

Valérian Lambert  
Julie Leroy  
Arsene Jerome  
Thomas Madrange-Maire  
Clément Berard



université PARIS-SACLAY

IUT de Vélizy-Rambouillet  
CAMPUS DE VÉLIZY-VILLACOUBLAY

## Compte rendu – Configuration des RPis

# Sommaire

<b>I. Matériel.....</b>	<b>2</b>
<b>II. Préparation au 1er démarrage du contrôleur.....</b>	<b>2</b>
<b>III. Configuration du système de cluster et du SSH.....</b>	<b>3</b>
<b>IV. Installation de la bibliothèque Python MPI.....</b>	<b>6</b>
<b>V. Installation du dossier partagé NFS : Rapport dédié sur le GitHub.....</b>	<b>7</b>
<b>VI. Installation VNC &amp; Tailscale : Rapport dédié.....</b>	<b>7</b>
<b>Conclusion.....</b>	<b>8</b>

## I. Matériel

Matériel prêté par Mr Hoguin ce jour :

- 5 cartes microSD de 16Go avec un adaptateur microSD vers SD chacune
- 1 Raspberry Pi 4B avec son support transparent noir installé
- 1 Raspberry Pi Cluster Hat (vissé sur le Raspberry Pi 4B sur 4 montants) avec 4 Raspberry Pi 0 montés dessus
- 1 câble d'alimentation murale vers USB-C pour le Raspberry Pi 4B
- 1 câble USB / microUSB reliant le Cluster Hat au Raspberry Pi 4B
- 1 câble microHDMI vers HDMI pour relier le Raspberry Pi 4B à un moniteur externe
- (1 carte microSD laissé dans le Raspberry Pi 0 attaché au port 1 du Cluster Hat, cela doit sûrement être un oubli)

## II. Préparation au 1er démarrage du contrôleur

Nous avons utilisé pour notre cluster de RPi un ensemble d'OS nommés ClusterCTRL qui permet d'utiliser le clusterhat qui fait le lien entre le RPi 4 et les RPi 0.

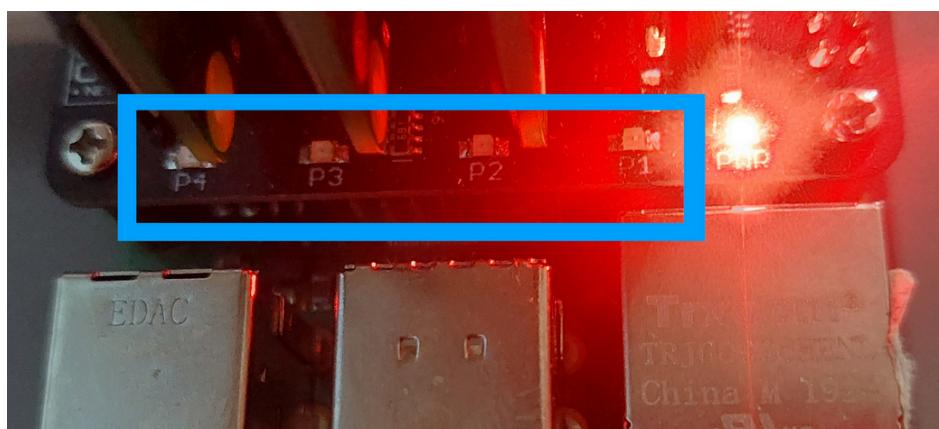
Toutes les images sont des images système 32bit car les RPi 0 ne supportent pas des images 64bit. Pour éviter des problèmes de compatibilité, nous avons fait le choix d'installer aussi un système 32bit sur le RPi 4.

Les images suivantes ont été sélectionnées sur [le site de ClusterCTRL](#), dans la catégorie 32bit :

- L'image CNAT - Desktop controller pour le RPi 4 (CNAT pour le type de communication entre le contrôleur et les RPi 0, Desktop psk si tu prend Lite t'as pas d'interface graphique et controller psk MDR le RPi 4 c'est un contrôleur)
- Les images P1, P2, P3 et P4 pour les 4 RPi 0

Nous avons choisi CNAT au lieu de CBRIDGE car ce dernier crée un pont SSH entre contrôleur et node, et est apparemment sensible au réseau du RPi 4 alors que CNAT fonctionne en réseau virtuel et est indépendant de la configuration réseau du RPi 4, et nous avons également choisi l'image Desktop pour avoir une interface graphique sur le contrôleur.

