

Dernière mise-à-jour : 2020/08/07 14:50

LDF302 - Gestion des Disques Avancée - Raid Logiciel

Contenu du Module

- **LDF302 - Gestion Avancée des Disques - Raid Logiciel**

- Contenu du Module
- Concepts RAID
 - Disques en miroir
 - Bandes de données
- Types de RAID
 - RAID 0 - Concaténation
 - RAID 0 - Striping
 - RAID 1 - Miroir
 - RAID 1+0 - Striping en Miroir
 - RAID 2 - Miroir avec Contrôle d'Erreurs
 - RAID 3 et 4 - Striping avec Parité
 - RAID 5 - Striping avec Parité Distribuée
 - Au delà de RAID 5
- RAID Logiciel sous Debian
 - Préparation du disque
 - Partitionnement
 - Mise en Place du RAID 5 Logiciel

Concepts RAID

Les solutions RAID ou *Redundant Array of Independent Disks* ou encore *Redundant Array of Inexpensive Disks* permettent la combinaison de plusieurs disques de façon à ce que ceux-ci soient vu comme un seul disque logique.

Les solutions RAID sont issues du travail fourni par l'université de Berkeley en Californie sur un projet de tolérances de pannes. Les systèmes RAID offre maintenant plusieurs avantages :

- Addition des capacités,
- Amélioration des performances,
- Apporter la tolérance de panne.

Deux concepts sont fondamentaux à la compréhension des solutions RAID.

Disques en miroir

La technique des disques en miroir consiste à dupliquer l'écriture des données sur plusieurs disques. Le miroir peut être géré par un logiciel ou par du matériel.

Bandes de données

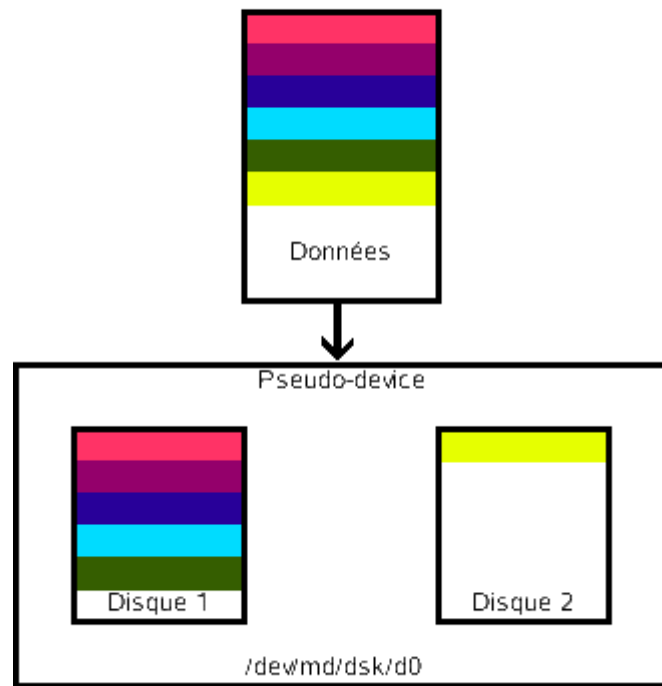
La technique des bandes de données, autrement appelée *data striping* consiste à couper les données à enregistrer en segments séquentiels et contigus pour les enregistrer sur plusieurs disques physiques. L'ensemble des segments constitue alors un disque logique ou *striped disk*. Cette technique peut être améliorée en déposant une bande de parité, calculée à partir des données des autres bandes, afin de pouvoir reconstituer une bande de données défaillante.

Types de RAID

RAID 0 - Concaténation

Création de volume par récupération de l'espace libre sur un ou plusieurs disques. Le principe de la concaténation est la création d'un volume à bandes où chaque bande est une tranche.

