Guide d'utilisation d' **Exocet**
le nouveau cluster de l'Université des Antilles
géré par le C3I
Centre **C**ommun de **C**alcul **I**ntensif

Les **exocets** sont des poissons volants très rapides des mers chaudes. On les observe facilement autour de la Gaudeloupe.

Pour toute aide, adressez-vous à `raphael.pasquier@univ-antilles.fr` l'ingénieur informaticien du C3I.

Mise à jour : 8 mars 2021

Présentation du cluster

Exocet se compose de plusieurs serveurs :

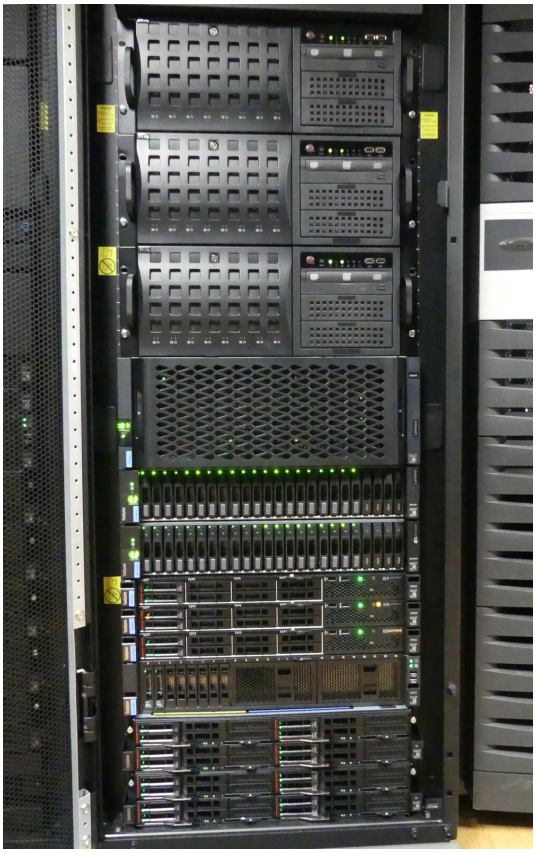
- Deux serveurs frontaux **exocet1** et **exocet2**. Quand on se connecte à **Exocet** , on se connecte en fait à l'un des deux.
- vingt cinq nœuds "Calcul" (**node01** à **node25**). Ils servent pour les calculs généraux sur processeurs Intel. Chaque nœud contient deux processurs Intel de 18 cœurs chacun, soit un total de 36 cœurs pour un nœud, 192 Go de mémoire RAM.
- Un nœud "grosse mémoire RAM" (**mem01**) ayant les mêmes deux processeurs Intel qu'un nœud "Calcul" mais avec 1 536 Go de mémoire RAM.
- Un nœud "V100" (**gpu01**) ayant les mêmes caractéristiques qu'un nœud "Calcul" mais avec en plus deux cartes graphiques **Nvidia TESLA V100**.
- Un nœud "T4" (**gpu02**) comme le "V100" mais avec deux cartes graphiques **NVidia T4**.
- Les trois anciens nœuds graphiques de **Wahoo** (**gpu03**, **gpu04**, **gpu05**). Ils sont aussi des serveurs bi-processeurs. Ils ont chacun 256 Go de mémoire RAM, et une carte graphique **QUADRO P5000**.

Tous les nœuds, excepté les anciens nœuds de Wahoo, ont un disque dur SSD `lscratch` réservé aux fichiers temporaires créés par certaines applications comme gaussian09.

