



**I106B**  
**14. inode**

# ls -l

```
professeur@Pinux:~$ ls -l
total 12
drwxrwxr-x 2 professeur professeur 4096 mai 26 09:49 Documents
drwxrwxr-x 2 professeur professeur 4096 mai 26 09:50 Images
drwxrwxr-x 2 professeur professeur 4096 mai 26 09:50 Musique
```

- ▶ Chaque fichier possède de nombreuses caractéristiques :
  - drwxrwxr-x
  - 2
  - professeur professeur
  - 4096
  - mai 26 09:49
- ▶ Ce sont les **propriétés** des fichiers, retenues dans un **inode**.

Un **nœud d'index** ou **inode** (contraction de l'anglais *index* et *node*) est une structure de données contenant des informations à propos d'un fichier ou répertoire stocké dans certains systèmes de fichiers (notamment de type Linux/Unix).

À chaque fichier correspond un **numéro d'inode** (i-number) dans le système de fichiers dans lequel il réside, unique au périphérique sur lequel il est situé.

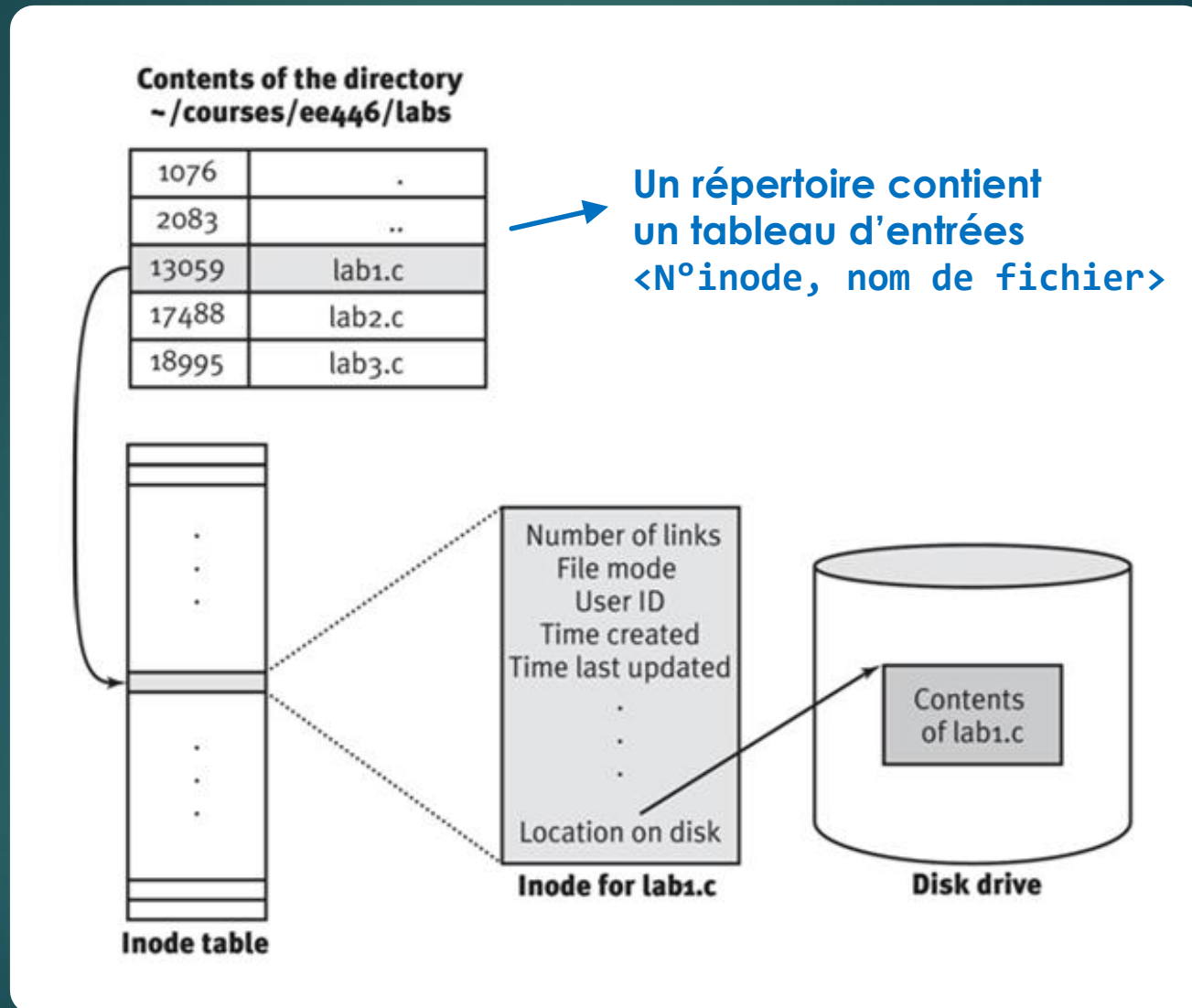
Chaque fichier a un seul inode, même si ce fichier peut avoir plusieurs noms (chacun de ceux-ci fait référence au même inode). Chaque nom est appelé **lien**.

