École nationale supérieure d'informatique pour l'industrie et l'entreprise

---

«Appel d'offre»

Élèves : **Aurélien LHUILLIER Evan BERROU**

Sommaire

1 Developpement

1.1 Conformité de la réponse technique au cahier des charges

	Partition	Type	Nbr de noeuds demandé	Go de mémoire par coeur demandé	Nbr de noeuds fournit par AzkarHPC	Nbr de coeurs fournit par AzkarHPC	Go de mémoire par coeur fournit par AzkarHPC
AC	AmdCpu	NA	40000	< 2	302	38656	256
AL	AmdLarge	NA	4000	< 4	50	4200	512
AX	AmdGpu	NA	3000	<2	50	1200	256
login	frontale	NA	24	<2	10	84	NA
admin	admin	10	NA	NA	20	NA	96

TABLE 1 – Comparaison cahier des charges et réponse technique

Partition AC

La réponse à l'appel d'offre propose 38656 coeurs sur les 40000 demandés avec 302 noeuds comprenant un processeur AMD 9754 de 128 coeurs, cela reste dans la limite des 5% demandés. Chaque noeud disposera de 256 Go de mémoire DDR5 ce qui correspond bien au 2 Go par noeuds du cahier des charges.

Partition AL

La partition AL propose 4200 coeurs répartis sur 50 noeuds comprenant un processeur AMD 9634 de 84 coeurs. Cela excède de 5% le nombre de coeurs demandés. Avec 256 Go de mémoire DDR5 par noeuds, il y a 6 Go de mémoire par coeurs ce qui excède beaucoup les 4 Go requis.

Partition AX

La partition AX possède seulement 50 noeuds avec des processeurs AMD 9224 de 24 coeurs soit 1200 coeurs sur les 3000 demandés. De plus avec une carte GPU NVIDIA A100 sur les 50 noeuds, les 60 cartes demandées ne sont pas atteintes. Pour ce qui est de la mémoire par noeuds, on atteint 10 Go avec 256 Go de DDR5 pour 24 coeurs.

Partition F

La partition F comprend bien les 10 noeuds demandés avec chacun des processeurs AMD 9634 de 84 coeurs pour 96 Go de mémoire DDR5 et 2 disques de 4 To, cela implique une mémoire par coeur d'environ 1 Go . Ces caractéristiques ne correspondent pas aux demandes, puisqu'il fallait 24 coeurs par noeuds avec chacun 2 Go de mémoire et des disques sécurisés de 3 To.

Partition Srv

La partition serveur comporte 3 types de noeuds : les noeuds maîtres, les noeuds de service et les noeuds routeurs I/O. Le cahier des charges requiert 10 noeuds de services comprenant 2 disques sécurisés de 1 To pour le système et deux disques sécurisés de 4 To pour les données système. Cela n'est pas respecté par la proposition de AzkarHPC puisque seuls deux disques de 3 To sont inclus pour chaque noeud. Il faut cependant prendre en compte le serveur de stockage comprenant 30 disques de 18 To pour les données système. En utilisant 4 de ces disques, on peut obtenir un stockage équivalent à 2 disques de 4 To pour chacun des noeuds. Cependant cela rend les disques d'autant plus critique, on pourrait potentiellement répartir ce stockage dans plus de 4 des 18 disques pour augmenter la résilience mais cela poserait des problèmes de sécurité puisque des données système pourrait être sur le même disque physique que des données gérées par les utilisateurs même si on partionne. Pour ce qui est du nombre de coeur par

noeuds et de la mémoire, les attentes d'au moins 16 coeurs et 256 Go de mémoire sont respectés avec les 2 fois 24 coeurs des 2 processeurs AMD9224 et les 256 Go de mémoire DDR5.

Offre logicielle

AzkarHPC propose bien une licence RHEL9 comme demandée. Leur offre comprend également des rpms Lustre pour les noeuds clients et routeurs, ainsi que des compilateurs Intel et amd, un debugger Totalview ainsi que des logiciels OpenSource. Les rpms Lustre et les compilateurs AMD sont bien requis au vu de l'organisation du cluster, cependant les compilateurs Intel sont surprenant car aucun des processeurs sélectionnés par AzkarHPC n'est Intel. Le debuggeur Totalview est bien adaptés pour la partition AX puisqu'il est destiné au GPU.

débit I/O

Les mêmes ports InfiniBand HDR200 sont présents sur les 3 partitions et offrent finalement un débit de 24 Go/s par noeud (limité par le nombre de routeurs et de top switchs dans le réseau d'interconnexion) avec 9 routeurs et 6 top switchs, on peut avoir jusqu'à 144(24×6) Go/s ce qui correspond à la demande de 140 Go/s du cahier des charges. Le réseau de stockage lustre est bien un réseau Infiniband HDR200. Pour ce qui est du débit vers le backbone, les 3 partitions passent à nouveau par les routeurs et empruntent cette fois un port Ethernet 10 Gb des routeurs vers le backbone. Les 140 Go qui parviennent aux routeurs sont donc limités à $9 \times 10 = 90$ Gb ce qui valide la demande du cahier des charges.

Racking

Un châssis AZ-B30 occupe 16 U et on dispose de racks 48 U, on peut donc mettre 3 châssis par rack. La partition AC contient 302 noeuds qui peuvent être réparti sur 11 châssis de 30 lames dans 4 racks. La partition AL contient 50 noeuds qui peuvent être réparti sur 2 châssis sur 1 rack. La partition AX a 50 des noeuds qui prennent 2U ce qui correspond à 3 châssis dont un qui comporte seulement 2 noeuds si les autres sont pleins. La partition Srv comprend au total 20 noeuds pour 40U ce qui peut être contenu dans un seul rack. La partition F compte 10 noeuds pouvant être compris dans un seul rack. Enfin, le serveur NFS compte pour 2U et peut être intégré dans un des racks de service.

AzkarHPC propose bien le nombre de rack attendu pour les partition AC, AL et AX et regroupe les partitions Srv et F, ainsi que le serveur NFS dans 2 racks. Ils ajoutent également un rack pour les switchs Ethernet et InfiniBand non intégrés dans les châssis.

Conclusion

L'utilisation de contrôleur RAID sur le stockage pour la résilience des donnés est un point positif. Il est également possible de rendre moins chère la partition AL qui coûte 50*10304=515 200 $ alors que l'on pourrait réduire le nombre de noeuds et faire une économire de 188 800 euros en prenant les processeurs AMD 9754S.Certe ceux-ci contiennent un cache L3 inférieur mais celui-ci n'est pas demandé dans le cahier des charges.
L'utilisation des technologies diskless sur les partitions AC, AL et AX permet de réduire les coûts du stockage mais impose des contraintes sur le réseau.
La partition F offre des des processeurs avec plus de coeurs que requis ce qui permet moins de mémoire par coeur que ce qui est demandé. On pourrait échanger les processeurs AMD 9634 contre des AMD 9224 ce qui permet de dégager 84 790$ de budget dans les autres partitions La partition AX est bien en deçà des attentes, on pourrait envisager réduire l'investissement sur la partition AL, qui elle les excède pour mettre plus de noeuds sur la partition AX.
En supprimant 2 noeuds sur la partition AC et 4 sur la partition AL, on reste dans les 5 % de tolérance les concernant et on libère 65 016$ de budget sur les processeurs que l'on peut réinvestir sur la partition AX. On suppose que le prix des autres composantes des noeuds retirés compensent ceux des noeuds achetés, sans prendre en compte la carte NVIDIA. Il faut au minimum 10 noeuds supplémentaires pour atteindre les 60 cartes NVIDIA. On peut consacrer le reste du budget dégagé, en comptant celui de la partition F, (169 580$) pour l'achat des 10 cartes manquantes et des autres composants que l'on a pas pu récupérer sur les autres partition. Cela permettra d'atteindre 60 cartes NVIDIA et 60 processeurs AMD 9224 pour 1440 coeurs, ce qui reste extrêmement insuffisant par rapport à la demande. Le fait que les racks ne soient pas pleins permet de faire évoluer le cluster.

1.2 Nommage du cluster et des composants matériels

1.2.1 Nom du cluster et Organisation logique des nœuds

Il devra comporter .univ-tlse3.fr car il fera partie du sous réseau de l'université. On remarque également que tous les noms sont en rapport avec les montagnes des pyrénées.Le nom choisi devra être explicite pour que l'on sache que c'est le cluster sans chercher.De plus celui-ci doit s'inscrire dans l'architecture déjà présente. On utilisera le nom de Montaigu car c'est un mont qui fût connu pour ses réserves d'argent et donc ses réserves minières. Pour les noeuds

- montaigu[0-10] noeuds admin/maitres
- montaigu[11-60] noeuds de login/srv
- montaigu[70-90] noeuds frontaux
- montaigu[100-700] pour les noeuds de calculs
- montaigu[701 -1000] pour les noeuds équipés de gpu

Nom du groupe @xxxx	Nodesets	Commentaire
@admin	montaigu[0-10]	Noeuds pour la gestion
@login	montaigu[11-25]	Gestion des connexions
@io	montaigu[26-40]	Gestion du stockage
@service	montaigu[41-70]	Gestion des services
@calc	montaigu[100-700]	Gestion des services
@gpu	montaigu[701-1000]	Gestion des services

Table 2 – Plan de connectivité externe au Backbone de l'université

Pour les numérotations, celles-ci contiendront des adresses libres pour garder la possibilité de l'ajout de noeuds plus tard. On garde de la place dans les adresses proportionnellement au nombre de noeuds qui seront potentiellement ajoutés. Par exemple, on a laissé beaucoup plus de place dans l'espace des noeuds de calculs par rapport aux noeuds de login. De plus la proportion de place laissée pour les noeuds équipés de GPU peut paraître démesurée mais avec l'avènement de l'IA dans plusieurs domaines ce choix est cohérent.

