

# Système d'exploitation

## Système d'exploitation - Utilisation

Mickael Rouvier

CERI – Avignon Université  
`mickael.rouvier@univ-avignon.fr`

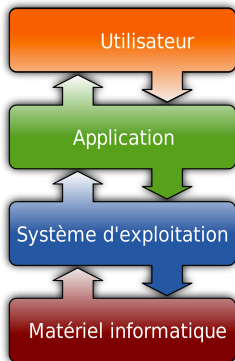


Section 1

# **Rappel : Système d'exploitation**



# Le rôle d'un SE



## • Quelle est le rôle d'un SE

- Permet de faire le lien entre les ressources matérielles d'un ordinateur et les applications de l'utilisateur (ex. donner une interface commune pour écrire sur un fichier qui peut se trouver sur un disque, clé USB, réseau. . . )
- Assure le démarrage et gérer l'accès à des ressources communes entre plusieurs utilisateurs ou programmes (ex. mémoire, disque, accès réseau...)

# Les services rendus par un SE

- **Gestion des fichiers**

- Gérer l'arborescence logique des fichiers
- Distribution physique sur le matériel de stockage (disque dur, clef usb...)

- **Gestion de la mémoire**

- Attribution de la mémoire à chaque application
- Partagée l'accès de la mémoire entre plusieurs applications

- **Gestion des applications**

- Ordonnancement des applications

- **Gestion des entrées/sorties**

- Carte réseau, carte son, imprimante...

- **Gestion d'une interface entre le matériel informatique et l'application**

- Interface en ligne de commande
- Interface graphique (window manager : composite window manager, tiling manager...)

Section 2

## **Gestion des processus**



# Définition d'un processus

- **Qu'est ce qu'un processus ?**
- **Processus** : Instance de programme s'exécutant à un instant donné ainsi que son contexte (ou environnement)
- **Contexte** : L'ensemble des ressources utilisées par le programme (mémoire, les fichiers ouverts, les droits associés, la priorité...)

# Les caractéristiques d'un processus

**Plusieurs processus peuvent être lancé. Le système d'exploitation doit être capable de les identifier. Un processus possède un certain nombre de caractéristiques :**

- **PID** : identifiant unique correspond au numéro du processus.
- **UID** : nom de l'utilisateur qui a lancé le processus.
- **PPID** : correspond au numéro du processus parent.
- **C** : au facteur de priorité : plus la valeur est grande, plus le processus est prioritaire
- **STIME** : correspond à l'heure de lancement du processus
- **TIME** : correspond à la durée de traitement du processus
- ...

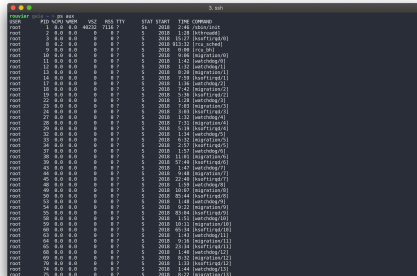
# Commande pour la gestion des processus

- **ps** : outils pour lister les processus (**p**rocess **s**tatus)
- **pstree** : outils pour lister les processus (display a tree of processes status)
- **htop** : outils pour lister les processus (interactive process viewer)
- **kill** : arrêter un processus
- **nice** : changer la priorité d'un processus
- **jobs** : lister les processus en tâche de fond



# PS - Outil pour lister les processus

- La commande **ps** permet de lister les processus de l'utilisateur.
  - sans option : affiche tous les processus associés au terminal
  - **-a** : affiche tous les processus (all)
  - **-e** : affiche tous les processus en cours d'exécution sur un ordinateur (execute)
  - **-l** : affiche les informations détaillées d'un processus (list)



USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1	0.0	0.0	4228	712	T	S	2018	2:46	init
root	2	0.0	0.0	0	0	T	S	2018	1:28	[kthreadd]
root	3	0.0	0.0	0	0	T	S	2018	18:52	[ksoftirqd/0]
root	8	0.2	0.0	0	0	T	S	2018	01:32	[rcu_sched]
root	9	0.0	0.0	0	0	T	S	2018	0:00	[rcu_bh]
root	10	0.0	0.0	0	0	T	S	2018	9:06	[migration/0]
root	11	0.0	0.0	0	0	T	S	2018	1:42	[migration/1]
root	12	0.0	0.0	0	0	T	S	2018	1:32	[migration/2]
root	13	0.0	0.0	0	0	T	S	2018	0:28	[migration/3]
root	14	0.0	0.0	0	0	T	S	2018	7:55	[ksoftirqd/1]
root	17	0.0	0.0	0	0	T	S	2018	1:18	[migration/4]
root	18	0.0	0.0	0	0	T	S	2018	7:42	[migration/5]
root	19	0.0	0.0	0	0	T	S	2018	5:36	[ksoftirqd/2]
root	22	0.0	0.0	0	0	T	S	2018	1:28	[migration/6]
root	23	0.0	0.0	0	0	T	S	2018	7:53	[migration/7]
root	24	0.0	0.0	0	0	T	S	2018	3:03	[ksoftirqd/3]
root	27	0.0	0.0	0	0	T	S	2018	1:32	[migration/8]
root	28	0.0	0.0	0	0	T	S	2018	7:31	[migration/9]
root	29	0.0	0.0	0	0	T	S	2018	5:18	[ksoftirqd/4]
root	32	0.0	0.0	0	0	T	S	2018	1:34	[migration/5]
root	33	0.0	0.0	0	0	T	S	2018	0:32	[migration/6]
root	34	0.0	0.0	0	0	T	S	2018	2:57	[ksoftirqd/5]
root	37	0.0	0.0	0	0	T	S	2018	1:57	[migration/7]
root	38	0.0	0.0	0	0	T	S	2018	11:01	[migration/8]
root	40	0.0	0.0	0	0	T	S	2018	57:49	[ksoftirqd/6]
root	43	0.0	0.0	0	0	T	S	2018	1:47	[migration/9]
root	44	0.0	0.0	0	0	T	S	2018	0:48	[migration/7]
root	45	0.0	0.0	0	0	T	S	2018	22:48	[ksoftirqd/7]
root	46	0.0	0.0	0	0	T	S	2018	1:39	[migration/8]
root	49	0.0	0.0	0	0	T	S	2018	10:07	[migration/9]
root	50	0.0	0.0	0	0	T	S	2018	05:44	[ksoftirqd/8]
root	54	0.0	0.0	0	0	T	S	2018	1:48	[migration/9]
root	54	0.0	0.0	0	0	T	S	2018	0:22	[migration/9]
root	55	0.0	0.0	0	0	T	S	2018	03:04	[ksoftirqd/9]
root	56	0.0	0.0	0	0	T	S	2018	1:51	[migration/10]
root	59	0.0	0.0	0	0	T	S	2018	10:31	[migration/10]
root	60	0.0	0.0	0	0	T	S	2018	09:34	[ksoftirqd/10]
root	63	0.0	0.0	0	0	T	S	2018	1:43	[migration/11]
root	64	0.0	0.0	0	0	T	S	2018	0:38	[migration/11]
root	66	0.0	0.0	0	0	T	S	2018	20:34	[ksoftirqd/11]
root	67	0.0	0.0	0	0	T	S	2018	1:48	[migration/12]
root	69	0.0	0.0	0	0	T	S	2018	0:32	[migration/12]
root	70	0.0	0.0	0	0	T	S	2018	1:33	[ksoftirqd/12]
root	74	0.0	0.0	0	0	T	S	2018	1:44	[migration/13]
root	75	0.0	0.0	0	0	T	S	2018	0:32	[migration/13]

