Cours de systèmes d'exploitation centralisés

Chapitre 5

Systèmes de gestion de fichiers

Séance 3

Mme L. BOUZAR

2021-2022

SGF EXT 4 (Linux)

Utilise des extents

- ensemble de blocs contigus. Exemple 128 Mo de blocs de 4Ko.
- Ils ne requierent pas de metadata (au niveau de l'inode) pour chaque bloc.
- Les opérations sont plus rapides.
- Accepte des fichiers plus grands que ses prédécesseurs Ex: pour des blocs de 1Ko, la taille max d'un fichier passe de 16 Go à 16 To. Et la taille max de la partition passe a 1 Exa octets.

Journaling file system on y enregistre toute modification dans les données ainsi que dans le SGF lui même de manière séquentielle. Pour contrer la perte de données après un bug.

Partition en groupe de blocs comme ext 2

Utilise 48 bits pour l'adresse de chaque extent.

4.4.6 Les liens

- Un lien permet de référencer un fichier avec plusieurs noms.
- On distingue deux sortes de liens :
  1. Lien physique (ou hard link),
  2. Lien symbolique (ou symbolic link).

4.4.6.1 Lien physique (ou hard link)

- ✓ Permet d'associer plusieurs noms(liens) à un fichier : le fichier doit se trouver dans le même système de fichie
- ✓ Au niveau implémentation, il y a un seul inode : c'est celui du fichier original.
- ✓ Tous les noms de liens pointent vers le même inode.
- ✓ Suppression du fichier ➔ Suppression d'un lien .
  Le fichier n'est supprimé que si le nombre de liens est égal à zéro.
- ✓ **Commande:** ln fichier_original fichier_lien_physique

4.4.6.2 Lien symbolique(symbolic link) (raccourci)

- ✓ Introduit dans la version 4.2BSD,
- ✓ Il permet d'associer un ou plusieurs noms(liens) à un fichier ou répertoire se trouvant dans le même système de fichiers ou dans un autre système de fichiers.
- ✓ C'est un fichier de type lien qui possède son propre inode,
- ✓ Les données d'un lien symbolique correspondent au chemin du fichier original.
- ✓ **Suppression** :
  - ❖ La suppression d'un lien symbolique n'a pas d'influence sur le fichier original auquel il se réfère.
  - ❖ La suppression ou le déplacement du fichier original "casse" le lien avec le lien symbolique:
    Le lien symbolique existe mais ne peut pas accéder au fichier.
- ✓ **Commande :**

```
ln -s  fichier_original  fichier_lien_symbolique
```

Exemples:

```
$ ln   cmd   cmdlp                /* liens physique sur un fichier*/
$ ln -s cmd  cmdls                /* liens symbolique sur un fichier*/
$ ls -li  cmd*
   135632 -rwxr-xr-x    3   bs   bs   2641   22 déc.  11:45 cmd
   135632 -rwxr-xr-x    3   bs   bs   2641   22 déc.  11:45 cmdlp
   136264 lrwxrwxrwx    1   bs   bs      3    25 mai   09:32 cmdls ->
   cmd

$ ln -s   ./test   testlsymbrep     /* liens symbolique sur un
   répertoire*/
$ ln   ./test   testlphrep          /* lien physique sur un
   répertoire*/
ln: « ./test »: lien direct non permis pour un répertoire,

drwxr-xr-x   2   bs  bs   4096   19 déc.  10:49 test
lrwxrwxrwx   1   bs  bs      6   20 mai   14:33 testlsymbrep -> ./test

$ cd testlsymbrep
```

```
$ ln -s /home   HOME
lrwxrwxrwx 1 bs bs 5 20 mai   14:57 HOME -> /home
 $ cd HOME
/HOME$ ls
bs

/HOME$ ln -s /usr   usr
ln: création d'un lien symbolique « usr »: Permission non accordée
/HOME$ ln -s /etc   etc
ln: création d'un lien symbolique « etc »: Permission non accordée

/HOME$ su
/HOME# ln -s /usr usr
lrwxrwxrwx  1 root root    4 20 mai   14:59 usr -> /usr
 /HOME# ls usr
bin  games  include  lib  lib64  local    sbin  share  src
/HOME# ls /usr
bin  games  include  lib  lib64  local    sbin  share  src
```

4.4.5 La protection des fichiers et répertoires Unix

4.4.5.1 Définition de la protection Unix

- Unix utilise la protection par classe d'utilisateurs ;
- Il existe trois classes d'utilisateurs:
  - ✓ Le propriétaire,
  - ✓ les membres du groupe et
  - ✓ les autres utilisateurs.
- Les opérations sur un fichier peuvent être:
  - ✓ lecture(R),
  - ✓ écriture (W),
  - ✓ exécution(X).