1.3 Architecture du réseau de contrôle électrique 1Gb du cluster

Noeuds	Port Switch AZ-W40-1g	Commentaire
nœud master1 ports Gb [1-3]	ewc1p[1-3]	Ils leur reste donc 3 ports Gb disponibles qui peuvent être mis sur les 3 switchs pour un maximum de résilience
nœud master2 ports Gb [1-3]	ewc2p[1-3]	Ils leur reste donc 3 ports Gb disponibles qui peuvent être mis sur les 3 switchs pour un maximum de résilience
1 serveur NFS admin et 2 serveur NFS , port BMC	ewc1p38 ewc2p38 ewc3p38	
9 nœuds de service, port BMC	ewc1p[4-6]ewc2p[3-5]ewc3p[2-4]	On répartit pour assurer la disponibilité du service
10 nœuds de login, port BMC	ewc1p[7-10]ewc2p[6-8]ewc3p[5-8]	Comme au dessus
9 nœuds routeur, port BMC	ewc1p[11-13]ewc2p[9-11]ewc3p[9-11]	
50 nœuds graphiques, portBMC	ewc[1-3]p[14-29]ewc2[2][12-13]	
Châssis AZ-B30, port BMC	ewc[1-3]p[30-36]	idem
Les switchs IB externes aux châssis et les switchs Ethernet	ewc[1-2]p[37]	On branche sur 2 différents pour essayer de garder la disponibilité même partiel

Table 3 – Plan de câblage du réseau de contrôle

On a cherché à séparer les différents nœuds occupant une même fonction sur différents switchs pour pouvoir quand même contrôler les éléments si un switch venait à tomber en panne. De plus, ne pas remplir totalement les switchs les uns après les autres permet d'éviter la congestion.
Il n'y a qu'un serveur NFS alors que c'est un serveur pour les files system donc si celui-ci ne fonctionne plus aucun des autres serveurs ne peut accéder aux données. Il manque la ligne sur les switch externe(tous les switchs IB et Ethernet externes via leurs alimentations reliées à des PDU. De plus, on pourrait ajouter les 10 cartes graphiques manquantes que l'on pourrait brancher.

1.4 Architecture du réseau d'administration 10 Gb du cluster

Noeuds	Port Switch AZ-W40-10g	Commentaire
nœud master1 ports 10Gb [1-2]	ewa[1-2]p1	Pour garder la disponibilité on met sur des switchs différents pour limiter les dégâts d'une panne sur un switch
nœud master2 ports 10Gb [1-2]	ewa[3-4]p1	idem
1 serveur NFS admin et 2 serveur NFS	ewa[1-4]p[2-4]	Pour limiter les SPOFs
9nœuds de service, 1 port 10Gb	ewa[1-6]p5ewa[5-6]p1ewa5p2	On répartit équitablement pour la disponibilité
10 nœuds de login , 1 port 10Gb	ewa[1-6]p6ewa[1-4]p7	On répartit équitablement pour la disponibilité
9 nœuds routeur, 1 port 10Gb	ewa[1-6]p8ewa[5-6]p7ewa1p9	On répartit équitablement pour la disponibilité
50 nœuds graphiques, 1port 10Gb	ewa[1-6]p[10-17]ewa[1-2]p[18]	On répartit équitablement pour la disponibilité et éviter les SPOFs
Châssis AZ-B30, 4 ports 10Gb	ewa[1-4]p[20-32]	Contenu

Table 4 – Plan de câblage du réseau d'administration 10Gb

On dénombre 134 ports occupés sur les 240 disponibles, des câblages supplémentaires pourront donc être aisément ajoutés. L'organisation permet que la panne d'un port n'ait pas d'influence sur les partitions AC et AL mais les nœuds des partitions AX et F étant reliés par un seul port au réseau, le nœud concerné sera nécessairement hors service. En cas de panne d'un switch, les services qui ne sont que sur un des deux de services ne seront plus disponibles, alors que les autres le seront. De plus, tous les nœuds des partitions F et AX connectés à ce switch seront affectés. On sépare donc les nœuds d'un même groupe d'utilisateur de la partition F sur différents switchs pour que les utilisateurs aient toujours accès à des nœuds. Au contraire, on regroupe les nœuds graphiques proches sur un même switch pour avoir une panne localisée puisque de toute façon le même nombre de nœuds sera impacté sur la partition.

1.5 Connectivité externe au Backbone de l'université

On va connecter 2 ports sur chaque noeud maitre car ceci ont 4 port 10GB et 2 sont déjà pris pour la connexion internet.Pour les noeuds de service on fait de même sauf que ceci sont 9.Pour les noeuds routeurs on connectera 4 et laissé 1 port libre pour des extensions.De plus,nous voulons connecter les noeuds de login au backbone ,nous prendrons 1 ports sur chacun.En tout On demandera finalement 68 connexions au backbone de l'université

Nodeset	Serveurs université	Flux/protocoles
@admin	LDAP NTP DNS NFS	Gestion des noeuds
@login	Salles TP,Bureaux des professeurs,Bureaux PME	Gestion des connexions
@io	NFS,idem au dessus	gestion du stockage
@service	LDAP,DNS,NTP	Gestion du flux de donné

Table 5 – Plan de connectivité externe au Backbone de l'université

Le serveur PXE ne sera pas utilisé ici, on en fera un autre au sein du cluster. Les services DNS et LDAP pour l'identification doivent absolument être reliés au cluster. Les nœuds de login sont reliés au

NFS pour pouvoir accéder à leur répertoire HOME. Les noeuds maîtres peuvent être connectés au serveur NTP de l'université et déléguer la gestion du service sur les autres noeuds en faisant d'un des noeuds de service un serveur NTP de couche inférieur. On conserve également des ports en cas d'extension futur

1.6 Architecture du réseau d'interconnexion Infiniband

Noeuds	Port IB	Commentaire
2 noeuds master	iwh1n[16-17]p1	Sur des switchs différents pour la résilience en cas de panne d'un switch
noeuds de service	iwh1n[16-17]p[2-5] et iwh1n16p6	
noeuds de routeurs	iwh1n17p6 et iwh1n[16-17]p[7-10]	
noeuds de login	iwh1n[16-17]p[11-15]	On mettra les noeuds de login d'un même groupe d'utilisateur sur différent switch pour qu'ils puissent se connecter en cas de panne d'un switch.