# PS : Outil pour lister les processus

- La commande **ps** permet de lister les processus de l'utilisateur.

**Lance mon programme loop :**

```
invite/> ./loop
```

**Lance l'application de visualisation d'outil :**

```
invite/> ps
UID PID PPID F CPU PRI NI SZ RSS WCHAN S ADDR TTY TIME CMD
501 89475 89474 4006 0 31 0 4299792 3960 - S 0 ttys003 0:00.14
-bash
501 89491 89475 4006 0 31 0 4267920 756 - R+ 0 ttys003 0:03.92
./loop
```

# PStree – Outil pour lister les processus

- La commande **pstree** permet d'afficher un arbre de processus

```
3. ssh
rouvier gaia ~ > pstree
systemd--ModemManager--{gdbus}
                        {gmain}
--accounts-daemon--{gdbus}
                   {gmain}
--acpid
--2*[agetty]
--automount--4*[{automount}]
--avahi-daemon--avahi-daemon
--cron
--2*[dbus-daemon]
--dbus-launch
--gconfd-2
--irqbalance
--2*[iscsid]
--lvmetad
--lxcfs--10*[{lxcfs}]
--mdadm
--ntpd
--polkitd--{gdbus}
          {gmain}
--rpc.idmapd
--rpc.mountd
--rpcbind
--rsyslogd--{in:imklog}
```