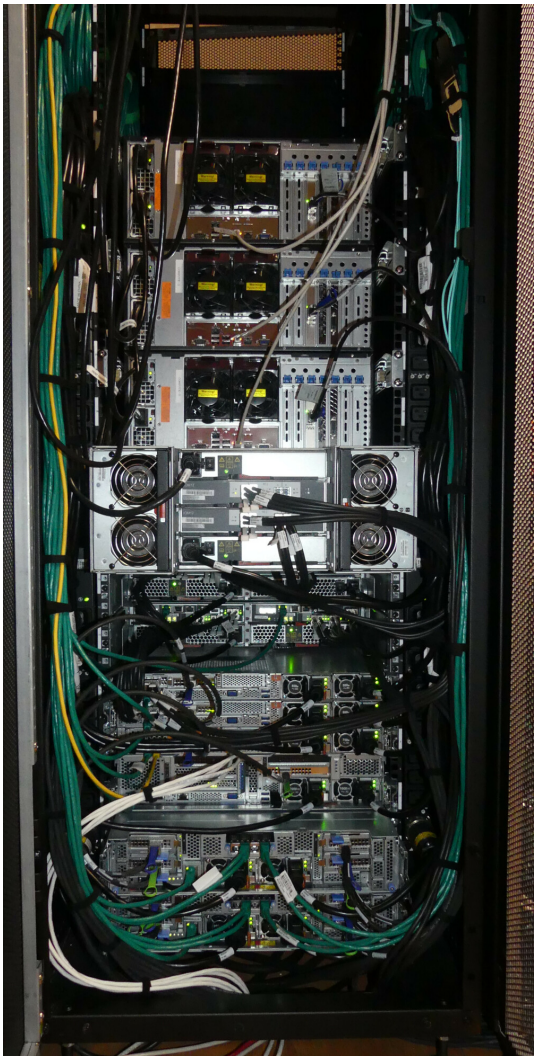
Cela représente presque 1000 cœurs au total et 7 349 Go de mémoire RAM.