# inode

- ▶ Identifiant du fichier sur la partition
  - le lien entre le nom du fichier et son inode est réalisé dans le répertoire
  - l'inode pointe vers le contenu du fichier
  - l'inode retient les propriétés du fichier (contient notamment les **métadonnées** des fichiers, et en particulier celles concernant les droits d'accès)
- ▶ Commande utile
  - `ls -li` → affiche les numéros d'inodes (i-numbers)

# Fichier dans un dossier

5



- Affiche les principales informations de l'inode d'un fichier

taille (en bytes)

numéro d'inode

nombre de liens physiques

```
anthony.legrand@courslinux:~$ stat find.sh
Fichier : find.sh
  Taille : 54          Blocs : 8          Blocs d'E/S : 4096      fichier
Périphérique : 801h/2049d      Inœud : 3016064      Liens : 1
Accès : (0755/-rwxr-xr-x)  UID : ( 1014/anthony.legrand)  GID : ( 1002/students)
  Accès : 2019-03-09 11:03:56.366448568 +0100
Modif. : 2019-02-20 11:47:04.945375801 +0100
Changt : 2019-02-20 11:47:04.945375801 +0100
  Créé : -
```

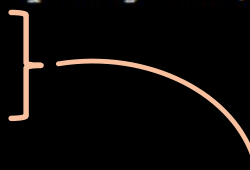
permissions

propriétaire

groupe propriétaire

- Affiche les principales informations de l'inode d'un fichier

```
anthony.legrand@courslinux:~$ stat find.sh
Fichier : find.sh
  Taille : 54          Blocs : 8          Blocs d'E/S : 4096    fichier
Périphérique : 801h/2049d      Inœud : 3016064      Liens : 1
Accès : (0755/-rwxr-xr-x)  UID : ( 1014/anthony.legrand)  GID : ( 1002/students)
Accès : 2019-03-09 11:03:56.366448568 +0100
Modif. : 2019-02-20 11:47:04.945375801 +0100
Changt  : 2019-02-20 11:47:04.945375801 +0100
Créé : -
```