Il faut également faire correspondre les images des RPi 0 avec les inscriptions du clusterhat présentes à côté de chaque emplacement comme on peut le voir sur cette photo ci dessous.



Une fois téléchargés et décompressés avec WinZip (le site les donne en format xz), j'utilise Balena Etcher pour flasher les images dans chaque carte.

Une fois les cartes flashées, il faut créer un fichier nommé "ssh" sans extension dans la partition de boot (appelée bootfs après un flash, c'est normalement la seule partition ouvrable par l'explorateur de fichiers Windows) pour donner la possibilité à chaque RPi de communiquer par SSH.

On place chaque carte SD dans le RPi qui lui correspond et on peut démarrer le RPi 4.

### III. Configuration du système de cluster et du SSH

On entre l'utilisateur et mot de passe de l'utilisateur principal pendant la configuration de l'OS :

- Username (RPi 4) : cluster-ctrl
- Password (RPi 4) : sae5-ctrl

On donne également l'accès à Internet au RPi 4 (par WiFi ou Ethernet). Après redémarrage de l'OS pour que les changements prennent effet, nous avons accès au bureau du RaspberryPi OS.

Pour pouvoir configurer les RPi 0, on doit les planter dans le bootfs comme le fichier ssh :

- Les mots de passes doivent être cryptés à l'aide d'un système Linux, d'où le démarrage du RPi 4 avant celle des RPi 0. On ouvre un terminal et on crypte les mots de passes souhaités pour chaque utilisateur :

```
echo 'mot-de-passe' | openssl passwd -6 -stdin
```

- On crée un fichier userconf.txt dans la partition bootfs des cartes SD dans lequel on entre l'utilisateur et le mot de passe crypté dans ce format précis :

```
username:password
```

Ce qui nous donne une ligne dans ce format :

```
pi:$6$Fc...tu.cczYy8v8x.axK38k6LFHe1Qhz4JuuQ6BpkGK13MVm86SLFhNpEyj5xkqbHMtZ8n6/80A0u8UCI1
```

Tous les utilisateurs des RPi 0 ont pour nom pi. Leurs mots de passe sont :

p1 : sae5-p1  
p2 : sae5-p2  
p3 : sae5-p3  
p4 : sae5-p4

On remet les cartes SD dans les RPis en se souvenant de l'emplacement défini par le clusterhat et on peut configurer la communication SSH.

La première chose à faire est d'ouvrir le terminal et de configurer le SSH en générant une clé SSH (laisser toutes les options de la commande vides) :

```
ssh-keygen -t rsa -b 4096
```

```
Your identification has been saved in /home/cluster-ctrl/.ssh/id_rsa
Your public key has been saved in /home/cluster-ctrl/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:B3xpLDLQmMQTh07kBzKW0J50g660Mlu6oB5JMXLQVXE cluster-ctrl@cnat
```