Table 1: Photos d'**Exocet**

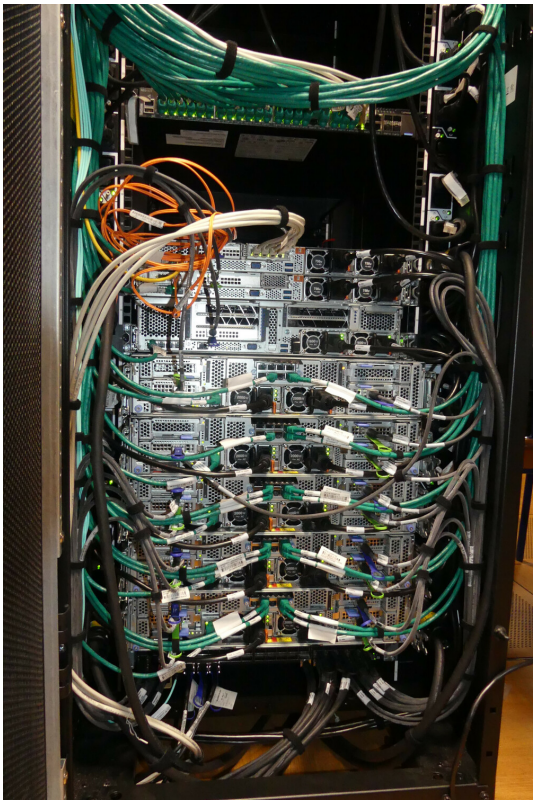
La première baie

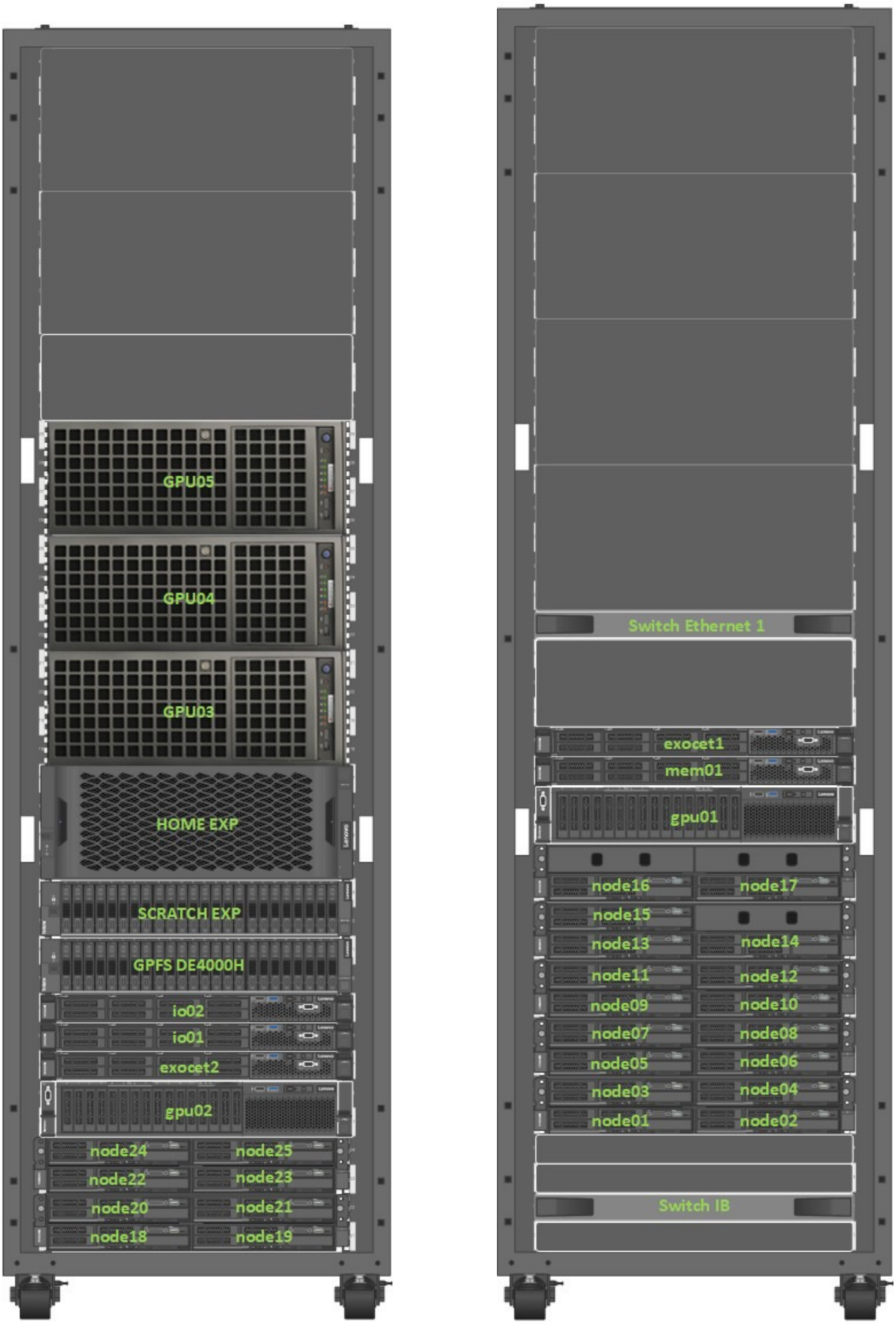


la deuxième



Faces arrières



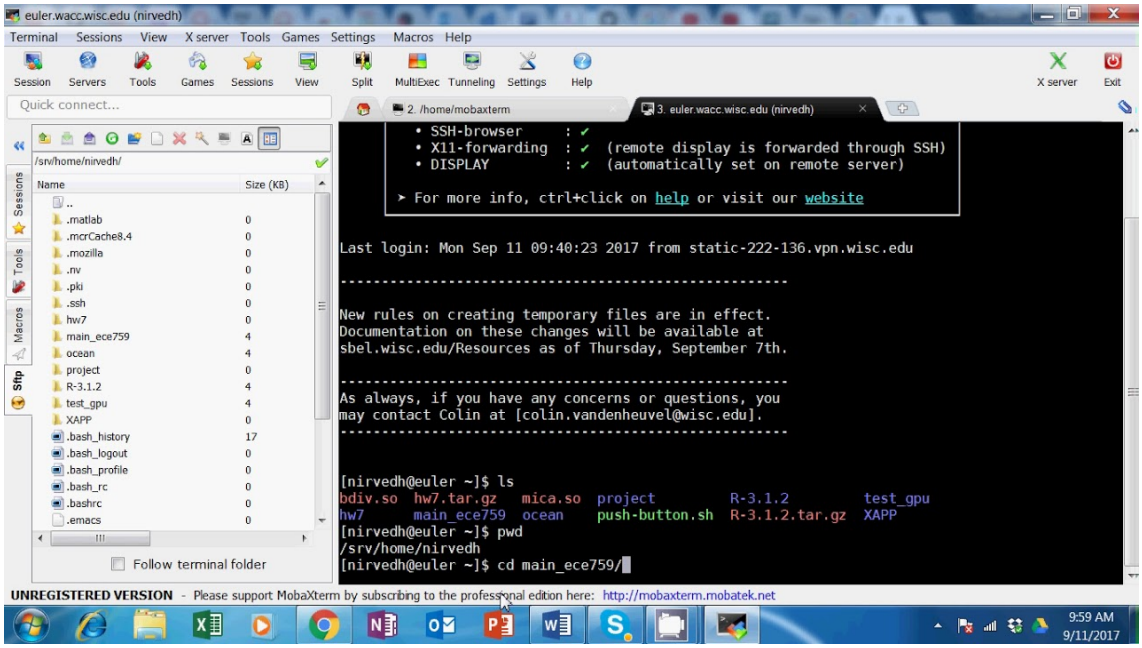

GPU05
GPU04
GPU03
HOME EXP
SCRATCH EXP
GPFS DE4000H
io02
io01
exocet2
gpu02
node24
node25
node22
node23
node20
node21
node18
node19
Switch Ethernet 1
exocet1
mem01
gpu01
node16
node17
node15
node13
node14
node11
node12
node09
node10
node07
node08
node05
node06
node03
node04
node01
node02
Switch IB

1 Se connecter, Copier ou récupérer des fichiers ou des répertoires

1.1 Connexion

1.1.1 Windows

Je vous conseille l'application graphique **mobaXterm** en version gratuite ("Home Edition"). Lancer l'application. Voici son interface :



Dans l'émulateur de terminal (zone noire), tapez la commande (comme pour Linux) :

```
$ ssh -X monlogin@exocet
```

si vous êtes sur le campus de l'Université. Remplacez "monlogin" par votre login sur **Exocet** . Ou sinon

```
$ ssh -X monlogin@exocet.univ-antilles.fr
```

Pour les usagers qui veulent se connecter à **Exocet** hors du campus, il faudra me fournir votre adresse IP publique. En effet les adresses IP sont filtrées pour des raisons de sécurité. Un exemple d'adresse IP est 89.157.98.209 (4 nombres entre 0 et 255 séparés par un point). Pour connaître son adresse IP publique, consulter par exemple le site web `https://www.whatismyip.com/`. Il vous affichera votre adresse.