**Accès** = date du dernier accès (*atime*)

**Modif.** = date de la dernière modification (*mtime*)

**Changt** = date du dernier changement de l'inode (*ctime*)



# Lien physique

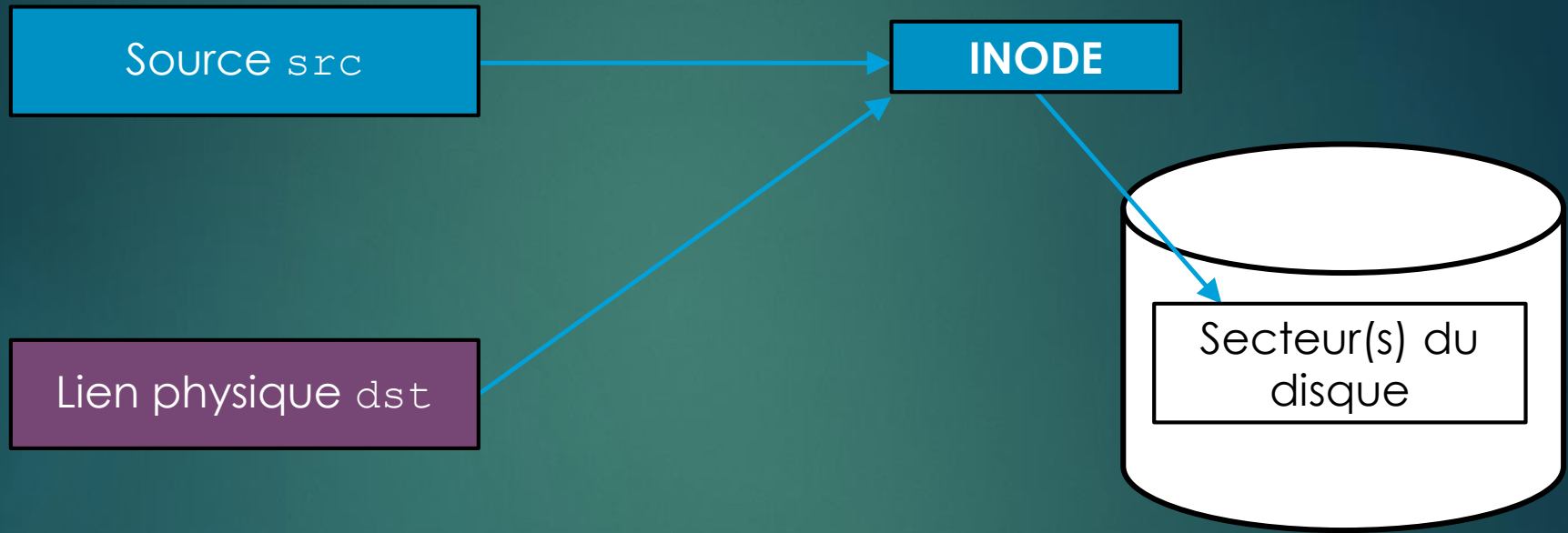
8

- ▶ Permet d'accéder à un fichier à partir d'un autre chemin d'accès (autre nom et/ou répertoire)
  - en créant un nouveau fichier dans un répertoire
  - mais pointant vers l'inode d'un fichier déjà existant
  - restreint aux fichiers d'une même partition
- ▶ Commande :  
**ln src dst**
- ▶ Les deux fichiers `src` et `dst` pointent vers le **même inode**, et donc vers le **même contenu**!



# Lien physique

9



→ Si on supprime la source `src`, le fichier est toujours présent via son lien physique `dst`.

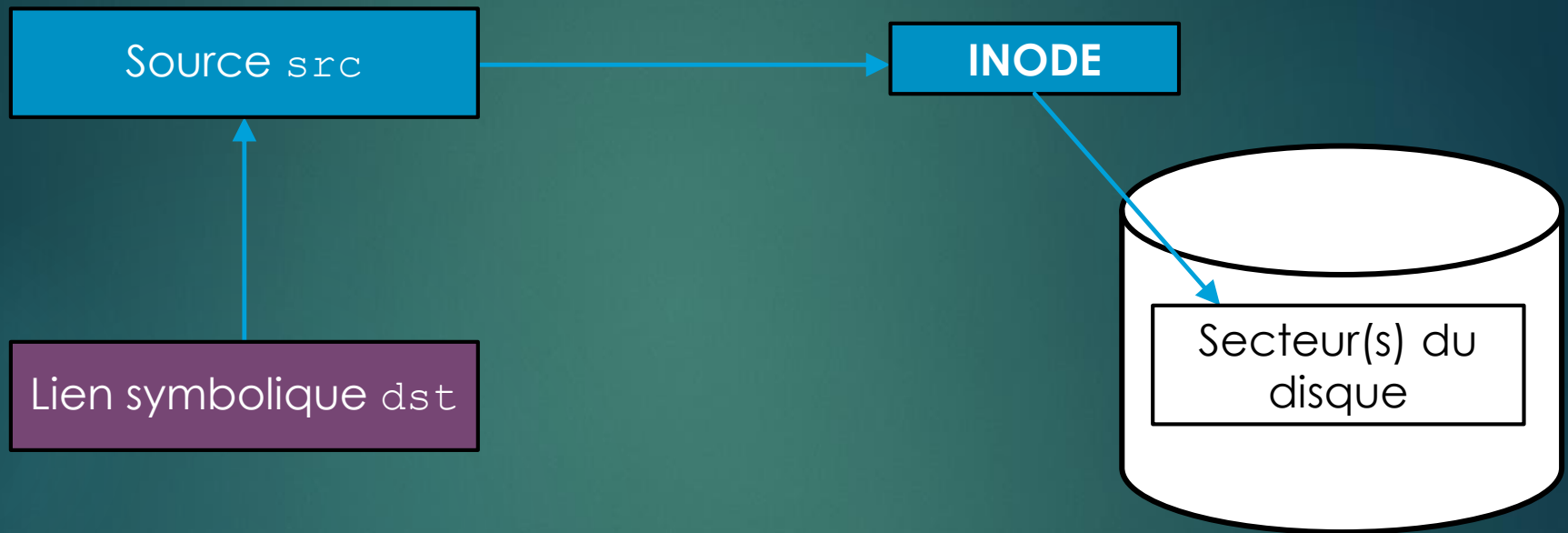
# Lien symbolique

10

- ▶ Permet d'accéder à un fichier à partir d'un autre chemin d'accès (autre nom et/ou répertoire)
  - en créant un nouveau fichier dans un répertoire
  - qui se contente de retenir le chemin du fichier d'origine
  - pas restreint aux fichiers d'une même partition
- ▶ Commande :  
**`ln -s src dst`**
- ▶ Lorsque l'OS tente de lire/modifier le contenu d'un lien symbolique, il effectue en réalité l'opération sur le fichier (chemin) pointé par le lien.