Table 6 – Plan de câblage du réseau d'administration 10Gb

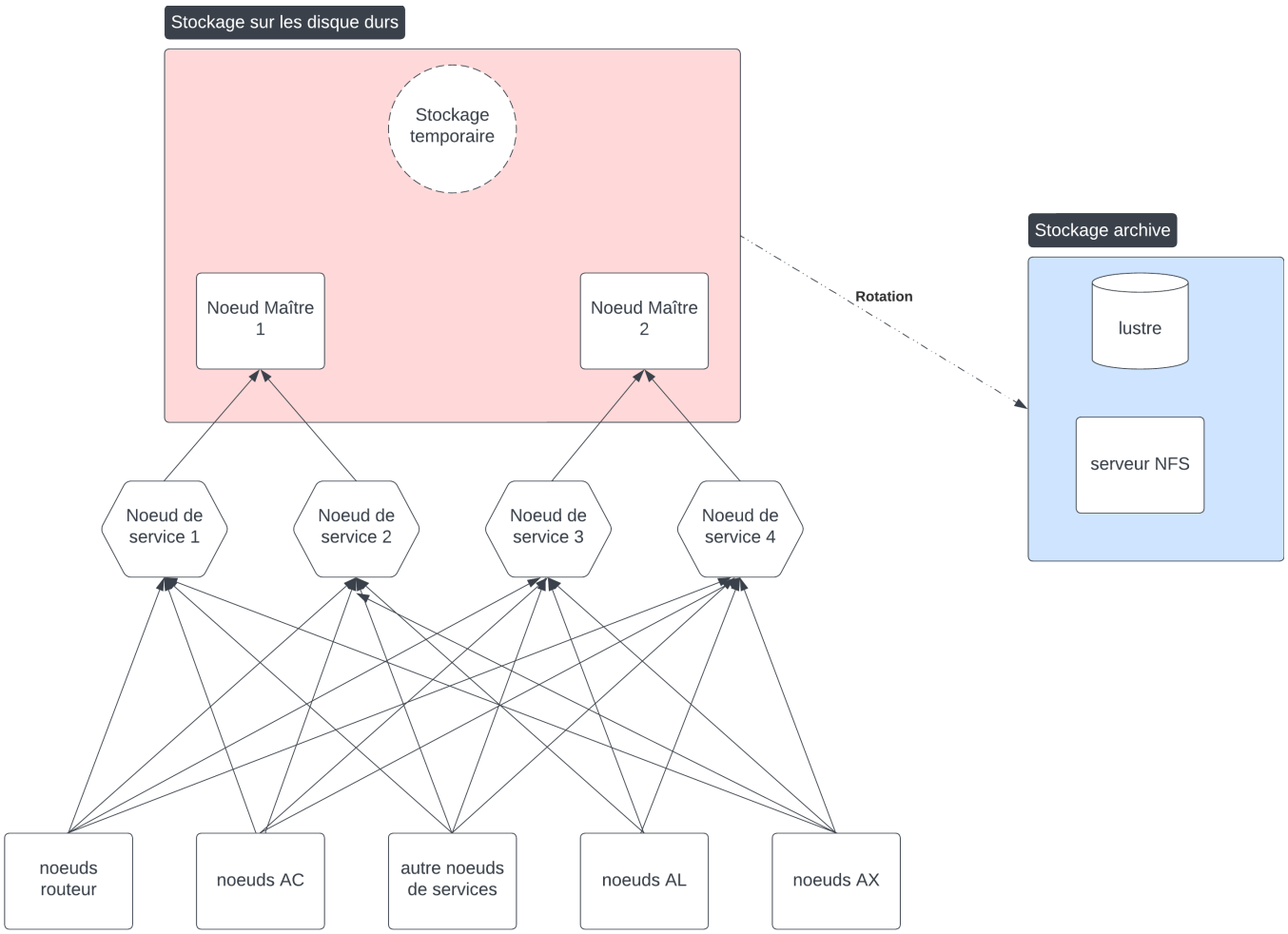
Ce plan est redondant et résilient puisqu'on a distribué les nodesets des services sur les deux switchs. En cas de panne d'un switch la moitié du nodeset sera toujours disponible. Pour ce qui est des groupes d'utilisateurs, il y a deux groupes sur 3 nœuds et deux groupes sur 2 nœuds. On sépare les nœuds d'un même groupe pour qu'en cas de panne d'un des switchs, les utilisateurs aient tout de même accès au cluster et à ses services sur le ou les nœuds restants.

1.7 Extensibilité de la solution

Sans rajouter de châssis, on peut ajouter 28 lames donc 28 nœuds dans la partition AC et 10 lames donc 10 nœuds dans la partition AL. Ces noeuds seront sur un châssis préexistant et ne changeront donc pas le réseau d'interconnexion InfiniBand.
Sans ajouter de racks, on peut ajouter 1 châssis soit encore 30 nœuds supplémentaires dans chacune des partitions AC et AL. Cela ajouterait deux switchs supplémentaires dans le réseau d'interconnexion, on augmenterait alors de 1 les numéros des switchs au-delà de 11 et de 2 ceux au-delà de 13, libérant le 12ème switch et 15ème switch pour respectivement les partitions AC et AL. Les top switch n'auront pas de problèmes à ajouter 2 autres connections puisqu'ils ont 13 ports disponibles.
Chaque nœud de la partition AX prend 2U dans ses propres racks, il y actuellement 44U disponibles (équivalents à 22 nœuds) dans ces racks. On les lie à des switchs L1 pour permettre leur appartenance au réseau d'interconnexion. Il y a 2 switchs dédiés aux 50 nœuds de la partition AX. Cela rajoute donc 2 switchs dans le rack switch, en plus des 3 switchs du réseau de contrôle et des 6 switchs du réseau d'administration. Chaque switch occupe 2U pour un total de 22U. Il reste donc assez de place pour 6 switchs. De plus l'ajout de serveur NFS permet d'agrandir facilement le cluster sans vraiment augmenter le prix.

1.8 Récupération des logs et des consoles

On va centraliser tous les logs des partitions AX, AC, AL, F, ainsi que celle des noeuds admin, routeurs et de services. On va utiliser rsyslog qui permet de transmettre les logs entre machine. On va transmettre tous les logs indépendaments de la facilité ou du niveau d'erreur. La politique est de stocker temporairement les logs dans les disques durs des serveurs maîtres pour que les logs récents soient facilement accessible et d'organiser des rotations des logs vers le serveur NFS pour éviter une perte des logs due au manque de stockage disponible. On utilisera 4 noeuds de service qui sont connecté sur les switchs ewa1 ewa2 ewa3 et ewa4 car le serveur NFS et les noeuds maîtres y sont connectés également. Ces noeuds sont : montaigu[44-47].



On utilise les deux disques de 3 To des noeuds maîtres pour l'arborescence /var/cluster/. Pour configurer les redirections via TCP (port 514 usuel pour rsyslog), j'ai besoin des adresses IP de la destination : 10.0.0.1 est l'adresse IP d'un noeud maître. Il faut faire la différence entre les fichiers de logs propres au noeud et ceux obtenus par TCP dans rsyslog.conf pour pouvoir respecter l'arborescence.
Contenu du fichier "/etc/rsyslog.conf" d'un des noeuds maîtres

```
# Configuration du répertoire de destination des logs
$WorkDirectory /var/lib/rsyslog

# Configuration de la réception des logs provenant des applications locales
$ModLoad imuxsock # Module pour les logs locaux
$ModLoad imklog   # Module pour les logs du noyau

# Configuration de la réception des logs provenant des autres nœuds du cluster
# Noeuds AC
$ModLoad imtcp # Module pour écouter en tcp
$InputTCPServerRun 514
# Pas UDP pour éviter les pertes

kern.* /var/cluster/consoles/%HOSTNAME%/messages
# Filtrage des logs propres aux noeuds et des logs reçus par TCP
*.* if $fromhost-ip == '127.0.0.1' then /var/cluster/logs/%HOSTNAME%/messages #dans le répertoire cor
& ~
*.* if $inputname == 'imtcp' then /var/cluster/logs/%FROMHOST%/messages
& ~
```

Contenu du fichier "/etc/rsyslog.conf" d'un des noeuds d'une partition ou routeur

```
# Configuration du répertoire de destination des logs
$WorkDirectory /var/lib/rsyslog
```

```
# Configuration de la réception des logs provenant des applications locales
$ModLoad imuxsock # Module pour les logs locaux
$ModLoad imklog   # Module pour les logs du noyau
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat
# Redirection des logs et consoles
kern.* @@10.0.0.1:514;*.* /var/cluster/consoles/%HOSTNAME%/messages

*.* @@10.0.0.1:514;*.* /var/cluster/logs/%HOSTNAME%/messages
```

On défini également des logs rotate. On ne peut pas utiliser olddir car cela permet uniquement de faire la rotation sur le même espace de stockage. On doit donc utiliser le mail. On fait des rotations tous les jours ou bien quand le stockage devient trop important et on conserve les versions les plus récentes des logs. Les utilisateurs non root n'ont pas de droits sur les logs anciens. C'est la politique de gestions des logs du cluster qui permet sécurité, conservation des données récentes pour le debugging immédiat et archivage pour ne pas entraver l'espace de stockage avec des données moins récentes.
Contenu de "/etc/logrotate.conf"

```
/var/cluster/logs/*/messages/*.log {
    daily
    size 15G
    rotate 2
    compress
    notifempty
    create 0640 root root
    mail admin@montaigu.univ-tlse3.fr
    postrotate
        systemctl restart rsyslog
    endscript
}
```

1.9 Synchronisation temporelle du cluster

Les serveurs NTP peuvent avoir plusieurs serveurs N-1, ce qui permet d'éviter les SPOFs. La capacité à pouvoir se référer à des nœuds de même niveau si les serveurs N-1 sont indisponibles permet a priori de limiter la désynchronisation du système. Cependant, dans notre cas, si le réseau backbone est indisponible ou que les deux nœuds du réseau backbone de l'université sont hors service, alors aucun des quatre nœuds n'aura de N-1. Il faudrait d'autres sources de référence hors du backbone de l'université pour résister à une coupure de ce réseau (ou bien utiliser les horloges locales moins précises). En s'intéressant au boîtier GPS-NTP, qui permet la synchronisation des serveurs du backbone de l'université, on se rend compte qu'il peut être connecté sur deux réseaux. S'il n'est pas déjà connecté sur un autre réseau, on pourrait vouloir le rendre disponible directement sur le réseau d'administration ou sur un réseau sur lequel les quatre nœuds NTP du cluster doivent être présents.

Le typage pour le boîtier GPS est 4 d'après `https://doc.ntp.org/documentation/4.2.8-series/refclock/` : Type 4 Spectracom WWVB/GPS Receivers (WWVB_SPEC) donc pour le définir comme serveur il faudra utiliser fudge avec 127.127.4.0 ou 127.127.4.1 si le 0 est déjà utilisé.