Si vous souhaitez le frontal exocet1 plus particulierement, tapez :

```
$ ssh -X monlogin@exocet1
```

Si vous êtes itinérant et que vous utilisez un ordinateur portable par exemple, vous devrez vous connecter à un ordinateur "passerelle" nommé sasc3i comme ceci :

```
$ ssh -X monlogin@sasc3i.univ-antilles.fr
```

Même login et mot de passe qu' **Exocet**
puis connectez-vous à **Exocet** :

```
$ ssh -X monlogin@exocet
```

1.1.2 Linux, Mac, et autre Unix like

Le plus simple est de se connecter depuis un terminal en tapant :

```
$ ssh -X monlogin@exocet
```

ou si vous êtes à l'extérieur du campus de l'Université, travallant avec un ordinateur de bureau :

```
$ ssh -X monlogin@exocet.univ-antilles.fr
```

Remplacez "monlogin" par votre login. Pour les usagers qui veulent se connecter à **Exocet** hors du campus, il faudra me fournir votre adresse IP publique. En effet les adresses IP sont filtrées pour des raisons de sécurité. Un exemple d'adresse IP est 89.157.98.209 (4 nombres entre 0 et 255 séparés par un point). Pour connaître son adresse IP publique, consulter par exemple le site web `https://www.whatismyip.com/`. Il vous affichera votre adresse.