# HTOP (ou TOP) – Outil pour lister les processus

- La commande **htop** (ou **top**) permet d'avoir une liste dynamique des processus

```
3. ssh

1 [ 0.0%] 13 [ 0.0%] 25 [ 0.0%] 37 [ 0.0%]
2 [ 0.0%] 14 [ 0.0%] 26 [ 0.0%] 38 [ 0.0%]
3 [ 0.0%] 15 [ 0.0%] 27 [ 0.0%] 39 [ 0.0%]
4 [ 0.0%] 16 [ 0.0%] 28 [ 0.0%] 40 [ 0.0%]
5 [ 0.0%] 17 [ 0.0%] 29 [ 0.0%] 41 [ 0.0%]
6 [ 0.0%] 18 [ 0.0%] 30 [ 0.0%] 42 [ 0.0%]
7 [ 0.0%] 19 [ 0.0%] 31 [ 0.0%] 43 [ 0.0%]
8 [ 0.0%] 20 [ 0.0%] 32 [ 0.0%] 44 [ 0.0%]
9 [ 0.0%] 21 [ 0.0%] 33 [ 0.0%] 45 [ 0.0%]
10 [ 0.0%] 22 [ 0.0%] 34 [ 0.0%] 46 [ 0.0%]
11 [ 1.3%] 23 [ 0.0%] 35 [ 0.0%] 47 [ 0.0%]
12 [ 0.0%] 24 [ 0.0%] 36 [ 0.0%] 48 [ 0.0%]
Mem [||||] 1.99G/504M Tasks: 72, 66 thr: 1 running
Swap 0K/0K Load average: 0.08 0.02 0.03
Uptime: 276 days(1), 19:58:12

PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
11502 marink 20 0 27928 5284 3300 S 1.3 0.0 16:47.82 htop -u marink
54589 rouvier 20 0 48372 4616 3428 S 1.3 0.0 0:00.10 htop
1558 root 10 0 4772 4712 2450 S 0.7 0.0 50:51.46 /sbin/lscsd
1710 root 20 0 15668 2180 1800 S 0.0 0.0 1:46:57 /usr/sbin/irqbalance --pid=/var/run/irqbalance.pid
1 root 20 0 40232 7116 3416 S 0.0 0.0 2:46.20 /sbin/init
481 root 20 0 80860 2852 1832 S 0.0 0.0 0:00.02 sshd: rouvier [rty]
408 rouvier 20 0 88690 2532 1412 S 0.0 0.0 0:14.32 sshd: rouvier@pts/52
409 rouvier 20 0 20308 4276 1724 S 0.0 0.0 0:01.37 -bash
548 rouvier 20 0 31008 4728 2216 S 0.0 0.0 0:23.92 ssh orion
729 root 20 0 84772 588 416 S 0.0 0.0 0:00.00 /sbin/lnetstat -f
1461 root 20 0 40696 1112 592 S 0.0 0.0 0:44.71 /sbin/rpcbind -f -u
1491 syslog 20 0 2500 4888 2060 S 0.0 0.0 0:20.88 /usr/sbin/rsyslogd -n
1492 syslog 20 0 2500 4888 2060 S 0.0 0.0 0:04.05 /usr/sbin/rsyslogd -n
1493 syslog 20 0 2500 4888 2060 S 0.0 0.0 0:49.21 /usr/sbin/rsyslogd -n
1464 syslog 20 0 2500 4888 2060 S 0.0 0.0 1:07.03 /usr/sbin/rsyslogd -n
1472 root 20 0 5484 1620 1508 S 0.0 0.0 0:00.48 /usr/sbin/acpid
1474 root 20 0 32112 4040 1764 S 0.0 0.0 5:33.59 /usr/sbin/cron -f
1498 avahi 20 0 46832 844 428 S 0.0 0.0 0:00.25 avahi-daemon: running [linux-4.local]
1495 root 20 0 2700 3664 2432 S 0.0 0.0 14:24.17 /usr/lib/accounts-service/accounts-daemon
1498 root 20 0 2700 3664 2432 S 0.0 0.0 0:00.00 /usr/lib/accounts-service/accounts-daemon
1482 root 20 0 2700 3664 2432 S 0.0 0.0 14:24.19 /usr/lib/accounts-service/accounts-daemon
1484 messagebus 20 0 45960 680 8 S 0.0 0.0 0:00.95 /usr/bin/dbus-daemon -system --address=systemd: --nofork --nopidfile --systemd-acti
1497 avahi 20 0 46832 344 8 S 0.0 0.0 0:00.00 avahi-daemon: chroot helper
1499 root 20 0 61584 3156 2428 S 0.0 0.0 0:07.77 /usr/sbin/sshd -D
1593 root 20 0 4040 4624 768 S 0.0 0.0 0:00.00 /usr/sbin/ModemManager
1586 root 20 0 4040 4624 768 S 0.0 0.0 0:00.00 /usr/sbin/ModemManager
1589 root 20 0 4040 4624 768 S 0.0 0.0 0:00.04 /usr/sbin/ModemManager
1557 root 20 0 1220 1576 448 S 0.0 0.0 0:00:35.46 /sbin/lscsd
F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Alice F8Police F9Kill F10Quit
```

# Kill – Arrêter un processus

## Arrêter un processus :

```
invite/> kill -9 <PID>
```

## Envoye un signal à un processus :

```
invite/> kill -<signal> <PID>
```

- 9 : termine le processus immédiatement
- 11 : violation de mémoire
- 19 : mettre un processus en pause
- ....

# Interruption clavier

- Les processus peuvent être interrompus par l'envoi d'un signal depuis le terminal
- CTRL+C
  - L'émission de ce signal est provoquée par la frappe de caractères particuliers que nous désignerons par CTRL + C
  - Ce signal à pour effet d'interrompre tous les processus attachés au terminal depuis lequel il est émis

# nice, renice - Contrôler la priorité des processus

- Chaque processus s'exécute avec une priorité, on désire changer la priorité des processus
- La valeur des priorités peut aller de -20 à 20
- La priorité maximale pour un processus est de -20
- Par défaut tous les processus s'exécute avec une priorité de 0

**nice : permet de changer la priorité d'une commande**

```
invite/> nice -n 19 dd if=/dev/cdrom of= /mdk1.iso
```

**renice : permet de changer la priorité d'un processus déjà lancé**

```
invite/> renice 20 -u pierre
```

```
invite/> renice 20 785
```

# Job - Commande pour placer les processus en premier- et arrière-plan

**Attention :** les commandes `jobs`, `bg` et `fg` sont des commandes propres au shell que vous avez lancé.

- **jobs** : pour connaître tous les jobs lancée en arrière-plan
- **bg %n** : met le job en arrière-plan, où *n* est le job ID
- **fg %n** : met le job en premier-plan, où *n* est le job ID
- **kill %n** : tue le job en, où *n* est le job ID
- **Ctrl+Z** : met le job en arrière-plan et stoppé



# Processus système vs utilisateur

**On peut distinguer deux types de processus :**

- **Le processus système**

- Ne sont attachés à aucun terminal
- Sont créés au lancement du système ou à des dates créées par l'administrateur du système
- Ne sont interrompus qu'à l'arrêt du système
- Exemple : le processus assurant le bon usage de l'imprimante, le processus cron qui permet de lancer des tâches à une date donnée

- **Le processus utilisateur**

- Lancés par un utilisateur particulier depuis un terminal donné à une date donnée
- Il s'agit souvent d'un processus correspondant à l'exécution d'un interpréteur de langage de commandes comme par exemple le Bash
- Le fait de se loguer sur un terminal, sous une identification donnée, provoque le lancement d'un processus correspondant à l'exécution d'un programme déterminé à l'avance pour chaque utilisateur

