matériel

<term RPI>un mini-PC de la taille d'une carte de crédit</term>

Mise en place d'un NAS avec Raid 1 sur un Raspberry Pi

Utilisation

Maintenance

C'est bien beau de mettre en œuvre une politique de tolérance de panne mais que fait on... en cas de panne ?

Comment détecter une panne ?

Le fichier système **/proc/mdstat** permet d'avoir un aperçu rapide de l'état de nos ensembles RAID.

```
$ sudo cat /proc/mdstat
Personalities : [raid1]
md0 : active raid1 sdb1[0] sdc1[1]
      976759936 blocks [2/2] [UU]
.
unused devices: <none>
```

Un coup d'oeil rapide sur le flag [UU] et nous savons que tout est OK. Si il y avait un problème sur un des disques nous aurions [_U] ou [U_]

Identifier le disque fautif

La ligne md0 : active raid1 sdb1[0] sdc1[1] peut apporter cette information : si une partition manque à l'appel alors nous aurons par exemple md0 : active raid1 sdc1[1] (il manque sdb1 !) ou encore ceci md0 : active raid1 sdb1[0](F) sdc1[1] (notez le (F) pour faulty !).

On peut également consulter la sortie de cette commande :

```
$ sudo mdadm --detail /dev/md0
```

Repérez la ligne State : si vous voyez ceci : State : clean, degraded ce n'est pas bon signe... l'un des disques pose problème. A la fin, vous avez un tableau listant les composants de l'ensemble RAID. Quand tout va bien, vous voyez vos périphériques (/dev/sdb1, /dev/sdc1, etc..) précédés de la mention active sync. Mais si ça va mal, on trouvera un commentaire différent, par exemple faulty spare, ou removed)

Reste enfin la commande système dmesg ou encore les fichiers journaux /var/log/syslog et

/var/log/messages qui peuvent apporter des informations précieuses en cas de problème.

Automatiser la détection de panne

- Soit vous scriptez en vous basant sur le fichier /proc/mdstat ou la sortie de mdadm –detail puis vous appelez ce script via une crontab. Pour ceux qui manqueraient d'inspiration, voici un exemple tout bête avec le célèbre Conky : Placez cette ligne quelque part dans la partie TEXT de votre .conkyrc.

.conkyrc

```
${texeci 3600 awk ' $2 == "blocks" && $4 != "[UU]" {print "DANGER ! Erreur RAID détectée !!!"}' /proc/mdstat}
```

Toutes les 3600s (1h) conky exécutera la commande awk qui vérifiera si le flag [UU] est bien présent. Si ce n'est pas le cas, le message “DANGER ! Erreur RAID détectée !!!” s'affichera sur votre bureau. Notez tout de même que cette commande n'est compatible que pour un seul ensemble RAID de 2 disques. Veuillez adapter cette commande si votre configuration est différente.

- Soit vous utilisez le mode monitor de mdadm. Mdadm intègre un système d'alerte automatique avec possibilité de passer en paramètre une adresse email (–mail) ou bien un programme à exécuter (–alert) (cf. man mdadm)

Exemple :

```
$ sudo mdadm --monitor -f --mail=votre@email.com --delay=3600 /dev/md0
```

-f permet de “daemoniser” le mode monitor (comprendre exécuter en arrière plan). Mais si par malheur le processus tombe, alors vous ne serez plus averti.

Cas 1 : panne d'un disque de l'array (ex : /dev/sdb)

Une fois le disque fautif identifié, il faut signaler à mdadm que nous voulons le retirer de notre ensemble RAID.

```
$ sudo mdadm --manage /dev/md0 --remove /dev/sdb1
```

En cas de grosse panne disque, ce dernier n'est peut être plus détecté par le système. Cette commande renverra donc une erreur signalant que /dev/sdb1 n'existe pas. Ce n'est pas génant, continuons...

/!\ CAS PARTICULIER /!\ : si votre disque n'est plus détecté et que le redémarrage de la machine est impossible (contrainte production). Allez voir les précisions de AnatomicJC ICI

Eteindre la machine et placer un nouveau disque de taille identique ou supérieure. Une fois redémarrée, vérifier la présence du nouveau disque avec # fdisk -l et recommencer les étapes de la partie 3 Préparations des disques (fdisk). Vous pouvez aussi recopier simplement la table des partitions de /dev/sdc sur /dev/sdb comme nous l'avons vu dans la partie 3.3 Préparons le deuxième disque...

/!\Attention/!\ NE TAPER PAS CETTE COMMANDE SANS VÉRIFIER A DEUX FOIS LE NOM DES DISQUES !!

```
$ sudo sfdisk -d /dev/sdc | sfdisk /dev/sdb
```

Ajoutons le disque à l'ensemble dégradé :

```
$ sudo mdadm --manage /dev/md0 --add /dev/sdb1
```

L'ensemble RAID est alors en reconstruction, cela peut être très long. Vous pouvez observer l'avancement en consultant le fichier /proc/mdstat. Vous remarquerez alors la mention “recovery” à droite de la jauge de progression.