Si vous souhaitez le frontal exocet1 plus particulierement, tapez :

```
$ ssh -X monlogin@exocet1
```

Si vous êtes itinérant et que vous utilisez un ordinateur portable par exemple, vous devrez vous connecter à un ordinateur "passerelle" nommé sasc3i comme ceci :

```
$ ssh -X monlogin@sasc3i.univ-antilles.fr
```

Même login et mot de passe qu' **Exocet**
puis connectez-vous à **Exocet** :

```
$ ssh -X monlogin@exocet
```

1.2 Déplacer des données

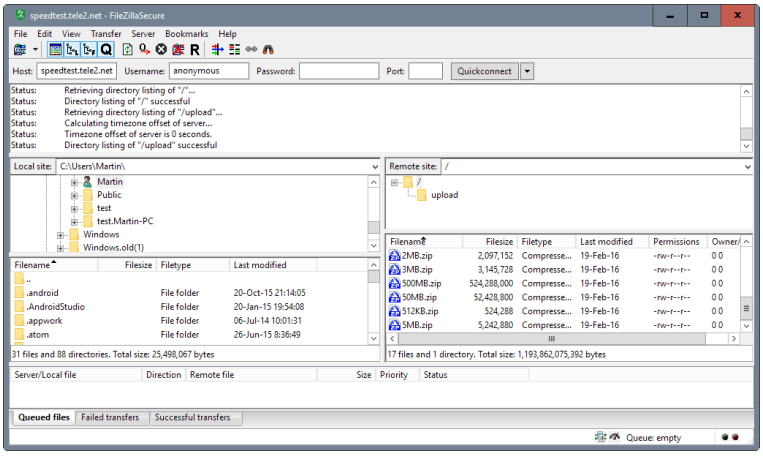
1.2.1 Windows

Si vous utilisez **mobaXterm**, vous pouvez glisser/déposer depuis l'explorateur de fichiers de Windows vers l'arborescence de fichiers de la fenêtre de **mobaXterm** (zone allongée à gauche). Cela fonctionne dans les deux sens.

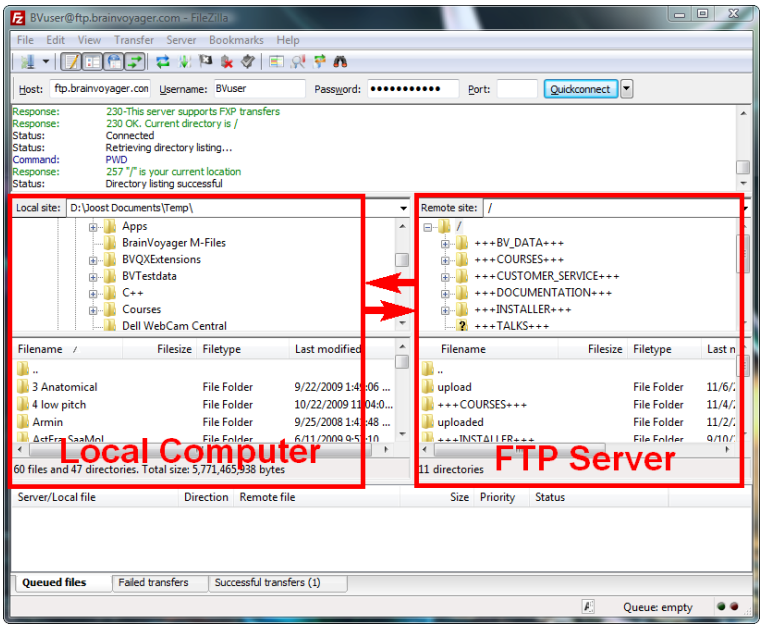
FileZilla est le logiciel le plus connu pour transférer, copier des répertoires ou des fichiers d'un ordinateur à un autre.


Attention! Il est très déconseillé de laisser ses mots de passe dans le logiciel **FileZilla**.

Voici son interface :



L'interface se divise en plusieurs zones dont les deux plus importantes sont encadrées de rouge :



La zone de gauche représente l'arborescence de fichiers de votre ordinateur et celle de droite celui de l'ordinateur distant. Celle-ci est vide au démarrage de Filezilla. Complétez les champs comme indiqué ci-dessous :

Host :	exocet.univ-antilles.fr
Username :	monlogin sur exocet
Password :	mon mot de passe sur exocet
Port :	22

1.2.2 Linux, Mac, et autre Unix like

Comme sur Windows, on peut utiliser **Filezilla** ou le vénérable **gftp**. Pour les utilisateurs qui utilisent la ligne de commande, le programme **scp** est très pratique (logiciel compris dans l'environnement SSH).
TODO