# Système de fichier virtuel : /proc

```
invite/> ls /proc
```

```
rouvier@proladite ~ > ls /proc/
1      11119  12      134      17268  1916    1939    2      22871  2428    26589  28385  2955    38319  31167  31959  48      54      6531  717    788    8181  9594      keys      timer_list
10     11128  128     135     17261  1917    19399   2019   22872  24657  26648  28397  29747  38320  31166  31980  41     559    6563  72      7816  82     96      kmsg      tty
100    11167  12020  13581  1738   1918    194     20201  22241  24687  26734  28398  29853  38321  31275  31982  42     562    66     7289  7817  8738  966      kpagecgroup  uptime
1805   112    12088  13627  17381  1919    1940    28332  22328  24866   27      2841   29855  38322  31276  32841  4212   564    6642  7384  7826  8276  98      kpagecount  version
181    113    121    13674  17382  192     1941    28344  22335  25062  27134  28580   30     38323  313   32842  428    566    6643  7313  7827  83     99      kpageflags   version_signature
1819   11334  122    13912  17545  1928    1942    28345  22348  25063  27182  28534  38112  38324  314   32183  447     567    6699  7314  7828  84     acpi        loadavg
182    11335  1226   13958  17546  1921    1943    28359  22350  25156  27183  28546  38113  38325  31469  32522  448     568    670   7344  783    86     buddyinfo   locks
1827   1138   1227   13954  177    19219   1944    28578  22355  25157  27286  28557  38138  38326  31470  33      45     569    6700  74     7877  8673     bus         ndstat
1834   1195   123    14      178     1922    1945    28579  22373  25158  27218  28563  38154  38458  31632  337     493    57    6729  7448  7885   87      cgroups     meminfo
18428  114    12345  1484   17882  19228   1946    286    22374  25598  27351  28570  38172  38510  31640  34      46     574    6776  7449  7886   8766     cmdline    misc
18483  1147   12372  14328  17883  1923    1947    28726  22378  25867  27368  28578  38192  38511  31752  348     467    58    6777  7468  790   85     consoles    modules
185    115    124    14321  17819  1924    1948    28727  22553  26    27364  28597  38295  38512  31753  341     47     5838  6883  75     791    89     cpuinfo     mounts
18551  11597  12451  14579  17822  1925    1949    28781  22785  26277  27531  28611  38796  38513  31761  3492    48     5839  6895  7568  7918  8985     mtrr
18552  116    1247   14792  17827  1926    1950    21     22712  26373  27532  28612  38297  38518  31764  35      4818  5869  6896  7576  792    9      devices     net
18593  11622  1248   15    17828  1927    1951    2100   23    26379  27549  28892  38298  38582  31765  3598    4819   59     69    76    7921  98     diskstats   pagetypeinfo
186    11623  125    15189  17866  19271   1952    21188  23811  26393  27563  28945  38299  38583  31766  36      4931  5912  6906  766    7931  980     dma         partitions
18610  1165    126    15315  17988  1928    1953    21189  23159  26485  27583  29    38308  38589  31767  3688    4932    60    6915  767    7932  9876     driver      sched_debug
18621  11669  12727  1558    17989  1929    1954    21200  23168  26486  27584  28831  38301  38593  31768  3619    5845  6855  6927  768    794    9891     execdomains  schedstat
18622  11668  12762  15582  17952  193    1955    21384  231    26423  27681  28832  38382  38699  31769  3644    5847  6212  6928  7681  795    910     fb           scsi
1863    117    12763  15665  1833    1938    1956    21385  23446  26443  27682  28833  38383  38700  31778  3645    51     6227  6956  77     796  9183   filesystems  self
18674  11738  1289    16    18452  1931    1957    21494  23880  26445  27636  28834  38318  38783  31779  3646    512    6228  6957  778  7990  91    fs          slabinfo
187    11739  1298    16433  18473  1932    196    21582  23858  26585  2778    29835  38311  38794  31812  3724    5196   624    6970  7718  8     93     interrupts  softirqs
1875   118    13    1649    18474  1933    197    21536  23859  26528  27798  29837  38312  38803  31902  3731    5197   625    6978  7711  8826  94     ioem       spl
188    18222  130    16586  18587  1934    19728  21537  23959  26552  27885  29358  38313  38839  31903  3783    52     6298  6991  7725  8827  95     ioparts     stat
18835  11823  131    16538  18589  1935    19740  21556  24    26558  28    29441  38314  38988  31904  3860    5211  6291  7     7732  8863  9529  ipmi       swaps
1891    11827  132    1654    1874    1936    198    21592  24178  28559  28925  29442  38315  38987  31905  3861    529    63    78    7783  8864  9538  irq         sys
11     11856  13267  1654    1890    19364  18876   216    24179  28566  28151  29466  38316  31846  31986  39      53     64    7843  7783  8866  9568  kallsyms    sysrq-trigger
11819  11881  13268  17     191    1937    18877  21760  24183  26584  28152  29489  38317  31124  31907  3958    535    65    7844  7795  81     9588     kcore       sysvipc
11882  119    133    17846  1912  1938    199    22    2426  26585  28376  29534  38318  31146  31908  4      5362  6538  71    78    8111  9581  key-users   thread-self
```