Pour réaliser cette protection, Unix utilise neuf (9) bits, trois bits par classe: Un bit pour la lecture, un bit pour l'écriture et le dernier pour l'exécution.

- ✓ L'opération est autorisée si le bit correspondant est positionné à '1'.
- ✓ L'opération est interdite, si le bit contient la valeur 0.

- Trois autres bits spéciaux sont également utilisés :
  - ✓ Le Sticky bit,
  - ✓ le Set GID,
  - ✓ le Set UID.

Bits Spéciaux			Propriétaire			Groupe			Autres		
Set UID	Set GID	Sticky bit	**r**	**w**	**x**	**r**	**w**	**x**	**r**	**w**	**x**

- L’inode :
  - Type de fichier,
  - Nombre de liens,
  - Identificateur du propriétaire(UID),
  - Identificateur du groupe(GID),
  - Taille du fichier en octets,
  - Date de création ou Date de la dernière modification,
  - Date de la dernière utilisation(dernier accès),
  - Date de la dernière modification de l’inode,
  - **Protection ou droit d'accès (12 bits),**
  - Table d’index d’allocation du fichier (13 mots de 32 bits),
  - *Autres informations selon les versions.*

	Fichier	Répertoire
R	Lecture autorisée	Lister(ls), copier(cp), lire les fichiers du répertoire.
W	Ecriture autorisée	Créer ou supprimer les fichiers du répertoire.
X	Exécution autorisée	Parcourir le répertoire et son arborescence. En l'absence de ce droit, aucun accès au répertoire et à son arborescence n'est possible.
Sticky Bit	Maintenir le programme en mémoire après la fin de son exécution.	Un fichier de ce répertoire ne peut être détruit que par son propriétaire.
Set GID	Permet d'exécuter le programme avec **les droits du groupe auquel il appartient.**	Tout fichier créé dans ce répertoire aura le groupe du répertoire et non le groupe du propriétaire.
Set UID	Permet d'exécuter le programme avec **les droits du propriétaire** de ce programme.	Non utilisé

Exemples : Protection des fichiers et répertoires

- Le fichier F1 a la protection suivante : « 111 101 000 » ⇒
  - ✓ le propriétaire a tous les droits,
  - ✓ les membres du groupe peuvent lire et exécuter,
  - ✓ les autres utilisateurs ne peuvent pas utiliser ce fichier.

- Le fichier F2 a la protection suivante : « 101 101 101 » ⇒
  - ✓ le propriétaire, les membres du groupe et autres utilisateurs ont les mêmes droits, ils peuvent lire et exécuter le fichier.

- Le répertoire R1 a la protection suivante : « 111 101 000 » ⇒
  - ✓ le propriétaire a tous les droits,
  - ✓ les membres du groupe peuvent lire, lister les fichiers du répertoire et parcourir le répertoire,
  - ✓ les autres utilisateurs ne peuvent pas utiliser ce répertoire.

4.4.5.2 Commande chmod pour modifier la protection

- La commande chmod peut être utilisée en deux modes :
  1. Mode symbolique,
  2. Mode absolu.

1) Mode symbolique

Chmod ***qui opération autorisations*** *fichier(s)*

- **Qui :** spécifie pour qui les autorisations doivent être modifiées.
  - ✓ **u** : propriétaire
  - ✓ **g** : membre du groupe
  - ✓ **o** : les autres utilisateurs
  - ✓ **a** : tous les utilisateur(u+g+o)
- **Opération :** Indique l'opération à effectuer
  - ✓ **=** : affecter,
  - ✓ **+** : ajouter,
  - ✓ **-** : retirer (ou supprimer)

- **Autorisations :** Spécifient les autorisations à modifier.
  - ✓ r: lecture
  - ✓ w: écriture
  - ✓ x: exécution
  - ✓ s : activé le bit setuid ou bit setgid
  - ✓ t : activé le sticky bit

`chmod g=rwx f`	les membres du groupe ont tous les droits : lecture, écriture, exécution.
`chmod g-w f`	Retirer aux membres du groupe le droit d'écriture.
`chmod o-rwx f`	Retirer aux autres utilisateurs tous les droits.
`chmod u=rx f`	Propriétaire = lecture + exécution.
`chmod o+r f`	Rajouter aux autres utilisateurs le droit de lecture.