Une fois la reconstruction terminée, ne pas oublier de mettre à jour le fichier de configuration :

```
$ sudo mdadm --detail --scan --verbose > /etc/mdadm/mdadm.conf
```

Cas 2 : panne du disque système, la grappe RAID est OK

Voici un cas un peu particulier mais qui m'est arrivé donc je tenais à en parler ici. Nous sommes donc dans le cas donné pour exemple dans cet article à savoir ; une Debian installée sur un seul disque et une grappe RAID1 pour les données construite avec mdadm. Imaginons que votre disque système tombe en rade mais que votre grappe reste OK. Il vous faut donc réinstaller votre Linux avec mdadm et lui faire comprendre qu'une array existe déjà.

Dans l'ordre de préférence faisons ceci :

Choix 1 - Si tout va bien, lors de la nouvelle installation de mdadm, il va détecter les “md superblock” et en déduire donc qu'une grappe RAID1 existe entre les partitions /dev/sdb1 et /dev/sdc1. Surveillez donc votre /proc/mdstat et attendez que la synchro soit terminée avant de redémarrer votre machine.

Choix 2 - Si vous aviez sauvegardé le fichier /etc/mdadm/mdadm.conf, vous pouvez le restaurer et taper la commande suivante :

```
$ sudo mdadm --assemble --scan
```

Ceci à pour effet de scanner le mdadm.conf et d'assembler toutes les arrays y figurant. En sachant qu'un simple redémarrage de la machine avec le mdadm.conf fraîchement restauré suffit à offrir le même résultat.

Choix 3 - S'il ne se passe rien et que vous n'avez plus votre mdadm.conf alors il vous faut connaître le

nom de votre array et les partitions qui la composent. Dans notre cas c'est pas bien dur ; nos deux partitions /dev/sdb1 et /dev/sdc1 sont en RAID1 dans l'array /dev/md0. On tape donc :

```
$ sudo mdadm --assemble /dev/md0 /dev/sdb1 /dev/sdc1
```

Choix 4 - Si l'architecture est plus complexe et que vous pataugez dans la semoule, il est judicieux de vous aider des commandes # fdisk -l et # mdadm –-examine /dev/<partition> pour identifier le bouzin.

Exemple :

```
$ sudo fdisk -l
...
...
   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1               1      121601   976760001   fd  Linux raid
autodetect
...
...
```

Je vois ici que /dev/sdb1 fait certainement partie d'un ensemble RAID. Alors j'enquête....

```
$ sudo mdadm --examine /dev/sdb1
mdadm: metadata format 00.90 unknown, ignored.
/dev/sdb1:
          Magic : a92b4efc
        Version : 00.90.00
           UUID : 3b2be7cf:1eca6c08:a962df05:773a6f64
  Creation Time : Sun Dec 14 20:41:21 2008
     Raid Level : raid1
  Used Dev Size : 976759936 (931.51 GiB 1000.20 GB)
     Array Size : 976759936 (931.51 GiB 1000.20 GB)
   Raid Devices : 2
  Total Devices : 2
Preferred Minor : 0
.
    Update Time : Fri May  7 17:06:49 2010
          State : clean
 Active Devices : 2
Working Devices : 2
 Failed Devices : 0
  Spare Devices : 0
       Checksum : f320b0b5 - correct
         Events : 556
.
.
      Number   Major   Minor   RaidDevice State
this     1       8       17        1      active sync   /dev/sdb1
.
   0     0       8        1        0      active sync   /dev/sdc1
```

```
   1     1       8       17        1      active sync   /dev/sdb1
```

Grâce à ces infos, je sais donc que la partition /dev/sdb1 fait partie d'un ensemble RAID1 et que sa compagne est la partition /dev/sdc1 (voir à la fin) On tape donc :

```
$ sudo mdadm --assemble /dev/md0 /dev/sdb1 /dev/sdc1
```

Une fois que tout est en ordre et bien synchro, tapez ceci :

```
$ sudo mdadm --detail --scan --verbose > /etc/mdadm/mdadm.conf
```

Réassembler un disque

C'est la commande **assemble** (**-A**) qui permet de reconstituer le raid. Suivant la façon dont il a été arrêté, il est possible qu'il faille des options complémentaires pour resynchroniser.

```
$ sudo mdadm -A /dev/md0 /dev/sdb1 /dev/sdc1
mdadm: /dev/md0 has been started with 2 drives.
```