On affiche ensuite la clé publique et on la copie pour l'utiliser ultérieurement :

```
cat ~/.ssh/id_rsa.pub
```

```
ssh-rsa
AAAAB3NzaC1yc2EAAAQABAAQADMS3+2arB3tTdPIp4Hfek+/YlmMvax4x2TEOCpI2jd
KAM96pRa0vBPmy0XH+y5TcnwB0W8P3s+J9wcF3hL/sd5+98rcvalL9M3CeeYy6bvysQHFJ+i
6B0+HalTI09Cm0SC0u2tYiDf9yIzUJjwayfzWE3b934aCuvhQmWKv9zQDEGoT18h9r0JytIY
y9fRqj8ZEKdBC2PpkSJCP17WhjUq/pc/zd4vby+Re0AJWIC6tVs2U7RLQ6Mz74gQIHalkd0
fPO9vnCsUFQXErMXoZ4Eu6wz+/f03hJuLcK9SdGecnaDDFL6aEdRAENJT+9xmFYRzBnJKd8
m3UGjREnDUAAnp/bLGFL7QUt2t/yXiVz9Vh7Y/xU1o4x0VUeKgib+A/+MEIcneRrQ8LZS1s
XFFKLQyBgmahUbKmpM0BDr3j/eMINvuyFU6cFQj/LOD8D0zlo27c03S+XxUm/10KpjqsAWE1
SHwi8U1DGcwoafIb/AqmLtXXZBik1BdB1eC1+0j9ZP2Z6Q+0xMNnbFRHJuYfwwl+p+vsMYt0C
H4sRFo9YXmTCMkf/VoJzEVdyTeyXDLgbvqjLn7rX0pypkMue4yG3WLJS9Twk8GL72EV1Le9K
vYDYuwEe3vfvQFMZaqPalsC7HVDL1zSTTaYyY8YF5QVja0Fj+2/vlp1RkoHaZpiDtQ==
cluster-ctrl@cnat
```

Dans ce même dossier de configuration dans lequel la clé SSH a été créée, on crée un fichier `~/.ssh/config` dans lequel on met la configuration suivante :

```
Host p1
  Hostname 172.19.181.1
  User pi

Host p2
  Hostname 172.19.181.2
  User pi

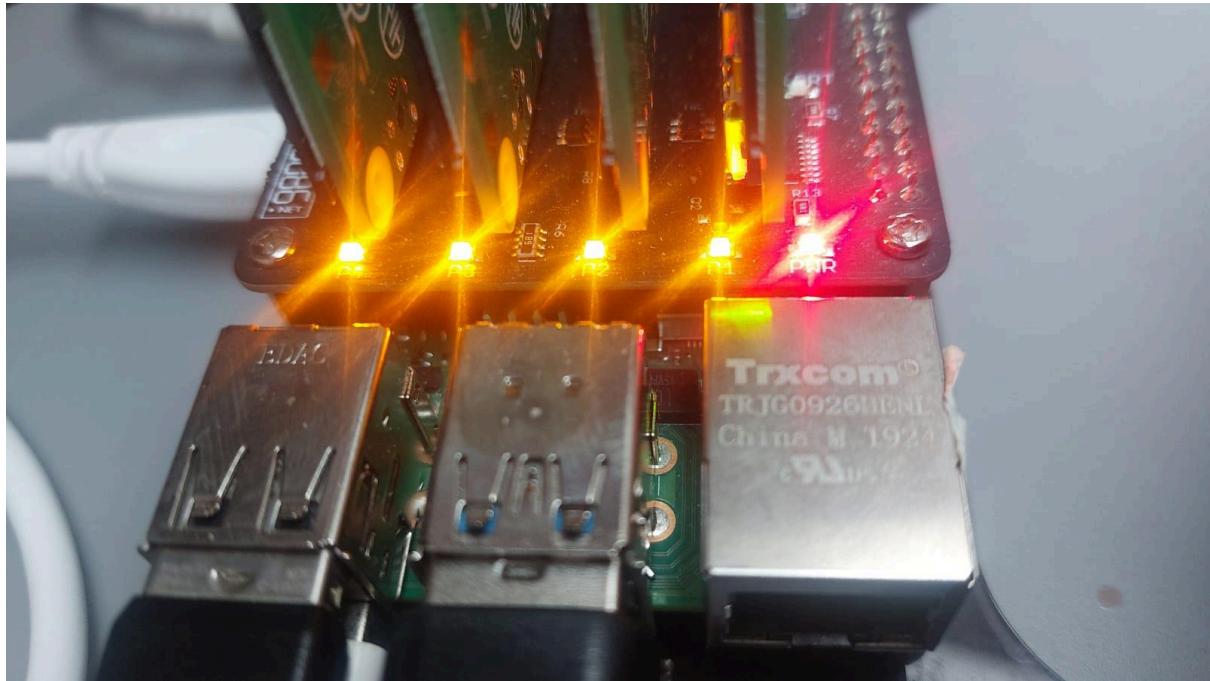
Host p3
  Hostname 172.19.181.3
  User pi

Host p4
  Hostname 172.19.181.4
  User pi
```