2) Mode absolu

Chmod valeur fichier(s)

- **Valeur :** valeur octale définissant les opérations autorisées sur le fichier.

```
# ls -l  f
-rwxr-xr--   1 bs bs 0 19 déc.  09:32  f

# chmod u+s  f          / *  chmod 4754  f  */
# ls -l  f
-rwsr-xr--    1 bs bs 0 19 déc.  09:32 f

# chmod g+s   f          /*  chmod 6754  f  <==> chmod  ug+s f */
# ls -l f
-rwsr-sr--    1 bs bs 0 19 déc.  09:32 f

# chmod ug-s  f          /*  chmod 0754  f */
# ls -l   f
-rwxr-xr--   1 bs bs 0 19 déc.  09:32  f
```

Exemple de chmod g+s répertoire (/home)

/home$ su /* mode administrateur*/

/home# chmod 777 /home /* tous les droits pour tous */

/home# su bs /* utilisateur bs*/

/home$ mkdir rep

/home$ ls -l

drwxr-xr-x 2 bs bs 4096 20 mai 16:18 rep

Le répertoire rep a comme propriétaire bs et groupe bs .

/home$ su /* mode administrateur*/

/home# **chmod g+s /home** /* ou **chmod 2777 /home** */

/* Les fichiers et repertoires créés dans /home auront le groupe du propriétaire de /home : le propriétaire et le groupe de /home est : **root** */

/home# su bs /* utilisateur bs*/

/home$ mkdir rep2 /* créer un répertoire rep2 */

/home$ ls -la

drwxrwsrwx 5 root root 4096 20 mai 16:19 .

drwxr-xr-x 23 root root 4096 28 févr. 2013 ..

drwxr-xr-x 2 bs bs 4096 20 mai 16:18 rep

drwxr-sr-x 2 bs root 4096 20 mai 16:19 rep2

/* groupe de rep2 = root et Set GID = 1*/

4.5 Système de fichiers FAT32

4.5.1 Structure d'un système de fichiers FAT32

- Une partition FAT32 est composée de:
  - Un secteur boot,
  - une zone réservée,
  - une table d'allocation FAT (deux copies),
  - l'espace réservé aux répertoires et fichiers : Cet espace est divisé en blocs appelés clusters. Le cluster est l'unité d'allocation de l'espace disque.

Structure d'une Partition FAT32

<table>
<tr><th colspan="4">MBR(Master Record Boot): 512 octets</th></tr>
<tr><td>Partition FAT32</td><td>Partition2</td><td>Partiton 3</td><td>Partition4</td></tr>
<tr><td>- Secteur Boot
- Zone réservée(avec secteur boot):
Octets 14-15 du secteur boot</td><td></td><td></td><td></td></tr>
<tr><td>- FAT1
- FAT2</td><td></td><td></td><td></td></tr>
<tr><td>Clusters
Répertoires et Fichiers</td><td></td><td></td><td></td></tr>
</table>

1. Secteur boot

- Le premier secteur d'une partition contient:
  - ✓ Le programme de démarrage appelé '**bootstrap' ou secteur boot**,
  - ✓ Une table contenant les paramètres du BIOS et des informations concernant le système de fichiers(section FAT).

Structure du secteur Boot d'une Partition FAT32

Dépl	Contenu	Taille
00	Instruction de saut : JMP; (EB**xx**90)	3
03	Nom fabricant et n° de version	8
11-35	Bloc de paramètres du BIOS (BIOS Parameters Block)	25
36-89	Informations concernant le système de fichiers (section FAT32)	54
xx-509	Suite du Programme de démarrage ou bootstrap	420
510-511	Signature hexa : AA55 (Little Endian : 55AA)	2

2. Bloc des paramètres du BIOS (BIOS Parameters Block)

Position	Contenu	Taille
11	Nombre d'octets par secteur	2
13	Nombre de secteurs par cluster	1
14	Nombre de secteurs réservés	2
16	Nombre de FAT	1
17	Nombre d'entrées pour le répertoire racine «\» (**FAT 12 et 16**)	2
19-20	Nombre de secteurs par partition(partition de petite capacité < 32Mo)	2
21	Descripteur du support (valeur hexadécimale, disque : dur F8)	1
22	Nombre de secteurs par FAT(FAT12 et 16; Pour FAT32 octets 36 à 39)	2
24	Nombre de secteurs par piste	2
26	Nombre de têtes de lecture-écriture	2
28	Nombre de secteurs avant la partition (0 si disque non partitionné)	4
32	Nombre de secteurs de la partition (si positions 19,20 = 0)	4