On suppose cependant ici que l'on n'a pas accès à ce boîtier GPS en dehors du backbone car il se trouve physiquement hors du cluster. Pour éviter une perte totale du service en cas de coupure du réseau Backbone, on va donc utiliser l'horloge interne d'un des nœuds de service en référence en cas d'urgence. On définit volontairement cette source avec une strate basse de 4 qui doit être inférieure à celle des serveurs de l'université qui sont des N+1 du boîtier GPS de strate 1. Ainsi, la strate du serveur avec l'horloge interne doit être supérieure à 3. On se trouve finalement avec une architecture très résistante aux pannes :

- si l'un des deux serveurs NTP de l'université est indisponible, l'autre prend le relais pour les nœuds de services qui ne sont plus desservis
- Si le réseau backbone n'est plus disponible, l'horloge interne du nœud de service 1 prend le relais. Si même le nœud de service 1 est indisponible, alors les 3 autres nœuds de service utilisent leur propre horloge interne.

Voici des fichiers de configurations : contenu du fichier srv 1 (montaigu41) "/etc/ntp.conf" :

```
# For more information about this file, see the man pages
```

```
# ntp.conf(5), ntp_acc(5), ntp_auth(5), ntp_clock(5), ntp_misc(5), ntp_mon(5).

driftfile /var/lib/ntp/drift

# Permit time synchronization with our time source, but do not
# permit the source to query or modify the service on this system.
# Pas de restriction nopeer ou nomodify
restrict default nomodify notrap

# Aucune restriction sur les noeuds master (utilisation du DNS)
restrict admin1.montaigu.univ-tlse3.fr
restrict admin2.montaigu.univ-tlse3.fr

# Hosts on local network are less restricted.
#restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap

# Use university server

#prefer sur le 15 pour les noeuds 1 et 2 et sur le 16 pour les noeuds 3 et 4

server 172.4.24.15 iburst prefer
server 172.4.24.16 iburst

#pour le noeud de service 1 uniquement
fudge 127.127.1.0 stratum 4
server admin41.montaigu.univ-tlse3.fr

#pour les 3 autres noeuds
#peer admin41.montaigu.univ-tlse3.fr

# Enable public key cryptography.
#crypto

includefile /etc/ntp/crypto/pw

# Key file containing the keys and key identifiers used when operating
# with symmetric key cryptography.
keys /etc/ntp/keys

# Disable the monitoring facility to prevent amplification attacks using ntpdc
# monlist command when default restrict does not include the noquery flag. See
# CVE-2013-5211 for more details.
# Note: Monitoring will not be disabled with the limited restriction flag.
disable monitor
```

contenu du fichier srv 3(montaigu43) "/etc/ntp.conf"

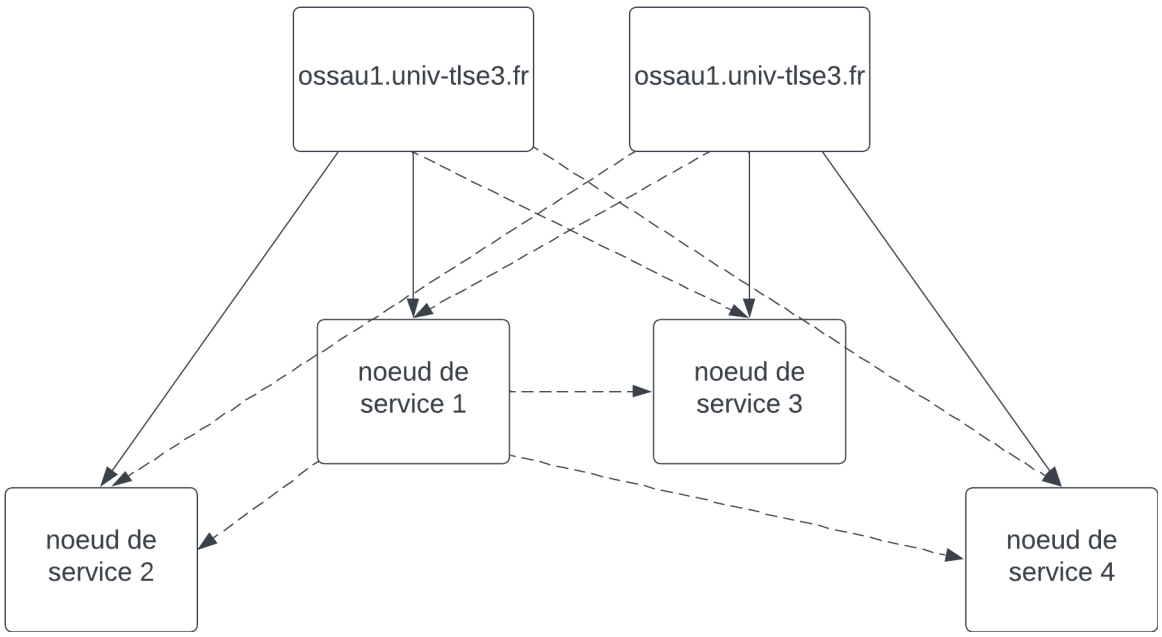
```
# For more information about this file, see the man pages
# ntp.conf(5), ntp_acc(5), ntp_auth(5), ntp_clock(5), ntp_misc(5), ntp_mon(5).

driftfile /var/lib/ntp/drift

# Permit time synchronization with our time source, but do not
# permit the source to query or modify the service on this system.
# Pas de restriction nopeer ou nomodify
restrict default nomodify notrap

# Aucune restriction sur les noeuds master (utilisation du DNS)
restrict admin1.montaigu.univ-tlse3.fr
restrict admin2.montaigu.univ-tlse3.fr
```

```
# Use university server
# prefer sur le 15 pour les noeuds 1 et 2 et sur le 16 pour les noeuds 3 et 4

server 172.4.24.15 iburst
server 172.4.24.16 iburst prefer

peer admin41.montaigu.univ-tlse3.fr
fudge 127.127.1.0 stratum 6
server admin43.montaigu.univ-tlse3.fr

# Enable public key cryptography.
#crypto

includefile /etc/ntp/crypto/pw

# Key file containing the keys and key identifiers used when operating
# with symmetric key cryptography.
keys /etc/ntp/keys
```

contenu du fichier client "/etc/ntp.conf"

```
# For more information about this file, see the man pages
# ntp.conf(5), ntp_acc(5), ntp_auth(5), ntp_clock(5), ntp_misc(5), ntp_mon(5).

driftfile /var/lib/ntp/drift

server admin41.montaigu.univ-tlse3.fr
server admin42.montaigu.univ-tlse3.fr
server admin43.montaigu.univ-tlse3.fr
server admin44.montaigu.univ-tlse3.fr

# Enable public key cryptography.
#crypto

includefile /etc/ntp/crypto/pw

# Key file containing the keys and key identifiers used when operating
# with symmetric key cryptography.
keys /etc/ntp/keys
```

On remarque que l'on a partout enlevé les "disable monitor" pour pouvoir ainsi que des "restrict" pour mieux contrôler le service. On a aussi séparé le serveur préférentiel des nœuds de service. Il n'y a pas besoin de modifier la configuration NTP de l'université puisque nos nœuds de service s'y connectent simplement comme client.



1.10 Architecture LDAP

Architecture

Dans le cadre de l'installation du service LDAP, nous voulons nous reposer sur l'architecture existante, c'est-à-dire utiliser les deux serveurs LDAP nommés montcalm1.univ-tlse3.fr et montcalm2.univ-tlse3.fr. Le service LDAP est un service d'annuaire qui permet de gérer les identités au sein du cluster ou de n'importe quel système informatique centralisé. Celui-ci est très peu gourmand en ressources malgré le nombre important de nœuds que nous avons.
L'objectif est de limiter les SPOF au sein de l'installation. Pour réaliser cela, nous allons intégrer de la redondance au sein de l'installation et installer la configuration sur les 2 nœuds d'administration en cas de défaut de l'un des deux. Pour ce faire, nous allons définir un nœud admin qui sera le maître de tous les autres pour profiter de la synchronisation et avoir celui-ci qui recopie la configuration sur le second nœud maître et 3 nœuds de services installés sur des switches différents pour assurer la disponibilité du cluster. Les nœuds de services seront ainsi les nœuds que le service utilisera vraiment et si on a vraiment besoin de l'un des nœuds maîtres.

Implémentation

Pour l'implémentation de ce service, on doit d'abord installer les paquets nécessaires. Pour les nœuds des partitions, il suffit d'installer **openldap-clients**, et pour les nœuds de services ainsi que les nœuds admin, les paquets à installer sont **openldap-clients openldap-servers**.