# Système de fichier virtuel : /proc

```
invite/> ls /proc
```

```
rouvier@protonmail: ~ - ssh -> ls /proc/
1      11119 12      134      17268 1916      1939      2      27871 2428      26589 28385 2955      38319 31167 31959 40      54      6531 717      788      8181 9594      keys      timer_list
10     11128 128      135      17261 1917      19399      2019 27872 24657 26648 28397 29747 38320 31166 31980 41      559 6563 72      7816 82      96      kmsg      tty
100    11167 12020 13581 1738 1918 194      20201 22241 24687 26734 28398 29853 38321 31275 31982 42      562 66      7289 7817 8738 966      kpagecgroup uptime
1805   112      12088 13627 17381 1919 1940      28332 22328 24866 27      2841 29855 38322 31276 32841 4212 564 6642 7384 7826 8276 98      kpagecount version
181    113    121    13674 17382 192      1941 28344 22335 25062 27134 28580 30      38323 313      32842 428 566 6643 7313 7827 83 99      kpageflags version_signature
1819   11334 122      13912 17545 1920 1942      28345 22348 25063 27182 28534 38112 38324 314      32183 447 567 6699 7314 7828 84      acpi      loadavg      vmallocinfo
182    11335 1226      13958 17546 1921 1943      28359 22350 25156 27183 28546 38113 38325 31469 32522 448 568 670 7344 783 86      buddyinfo locks      vmstat
1827   1138 1227      13954 177      19219 1944      28578 22355 25157 27286 28557 38138 38326 31470 33      45 569 6700 74      7877 8673      bus      ndstat      zoneinfo
1834   1195 123      14      178      1922      1945 28579 22373 25158 27218 28563 38154 38458 31632 337      493 57      6729 7448 7885 87      cgroups      meminfo
18428  114      12345 1484      17882 19220 1946 286      22374 25159 27251 28570 38172 38510 31640 34      46 574 6776 7449 7886 8766      cmdline      misc
18483  1147 12372 14328 17883 1923 1947      28726 22378 25867 27368 28578 38192 38511 31752 348 467 58 6777 7460 790 85      consoles      modules
185    115    124      14321 17819 1924 1948      28727 22553 26      27364 28597 38295 38512 31753 341 47 5838 6883 75 791 89      cpuinfo      mounts
18551  11597 12451 14579 17822 1925 1949      28781 22785 26277 27531 28611 38796 38513 31761 3492 48 5839 6895 7568 7918 8985      crypto      mtrr
18552  116      1247 14792 17827 1926 1950 21      22712 26373 27532 28612 38297 38518 31764 35      4818 5869 6896 7576 792 9      devices      net
18593  11622 1248 15      17828 1927      1951 2100 23      26379 27549 28892 38298 38582 31765 3598      4819 59 69 76 7921 90      diskstats      pagetypeinfo
186    11623 125      15189 17866 17271 1952 21108 23811 26393 27563 28945 38299 38583 31766 36      4931 5912 6906 766 7931 980      dma      partitions
18610  1165 126      15315 17988 1928      1953 21109 23159 26485 27583 29      38308 38589 31767 3688      4932 60 6915 767 7932 9876      driver      sched_debug
18621  11669 12727 1559 17989 1929 1954 21320 23160 26486 27584 28831 38301 38593 31768 3619 5845 6855 6927 768 794 9891      execdomains      schedstat
18622  11668 12762 15582 17952 193 1955 21384 231      26423 27681 28832 38382 38699 31769 3644 5847 6212 6928 7681 795 910      fb      scsi
1863    117      12763 15665 1833 1938 1956 21385 23446 26443 27682 28833 38383 38700 31778 3645 51 6227 6956 77 796 9183      filesystems      self
18674  11738 1289 16      18452 1931 1957 21494 23880 26445 27636 28834 38318 38783 31779 3646 512 6228 6957 778 7990 919      fs      slabinfo
187    11739 1298 16433 18473 1932 196      21582 23858 26585 2778 28835 38311 38794 31812 3724 5196 624 6970 7718 8 93      interrupts      softirqs
1875   118 13      1649 18474 1933 197 21536 23859 26528 27798 28837 38312 38803 31902 3731 5197 625 6978 7711 8826 94      iomem      spl
188    18222 130 1656 18567 1934 19728 21537 23959 26552 27885 28358 38313 38839 31903 3783 52 6298 6991 7725 8827 95      ioparts      stat
18835  11823 131 16538 18589 1935 19740 21556 24      26558 28      29441 38314 38988 31984 3860 5211 6291 7 7732 8863 9529      ipmi      swaps
1891    11827 132 1654 1874 1936 198      21592 24178 28559 28825 29442 38315 38987 31985 3861 529 63 78 7783 8864 9538      irq      sys
11      11856 13267 1654 1890 19364 18876 216 24179 28566 28151 29466 38316 31846 31986 39 53 64 7843 7783 8866      kallsyms      sysrq-trigger
11019  11881 13268 17      191 1937 18877 21760 24183 26584 28152 29489 38317 31124 31987 3958 535 65 7844 7795 81 9588      kcore      sysvipc
11882  119 133 17846 1912 1938 199 22      2426 26585 28376 29534 38318 31146 31988 4 5362 6538 71 78 8111 9581      key-users      thread-self
```

# Système de fichier virtuel : `/proc`

```
invite/> ls /proc/23342
```

```
rouvier@phrodite ~ -> ls /proc/23342
attr          clear_refs    cpuset        fd            limits       mem           net           oom_score     personality  schedstat    snaps_rollop  status        timerslack_ns
autogroup     cmdline       cwd           fdinfo        loginuid     mountinfo    ms           oom_score_adj projid_map    sessionid    stack         syscalls      uid_map
auxv          comm          environ       gid_map       map_files    mounts       numa_maps    pagemap       root         setgroups    stat          task          wchan
cgroup        coredump_filter exe           io           maps         mountstats   oom_adj      patch_state   sched         snaps         state         timers
```

Section 3

## **Ordonnancement des processus**

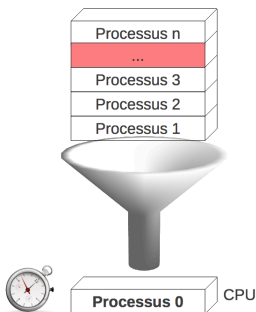


# Multitâche

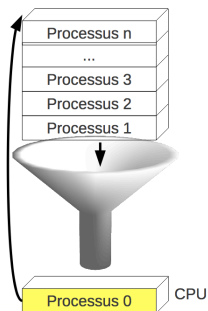
- **Problématique :** Comment un système d'exploitation peut être multitâche, alors qu'un processeur ne peut exécuter qu'un seul processus à la fois ?
- **Solution :** L'ordonnanceur permet d'alterner l'exécution des  $n$  tâches permettant d'avoir une simultanéité apparente
  - Le système d'exploitation (l'ordonnanceur) répartit les tranches de temps CPU à chaque processus
  - Un seul processus doit être exécuté à la fois
  - Globalement tous les processus doivent être exécutés