2 Le programme *module*

2.1 Présentation

module est un programme permettant de gérer rationnellement et facilement les variables d'environnements d'un terminal. En effet UNIX associe au `schell` (ligne de commande) une liste de variables d'environnement. Cette liste constitue le **contexte d'exécution** "UNIX" des programmes, des processus exécutés depuis le `schell`. La commande Bash `env` affiche cette liste de variables avec leur valeur. La valeur d'une variable est une chaîne de caractères. Cette liste et la valeur des variables dépendent du `schell`. On peut créer ses propres variables.
module a principalement un effet sur les très importantes variables `PATH` et `LD_LIBRARY_PATH`. `PATH` vaut une liste de répertoires séparés par le séparateur deux points :. Quand on tape une commande, le nom d'un programme, l'interpréteur de commandes cherchera le programme dans cette liste. On peut donc adapter la liste des programmes accessibles depuis le `schell`. `LD_LIBRARY_PATH` vaut aussi une liste de répertoires séparés par le séparateur deux points :. Quand on lance un programme ou une commande, le chargeur (loader) va copier le programme en mémoire RAM et si ce programme a été lié à des librairies dynamiques, le chargeur va chercher les librairies dont dépend le programme dans la liste de `LD_LIBRARY_PATH`. De nouveau, on peut contrôler les librairies utilisées par le programmes. Le chargeur fera d'autres petites tâches nécessaires et si l'édition des liens se termine bien, il proposera le

programme au processeur pour exécution.

Pour connaître la valeur de la variable `$LD_LIBRARY_PATH`, tapez :

```
$ echo $LD_LIBRARY_PATH
```

avec le symbole dollar $ devant le nom pour désigner sa valeur.

2.2 Utilisation

Le porgramme *module* gère des "modules". Pour afficher ceux disponibles, tapez :

```
$ module av
```

ou

```
$ module avail
```

On peut voir qu'ils sont actuellement regroupés dans trois listes :

- `/opt/ohpc/pub/modulefiles`
- `/opt/ohpc/pub/moduledeps/gnu9`
- `/opt/ohpc/pub/moduledeps/gnu9-openmpi4`

Un module chargé est suivi de `(L)`. La version de la librairie ou du programme associé au module suit son nom. Un module peut dépendre d'au autre module, comme par exemple `py3-mpi4py` dépend de `openmpi4`.

Pour charger un module, on tape :

```
$ module load nom_du_module
```

La valeur de `LD_LIBRARY_PATH` et `PATH` changeront probablement. Voici la liste des utilisations de *module* :

Table 2: Liste des options de `module`

`module av`	Liste les modules disponibles
`module whatis` *nom_du_module*	Affiche des informations sur le module
`module list`	Liste les modules chargés
`module load` *nom_du_module*	Charge le module
`module unload` *nom_du_module*	Décharge le module
`module purge`	Décharge tous les modules
`module help` *nom_du_module*	Donne des informations complémentaires sur le module

3 Le gestionnaire SLURM

3.1 Présentation

Le cluster **Exocet** se compose de nombreux serveurs utilisables par plusieurs personnes en même temps donc il est nécessaire d'utiliser un gestionnaire de ressources pour exploiter efficacement la machine. Le gestionnaire d'**Exocet** est **SLURM** (Simple Linux Utility for Resource Management). Il remplace avantageusement **SGE**, le gestionnaire de **Wahoo**, tout en gardant les mêmes principes de base.

Le principe général est le suivant : l'utilisateur fait une demande de ressources auprès de **SLURM** . On parle de soumission d'un "job".

Les nœuds d'**Exocet** sont regroupés dans différentes groupes, formant une partition des nœuds. **SLURM** associe à chaque groupe une file d'attente nommé "partition" dans **SLURM** . Lorqu'un utilisateur fait une soumission, il doit préciser dans quel groupe de nœuds son programme doit être exécuté et sa soumission sera enregistrée dans la file d'attente ou "partition" correspondante. S'il oublie de préciser le groupe, sa soumission sera enregistrée dans la file d'attente par défaut.