Dans un premier temps, la configuration du nœud d'administration, qui sera le nœud maître, c'est-à-dire le référentiel du cluster, se trouve dans le fichier /etc/openldap/slapd.conf. Pour la mise en place de ce service, il ne faut pas oublier de configurer la base de données ainsi que le fichier de log. Il est également important de configurer la rotation des logs LDAP, car ceux-ci peuvent devenir assez volumineux rapidement. fichier /etc/openldap/slapd.conf

```
include /etc/openldap/schema/core.schema
include /etc/openldap/schema/cosine.schema
#inclusion de du schéma nis parce que dans celui-ci il y a les users linux
include /etc/openldap/schema/nis.schema


argsfile /var/run/openldap/slapd.args
pidfile /var/run/openldap/slapd.pid
#setup des logs
loglevel -1
local4.*  /var/log/slapd.log
```

```
# Droits
# comme par exemple des administrateurs
access to *
by dn.children="ou=admin,dc=montaigu,dc=univ-tlse3,dc=fr" write
# Autres droits...
by * read

database bdb
suffix "dc=montaigu,dc=univ-tlse3,dc=fr"
#fichier dans lequel le service sera
directory /var/lib/ldap
rootdn "cn=admin,dc=montaigu,dc=univ-tlse3,dc=fr"
#On oublie pas de changer le mot de passe root avent la mise en prod
rootpw "admin"
database monitor
```

Dans un second temps, la configurations des noeuds esclave est assez similaire,seul l'ajout des lignes pour la synchronisation est nécessaire.

```
moduleload syncprov.la
include /etc/openldap/schema/core.schema
include /etc/openldap/schema/cosine.schema
#inclusion de du schéma nis parce que dans celui-ci il y a les users linux
include /etc/openldap/schema/nis.schema


argsfile /var/run/openldap/slapd.args
pidfile /var/run/openldap/slapd.pid
#setup des logs
loglevel -1
local4.*  /var/log/slapd.log
# Droits
access to *
by dn.children="ou=admin,dc=montaigu,dc=univ-tlse3,dc=fr" write
# droits autres...
by * read

database bdb
overlay syncprov
suffix "dc=montaigu,dc=univ-tlse3,dc=fr"
#fichier dans lequel le service sera
directory /var/lib/ldap
rootdn "cn=admin,dc=montaigu,dc=univ-tlse3,dc=fr"
#On oublie pas de changer le mot de passe root avent la mise en prod
rootpw "admin"
database monitor
```

Pour les noeuds et les clients des machines de l'université,afin d'utiliser le cluster il faut configurer les fichier **/etc/openldap/ldap.conf** et intégrer les urls des serveurs des partitions de service.

```
#
# LDAP Defaults
#
# See ldap.conf(5) for details
# This file should be world readable but not world writable.
BASE dc=montaigu,dc=univ-tlse3,dc=fr
URI ldap://montaigu20.admin.univ-tlse3.fr
URI ldap://montaigu21.univ-tlse3.fr
URI ldap://montaigu.service.montaigu.univ-tlse3.fr # Esclave 1 Zone Service
URI ldap://knuth26.service.knuth.montaigu.univ-tlse3.fr # Esclave 2 Zone Service
# Serveurs LDAP exterieurs de l'universite .univ-tlse3.fr
```

```
BASE dc=u-bordeaux,dc=fr
URI ldap://montcalm1.univ-tlse3.fr
URI ldap://montcalm2.univ-tlse3.fr
#SIZELIMIT 12
#TIMELIMIT 15
#DEREF never
TLS_CACERTDIR /etc/openldap/certs
# Turning this off breaks GSSAPI used with krb5 when rdns = false
SASL_NOCANON on
```

Sur le LDAP de l'université,il faut créer de nouveau users et de nouveau posixgroup pour les communautés d'utilisateurs et ainsi on pourrait imaginer une restriction des utilisateurs grâce au ldap.

1.11 Architecture DNS

Le DNS (Domain Name System) au sein d'un réseau local est une composante essentielle de l'infrastructure informatique qui permet de résoudre les noms de domaine en adresses IP au sein de ce réseau restreint. Celui-ci permet de rediriger les connexions et d'orchestrer l'ensemble du réseau. Le service DNS, dans notre cas, est très important car tous les acteurs vont l'utiliser, peu importe leur place dans l'architecture de l'université. Nous voulons donc avoir un DNS maître qui gérera les DNS pour chaque partition. De plus, pour ajouter de la redondance, nous pouvons configurer l'autre nœud admin en tant que DNS esclave. Chaque serveur de partition a besoin d'avoir un nœud de service car le service DNS est très gourmand.

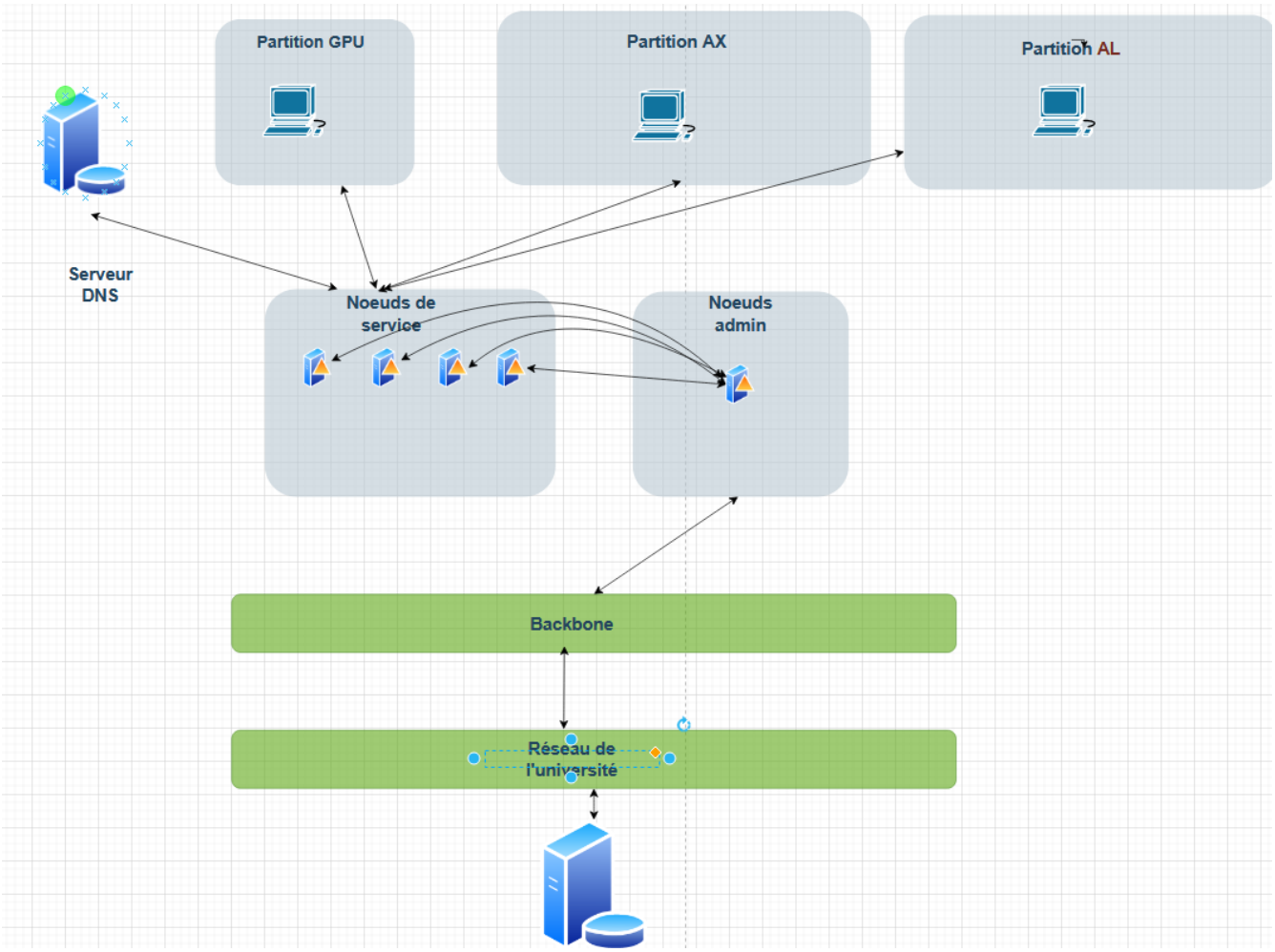


FIGURE 1 – Architecture DNS

Nous devons définir des zones et pour cela, nous allons établir des listes de contrôle d'accès (ACL). Pour chaque partition, nous appellerons l'ACL "<nom-partition>.montaigu.univ-tlse3". Nous allons également