# Ordonnancement

- Qu'est-ce que l'ordonnanceur :
  - Une queue de processus en attente (question : comment ordonne-t-on les processus dans cette queue ?)
  - **Commutation de contexte** : Un processus actif sur un **quanta de temps**



# Ordonnancement





# Commutation de contexte

Par exemple, avec 2 tâches cela peut se décomposer par :

- **Commutation de contexte** et élection de la nouvelle tâche n°1
- **Chargement** par le noyau du contexte de la tâche 1.
- **Exécution** des instructions de la tâche 1 pendant x millisecondes
- **Sauvegarde** du contexte de la tâche 1
  
- **Commutation de contexte** et élection de la nouvelle tâche n°2
- **Chargement** par le noyau du contexte de la tâche 2.
- **Exécution** des instructions de la tâche 2 pendant x millisecondes
- **Sauvegarde** du contexte de la tâche 2
  
- **Commutation de contexte** et élection de la nouvelle tâche n°1
- ...

# Algorithmes d'ordonnancement

- Un bon algorithme doit être capable de :
  - **Équité** : s'assurer que chaque processus reçoit sa part du temps CPU
  - **Efficacité** : utiliser le temps de processeur à 100% : efficacité
  - **Temps de réponse** : minimiser le temps de réponse en mode interactif
  - **Temps d'exécution** : minimiser le temps d'attente en traitement par lots
  - **Rendement** : maximiser le nombre de travaux effectués en une heure
- Un **algorithme d'ordonnancement** sert à choisir lequel de plusieurs processus sera traité en premier par le processeur.
- Un système d'exploitation multitâche est préemptif lorsque celui-ci peut arrêter (réquisition) à tout moment n'importe quel processus pour passer la main.

# Exemple d'ordonnancement

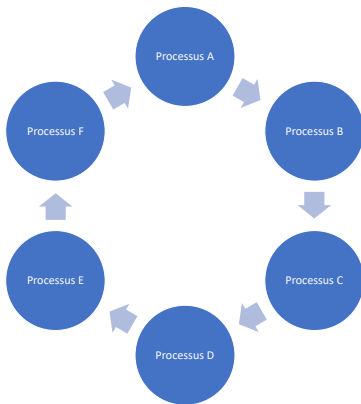
- **Ordonnanceur circulaire**
- **Ordonnanceur avec priorité**

# Ordonnancement circulaire

- L'algorithme du tourniquet circulaire (ou round robin) est un algorithme ancien, simple, fiable et très utilisé
- Il mémorise dans une file du type FIFO (First In First Out) la liste des processus prêts, c'est-à-dire en attente d'exécution
- Il alloue le processeur au processus en tête de file, pendant un quantum de temps
- L'algorithme round robin permet une répartition équitable du processeur
- Le choix de la valeur du quantum se révèle très importante
  - Un quantum trop petit provoque trop de commutations de processus et abaisse l'efficacité du processeur
  - Un quantum trop élevé augmente le temps de réponse des courtes commandes en mode interactif

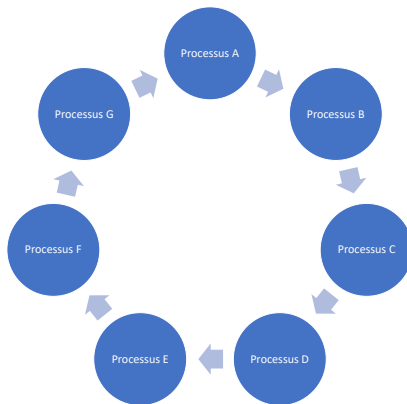
# Ordonnancement circulaire

On boucle sur l'ensemble des processus



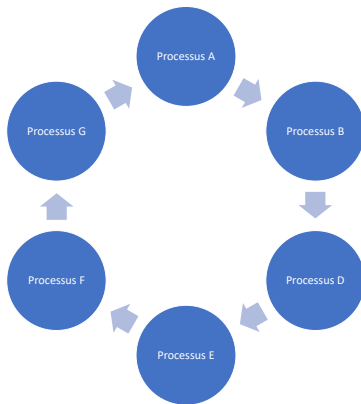
# Ordonnancement circulaire

## Ajout du Processus G dans l'ordonnanceur



# Ordonnancement circulaire

## Suppression du Processus C dans l'ordonnanceur



# Ordonnancement circulaire

## Algorithme

---

```
1
2 void circulaire () {
3     int max = 5;
4     int tab[max];
5     int i = 0;
6     while(1) {
7         commutation_context(tab[i]);
8         execution(tab[i]);
9         sauvegarde(tab[i]);
10        i++;
11        if (i < max)
12            i=0;
13    }
14 }
```

---

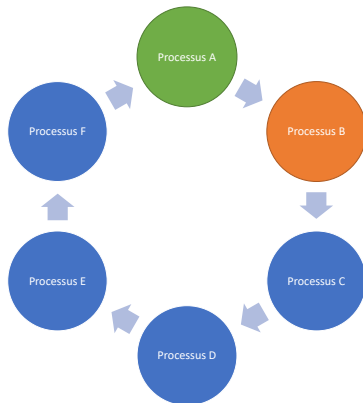
Listing 1 – "Exemple algorithme circulaire"



# Ordonnancement avec priorité

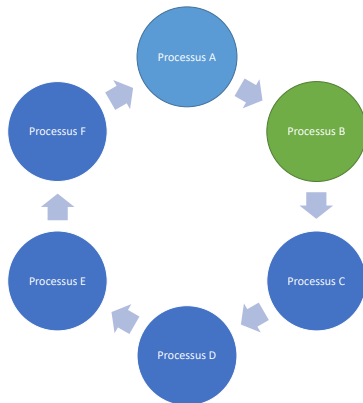
- L'ordonnanceur à priorité attribue à chaque processus une priorité
- Le choix du processus à élire dépend donc des priorités des processus prêts.
- Les processus de même priorité sont regroupés dans une file du type FIFO. Il y a autant de files qu'il y a de niveaux de priorité. L'ordonnanceur choisit le processus le plus prioritaire qui se trouve en tête de file
- Pour empêcher les processus de priorité élevée de s'exécuter indéfiniment, l'ordonnanceur diminue régulièrement la priorité du processus en cours d'exécution.