RAID 0 – Concaténation

**Avantages**

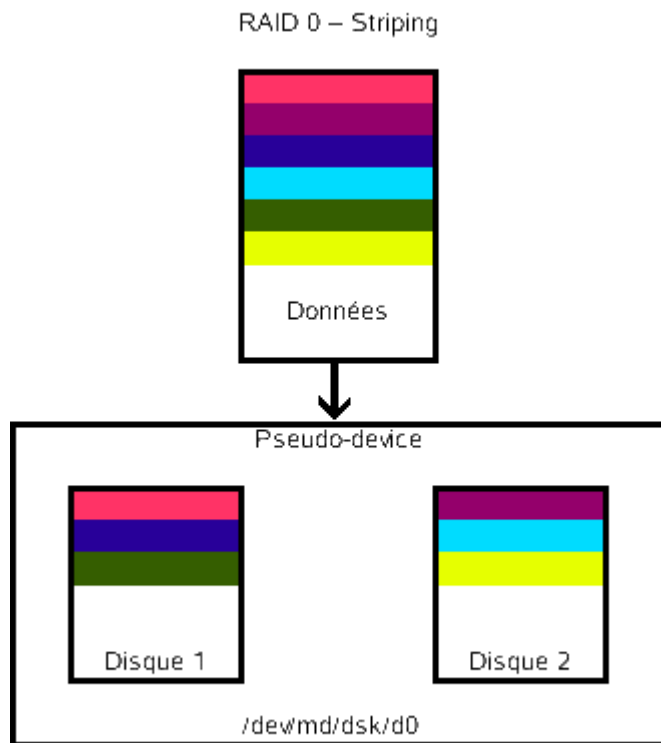
- Récupération de l'espace disque.

Inconvénients

- Pas de protection des données,
- Pas d'augmentation des performances d'E/S.

RAID 0 - Striping

Création de volume sur plusieurs disques afin d'augmenter les performances d'E/S. Le principe du striping est la création d'un volume à bandes réparties sur plusieurs tranches. La taille de la bande doit être fonction des données à écrire sur le volume (16k, 32k, 64k, etc.) Cette taille est choisie à la création du volume.



Avantages

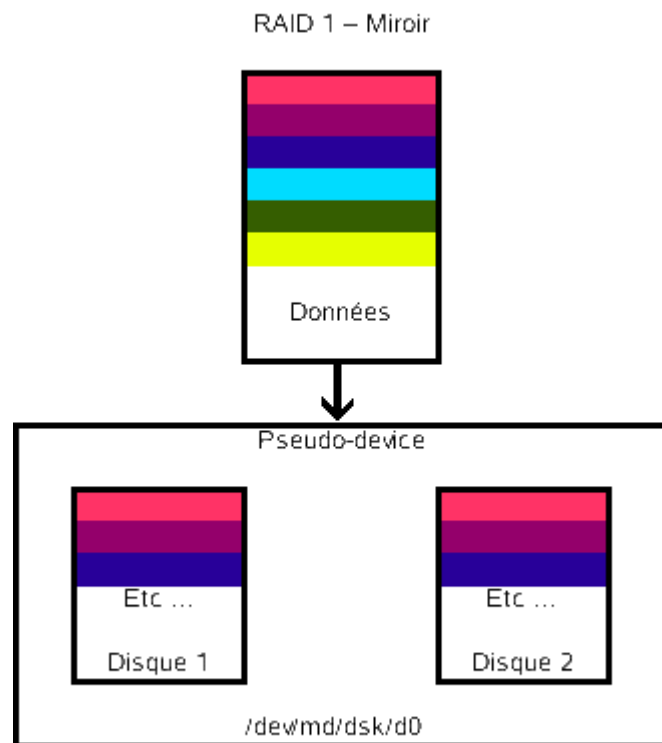
- Augmentation des performances d'E/S par écriture en parallèle sur les disques.

Inconvénients

- Pas de protection des données.

RAID 1 - Miroir

Création d'un volume où les disques sont en miroir. Quand les deux disques sont connectés à des contrôleurs de disques différents, on parle de *duplexing* :