déterminer des sous-domaines pour chaque partie des nœuds autres.
Pour que tout cela réussisse à communiquer, on doit définir le named.conf avec les noeuds admin et les noeuds esclave. La définition des zones avec les noms dans le dossier /var/named/ va permettre de définir toutes les zones du DNS. On définit aussi. Côté client, il faut compléter le fichier etc/resolv.conf sur les noeuds du cluster et sur les noeuds de l'université pour qu'ils puissent communiquer à travers ce service. Pour éviter la multiplication des fichiers de configuration, on donnera juste un exemple de chacun d'eux que l'on adaptera en fonction du noeud.
fichier /etc/named.conf du noeud admin

```
    //
// named.conf
//
// Provided by Red Hat bind package to configure the ISC BIND named(8) DNS
// server as a caching only nameserver (as a localhost DNS resolver only).
//
// See /usr/share/doc/bind*/sample/ for example named configuration files.
//
// See the BIND Administrator's Reference Manual (ARM) for details about the
// configuration located in /usr/share/doc/bind-{version}/Bv9ARM.html

acl     universite      {172.43.0.0/16;
                         172.44.0.0/16;};
acl     PME             {172.45.0.0/16


acl     login           {10.1.0.0/10;};
acl     io              {10.2.0.0/15;};
acl     service         {10.3.0.0/30;};
acl     front           {10.4.0.0/20;};
acl     AC              {10.5.0.0/400;};
acl     AL              {10.6.0.0/100;};
acl     gpu             {10.7.0.0/100;};
acl     nfs             {10.8.0.1;};
acl     serverdns       {172.4.24.13;
                         172.4.24.14;};
acl     esclave         {10.3.0.1;
                         10.3.0.2;
                         10.3.0.3;
                         10.3.0.4}:
acl     admin           {10.0.0.0/16;};
acl     master          {10.0.0.1;};
masters master          {10.0.0.2;};


options {
        listen-on port 53 { 127.0.0.1; master };
        listen-on-v6 port 53 { ::1; };
        directory       "/var/named";
        dump-file       "/var/named/data/cache_dump.db";
        statistics-file "/var/named/data/named_stats.txt";
        memstatistics-file "/var/named/data/named_mem_stats.txt";
        recursing-file  "/var/named/data/named.recursing";
        secroots-file   "/var/named/data/named.secroots";
        allow-query     { localhost; AC; AL; login; io; service; front; gpu;};
        allow-transfer  { esclave; };
        also-notify     { esclave; };
        forward only;
        forwarder { universite;} ;
```

```
        /*
         - If you are building an AUTHORITATIVE DNS server, do NOT enable recursion.
         - If you are building a RECURSIVE (caching) DNS server, you need to enable
           recursion.
         - If your recursive DNS server has a public IP address, you MUST enable access
           control to limit queries to your legitimate users. Failing to do so will
           cause your server to become part of large scale DNS amplification
           attacks. Implementing BCP38 within your network would greatly
           reduce such attack surface
        */
        recursion yes;

        dnssec-enable yes;
        dnssec-validation yes;

        /* Path to ISC DLV key */
        bindkeys-file "/etc/named.root.key";

        managed-keys-directory "/var/named/dynamic";

        pid-file "/run/named/named.pid";
        session-keyfile "/run/named/session.key";
};

logging {
        channel my_file {
                file "/var/log/named.log";
                severity dynamic;
                print-category yes;
                print-time yes;
        };
        category default { my_file; };
        category queries { my_file; };
        category config  { my_file; };
        category network { my_file; };
        category xfer-out { my_file; };
        category xfer-in { my_file; };
};


zone "login.montaigu.univ-tlse3.fr" IN {
        type master;
        file "/var/named/named.login.montaigu.univ-tlse3";
};

zone "0-15.0.1.10" IN {
        type master;
        file "/var/named/named.login.montaigu.univ-tlse3_rev";
};
zone "io.montaigu.univ-tlse3.fr" IN {
        type master;
        file "/var/named/named.io.montaigu.univ-tlse3";
};

zone "0-15.0.2.10" IN {
        type master;
        file "/var/named/named.io.montaigu.univ-tlse3_rev";
};
zone "service.montaigu.univ-tlse3.fr" IN {
```

```
        type master;
        file "/var/named/named.service.montaigu.univ-tlse3";
};

zone "0-30.0.3.10" IN {
        type master;
        file "/var/named/named.service.montaigu.univtlse3_rev";
};
zone "front.montaigu.univ-tlse3.fr" IN {
        type master;
        file "/var/named/named.front.montaigu.univ-tlse3";
};

zone "0-19.0.4.10" IN {
        type master;
        file "/var/named/named.front.montaigu.univ-tlse3_rev";
};
zone "nfs.montaigu.univ-tlse3.fr" IN {
        type master;
        file "/var/named/named.nfs.montaigu.univ-tlse3";
};

zone "1.0.8.10" IN {
        type master;
        file "/var/named/named.nfs.montaigu.univ-tlse3_rev";
};
zone "AC.montaigu.univ-tlse3.fr" IN {
        type master;
        file "/var/named/named.AC.montaigu.univ-tlse3";
};

zone "0-400.0.5.10" IN {
        type master;
        file "/var/named/named.AC.montaigu.univ-tlse3_rev";
};
zone "AL.montaigu.univ-tlse3.fr" IN {
        type master;
        file "/var/named/named.AL.montaigu.univ-tlse3";
};

zone "0-99.0.6.10" IN {
        type master;
        file "/var/named/named.AL.montaigu.univ-tlse3_rev";
};
zone "gpu.montaigu.univ-tlse3.fr" IN {
        type master;
        file "/var/named/named.gpu.montaigu.univ-tlse3";
};

zone "0-100.0.7.10" IN {
        type master;
        file "/var/named/named.gpu.montaigu.univ-tlse3_rev";
};
include "/etc/named.rfc1912.zones";
include "/etc/named.root.key";
```

fichier /var/named/named.login.montaigu.univ-tlse3

```
    $TTL 1D
@       IN      SOA montaigu11.login.montaigu.univ-tlse3. root.login.montaigu.univ-tlse3. (
        2023020300      ; serial
```

```
        3H              ; refresh every 3 hours
        1H              ; retry every hour
        3W              ; expire after 3 weeks
        1D )            ; minimum TTL of 1 day
        IN      NS      montaigu11.login.montaigu.univ-tlse3.

$ORIGIN login.montaigu.univ-tlse3.

montaigu11     IN      A       10.1.0.1
montaigu12     IN      A       10.1.0.2
...
montaigu25    IN      A       10.1.0.15
ns      IN      CNAME   montaigu11.login.montaigu.univ-tlse3.
```

fichier /var/named/named.login.montaigu.univ-tlse3_rev

```
    $TTL 1D
@       IN              SOA             montaigu11.login.montaigu.univ-tlse3.       root.login.montai
        2023020300      ; serial
        3H              ; refresh every 3 hours
        1H              ; retry every hour
        3W              ; expire after 3 weeks
        1D )            ; minimum TTL of 1 day
        IN      NS     montaigu11.login.montaigu.univ-tlse3.

$ORIGIN 0-15.0.1.10.in-addr.arpa.

1.0       IN      PTR     montaigu11.login.montaigu.univ-tlse3.
...
15.0       IN      PTR     montaigu25.login.montaigu.univ-tlse3.
```

fichier /etc/resolv.conf

```
    search knuth.u-bordeaux.fr.
    nameserver 10.3.3.1 # DNS Maitre
    nameserver 10.4.4.1 # Premier DNS Esclave de la partition S
    nameserver 10.4.4.2 # Deuxieme DNS Esclave de la partition S
    search u-bordeaux.fr.
    nameserver 172.23.0.1 # carnet1
    nameserver 172.23.0.2 # carnet2
```

On réalise la même chose avec les autres noms définit dans le fichier conf du master.

1.12 Configuration du contrôleur SLURM

La base de données est configurée dans "/etc/slurm/slurmdbd.conf" mais nous ne l'aborderons pas ici du fait du nom de la partie. Elle doit toutefois utiliser l'authentification munge pour assurer la sécurité des communications. Les logs iront dans /var/log/slurm/slurmdbd.log du premier nœud maître. On utilisera l'un des nœuds maîtres en tant que nœud de contrôle et la base de données associée à Slurm sera stockée sur les disques associés aux contrôleurs RAID. Le deuxième nœud maître servira de sauvegarde au premier pour éviter les SPOFs. On utilisera une architecture d'arbre et on indiquera à Slurm les switchs.

contenu du fichier "/etc/slurm/topology.conf"

```
# Slurm's network topology configuration file for use with the
# topology/tree plugin
SwitchName=chassisAC0 Nodes=montaigu[100-129]
SwitchName=chassisAC1 Nodes=montaigu[130-159]
SwitchName=chassisAC2 Nodes=montaigu[160-189]
SwitchName=chassisAC3 Nodes=montaigu[190-219]
SwitchName=chassisAC4 Nodes=montaigu[220-249]
SwitchName=chassisAC5 Nodes=montaigu[250-279]
SwitchName=chassisAC6 Nodes=montaigu[280-309]
```

```
SwitchName=chassisAC7 Nodes=montaigu[310-339]
SwitchName=chassisAC8 Nodes=montaigu[340-369]
SwitchName=chassisAC9 Nodes=montaigu[370-399]
SwitchName=chassisAC10 Nodes=montaigu[400-401]
SwitchName=chassisAL0 Nodes=montaigu[600-629]
SwitchName=chassisAL1 Nodes=montaigu[630-649]

SwitchName=ewa1 Nodes=montaigu[700-708] Switchs=chassisAC[0-10],chassisAL[0,1]
SwitchName=ewa2 Nodes=montaigu[709-717] Switchs=chassisAC[0-10],chassisAL[0,1]
SwitchName=ewa3 Nodes=montaigu[718-725] Switchs=chassisAC[0-10],chassisAL[0,1]
SwitchName=ewa4 Nodes=montaigu[726-733] Switchs=chassisAC[0-10],chassisAL[0,1]
SwitchName=ewa5 Nodes=montaigu[734-741] Switchs=chassisAC[0-10],chassisAL[0,1]
SwitchName=ewa6 Nodes=montaigu[742-749] Switchs=chassisAC[0-10],chassisAL[0,1]
```