# Ordonnancement avec priorité



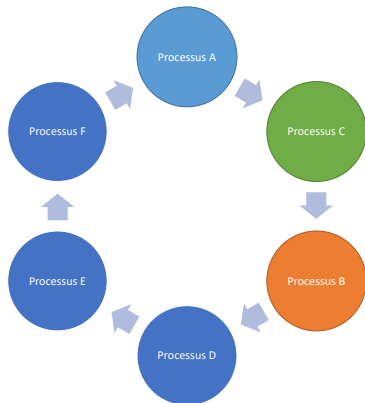
- Le processus vert : exécuté par la machine
- Le processus orange : a une priorité plus élevée que les autres
- Le processus bleu : a une priorité normale

# Ordonnancement avec priorité



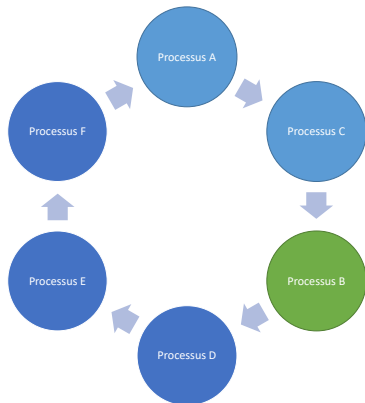
- Le processus vert : exécuté par la machine
- Le processus orange : a une priorité plus élevée que les autres
- Le processus bleu : a une priorité normale

# Ordonnancement avec priorité



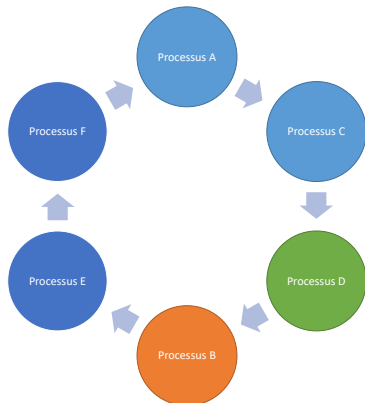
- Le processus vert : exécuté par la machine
- Le processus orange : a une priorité plus élevée que les autres
- Le processus bleu : a une priorité normale

# Ordonnancement avec priorité



- Le processus vert : exécuté par la machine
- Le processus orange : a une priorité plus élevée que les autres
- Le processus bleu : a une priorité normale

# Ordonnancement avec priorité



- Le processus vert : exécuté par la machine
- Le processus orange : a une priorité plus élevée que les autres
- Le processus bleu : a une priorité normale

Section 4

## **Gestion des fichiers**



# Couche d'abstraction matérielle

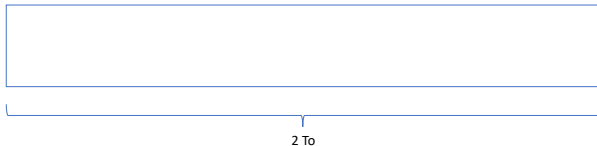
- Comment stocker les informations sur un disque dur ?
- Comment organiser les informations sur un disque dur ?





# Disque dur

- Un disque dur est une mémoire de masse pour écrire nos données.

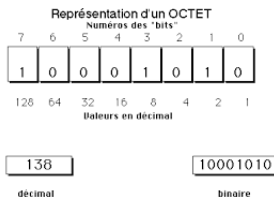


# Les bits

- Bit : la plus petit unité numérique définissable
- L'état du bit (0 ou 1) correspond à l'état du signal électrique (arrêt ou marche)
- **Problème** : Un seul bit ne correspond à rien et on ne peut pas définir de donnée plus complexe que OUI ou NON
- **Solution** : Regrouper les bits en bloc nommés **octet**

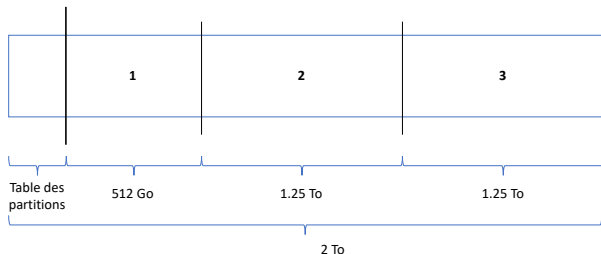
# Les octets

- **GECOS** : premiers ordinateurs – 1962
  - Les octets étaient regroupés par bloc de 6
  - L'information était inscrit sur un fichier en carton appelé : carte perforée
- **Standard ASCII** – 1963
  - Les octets étaient regroupés par bloc de 8
- **Unicode** – 2016
  - Conçu pour coder l'ensemble des caractères du monde
  - On utilise le plus souvent le standard : UTF-8