Avantages

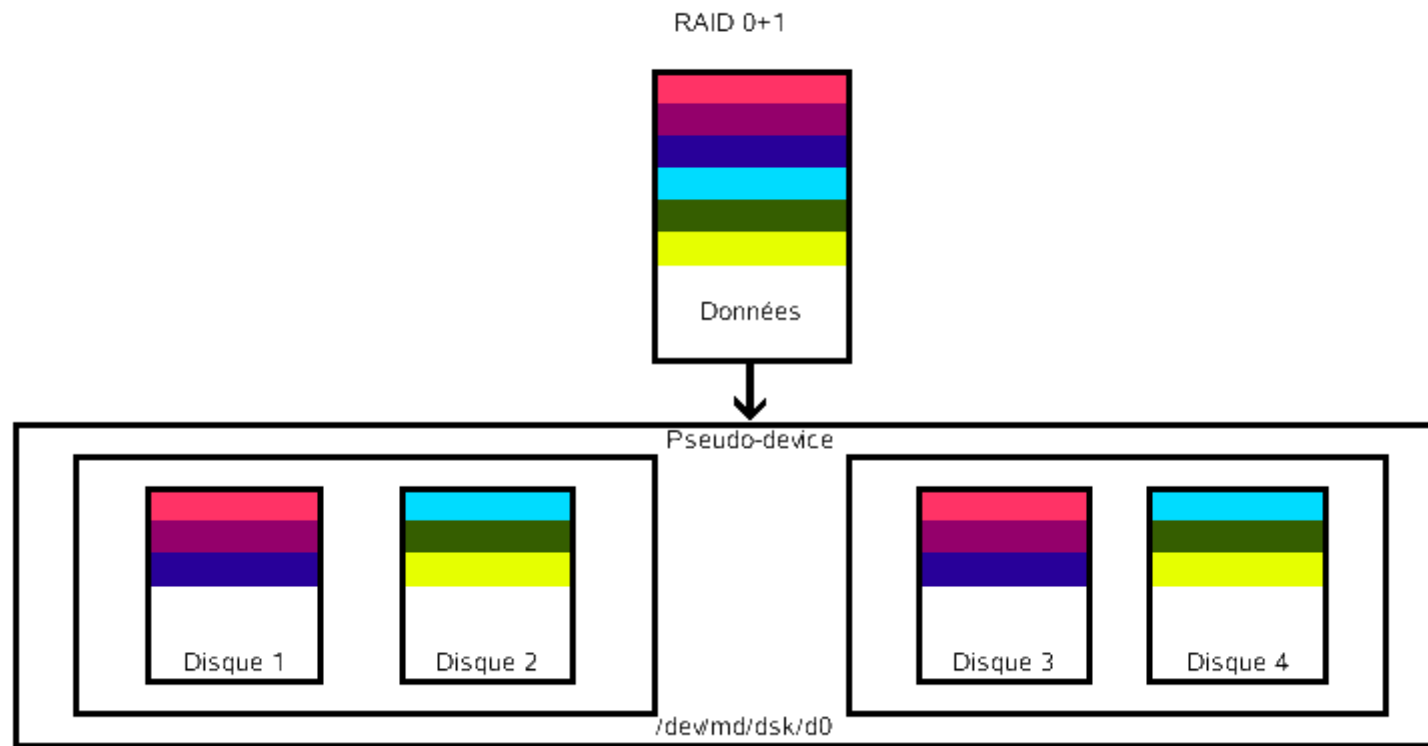
- Protection des données contre une défaillance d'un disque.

Inconvénients

- Coûteux à cause de l'augmentation du nombre de disques.

RAID 1+0 - Striping en Miroir

Le RAID 1+0 ou encore 0+1 est une technique qui réunit le RAID 0 et le RAID 1. On l'appelle aussi un RAID **exotique**:



Avantages

- Protection des données contre une défaillance d'un disque.
- Augmentation des performances d'E/S par écriture en parallèle sur les disques.

Inconvénients

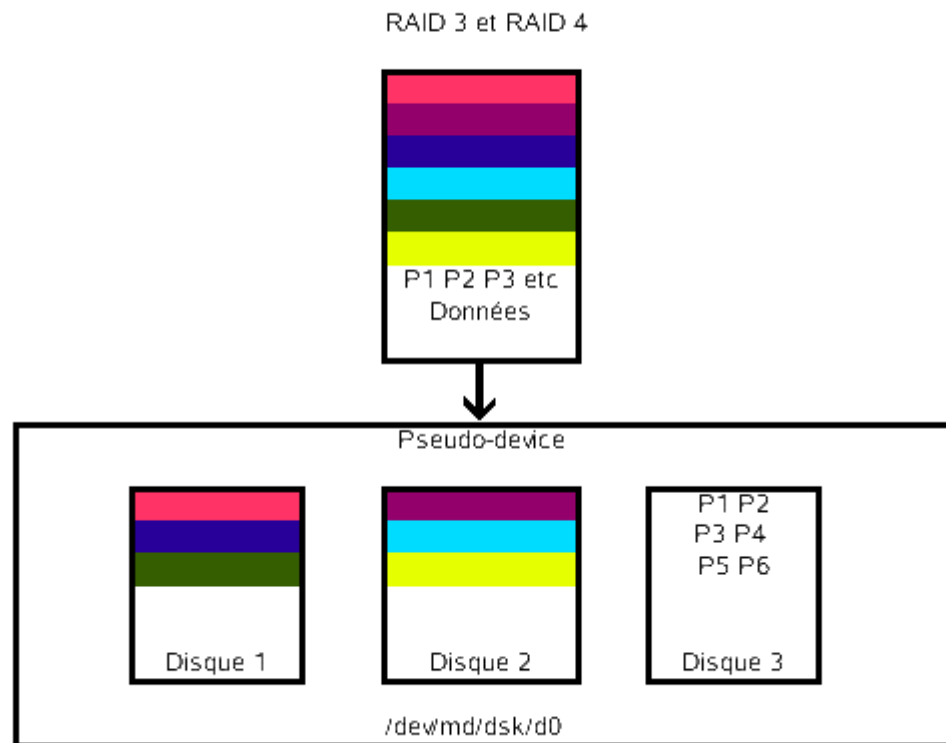
- Coûteux à cause de l'augmentation du nombre de disques.