Voici la liste des partitions d'**Exocet** avec quelques caractéristiques :

Table 3: Liste des partitions

Nom	nom des nœuds de la partition	nombre de GPU par nœud
`cpu` (partition par défaut)	`node01` à `node25`	0
`gpu-v100`	`gpu01`	2
`gpu-t4`	`gpu02`	2
`gpu-p5000`	`gpu03` à `gpu05` (nœuds de Wahoo)	1
`mem`	`mem01`	0

Les partitions `gpu-v100`, `gpu-t4` et `mem` n'ont qu'un nœud.

Account est un concept nouveau de **SLURM** pour les utilisateurs habitués à SGE. Il correspond à la notion de projet ou de groupe comme les groupes UNIX. Ainsi chaque utilisateur d'**Exocet** appartient à un ou plusieurs **Account**. Pour faire simple sur **Exocet** , on considère **Account** comme un groupe dont le nom est le même que celui du groupe UNIX auquel appartient l'utilisateur. Par exemple, les utilisateurs du laboratoire Large de l'Université des Antilles sont dans l'**Account** `large`.

Tous les programmes de **SLURM** commencent par un s. Par exemple `sinfo` donne l'état des partitions.

3.2 Utilisation

3.2.1 État des files d'attentes

Pour voir les jobs qui tournent, qui sont en attente, on utilise la comamnde `squeue`. Voici un exemple de sortie :

```
JOBID PARTITION  NAME      USER      ST  TIME          NODES NODELIST(REASON)
2441  cpu        AD-mu9-t  lbatteux  R   7-00:03:22    1     node01
2447  cpu        AD-biflu  lbatteux  R   1-13:59:01    1     node01
```

Le champs "ST" (state) donne l'état du job. Ici les jobs sont exécutés ("R" pour running).

Si on ne veut afficher que ses jobs, on tape `squeue -u` *son_login*.

3.2.2 Activité d'Exocet

La commande `sinfo` affiche l'information :

```
PARTITION AVAIL  TIMELIMIT   NODES  STATE NODELIST
gpu-v100     up 1-00:00:00       1   idle gpu01
```

```
gpu-t4          up  1-00:00:00      1   idle  gpu02
gpu-p5000       up  1-00:00:00      3   idle  gpu[03-05]
cpu*            up    infinite      1    mix  node01
cpu*            up    infinite     24   idle  node[02-25]
mem             up    infinite      1   idle  mem01
```

L'état "idle" siginifie que le nœud ne fait rien et "mix" qu'il est partiellement utilisé. Le symbole étoile * à côté du nom de la partition signifie que c'est la partition par défaut.

Dans cet exemple, on voit que c'est le bon moment pour soumettre ses jobs. La place est libre.

Voici d'autres commandes utilies :

- Utilisation des cœurs CPU :

  ```
  $ sinfo -o%C
  ```

  affiche par exemple :

  ```
  CPUS(A/I/O/T)
  368/676/0/1044
  ```

  A désigne le nombre de cœurs utilisés, I (idle) le nombre de cœurs libres, O le nombre de cœurs en panne et T le nombre total de cœurs.

- Variante avec plus de détails :

  ```
  $ sinfo -o"%C %R"
  ```

  affiche par exemple :

  ```
  CPUS(A/I/O/T) PARTITION
  0/36/0/36 gpu-v100
  0/36/0/36 gpu-t4
  0/36/0/36 gpu-p5000
  368/532/0/900 cpu
  0/36/0/36 mem
  ```

- Utilisation par utilisateurs :

  ```
  $ squeue -o"%C %u"
  ```

3.2.3 Suppression d'un job

La syntaxe est : `scancel` *JOBID*
La commande `squeue` vous donnera l'identifiant du job à supprimer.

3.2.4 Mode

Comme pour SGE, il y a deux modes d'utilisation :

- mode "**intéractif**" utilisable par la commande `srun` (qui correspond à qrsh dans SGE),
- mode "**batch**" (ou différé) utilisable par la commande `sbatch` (qui correspond à qsub dans SGE).