On a ici la topologie d'arbre désirée qui comprend l'ensemble des nœuds de nos 3 partitions de calcul. Cependant, ce n'est pas tout à fait un arbre en réalité puisqu'il n'y a pas de top switch et que les switchs ewa[1-6] sont en réalité connectés entre eux. Concernant la configuration générale, on utilisera bien sûr des partitions pour répartir les différents types de nœuds de calcul. L'architecture est très résiliente grâce au découpage des nœuds de Ax sur tous les switchs et grâce au fait que les switchs des châssis soient tous reliés à quatre switchs ewa. Pour définir les GPUs comme tel, on doit faire appel au Slurm Generic Ressources (GRES) et définir un nouveau fichier de configuration "gres.conf".

```
Name=gpu Type=a100 File=/dev/nvidia0 Flags=nvidia_gpu_env
```

Le flag nvidia_gpu_env sert à rendre le processeur visible pour CUDA. On doit définir également un nouveau paramètre AccountingStorageTRES dans slurm.conf. Le fichier de configuration slurm.conf a été réalisé avec l'aide de `https://slurm.schedmd.com/configurator.html`. On a choisi la sélection des noeuds complets pour éviter que des coeurs d'un même noeuds soient attribués à différents jobs. Le backfiling est plus efficace que le fifo en terme de politique d'ordonnancement. On donnera aussi une forte importance à la quality of service et une importance plus faible à la taille du job et au fairshare pour la priorité des jobs.

contenu du fichier montaigu0 "/etc/slurm/slurm.conf"

```
# slurm.conf file generated by configurator.html.
# Put this file on all nodes of your cluster.
# See the slurm.conf man page for more information.
#
ClusterName=montaigu
SlurmctldHost=montaigu0.admin.montaigu.univ-tlse3.fr
SlurmctldHost=monaigu1.admin.montaigu.univ-tlse3.fr
AuthType=auth/munge
ControlMachine=montaigu0admin.montaigu.univ-tlse3.fr
SlurmUser=admin
#
#DisableRootJobs=NO
#EnforcePartLimits=NO
#Epilog=
#EpilogSlurmctld=
#FirstJobId=1
#MaxJobId=67043328
#GresTypes=
#GroupUpdateForce=0
#GroupUpdateTime=600
#JobFileAppend=0
#JobRequeue=1
#JobSubmitPlugins=lua
#KillOnBadExit=0
#LaunchType=launch/slurm
#Licenses=foo*4,bar
#MailProg=/bin/mail
#MaxJobCount=10000
```

```
#MaxStepCount=40000
#MaxTasksPerNode=512
#MpiDefaultmpi/pmix=
#MpiParams=ports=#-#
#PluginDir=
#PlugStackConfig=
#PrivateData=jobs
ProctrackType=proctrack/cgroup
#Prolog=
#PrologFlags=
#PrologSlurmctld=
#PropagatePrioProcess=0
#PropagateResourceLimits=
#PropagateResourceLimitsExcept=
#RebootProgram=
ReturnToService=0
SlurmctldPidFile=/var/run/slurmctld.pid
SlurmctldPort=6817
SlurmdPidFile=/var/run/slurmd.pid
SlurmdPort=6818
SlurmdSpoolDir=/var/spool/slurmd
SlurmUser=test_user
#SlurmdUser=root
#SrunEpilog=
#SrunProlog=
StateSaveLocation=/var/spool/slurmctld
#SwitchType=
#TaskEpilog=
TaskPlugin=task/affinity,task/cgroup
#TaskProlog=
#TopologyPlugin=topology/tree
#TmpFS=/tmp
#TrackWCKey=no
#TreeWidth=
#UnkillableStepProgram=
#UsePAM=0
#
#
# TIMERS
#BatchStartTimeout=10
#CompleteWait=0
#EpilogMsgTime=2000
#GetEnvTimeout=2
#HealthCheckInterval=0
#HealthCheckProgram=
InactiveLimit=0
KillWait=30
#MessageTimeout=10
#ResvOverRun=0
MinJobAge=300
#OverTimeLimit=0
SlurmctldTimeout=120
SlurmdTimeout=300
#UnkillableStepTimeout=60
#VSizeFactor=0
Waittime=0
#
#
# SCHEDULING
```

```
#DefMemPerCPU=0
#MaxMemPerCPU=0
#SchedulerTimeSlice=30
SchedulerType=sched/backfill
SelectType=select/linear
#
#
# JOB PRIORITY
#PriorityFlags=
PriorityType=priority/multifactor
#PriorityDecayHalfLife=
#PriorityCalcPeriod=
#PriorityFavorSmall=
#PriorityMaxAge=
#PriorityUsageResetPeriod=
#PriorityWeightAge=
PriorityWeightFairshare=1000
PriorityWeightJobSize=1000
#PriorityWeightPartition=
PriorityWeightQOS=10000
#
#
# LOGGING AND ACCOUNTING
AccountingStorageEnforce=Warn
AccountingStorageHost=montaigu0
#AccountingStoragePass=
#AccountingStoragePort=
AccountingStorageType=accounting_storage/slurmdbd
#AccountingStorageUser=
AccountingStorageTRES=gres/gpu,gres/gpu:a100
#AccountingStoreFlags=
#JobCompHost=
#JobCompLoc=
#JobCompParams=
#JobCompPass=
#JobCompPort=
JobCompType=jobcomp/none
#JobCompUser=
#JobContainerType=
JobAcctGatherFrequency=30
#JobAcctGatherTypejobacct_gather/linux=
SlurmctldDebug=info
SlurmctldLogFile=/var/log/slurmctld.log
SlurmdDebug=info
SlurmdLogFile=/var/log/slurmd.log
#SlurmSchedLogFile=
#SlurmSchedLogLevel=
#DebugFlags=
#
#
# COMPUTE NODES
NodeName=montaigu[600-649] RealMemory=384 Sockets=2 CoresPerSocket=42 ThreadsPerCore=2 State=UNKNOWN
PartitionName=AL Nodes=montaigu[600-649] Default=NO MaxTime=INFINITE State=UP
NodeName=montaigu[700-749] RealMemory=64 Sockets=2
NodeName=montaigu[100-401] RealMemory=256 Sockets=2 CoresPerSocket=64 ThreadsPerCore=2 State=UNKNOWN
PartitionName=AC Nodes=montaigu[100-401] Default=YES MaxTime=INFINITE State=UP
GresTypes=gpu:a100
NodeName=montaigu[700-749] Gres=gpu:a100 RealMemory=64 Socket=2 CoresPerSocket=12 ThreadsPerCore=2 St
PartitionName=AX Nodes=montaigu[700-749] Default=NO MaxTime=INFINITE State=UP
```

On mit l'AccountingStorageEnforce à Warn pour ne pas ignorer les problèmes mais sans faire disfonctionné totalement le service. La politique de priorité des jobs. L'accounting se fait dans la base de données des disques durs avec contrôleurs RAID des noeuds maîtres. On utilise munge pour sécuriser les communications. Pour contraindre l'utilisation des services et gérer les groupes d'utilisateurs, on utilise des commandes sacctmgr

```
# Création du cluster
sacctmgr create -i cluster=montaigu

#Création des qos
sacctmgr -i add qos Name=normal Priority=10 MaxJobs=15 MaxWall=04:00:00
sacctmgr -i add qos Name=debug Priority=100 MaxJobs=5 MaxWall=00:30:00

#upg[0-2] : projet0 , 2 noeuds de login, 40 personnes
sacctmgr create -i account name=projet0 fairshare=15 cluster=montaigu qos=normal

sacctmgr create -i account name=upg0projet0 parent=projet0
sacctmgr create -i account name=upg1projet0 parent=projet0
sacctmgr create -i account name=upg2projet0 parent=projet0
#upg[3-4] : projet1 , 3 noeuds de login, 80 personnes
sacctmgr create -i account name=projet1 fairshare=20 cluster=montaigu qos=normal,debug

sacctmgr create -i account name=upg3projet1 parent=projet1
sacctmgr create -i account name=upg4projet1 parent=projet1

#upg[5-7] : projet2 , 2 noeuds de login, 30 personnes
sacctmgr create -i account name=projet2 fairshare=40 cluster=montaigu qos=normal,debug

sacctmgr create -i account name=upg5projet2 parent=projet2
sacctmgr create -i account name=upg6projet2 parent=projet2
sacctmgr create -i account name=upg7projet2 parent=projet2
#upg[8-9] : projet3 , 3 noeuds de login, 70 personnes
sacctmgr create account name=projet3 fairshare=25 cluster=montaigu qos=normal

sacctmgr create -i account name=upg8projet3 parent=projet3
sacctmgr create -i account name=upg9projet3 parent=projet3

# Retirer la partition AX aux membre du projet0
sacctmgr modify account name=projet0 set partition=ALL,~AC

sacctmgr create -i user name=user0ugp0projet0 cluster=montaigu account=projet0
sacctmgr create -i user name=user0ugp3projet1 cluster=montaigu account=projet1
sacctmgr create -i user name=user0ugp6projet2 cluster=montaigu account=projet2
sacctmgr create -i user name=user0ugp8projet3 cluster=montaigu account=projet3
```

On a utilisé les qos pour définir le nombre de noeuds est le temps maximal des utilisateurs. Pour vérifier la politique mise en place on peut utiliser :

```
sacctmgr show qos
sacctmgr show account
scontrol show topo
sinfo
```

1.13 Configuration de la surveillance

Pour vérifier le premier point on utilise la sonde verif-nodeset-ssh pour s'assurer que les utilisateurs peuvent bien se connecter en ssh sur les noeuds de login. Pour le deuxième point, on utilise la sonde verif-lustre-routeur directement sur les noeuds I/O et pour les 3 derniers points on se sert de la dernière sonde : verif-noeud. Un noeud routeur assure un débit maximal de 24 Go/s donc deux sont suffisants pour 32 Go/s. Les 10 noeuds routeurs sont numérotés de 26 à 35.