RAID 2 - Miroir avec Contrôle d'Erreurs

Le RAID 2 est une technique de miroir avec contrôle de correction d'erreurs (EEC). De nos jours cette technique est peu utilisée, ayant été remplacée par les RAID 3, 4 et 5.

RAID 3 et 4 - Striping avec Parité

Les RAID 3 et 4 sont des technologies avec bandes de parité distribuées sur un seul disque :



En RAID 3, la taille des segments n'est pas modifiable et est fixée à 512 octets (en RAID 3 : un segment = un secteur de disque dur = 512 octets).

En RAID 4, la taille des segments est variable et se modifie en temps réel. Cela implique que les informations de parité doivent être mise à jour à chaque écriture afin de vérifier si la taille des segments a été modifiée.

Avantages

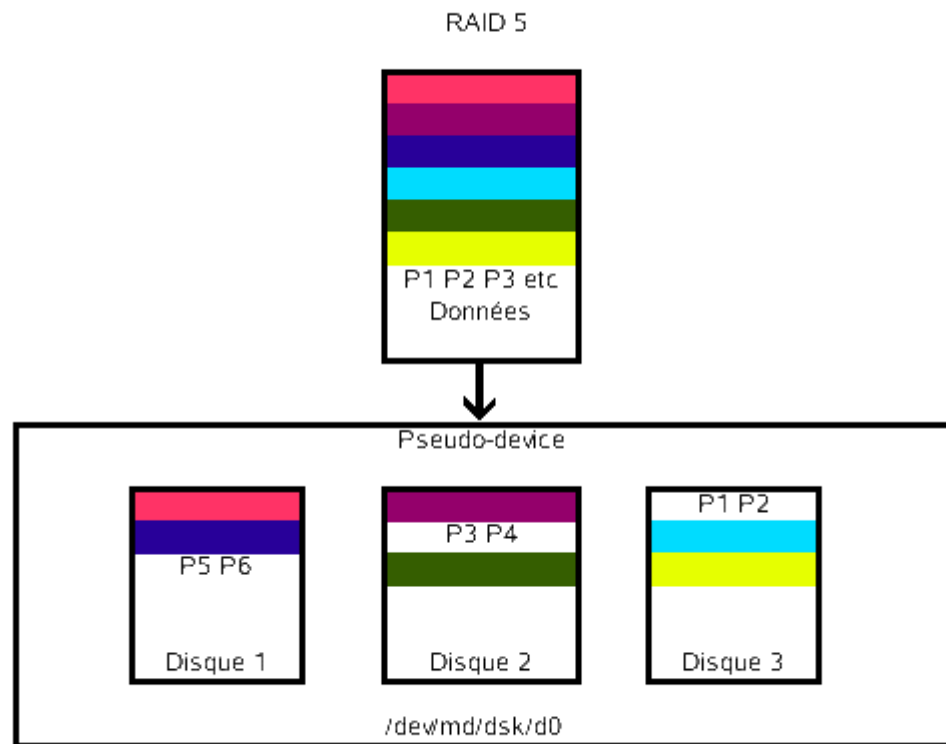
- Protection des données contre une défaillance d'un disque.

Inconvénients

- Création d'un goulot d'étranglement des données à cause de l'écriture des données de parité sur un seul disque.

RAID 5 - Striping avec Parité Distribuée

Le RAID 5 est une technologie avec bandes de parité distribuées sur plusieurs disques :



Avantages

- Protection des données contre une défaillance d'un disque,
- Evite le goulot d'étranglement d'un seul disque de parité.

Inconvénients

- Lecture moins performante qu'avec RAID 3 et 4.