Pour voir si tout s'est bien passé :

```
$ cat /proc/mdstat
Personalities : [raid1]
md0 : active (auto-read-only) raid1 sdb1[0] sdc1[1]
      131005440 blocks super 1.2 [2/2] [UU]
      bitmap: 0/1 pages [0KB], 65536KB chunk

unused devices: <none>
```

et :

```
$ sudo mdadm -D /dev/md0
/dev/md0:
        Version : 1.2
  Creation Time : Sat Aug 26 15:01:00 2017
     Raid Level : raid1
     Array Size : 131005440 (124.94 GiB 134.15 GB)
  Used Dev Size : 131005440 (124.94 GiB 134.15 GB)
   Raid Devices : 2
  Total Devices : 2
    Persistence : Superblock is persistent

  Intent Bitmap : Internal

    Update Time : Wed Aug 30 07:48:03 2017
          State : clean
 Active Devices : 2
Working Devices : 2
 Failed Devices : 0
```

```
  Spare Devices : 0

           Name : framboise:0  (local to host framboise)
           UUID : 63f3b6ad:4f7d815c:adfec463:e03739cd
         Events : 23072

    Number   Major   Minor   RaidDevice State
       0       8       17        0      active sync   /dev/sdb1
       1       8       33        1      active sync   /dev/sdc1
```

La grappe est en place avec les deux disques.

Aller plus loin...

En guise de conclusion, je voudrais donner quelques pistes pour aller plus loin :

Beaucoup d'entre vous voudront certainement mettre en place un miroir sur leur système plutôt que sur les données. On retrouve d'ailleurs souvent sur des serveurs une configuration du type : - 2 disques en RAID1 pour le système - 3(ou+) disques en RAID5 pour les données

RAID1 système : il est possible d'utiliser l'installeur debian pour mettre en place une architecture de type RAID1+LVM sur le système. Cependant, la partition /boot doit être en dehors du LVM à cause d'une limitation de GRUB premier du nom. GRUB2 supporte maintenant le boot sur LVM (avis aux experts pour compléments d'infos, merci).

Avec notre cher collaborateur lol ;) nous avons testé ceci grâce à ces explications : http://blog.le7.net/linux/debian/installation-debian-ubuntu-avec-raid-1/ Merci en passant à son auteur. Ce tutoriel fonctionne au poil avec une install lenny, j'ai effectué également des tests de panne et de récupération avec succès.

RAID5 de données : sur la question du stockage des données, il est plus intéressant sur un rapport prix/capacité d'opter pour un RAID5 plutôt qu'un RAID1. En effet, un RAID1 offre une capacité de stockage correspondant à la moitié de la taille totale des deux disques, alors qu'avec un RAID5 à 3 disques, nous avons les 2/3 de la taille totale des 3 disques.

Exemple : prenons des disques de 1To à 100 euros RAID1 = 200 euros pour 1To de données = 0.2 euro / Go RAID5 = 300 euros pour 2To de données = 0.15 euro / Go La mise en place d'un RAID5 se fait sensiblement de la même manière qu'un RAID1. Evidemment, la commande mdadm de création de l'array sera différente :

Exemple :

```
$ sudo mdadm --create --verbose /dev/md0 --level=5 --raid-devices=3
/dev/sdb1 /dev/sdc1 /dev/sdd1
```

Cependant, je tiens à signaler qu'on trouve des personnes se plaignant des performances RAID5 avec mdadm. Je vous conseille donc vivement de jeter un oeil à la fin de cet article qui est en anglais mais se comprend assez facilement : https://raid.wiki.kernel.org/index.php/RAID_setup

En bref, voici les paramètres qui permettraient d'offrir des performances dites “normales” dans la

plupart des cas pour un RAID5 à 3 disques :

- Construire son RAID5 en positionnant la taille du chunk à 128KB :

```
$ sudo mdadm --create --verbose /dev/md0 --level=5 --chunk=128 --raid-devices=3 /dev/sdb1 /dev/sdc1 /dev/sdd1
```

- Choisir la taille des blocs du système de fichiers résidant sur /dev/md0 : prenons par exemple 4KB qui est une valeur adaptée pour l'utilisation de gros fichiers.

- La taille du chunk et des blocs sur le système de fichiers vont déterminer deux autres paramètres : stride et stripe-width stride = chunk / blocs = 128 / 4 = 32KB stripe-width = stride x (nbr disques - 1) = 32 x (3 - 1) = 64KB

- Créons le système de fichiers avec ces valeurs :

```
$ sudo mkfs.ext3 -v -b 4096 -E stride=32,stripe-width=64 /dev/md0
```

Enfin, d'après cet article, nous pouvons jouer sur la valeur de stripe_cache_size. La valeur optimale semble être 8192 :

/!\ Attention, cette commande concerne l'ensemble md0 ! Adapter en conséquence...

```
$ sudo echo 8192 > /sys/block/md0/md/stripe_cache_size
```

Voir aussi

- **(fr)** https://wiki.debian-fr.xyz/Raid_logiciel_(mdadm)#Aller_plus_loin...
- **(fr)** https://xaviermichel.github.io/tutoriel/2013/06/17/mise-en-place-d%27un-NAS-avec-raid-sur-mon-raspberry-pi

---

Basé sur « https://xaviermichel.github.io/tutoriel/2013/06/17/mise-en-place-d%27un-NAS-avec-raid-sur-mon-raspberry-pi » par Xavier MICHEL