Dans les deux cas, il faudra fournir des informations à **SLURM** sous forme de paramètres aux programmes `srun` et `sbatch` comme la partition souhaitée, le nombre de cœurs, le nombre de nœuds, etc... Il existe de nombreux paramètres dont voici les plus importants :

Table 4: Liste des paramètres ou options

Option	Commentaire
`-p` *nom_partition*	Choix de la partition
`-n` *nombre_entier*	nombre de processus MPI ou thread
`-N` *nombre_entier*	nombre de nœuds
`-A` *nom_Account*	Nom de votre Account
`-J` *nom_du_job*	
`-o` *filename*	Fichier de sortie
`-e` *filename*	Fichier des erreurs
`-mail-type` *type* `-mail-user` *Bugs.Bunny@gmail.com*	*type* est l'évènement provoquant l'envoi de l'email parmi les choix : `NONE`, `BEGIN`, `END`, `FAIL`, `REQUEUE`, `ALL`
`-D` *répertoire*	Répertoire de travail

Certains paramètres sont optionnels comme `-J`.

3.2.5 Mode "intéractif"

Dans ce mode, **SLURM** essaye de servir l'utilisateur immédiatement et la commande `srun` ne se termine que lorque la tâche est terminée. S'il n'y a pas de ressource disponbible, votre soumission sera refusée et il faudra revenir plus tard.
C'est le mode pour se connecter à un nœud pour développer un programme, utiliser un programme qui a une interface graphique comme MATLAB, Maple, etc...

Quelques exemples d'utilisation :

- `srun -n 1 -p gpu-v100 --gres=gpu:2 nvidia-smi`

  Exécution du programme `nvidia-smi` sur le nœud `gpu01` en réservant les deux cartes graphiques (`gpu:2`).

- `srun -p gpu-p5000 --x11 --gres=gpu:1 --pty bash`

  Exécution d'un terminal sur un des anciens nœuds de Wahoo avec réservation du GPU et renvoie X11.

- `srun -p cpu -N 1 -n 36 --pty bash`

  Exécution d'un terminal sur un nœud "calcul" en prenant tous les cœurs du nœud.

3.2.6 Mode "batch"

Dans ce mode, la soumission est mise dans la file d'attente de la partition mais jamais rejetée. Elle sera exécutée dès que les ressources nécessaires seront disponibles. Cela peut se produire juste après la soumission. Vérifiez l'état de votre soumission avec la commande `squeue`.

Il est intéressant d'enregistrer les options dans un fichier Bash pour une réutilisation. IMPORTANT : placez les paramètres de **SLURM** au début du fichier juste après le shebang, un paramètre par ligne et précédét du mot clé `#SBATCH`. Les commentaires Bash commencent par le signe dièse # et finissent à la fin de la ligne. Ainsi un paramètre **SLURM** sera interprété par Bash comme un commentaire et comme un paramètre pour le programme `sbatch`. Voici un exemple générique que vous mettrez dans un fichier :

```
#!/bin/bash

### Nom du job

#SBATCH -J test_openmpi

### Choix de la partition (file d'attente dans le vocabulaire SLURM) parmi :
### cpu        <=>  noeud "calcul" pour calcul sur CPU
### gpu-v100   <=>  noeud avec les deux GPU NVidia V100
### gpu-t4     <=>  noeud avec les deux GPU NVidia T4
### gpu-p5000  <=>  noeud avec un GPU NVidia QUADRO P5000
### mem        <=>  noeud avec beaucoup de memoire RAM

#SBATCH -p cpu

### Choix du "groupe"

#SBATCH -A lamia

### Choix du nombre de noeuds

#SBATCH -N 1

### Choix du nombre de processus

#SBATCH -n 900

### Envoi d'un courriel a la fin du job

#SBATCH --mail-type END
#SBATCH --mail-user Bugs.Bunny@gmail.com

### lancement du programme

mpirun -np 1 ./master_mpi : -np 899 ./slave_mpi
```

3.3 Utilisation avancée

TODO

4 Les conteneurs Singularity

4.1 Présentation

TODO

4.2 Utilisation

TODO

4.3 Utilisation avancée

TODO

5 Application Web LICO

5.1 Présentation

TODO

5.2 Utilisation

TODO

5.3 Utilisation avancée

TODO

6 Logiciels installés sur Exocet

6.1 Logiciels propriétaires

TODO

6.2 Logiciels libres

TODO