3. Section FAT

Position	Contenu	Taille
36	Nombre de secteurs par FAT32 (remplace positions 22-23)	4
40	Flags : utilisé pour le « mirroring » de la FAT	2
42	Version du système de fichiers	2
44	Premier cluster du répertoire racine(\) : **cluster 2** (cette valeur peut changer si cluster 2 est défectueux)	4
48	Numéro de Secteur dans la zone réservée (secteur 1) du fichier d'information système : (Fsinfo sector)	2
50	Copie du Secteur boot (secteur 6 de la zone réservée)	2
52	12 octets réservés (12*0)	12
64	BIOS driver : valeur hexa : 8x (exemple : 80)	1
65	Réservé : (utilisé par Windows NT)	1
66	Extension de la signature boot (0x29) : cela signifie que les **3** champs suivants existent dans le secteur boot.	1
67	Numéro de série du volume	4
71	Label de volume(nom donné lors du formatage de la partition)	11
82	Système de fichiers : 'FAT32 '	8

4.5.2 Les répertoires

- **Le répertoire racine '\'**

  Il est créé lors du formatage du disque logique(partition).

4.5.2.1 Entrée d'un répertoire

- Une entrée d'un répertoire peut être:
  - ✓ Un fichier,
  - ✓ Un répertoire

- A la création d'un répertoire, le système crée deux répertoires '.' et '..'
  - '.' : Répertoire courant,
  - '..' : Répertoire père.

Chaque entrée est constituée de **32 octets**.

L**e nombre maximal d'entrées des répertoires** FAT32 est égal à 65534.

On distingue deux types d'entrées :

1- Entrée pour nom en format court(11 caractères(8.3)) : nom.extension ;

2- Entrée pour nom en format long(255 caractères).

- Dans la zone (adresse d'implantation) réservée au numéro du premier cluster on trouve:
  - Le N° du premier cluster du répertoire courant pour le répertoire '.' ,
  - Le N° du premier cluster du père pour le répertoire '..' ; ou
    Zéro si le répertoire parent est '\'.
  - La zone réservée à la taille du fichier contient zéro.
- Ces répertoires(. et ..) permettent de remonter dans l'arborescence.

4.5.2.2 Structure d'une entrée format court

- Le nom d'un fichier est composé de deux parties:
  - ✓ le nom sur 1 à 8 caractères ascii ,
  - ✓ l'extension de 0 à 3 caractères ascii.
- **Exemple** : command.com prog.c ; Le point séparant le nom de l'extension n'est pas représenté.
- **Remarque** :
  - ➢ la suppression d'une entrée (descripteur de fichier) du répertoire est réalisée en affectant la valeur **0xE5** au premier octet de l'entrée : c'est une suppression logique.

Structure d'une entrée format court

dépl	Taille	Contenu
00	8	Nom : Caractères ASCII
08	3	Extension : caractères ASCII
11	1	Attribut (Bit0: lecture seule, Bit1: caché, Bit2: système, Bit3: label de volume, Bit4:répertoire, Bit5:archive, 2 bits non utilisés)
12	1	Réservé pour Windows NT
13	1	Partie « millième de seconde » du temps de création
14	2	Heure de création: secondes(0 à 29), minute(0 à 59) , heure(0 à 23)
16	2	Date de création(date calculée à partir de 01/01/1980)
18	2	Date du dernier accès
20	2	**Numéro du premier cluster dans la FAT (poids fort pour FAT32)**
22	2	Heure de la dernière modification
24	2	Date de la dernière modification
26	2	Numéro du premier cluster dans la FAT (poids faible)
28	4	**Taille du fichier en octets(taille maxi= 4Giga octets)**

4.5.2.2 Structure d'une entrée format long

- Le nom d'un fichier est composé de 1 à 255 caractères.
- Un fichier est représenté dans le répertoire au moyen de :
  - **Une à vingt entrées de format long**
    - ✓ Une entrée = 32 octets.
    - ✓ Chaque entrée peut contenir au plus 13 caractères du nom.
    - ✓ Les caractères constituant les noms en format long sont représentés en UNICODE (UTF16: 2 octets par caractère),
  - **Une entrée format court.**

    Dans l'entrée en format court, le nom est constitué des six premiers caractères(lettres ou chiffres) suivi du caractère **~** qui est suivi d'un chiffre(numéro) et enfin l'extension(sans le point).
    - ✓ Le nom est converti en Majuscule.
    - ✓ Les caractères constituant les noms en format court sont représentés en ASCII (1 octet par caractère).

