : elle permettent de définir l'environnement dans lequel les travaux cron doivent être exécutés. Affectation de variables :
  - Pour initialiser une variable d'environnement dans le fichier **crontab**, il suffit d'utiliser la syntaxe : **VAR=VALEUR**.
  - Les variables `$LOGNAME`, `$HOME` et `$SHELL` sont prédéfinies et contiennent respectivement le nom du compte et le répertoire personnel indiqués dans `/etc/passwd` pour le propriétaire de la table CRON et `/bin/sh` comme interpréteur de commandes.
  - Variable PATH définit le chemin d'accès utilisé pour l'exécution des commandes. Le résultat des tâches cron est envoyé par e-mail au nom d'utilisateur défini par la variable **MAILTO**. Si la variable `$MAILTO` est une chaîne vide (**MAILTO=""**), aucun e-mail ne sera envoyé.



CRON table "crontab" utilisateur

- Les lignes de commandes pour les démons cron : elles définissent les travaux à lancer périodiquement.
- **Tâche CRON** : une ligne de **commande cron** est constituée de deux informations :
  - Une **périodicité d'exécution** de la tâche (les cinq premiers champs de la ligne permettent de spécifier cette période).
  - La ligne de **commande shell** à interpréter pour lancer la tâche.

- Les actions à lancer périodiquement sont indiquées au format suivant : **minutes heures jour-du-mois mois jour-de-semaine commande**
  - **Champ 1 - Minutes** : nombre entier entre 0 et 59.
  - **Champ 2 - Heures** : nombre entier entre 0 et 23.
  - **Champ 3 - Jour du mois** : nombre entier entre 1 et 31 (si le mois est spécifié, le jour doit être valide).
  - **Champ 4 - Mois** : nombre entier entre 1 et 12 (ou abréviation du nom du mois).
  - **Champ 5 - Jour de la semaine** : nombre entier entre 0 et 7, 0 ou 7 représentant le dimanche (ou l'abréviation du jour de la semaine).
  - **Champ 6 - Commande** : commande à exécuter.

CRON table "crontab" utilisateur

Il est possible de donner plusieurs valeurs à ces champs par des caractères spéciaux :

- Le caractère «*****» sert de caractère joker pour n'importe quel des champs 1 à 5.
- Le caractère «**,**» sépare les éléments d'une liste de valeurs. **Par exemple** : 1,3,4,6.
- Le caractère «**-**» désigne un intervalle entre deux valeurs. **Par exemple** : 1-4 → valeurs 1, 2, 3, 4.
- Le caractère «/» permet d'indiquer une fréquence. **Par exemple** :
  - ***/10** → valeurs 0, 10, 20, 30, 40, etc.
  - **1/2** → valeurs 1, 3, 5, 7, 9, 11, etc.

```
15  *  *  *  *  /chemin/vers/ntpdate.sh
0  *  *  *  *  /chemin/vers/check-devnull.sh
15  *  *  *  *  /chemin/vers/check-devnull.sh
30  *  *  *  *  /chemin/vers/check-devnull.sh
45  *  *  *  *  /chemin/vers/check-devnull.sh
0  21  *  *  1  /chemin/vers/sauvegarde.sh
0  21  *  *  3  /chemin/vers/sauvegarde.sh
0  21  *  *  5  /chemin/vers/sauvegarde.sh
```

- La **crontab** ci-dessus est donc équivalente à :

```
15  *  *  *  *  /chemin/vers/ntpdate.sh
*/15  *  *  *  *  /chemin/vers/check-devnull.sh
0  21  *  *  1,3,5  /chemin/vers/sauvegarde.sh
```


CRON table "crontab" utilisateur

min	heure	jour/mois	mois	jour/semaine	Périodicité
*	*	*	*	*	Toutes les minutes
30	0	1	1,6,12	*	à 00:30 le premier janvier, juin et décembre
0	20	*	10	1-5	à 20:00 chaque jour de la semaine (du lundi au vendredi) d'octobre
0	0	1,10,15	*	*	à minuit les premiers, dixièmes, et quinzième jours de chaque mois
5,10	0	10	*	1	à 00:05 et 00:10 chaque lundi et le 10 de chaque mois

- Les fichiers **crontab** ont beau être au format texte.
- **MAIS.** Il ne faut pas les éditer manuellement.
- Il faut passer par l'intermédiaire de la commande «`crontab`».
- `crontab -e` : cela lance l'éditeur de texte indiqué par la variable d'environnement «**EDITOR**» ou bien «`nano`» par défaut.