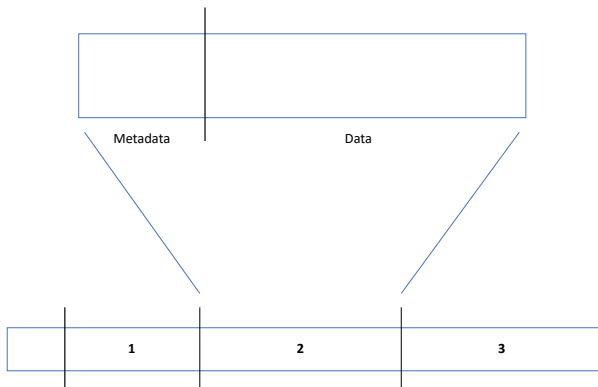
# Partition

- **Partition** : Consiste à créer des zones sur le disque dur dont les données ne seront pas mélangées
  - Organiser les données (sauvegarde...)
  - Sécuriser ses données



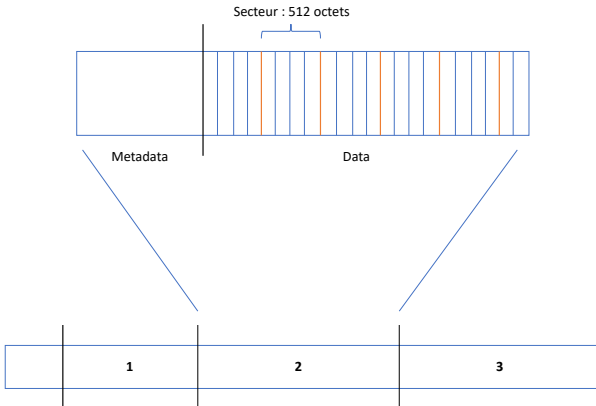
# Métadata/Data

- **Data** : contient le contenu du fichier
- **Métadata** : contient les méta-informations du fichier (droits d'accès en lecture, écriture..., taille du fichier...)



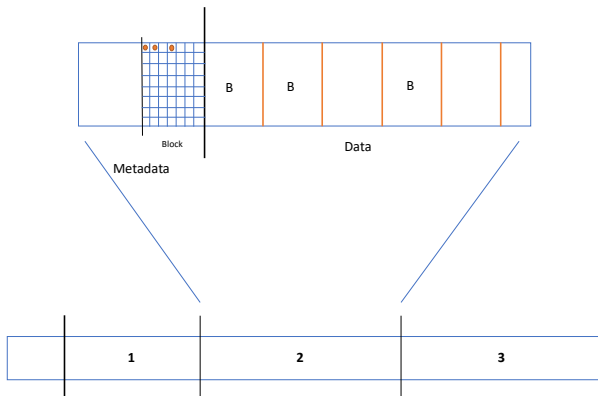
# Secteur

- **Secteur** : Le secteur est la plus petite unité physique de stockage sur un support de données (hardware)
- **Bloc** : Le bloc est la plus petite unité physique de stockage (logiciel)



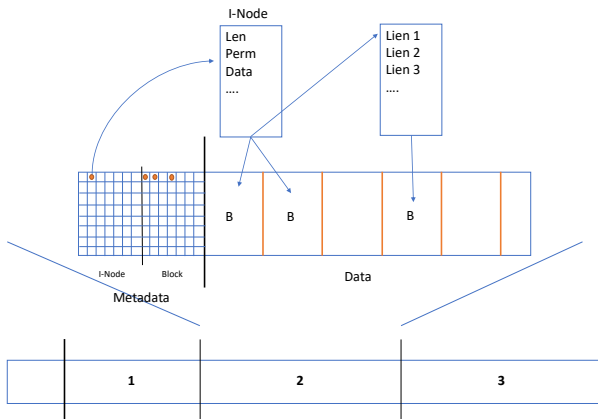
# Block

- **Block :**



# I-Node

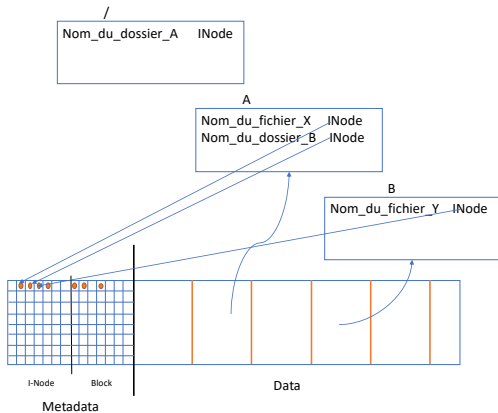
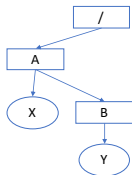
- **I-Node** : structure de donnée sur disque (lien directe vs lien indirecte)
- **Attention** : i-node ne contient pas le nom du fichier





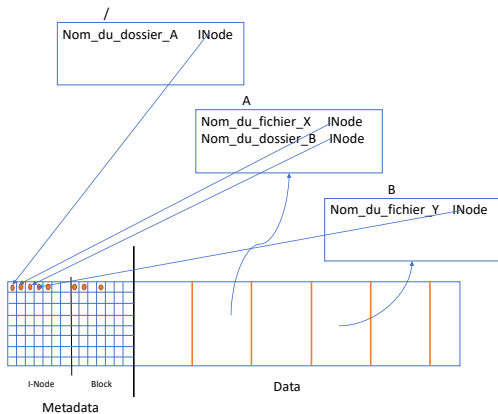
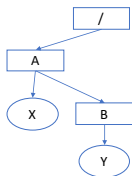
# Dossier

- **Dossier :**



# Root

- **Root :**



- **Système de gestion de fichier sous Linux :**
  - **Redimensionnement à chaud** : Permet de redimensionner à chaud la taille du système de fichiers
  - **RAID** : Permet la mise en oeuvre de plusieurs fonctionnalités de RAID (logiciel)
  - **Compression des données** : Permet la compression des données stockées
  - **Copy-on-write** : Garantit que tout le système de fichiers choisi sera synchronisé, archivé ou sauvegardé dans un état cohérent avec ce qu'il était au début de l'opération