Entrées composant un nom en format long

Dernière entrée format long d'un fichier f
Avant dernière entrée format long d'un fichier f
… … …
Deuxième entrée format long d'un fichier f
Première entrée format long d'un fichier f
Entrée format court d'un fichier f

- **Contenu d'une entrée format Long :**

dépl	Taille	Contenu
00	1	N° de l'entrée ; bit 6 (à partir de 0) de la dernière entrée = 1
01	10	Nom : Caractères 1-5 (UNICODE : UTF16)
11	1	Attribut LFN = '**0F** ' : Nom en format long
12	1	Réservé
13	1	Caractère de contrôle : checksum
14	12	Suite du nom : caractères 6-11 (UNICODE)
26	2	0000
28	4	Suite du nom : caractères 12-13 (UNICODE)

- L'attribut LFN=**0F** ➔ entrée d'un nom en format long.

- **Exemple de nom de fichier : "nom en format long.exemple.doc"**
  - ✓ Nom de fichier = 30 caractères ➔ 3 entrées en format long
  - ✓ Nom en Format court = **NOMENF~1DOC** (le point '.' n'est pas représenté)

Entrées format long	43	. 2E00	d 6400	o 6F00	c 6300	0000	0F	00	33	FFFF	FFFF	FFFF	FFFF	FFFF	FFFF	0000	FFFF	FFFF
	02	2000	l 6C00	o 6F00	n 6E00	g 6700	0F	00	33	. 2E00	e 6500	x 7800	e 6500	m 6D00	p 7000	0000	l 6C00	e 6500
	01	n 6E00	o 6F00	m 6D00	2000	e 6500	0F	00	33	n 6E00	2000	f 6600	o 6F00	r 7200	m 6D00	0000	a 6100	t 7400
Entrée format court		N 4E	O 4F	M 4D	E 45	N 4E	F 46	~ 7E	1 31	D 44	O 4F	C 43	attribut 20	00	1/10 sec B5	Date et heure création ECAE	D234	Date acces D234	cluster P.fort 0500	Date et heure modification 78B0	D234	cluster P.faible D4D8	Taille en octets 00920000

- **Une seule entrée en format court :**
  - Pour les noms de fichiers/répertoires en lettres majuscules ou en lettres minuscules et de longueur inférieure ou égale à **8 caractères**,
  - Pour les répertoire **.** et **..**
- Le répertoire racine \ ne contient pas de répertoires **.** et **..**

4.5.3 Allocation dans le système de fichiers FAT32

- L'espace disque réservé aux fichiers et aux répertoires est divisé en n clusters numérotés de 2 à n+1.
- Chacun d'eux est composé de 2^k secteurs ($k \geq 0$).
- L'unité d'allocation est le cluster.
- L'espace est alloué dynamiquement au fichier: un cluster à la fois.
- L'espace disque est représenté dans un fichier d'allocation que l'on appelle FAT(File Allocation Table).
- L'entrée numéro i de la FAT correspond au cluster numéro i de l'espace disque.
- Une entrée peut être sur 12 bits (FAT12), sur 16 bits (FAT16) ou 32 bits (FAT32).
- Les deux premières entrées sont réservées au système:
  Le premier octet contient le code caractéristique du support (ex : F8 disque dur) le reste des octets contiennent la valeur 'FF'.

- Une entrée de la FAT32 peut contenir les valeurs suivantes:

Valeur	Signification
0x00000000	Cluster libre
0x0FFFFFF0 – 0x0FFFFFF6	Cluster réservé
0x0FFFFFF7	Cluster défectueux
0x0FFFFFF8 – 0x0FFFFFFF	Dernier cluster du fichier(EOF)
Autres valeurs supérieures à 2 : (28 bits de droites)	N° du cluster suivant dans le fichier. **Premier cluster da la FAT = 2**

- Les entrées correspondant aux clusters alloués aux fichiers sont chainées entre elles.