Au delà de RAID 5

Il existe aussi deux autres technologies RAID, toutes deux issues de la technologie RAID 5 :

- RAID 6
 - *Disk Striping with Double Distributed Parity*
- RAID TP
 - *Disk Striping with Triple Distributed Parity*

RAID Logiciel sous Debian

Préparation du disque

Partitionnement

Suite à votre LAB de la leçon **Gestion des Disques et le Swap**, votre disque comporte 12 partitions :

```
root@debian8:~# fdisk -l
```

```
Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors
```

```
Units: sectors of 1 * 512 = 512 bytes
```

```
Sector size (logical/physical): 512 bytes / 512 bytes
```

```
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

```
Disklabel type: dos
```

```
Disk identifier: 0xac1e8183
```

Device	Boot	Start	End	Sectors	Size	Id	Type
/dev/sda1	*	2048	20482047	20480000	9.8G	83	Linux
/dev/sda2		20482048	41938943	21456896	10.2G	5	Extended
/dev/sda5		20484096	24578047	4093952	2G	82	Linux swap / Solaris

/dev/sda6	24610816	25587711	976896	477M	fd	Linux	raid	autodetect
/dev/sda7	25589760	25980927	391168	191M	8e	Linux	LVM	
/dev/sda8	25982976	26568703	585728	286M	8e	Linux	LVM	
/dev/sda9	26570752	27547647	976896	477M	fd	Linux	raid	autodetect
/dev/sda10	27549696	28329983	780288	381M	8e	Linux	LVM	
/dev/sda11	28332032	29308927	976896	477M	fd	Linux	raid	autodetect
/dev/sda12	29310976	30287871	976896	477M	fd	Linux	raid	autodetect
/dev/sda13	30289920	30699519	409600	200M	83	Linux		

Mise en Place du RAID 5 Logiciel

Dans le cas de cet exemple les quatre partitions concernées par la mise en place d'un RAID 5 sont :

/dev/sda6	24610816	25587711	976896	477M	fd	Linux	raid	autodetect
/dev/sda9	26570752	27547647	976896	477M	fd	Linux	raid	autodetect
/dev/sda11	28332032	29308927	976896	477M	fd	Linux	raid	autodetect
/dev/sda12	29310976	30287871	976896	477M	fd	Linux	raid	autodetect

La création d'une unité RAID avec la commande **mdadm** se fait grâce aux options passées en arguments à la commande :

```
mdadm --create <unité RAID> [options] <unités physiques>
```

Sous Debian 8, mdadm n'est pas installé par défaut :

```
root@debian8:~# apt-get install mdadm
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  mdadm
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 432 kB of archives.
```

```
After this operation, 1,233 kB of additional disk space will be used.
Get:1 http://ftp.fr.debian.org/debian/ jessie/main mdadm amd64 3.3.2-5+deb8u1 [432 kB]
Fetched 432 kB in 3s (123 kB/s)
Preconfiguring packages ...
Selecting previously unselected package mdadm.
(Reading database ... 82568 files and directories currently installed.)
Preparing to unpack .../mdadm_3.3.2-5+deb8u1_amd64.deb ...
Unpacking mdadm (3.3.2-5+deb8u1) ...
Processing triggers for man-db (2.7.0.2-5) ...
Processing triggers for systemd (215-17+deb8u4) ...
Setting up mdadm (3.3.2-5+deb8u1) ...
Generating mdadm.conf... done.
update-initramfs: deferring update (trigger activated)
update-rc.d: warning: start and stop actions are no longer supported; falling back to defaults
update-rc.d: warning: start and stop actions are no longer supported; falling back to defaults
Processing triggers for systemd (215-17+deb8u4) ...
Processing triggers for initramfs-tools (0.120+deb8u1) ...
update-initramfs: Generating /boot/initrd.img-3.16.0-4-amd64
W: mdadm: /etc/mdadm/mdadm.conf defines no arrays.
W: mdadm: no arrays defined in configuration file
```

Saisissez maintenant la commande suivante :

```
root@debian8:~# mdadm --create /dev/md1 --level=5 --raid-devices=3 /dev/sda6 /dev/sda9 /dev/sda11
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md1 started.
```

Les options dans la ligne de commande sont :

Option Courte	Option Longue	Description
-l	- -level	Le niveau RAID - linear, 0,1,2,4 ou 5
-n	- -raid-devices=<nombre>	Le nombre de périphériques actifs dans le RAID

Les autres options de la commande **mdadm** peuvent être visualisées grâce à la commande suivante :

```
root@debian8:~# mdadm --help-options
```

Any parameter that does not start with '-' is treated as a device name or, for --examine-bitmap, a file name.

The first such name is often the name of an md device. Subsequent names are often names of component devices.

Some common options are:

```
--help      -h      : General help message or, after above option,
                    mode specific help message
--help-options : This help message
--version    -V      : Print version information for mdadm
--verbose    -v      : Be more verbose about what is happening
--quiet      -q      : Don't print un-necessary messages
--brief      -b      : Be less verbose, more brief
--export     -Y      : With --detail, --detail-platform or --examine use
                    key=value format for easy import into environment
--force      -f      : Override normal checks and be more forceful