# Lien symbolique

11



→ Si on supprime la source `src`, le fichier aura disparu et le lien `dst` pointera vers un fichier inexistant. Le lien sera alors « mort ».

# Exemple

12

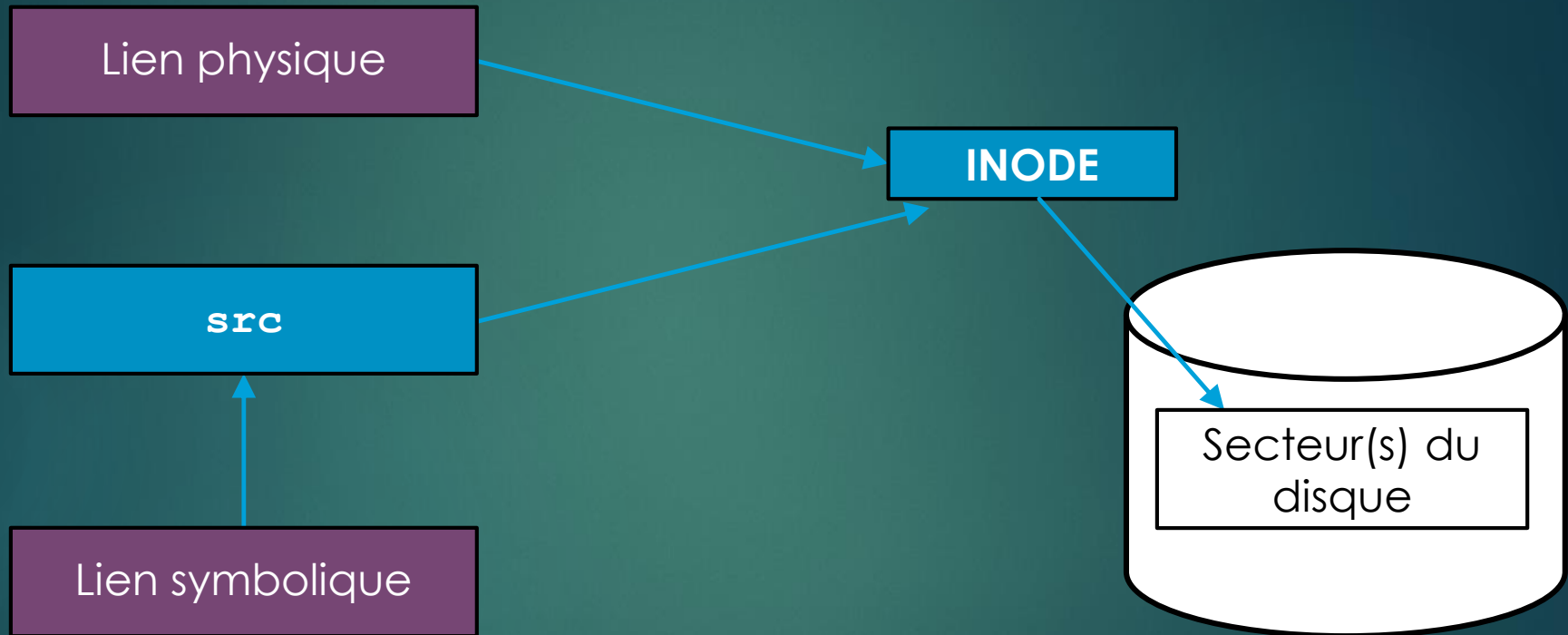
**Lien physique:** même inode que src

```
anthony@LAPTOP-GROTO:tmpdir$ ln src dstPhys
anthony@LAPTOP-GROTO:tmpdir$ ln -s src dstSym
anthony@LAPTOP-GROTO:tmpdir$ ls -li
total 8
47062 -rw-r--r-- 2 anthony anthony 13 Mar 19 15:25 dstPhys
47063 1rwxrwxrwx 1 anthony anthony  3 Mar 19 15:27 dstSym -> src
47062 -rw-r--r-- 2 anthony anthony 13 Mar 19 15:25 src
```

