Comment faire pour éviter un soudain surplus d’accès disque de la part d'un script Shell ou un programme

Un surplus soudain des E/S disque violente peut faire baisser les performances de votre serveur de mail ou de votre serveur Web.
Pour éviter ce genre de problème d'explosion soudaine, exécutez votre script avec une classe de d'ordonnancement et de priorité.

Linux est livré avec divers utilitaires pour gérer ce genre d'incident.

L'ordonnanceur CFQ

Vous devez avoir un noyaux Linux 2.6.13+ avec l'ordonnanceur CFQ IO.
CFQ (Completely Fair Queuing) est un ordonnanceur d'E/S pour le noyau Linux, qui est par défaut dans le noyau 2.6.18+.
Comme son nom l'indique, il permet d'ordonner les entrées et sorties des périphériques.
Pour la définition exacte Wikipedia.org
Pour savoir quel ordonnanceur vous utilisez, entrez:

```
for d in /sys/block/sd[a-z]/queue/scheduler; do echo "$d => $(cat $d)" ;
done
```

Voici, le résultat :

```
/sys/block/sda/queue/scheduler => noop deadline [cfq]
```

L'ordonnanceur entre crochet `[cfq]` est celui qui est actif.

CFQ est par défaut et recommandé pour une bonne performance.

Le changement est actif jusqu'au prochain redémarrage, ou nouveau changement de votre part.
Vous pouvez rendre le changement permanent à l'aide d'une règle “udev”, de manière indépendante pour chaque disque (rotationel ou SSD).

la commande ''nice''

Vous pouvez exécuter un programme avec une priorité d'ordonnancement modifiée en utilisant la commande `nice`.
La gestion de la rpiorité sous Linux est celle-ci :

- de -1 à -20 ce sont des priorités hautes, -20 étant la plus haute.
- de 0 à 19 ce sont priorités basses, 19 la plus basse.

Attention à ne pas monter trop haute la priorité des processus.

Exemple :

```
nice -n19 /chemin/vers/script.sh
```

Dans un crontab :

```
 00 00 * * * /bin/nice -n19 /chemin/vers/script.sh
```

la commande ''ionice''

La commande `ionice` fournit un meilleur contrôle que la commande `nice` pour les E/S de classe d'ordonnancement et la priorité d'un programme ou d'un script.
Un processus peut appartenir à une des trois classes d'ordonnancement possible (je cite le man man-linux-magique.net):

- **Idle** :
  Un programme s'exécutant avec une priorité d'entrées/sorties "idle" obtiendra du temps pour accéder au disque quand aucun
  autre programme n'a demandé d'entrées/sorties sur les disques dans une période donnée.
  L'impact des processus avec une classe d'ordonnancement d'entrées/sorties "idle" sur l'activité normale du système devrait être nul.
  Cette classe d'ordonnancement de processus ne prend pas de priorité en paramètre.
  Cette classe d'ordonnancement est permise pour un simple utilisateur (depuis Linux 2.6.25).
- **Best effort** :
  C'est la classe par défaut d'ordonnancement pour chaque processus qui n'a pas demandé une priorité spécifique d'entrées/sorties.
  Les programmes héritent des paramètres de politesse "nice" du processeur pour les priorités d'entrées/sorties.
  Cette classe prend une priorité en paramètre dans la gamme 0-7, où le nombre le plus bas sera d'une priorité plus haute.
  Les programmes en cours ayant la même priorité "best effort" sont servis l'un après l'autre.
  Un processus qui n'a pas demandé de priorité d'E/S utilise la classe d'ordonnancement none mais l'ordonnanceur d'E/S traitera un tel processus comme s'il était de la classe best effort.
  La priorité dans la classe best effort sera dynamiquement dérivée du niveau de politesse CPU de la priorité du processus d'E/S (égale à (politesse_cpu + 20) / 5).
- **Real time** :
  La classe d'ordonnancement RT donne en premier l'accès au disque, sans se soucier des autres exécutions sur le système.
  De ce fait, la classe RT doit être utilisée avec attention, car elle peut "affamer" d'autres processus.
  Comme la classe "best effort", 8 niveaux de priorité sont définis dénotant la période de temps qu'un processus donné recevra dans chaque fenêtre d'ordonnancement.
  Cette classe d'ordonnancement n'est pas permise pour un simple utilisateur (c'est-à-dire, non-superutilisateur).

Syntax de la commande :

```
ionice options  PID
ionice options -p PID
ionice -c1 -n0  PID
```

Comment puis-je utiliser la commande ionice sur Linux?

Classe d’ordonnancement	nombre	priorité possible
real time	1	8 niveaux de priorité sont définis pour indiquer la taille d'une tranche de temps que recevra un processus donné à chaque fenêtre de planification
best-effort	2	0-7 avec le numéro de priorité le plus élevée étant inférieure
idle	3	ne prend pas d'argument de priorité

Exemple :

```
onice -c3 -p 89
```

Définit le processus avec le PID 89 comme un processus de la classe d’entrées/sorties idle.

```
ionice -c2 -n 0 /chemin/vers/script.sh
```

Exécute “/chemin/vers/script.sh” comme un programme best-effort avec la priorité la plus élevée.

```
ionice -p 89 91
```

Renvoie la classe et la priorité des processus de PID 89 et 91.
Voici, le résultat

```
none : priorité 4
none : priorité 4
```

Enfin, vous pouvez combiner les deux `nice` et `ionice` ensemble:

```
nice -n 19 ionice -c2 -n7 /chemin/vers/script.sh
```

Autre suggestion pour améliorer les E/S disque :

1. Utilisez un contrôleur RAID matériel.
2. Utilisez SCSI rapide/SA-SCSI/des disques SAS 15K de vitesse.
3. Utilisez un stockage basé sur des disque SSD rapide (option coûteuse pour le moment).
4. Utiliser un esclave, un serveur passif à la sauvegarde de MySQL.