--assemble  -A      : Assemble an array
--build      -B      : Build an array without metadata
--create     -C      : Create a new array
--detail     -D      : Display details of an array
--examine    -E      : Examine superblock on an array component
--examine-bitmap -X : Display the detail of a bitmap file
--examine-badblocks: Display list of known bad blocks on device
--monitor    -F      : monitor (follow) some arrays
--grow       -G      : resize/ reshape and array
--incremental -I      : add/remove a single device to/from an array as appropriate
--query      -Q      : Display general information about how a
                    device relates to the md driver
--auto-detect : Start arrays auto-detected by the kernel
```

Les *modes majeurs* de la commande **mdadm** peuvent être visualisés grâce à la commande suivante :

```
root@debian8:~# mdadm --help
mdadm is used for building, managing, and monitoring
Linux md devices (aka RAID arrays)
Usage: mdadm --create device options...
        Create a new array from unused devices.
mdadm --assemble device options...
        Assemble a previously created array.
mdadm --build device options...
        Create or assemble an array without metadata.
mdadm --manage device options...
        make changes to an existing array.
mdadm --misc options... devices
        report on or modify various md related devices.
mdadm --grow options device
        resize/reshape an active array
mdadm --incremental device
        add/remove a device to/from an array as appropriate
mdadm --monitor options...
        Monitor one or more array for significant changes.
mdadm device options...
        Shorthand for --manage.
Any parameter that does not start with '-' is treated as a device name
or, for --examine-bitmap, a file name.
The first such name is often the name of an md device. Subsequent
names are often names of component devices.

For detailed help on the above major modes use --help after the mode
e.g.
    mdadm --assemble --help
For general help on options use
    mdadm --help-options
```

Dernièrement, chaque *mode majeur* dispose de son propre aide :

```
root@debian8:~# mdadm --assemble --help
Usage: mdadm --assemble device options...
       mdadm --assemble --scan options...
```

This usage assembles one or more raid arrays from pre-existing components.

For each array, mdadm needs to know the md device, the identity of the array, and a number of sub devices. These can be found in a number of ways.

The md device is given on the command line, is found listed in the config file, or can be deduced from the array identity.

The array identity is determined either from the --uuid, --name, or --super-minor commandline arguments, from the config file, or from the first component device on the command line.

The different combinations of these are as follows:

If the --scan option is not given, then only devices and identities listed on the command line are considered.

The first device will be the array device, and the remainder will be examined when looking for components.

If an explicit identity is given with --uuid or --super-minor, then only devices with a superblock which matches that identity is considered, otherwise every device listed is considered.

If the --scan option is given, and no devices are listed, then every array listed in the config file is considered for assembly.

The identity of candidate devices are determined from the config file. After these arrays are assembled, mdadm will look for other devices that could form further arrays and tries to assemble them. This can be disabled using the 'AUTO' option in the config file.

If the --scan option is given as well as one or more devices, then Those devices are md devices that are to be assembled. Their identity

and components are determined from the config file.

If mdadm can not find all of the components for an array, it will assemble it but not activate it unless `--run` or `--scan` is given. To preserve this behaviour even with `--scan`, add `--no-degraded`. Note that "all of the components" means as many as were present the last time the array was running as recorded in the superblock. If the array was already degraded, and the missing device is not a new problem, it will still be assembled. It is only newly missing devices that cause the array not to be started.

Options that are valid with `--assemble (-A)` are:

```
--bitmap=          : bitmap file to use with the array
--uuid=            -u : uuid of array to assemble. Devices which don't
                        have this uuid are excluded
--super-minor= -m  : minor number to look for in super-block when
                        choosing devices to use.
--name=            -N : Array name to look for in super-block.
--config=          -c : config file
--scan             -s : scan config file for missing information
--run              -R : Try to start the array even if not enough devices
                        for a full array are present
--force            -f : Assemble the array even if some superblocks appear
                        : out-of-date. This involves modifying the superblocks.
--update=          -U : Update superblock: try '-A --update=?' for option list.
--no-degraded      : Assemble but do not start degraded arrays.
--readonly         -o : Mark the array as read-only. No resync will start
```

Constatez maintenant les informations concernant le RAID 5 créé :

```
root@debian8:~# cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
md1 : active raid5 sda11[3] sda9[1] sda6[0]
      974848 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/3] [UUU]
```

```
unused devices: <none>
```

Grâce à la commande mdadm, il est possible d'obtenir d'avantage d'informations :