**Lien symbolique:** commence par la lettre “1” ;  
affiché en bleu clair ; la cible du lien est indiquée  
par une flèche

# Lien Symbolique et Physique : si rename de la source ?

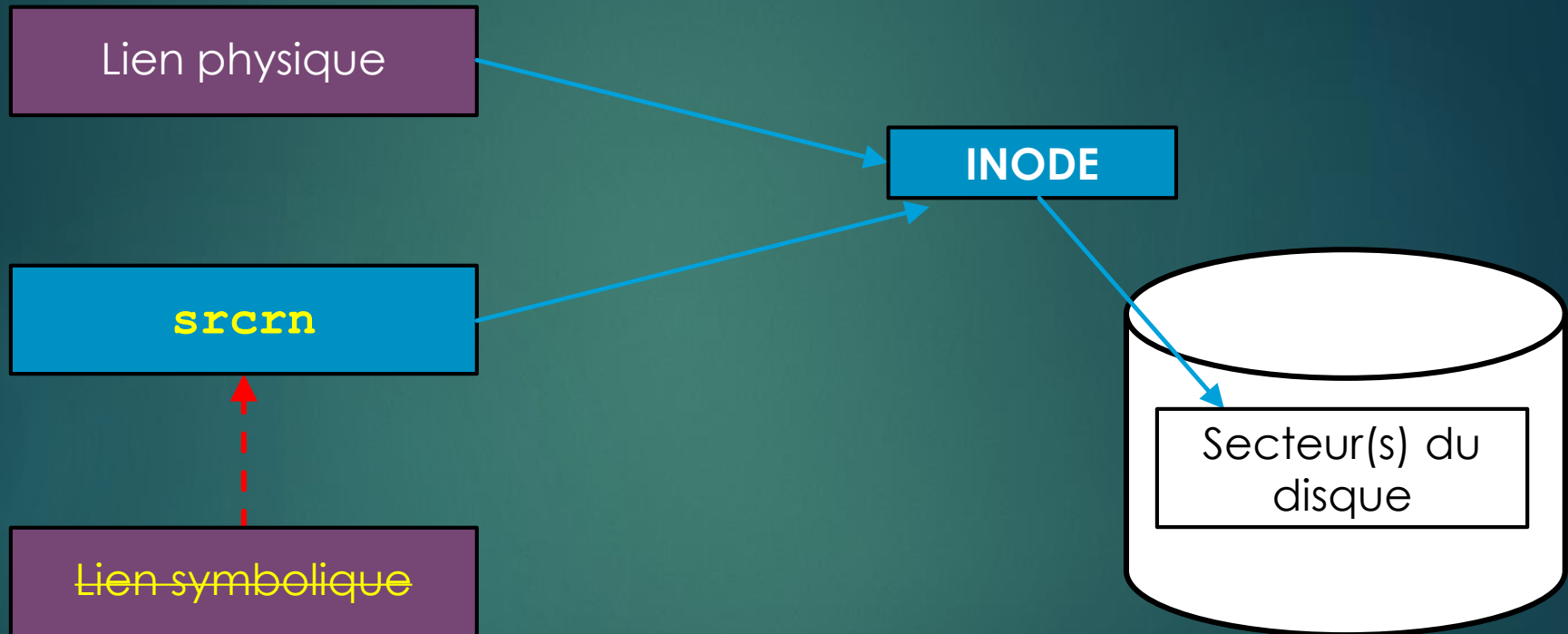
13



- Si on renomme la source `src` en `srcrn`, le lien symbolique sera perdu car il ne pointera plus vers le nom (chemin) initial (attribué lors de sa création).
- Par contre, aucun souci avec le lien physique.

# Lien Symbolique et Physique : si rename de la source ?

14



- Si on renomme la source `src` en `srcrn`, le lien symbolique sera perdu car il ne pointera plus vers le nom (chemin) initial (attribué lors de sa création).
- Par contre, aucun souci avec le lien physique.

# Commandes utiles

- ▶ **ln** : crée un lien physique
- ▶ **ln -s** : crée un lien symbolique
- ▶ **stat** : affiche les infos de l'inode d'un fichier
- ▶ **touch** : met à jour les *timestamps* d'un fichier
- ▶ **ls** : liste le contenu d'un répertoire
- ▶ **ls -i** : affiche le numéro d'inode des fichiers
- ▶ **ls -t** : trie selon *mtime*
- ▶ **ls -tc** : trie selon *ctime*
- ▶ **ls -tu** : trie selon *atime*