CRON table "crontab" utilisateur

- Un utilisateur ne peut **consulter** / **modifier** / **supprimer** que son fichier **CRONTAB**.

```
saber@hostname : $crontab -l
```

```
saber@hostname : $crontab -e
```

```
saber@hostname : $crontab -r
```

- L'administrateur peut **consulter** / **modifier** / **supprimer** tout fichier **CRONTAB** d'utilisateur.

```
root@hostname : #crontab -l [-u utilisateur]
```

```
root@hostname : #crontab -l [-u e]
```

```
root@hostname : #crontab -r [-u utilisateur]
```


Plan de chapitre

1 Introduction
2 Planification par CRON
3 CRON utilisateur
4 Contrôle des accès à CRON
5 CRON système

Contrôle des accès à CRON

- L'utilisation du fichier `/etc/cron.allow` ou `/etc/cron.deny` permet d'indiquer la liste des utilisateurs autorisés à employer le service CRON.
- Si les deux fichiers n'existent pas, il existe deux cas de figure selon la configuration du système par défaut : l'administrateur «**root**» est autorisé à exécuter la commande **crontab** et (ou) tous les utilisateurs.
- Si le fichier **cron.allow** existe, les utilisateurs dont il contient les noms, login un par ligne, sont autorisés à exécuter la commande **crontab**, les autres non.
- Si le fichier **cron.deny** existe, les utilisateurs dont il contient les noms, un par ligne, n'ont pas le droit d'exécuter la commande **crontab**, les autres oui.
- Les fichiers **cron.allow** et **cron.deny** contrôlent l'utilisation de la commande **crontab**.
- Il est conseillé de ne laisser l'accès à «**crontab**» qu'à l'utilisateur root.

cron.allow	cron.deny	Utilisateurs autorisés
Présent	Présent	Ceux explicitement dans `cron.allow`
Présent	–	Ceux explicitement dans `cron.allow`
–	Présent	Tous sauf ceux dans `cron.deny`
–	–	Uniquement root

Plan de chapitre

1 Introduction
2 Planification par CRON
3 CRON utilisateur
4 Contrôle des accès à CRON
5 CRON système

CRON table système

- En plus des **crontab utilisateur**, il existe une notion de **crontab système**.
- Celle-ci permet de planifier des tâches d'administration comme le nettoyage régulier de `/tmp` ou la mise à jour automatique des paquetages du système.
- `/etc/crontab` traite des répertoires spécifiques sur une périodicité horaire, quotidienne, mensuelle, annuelle.
- Des tables CRON peuvent également se trouver dans `/etc/cron.d`.
- Le fichier **crontab système** est : `/etc/crontab`, sa syntaxe est la même que précédemment, avec la possibilité de spécifier le nom d'utilisateur juste avant la ligne de la commande à lancer (en sixième champ).

CRON table système

- Structure **crontab système** quasi identique **crontab utilisateur** sauf :
  - **Champ 1 - Minutes** : nombre entier entre 0 et 59.
  - **Champ 2 - Heures** : nombre entier entre 0 et 23.
  - **Champ 3 - Jour du mois** : nombre entier entre 1 et 31 (si le mois est spécifié, le jour doit être valide).
  - **Champ 4 - Mois** : nombre entier entre 1 et 12 (ou abréviation du nom du mois).
  - **Champ 6 - login** : Utilisateur sous lequel lancer la commande.
  - **Champ 7 - Commande** : commande à exécuter.

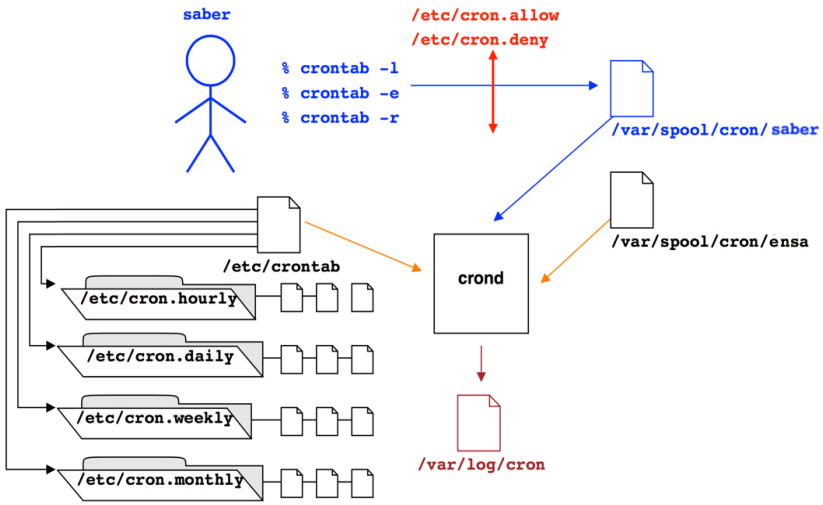
CRON table système

- **Exemple** :

```
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
HOME=/
# run-parts
01 * * * * root nice -n 19 run-part /etc/cron.hourly
02 4 * * * root nice -n 19 run-part /etc/cron.daily
22 4 * * 0 root nice -n 19 run-part /etc/cron.weekly
42 4 1 * * root nice -n 19 run-part /etc/cron.monthly
```

- La commande **run-parts <répertoire>** exécute tous les scripts et programmes présents dans le répertoire spécifié.
- Dans ce fichier les entrées faisant référence aux répertoires :
  - `/etc/cron.hourly` : répertoire de scripts horaires.
  - `/etc/cron.daily` : répertoire de scripts quotidiens.
  - `/etc/cron.weekly` : répertoire de scripts hebdomadaires.
  - `/etc/cron.monthly` : répertoire de scripts mensuels.
- **Intérêt** : plus simple à manipuler, il y a juste à copier un script dans un de ces répertoires.

CRON table système



QUESTIONS ?