```
root@debian8:~# mdadm --query /dev/md1
/dev/md1: 952.16MiB raid5 3 devices, 0 spares. Use mdadm --detail for more detail.
```

L'option - **-detail** produit le résultat suivant :

```
root@debian8:~# mdadm --query /dev/md1
/dev/md1: 952.16MiB raid5 3 devices, 0 spares. Use mdadm --detail for more detail.
root@debian8:~# mdadm --detail /dev/md1
/dev/md1:
    Version : 1.2
    Creation Time : Wed Aug  3 13:20:07 2016
    Raid Level : raid5
    Array Size : 974848 (952.16 MiB 998.24 MB)
    Used Dev Size : 487424 (476.08 MiB 499.12 MB)
    Raid Devices : 3
    Total Devices : 3
    Persistence : Superblock is persistent

    Update Time : Wed Aug  3 13:20:16 2016
    State : clean
    Active Devices : 3
    Working Devices : 3
    Failed Devices : 0
    Spare Devices : 0


    Layout : left-symmetric
    Chunk Size : 512K


    Name : debian8:1 (local to host debian8)
    UUID : 41b8799a:e24feb35:39e9aa3a:ec920626
```

Events : 18

Number	Major	Minor	RaidDevice	State	
0	8	6	0	active sync	/dev/sda6
1	8	9	1	active sync	/dev/sda9
3	8	11	2	active sync	/dev/sda11

Notez la ligne **Persistence : Superblock is persistent**. En effet, cette implémentation de RAID inscrit les caractéristiques du volume dans un *super bloc* persistant en début de chaque unité de type bloc dans le volume.



Recherchez la signification des termes Layout, Chunk size, Major et Minor.

Cependant, il est nécessaire de renseigner le fichier **/etc/mdadm/mdadm.conf** afin que le RAID soit construit à chaque démarrage :

```
root@debian8:~# echo 'DEVICES /dev/sda6 /dev/sda9 /dev/sda11' > /etc/mdadm/mdadm.conf
root@debian8:~# mdadm --detail --scan >> /etc/mdadm/mdadm.conf
root@debian8:~# cat /etc/mdadm/mdadm.conf
DEVICES /dev/sda6 /dev/sda9 /dev/sda11
ARRAY /dev/md1 metadata=1.2 name=debian8:1 UUID=41b8799a:e24feb35:39e9aa3a:ec920626
```

Chaque unité peut être examinée individuellement :

```
root@debian8:~# mdadm --examine /dev/sda6
/dev/sda6:
    Magic : a92b4efc
    Version : 1.2
    Feature Map : 0x0
    Array UUID : 41b8799a:e24feb35:39e9aa3a:ec920626
    Name : debian8:1 (local to host debian8)
    Creation Time : Wed Aug 3 13:20:07 2016
    Raid Level : raid5
```

```
Raid Devices : 3
```

```
Avail Dev Size : 974848 (476.08 MiB 499.12 MB)
```

```
Array Size : 974848 (952.16 MiB 998.24 MB)
```

```
Data Offset : 2048 sectors
```

```
Super Offset : 8 sectors
```

```
Unused Space : before=1960 sectors, after=0 sectors
```

```
State : clean
```

```
Device UUID : be46c991:3ad0ce5e:0bf2ec4e:1de865aa
```

```
Update Time : Wed Aug 3 13:20:16 2016
```

```
Bad Block Log : 512 entries available at offset 72 sectors
```

```
Checksum : 16f72803 - correct
```

```
Events : 18
```

```
Layout : left-symmetric
```

```
Chunk Size : 512K
```

```
Device Role : Active device 0
```

```
Array State : AAA ('A' == active, '.' == missing, 'R' == replacing)
```

A ce stade il est intéressant de noter comment réagir lors d'une défaillance d'un disque. Dans notre cas nous allons indiquer au système que la partition /dev/sda6 est devenue défaillante :

```
root@debian8:~# mdadm --manage --set-faulty /dev/md1 /dev/sda6
mdadm: set /dev/sda6 faulty in /dev/md1
```

L'utilisation de la ligne de commande suivante nous confirme le statut de /dev/sda6 :

```
root@debian8:~# mdadm --detail /dev/md1
/dev/md1:
  Version : 1.2
  Creation Time : Wed Aug 3 13:20:07 2016
  Raid Level : raid5
```

```
Array Size : 974848 (952.16 MiB 998.24 MB)
Used Dev Size : 487424 (476.08 MiB 499.12 MB)
Raid Devices : 3
Total Devices : 3
Persistence : Superblock is persistent