On allume le clusterhat et les RPi 0 qui sont dessus :

```
sudo clusterhat on
```

Sur le clusterhat, on observe que les LED des emplacements de RPi 0 s'allument l'une après l'autre :



On peut donc se connecter en SSH avec chaque node du clusterhat :

```
ssh p1
```

Maintenant que nous avons accès aux 4 RPi0, il faut les configurer comme le RPi contrôleur :

```
ssh p1
```

Dans le RPi 0 :

```
mkdir ~/.ssh  
touch ~/.ssh/authorized_keys  
echo mettre_ici_la_clé_ssh_publique >> ~/.ssh/authorized_keys
```

Maintenant, on peut se connecter avec le mot de passe :

```
ssh p1
```

ou sans, car la clé publique mise dans chaque RPi 0 sert d'authentification :

```
ssh pi@p1
```

On entre ensuite dans la configuration du RPi 0 :

```
sudo raspi-config
```

Une fois dans le raspi-config, il faut impérativement configurer la timezone dans le menu de

localisation pour avoir les mêmes heures entre les RPi.

## IV. Installation de la bibliothèque Python MPI

On commence par installer MPI et le package Python qui correspond sur chaque RPi (node et contrôleur) :

```
sudo apt install mpich python3-mpi4py
```

Tester que l'on peut lancer une tâche parallèle depuis le contrôleur :

```
mpiexec -n 4 --host p1,p2,p3,p4 hostname
```

Il faut maintenant faire un test avec un vrai programme, ici nous prendrons le code prime.py fourni par Mr Hoguin. Avant de lancer directement le programme depuis le contrôleur, il faut que le code soit présent aux mêmes emplacements et identiques sur tous les RPi.

Sur chaque RPi (contrôleur + nodes) :

```
mkdir -p ~/Documents/cluster_programs/
```

Sur le contrôleur :

```
cp -r chemin/absolu/vers/les/deux/codes ~/Documents/cluster_programs/
```

Sur le contrôleur, pour chaque node (changer pi@p1 pour chaque node) :

```
scp -r ~/Documents/cluster_programs/*  
pi@p1:~/Documents/cluster_programs/
```

Maintenant que le programme prime.py est installé aux mêmes emplacements sur tous les RPi, on peut déjà tester de le faire fonctionner sur le contrôleur :

```
mpiexec -n 1 python3 ~/Documents/cluster_programs/prime.py 10000
```

On reçoit le résultat en environ 1.58 secondes selon la commande.

On exécute maintenant le code sur un seul node :

```
mpiexec -n 1 -host p1 python3  
/home/pi/Documents/cluster_programs/prime.py 10000
```

On reçoit le résultat en 21.74 secondes selon la commande.

Maintenant, essayons de paralléliser la même commande sur les 4 nodes :

```
mpiexec -n 4 -host p1,p2,p3,p4 python3
```

```
/home/pi/Documents/cluster_programs/prime.py 10000
```

On reçoit le résultat en 5.0 secondes.

## V. Installation du dossier partagé NFS : [Rapport dédié sur le GitHub](#)

Le dossier NFS est un dossier partagé entre tous les RPi, ce qui s'est révélé utile pour exécuter des programmes identiques sur chaque node du cluster, ainsi que sur le contrôleur.

Nous avons donc stocké l'intégralité des programmes parallèles de calcul dans ce dossier partagé pour ne pas se soucier des problèmes de version.

## VI. Installation VNC & Tailscale : [Rapport dédié](#)

Tailscale crée un groupe d'appareils dans lequel il est possible de SSH par tunnel quelle que soit sa configuration réseau. Ainsi, chaque appareil ajouté au groupe du RPi peut travailler dessus de chez lui.

On peut y ajouter un logiciel de bureau à distance (VNC) comme RealVNC pour utiliser l'interface graphique du RPi à distance.