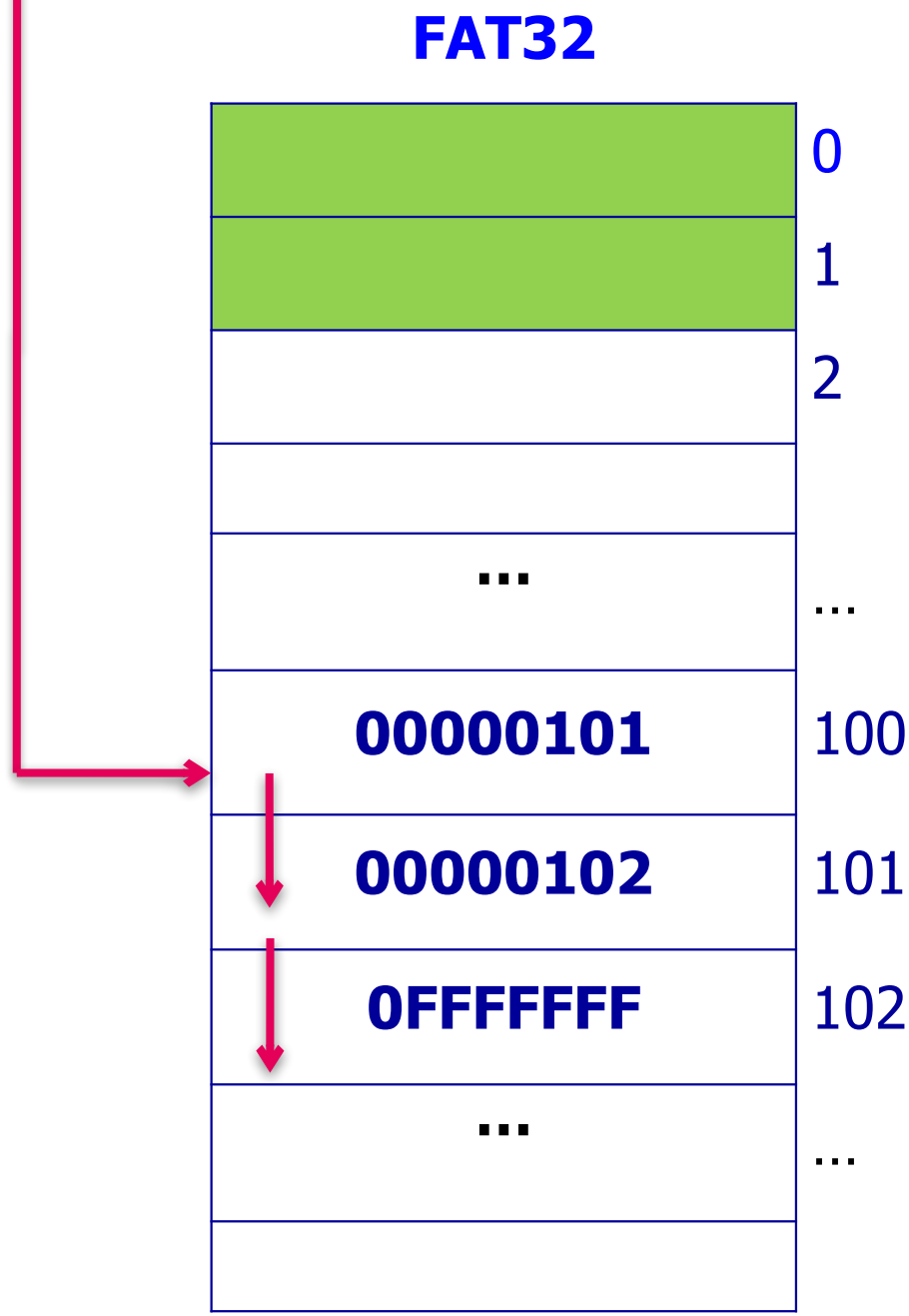
➢ Adresse LBA de la FAT = Adresse début LBA de la partition + nombre de secteurs réservés.

➢ Adresse LBA premier cluster = Adresse début LBA de la partition + nombre de secteurs réservés + Nombre FAT * Nombre secteurs par FAT.

➢ Adresse LBA d'un cluster i = Adresse LBA premier cluster + (i - 2) * nombre secteurs par cluster.

Fichier1	...	1er Cluster 0100					

Fichier1 avec 3 clusters



La taille d'un cluster dépend de la capacité des disques:

Capacité disque	Taille du cluster
513 Mo-8 Go	4 Ko
8 Go - 16 Go	8 Ko
16 Go - 32 Go	16 Ko
32 Go et plus	32 Ko

==========