Update Time : Wed Aug 3 13:27:53 2016
State : clean, degraded
Active Devices : 2
Working Devices : 2
Failed Devices : 1
Spare Devices : 0

Layout : left-symmetric
Chunk Size : 512K

Name : debian8:1 (local to host debian8)
UUID : 41b8799a:e24feb35:39e9aa3a:ec920626
Events : 20

Number Major Minor RaidDevice State
0 0 0 0 removed
1 8 9 1 active sync /dev/sda9
3 8 11 2 active sync /dev/sda11

0 8 6 - faulty /dev/sda6
```

Il est maintenant nécessaire de supprimer /dev/sda6 de notre RAID 5 :

```
root@debian8:~# mdadm --manage --remove /dev/md1 /dev/sda6
mdadm: hot removed /dev/sda6 from /dev/md1
```

A l'examen de notre RAID, on constate que /dev/sda6 a été supprimé :

```
root@debian8:~# mdadm --detail /dev/md1
/dev/md1:
    Version : 1.2
  Creation Time : Wed Aug  3 13:20:07 2016
    Raid Level : raid5
    Array Size : 974848 (952.16 MiB 998.24 MB)
  Used Dev Size : 487424 (476.08 MiB 499.12 MB)
    Raid Devices : 3
  Total Devices : 2
    Persistence : Superblock is persistent

    Update Time : Wed Aug  3 13:29:30 2016
      State : clean, degraded
  Active Devices : 2
Working Devices : 2
  Failed Devices : 0
  Spare Devices : 0


    Layout : left-symmetric
  Chunk Size : 512K


    Name : debian8:1 (local to host debian8)
    UUID : 41b8799a:e24feb35:39e9aa3a:ec920626
    Events : 21


   Number  Major   Minor  RaidDevice State
     0         0         0         0    removed
     1         8          9         1  active sync  /dev/sda9
     3         8        11         2  active sync  /dev/sda11
```

Constatez maintenant l'existence de votre RAID :

```
root@debian8:~# cat /proc/mdstat
Personalities : [raid6] [raid5] [raid4]
```

```
md1 : active raid5 sda11[3] sda9[1]
      974848 blocks super 1.2 level 5, 512k chunk, algorithm 2 [3/2] [_UU]
unused devices: <none>
```



Notez que le RAID a été démarré avec 2 unités au lieu de trois.

Pour ajouter un autre disque à notre RAID afin de remplacer /dev/sda6 il convient d'utiliser l'option **-add** :

```
root@debian8:~# mdadm --manage --add /dev/md1 /dev/sda12
mdadm: added /dev/sda12
```

L'examen du RAID indique que /dev/sda12 a été ajouté en tant que *spare* et à l'issue de quelques secondes le RAID 5 a été reconstruite :

```
root@debian8:~# mdadm --manage --add /dev/md1 /dev/sda12
mdadm: added /dev/sda12
root@debian8:~# mdadm --detail /dev/md1
/dev/md1:
    Version : 1.2
  Creation Time : Wed Aug  3 13:20:07 2016
    Raid Level : raid5
    Array Size : 974848 (952.16 MiB 998.24 MB)
  Used Dev Size : 487424 (476.08 MiB 499.12 MB)
    Raid Devices : 3
    Total Devices : 3
 Persistence : Superblock is persistent

    Update Time : Wed Aug  3 13:35:24 2016
      State : clean
 Active Devices : 3
Working Devices : 3
 Failed Devices : 0
```

Spare Devices : 0

Layout : left-symmetric

Chunk Size : 512K

Name : debian8:1 (local to host debian8)

UUID : 41b8799a:e24feb35:39e9aa3a:ec920626

Events : 40

Number	Major	Minor	RaidDevice	State	
4	8	12	0	active sync	/dev/sda12
1	8	9	1	active sync	/dev/sda9
3	8	11	2	active sync	/dev/sda11

Naturellement, il nécessaire de renseigner le fichier **/etc/mdadm/mdadm.conf** du changement afin que le RAID soit construit à chaque démarrage :

```
root@debian8:~# echo 'DEVICES /dev/sda12 /dev/sda9 /dev/sda11' > /etc/mdadm/mdadm.conf
root@debian8:~# mdadm --detail --scan >> /etc/mdadm/mdadm.conf
root@debian8:~# cat /etc/mdadm/mdadm.conf
DEVICES /dev/sda12 /dev/sda9 /dev/sda11
ARRAY /dev/md1 metadata=1.2 name=debian8:1 UUID=41b8799a:e24feb35:39e9aa3a:ec9206260
```

<html>

Copyright © 2020 Hugh Norris.

</html>