— Faire 4 appels à verif-nodeset-ssh avec les noeuds corresponds à chaque communauté successivement (les noeuds à passer en argument vont de montaigu26.front.montaigu.univ-tlse3.fr à montaigu35.front.montaigu.univ-tlse3.fr en les regroupant par communauté). verif-nodeset-ssh -w 50 -c 40 -H montaigu26.front.montaigu.univ-tlse3.fr montaigu27.front.montaigu.univ-tlse3.fr
verif-nodeset-ssh -w 60 -c 30 -H montaigu28.front.montaigu.univ-tlse3.fr montaigu29.front.montaigu.univ-tlse3.fr montaigu30.front.montaigu.univ-tlse3.fr
verif-nodeset-ssh -w 50 -c 40 -H montaigu31.front.montaigu.univ-tlse3.fr montaigu32.front.montaigu.univ-tlse3.fr
verif-nodeset-ssh -w 60 -c 30 -H montaigu33.front.montaigu.univ-tlse3.fr montaigu34.front.montaigu.univ-tlse3.fr montaigu35.front.montaigu.univ-tlse3.fr
Si un des appels est à Critical alors le test échoue.
— Passer la liste de tous les noeuds routeur en argument (sous la forme montaigu26.io.montaigu.univ-tlse3.fr)à verif-lustre-routeur.
verif-lustre-routeur -w 40 -c 20 -H montaigu[26-35].io.montaigu.univ-tlse3.fr
— Passer la liste de tous les noeuds de la partition AC en argument (sous la forme montaigu100.AC.montaigu.univ-tlse3.fr)à verif-noeud.
verif-noeud -w 95 -c 90 -H montaigu[100-402].AC.montaigu.univ-tlse3.fr
— Passer la liste de tous les noeuds de la partition AL en argument (sous la forme montaigu600.AL.montaigu.univ-tlse3.fr)à verif-noeud.
verif-noeud -w 85 -c 80 -H montaigu[600-649].AL.montaigu.univ-tlse3.fr
— Passer la liste de tous les noeuds de la partition AX en argument (sous la forme montaigu700.gpu.montaigu.univ-tlse3.fr) à verif-noeud.
verif-noeud -w 70 -c 60 -H montaigu[600-649].gpu.montaigu.univ-tlse3.fr

Pour que les utilisateurs des nœuds de la partition frontale aient une bonne expérience, il faut surveiller que les nœuds frontaux sont bien accessibles pour eux, que les nœuds de calcul sont accessibles depuis leur nœud de login et aussi que leurs nœuds ne sont pas surchargés par d'autres utilisateurs. Les services critiques à surveiller sont SLURM et le service NFS. Pour assurer une surveillance plus fine, il ne faudrait pas seulement surveiller les nœuds mais aussi les switchs. WebUI permettra également une lecture plus claire des problèmes et des pannes. Enfin, surveiller les nœuds de service et maîtres est également critique.

1.14 Configuration Puppet

Le choix est de mettre Puppet sur les nœuds maîtres et de pousser les changements sur tous les autres nœuds, car le nœud maître peut utiliser tous les nœuds. Pour utiliser Puppet, il faut installer et configurer le paquet **puppet-master** sur le nœud maître et **puppet-agent** sur tous les autres nœuds. Ceux-ci seront installés après la mise en place des services réseau car Puppet utilise le protocole DNS pour communiquer. Ensuite, il suffit de générer le certificat sur chaque nœud et de le signer sur le nœud maître.

Group/Role	Profiles	Nodeset	Commentaires
Noeud admin	— Serveur PXE — Connexion Backbone — Serveur DNS maître — Serveur DHCP — Serveur LDAP — Noeud connecté IB — Serveur NTP	montaigue[0-1]	
Noeuds de login	— Noeud connecté IB — Connexion Backbone	montaigu[11-21]	
Noeuds IO	— Noeud connecté IB — Connexion Backbone — Serveur NFS	montaigu[26-34]	
Noeuds service	— Serveur PXE — Connexion Backbone — Serveur DNS esclave — Serveur DHCP esclaves — Serveur LDAP esclave — DB — Serveur Rsyslog — Noeud connecté IB	montaigu[41-50]	
Noeuds AC-AL	— Noeud AX — Connexion Backbone	montaigu[100-453]	
Noeuds GPU	— Noeud AX — Connexion IB	montaigu[700-760]	

TABLE 7

Voici quelques dépôts distants qui contiennent certains de ces modules :
— Pour les services NTP, il y a https://forge.puppet.com/modules/puppetlabs/ntp/, qui contient l'essentiel pour configurer le client, etc.
— Pour les services LDAP, il y a https://forge.puppet.com/modules/puppet/openldap.
— Pour les services DHCP, vous pouvez utiliser le module Puppet DHCP disponible à l'adresse suivante : https://forge.puppet.com/modules/puppet/dhcp.
— Pour les services NFS, il y a https://forge.puppet.com/modules/simp/nfs/.
— Pour les services RSYSLOG, il y a https://forge.puppet.com/modules/puppet/rsyslog.

Il n'y a pas de module à développer car, a priori, les services requis pour le cluster sont assez standards. Mais on peut remarquer que le module Shinken est déprécié, donc il faudra le coder pour qu'il prenne en compte les dernières versions.

1.15 Environnement Utilisateur

Toolchains pour l'environnement Intel ICTCE : Il s'agit d'une toolchain Intel Compiler Toolset for C/C++/Fortran. Elle est spécifiquement conçue pour utiliser les compilateurs Intel (ICC, IFORT) et les bibliothèques Intel (MKL, IPP, etc.). Cet environnement est optimisé pour les processeurs Intel et fournit des performances optimales lors de l'exécution sur du matériel Intel.

Toolchain pour l'environnement GNU GOMPI

Toolchain pour l'environnement MATLAB=IOMKL : Cette option contient les bibliothèques Intel Math Kernel Library (MKL), ce qui permet des calculs plus rapides. Comme le calculateur est utilisé pour des calculs scientifiques, cela semble raisonnable.

Toolchain pour les GPU IOMKLC : Elle prend en charge CUDA et GCC, et de plus ajoute la bibliothèque Intel précédente pour améliorer les opérations sur les nombres. On ne sait jamais.

1.16 Déploiement

MOFED -2310213 LTS =Mellanox OpenFabrics Enterprise Distribution 23.10-2.1.3.0 Long-Term Support cela veut dire que la version du logiciel sera supportée donc mis à jour plusieurs années(5 ans par exemple pour les versions d'ubuntu) et sera donc plus stable et permet de limiter les conflits que l'on pourrait avoir avec d'autres logiciels.

RedHat	Lustre	MOFED	Firmware ConnectX-6
— RHEL CentOS8.4 — RHEL CentOS8.5 — RHEL Rocky8.5 — RHEL Rocky8.6 — RHEL Rocky8.7	— Lustre 2.15.2 — Lustre 2.12.9	MLNX_OFED 23.10-2.1.3.1 LTS	20.39.3004

Table 8

Nous allons donc installer la version 2.15.2 de lustre car elle coche toutes les cases malgré que l'OS ne coche pas toutes les case pour le support de fonctionnalité, celui-ci est le meilleur choix pour la compatibilité avec lustre 2.15.2.

1.16.1 Protocol d'installation

Pour installer un OS sur un serveur diskless, il faut dans un premier temps faire la configuration tftp sur les noeuds clients en installant le paquet tftp-server. Grâce au service DHCP au sein de l'université et l'accès à un serveur NFS,on peut commencer l'installation. Pour cela il suffit de suivre la procédure sur

en l'adaptant à notre problème. Pour installer et configurer le service tftp, il faut que le serveur PXE soit configurer. Il faut d'abord réaliser l'installation sur les noeuds admin puis mettre l'OS dont on a besoin car ils ont du stockage. Pour l'installation sur les noeuds clients, il suffit de :

- configurer le service tftp en réalisant dans un premier temps la configuration pxe
- configurer le service DHCP pour l'obtention d'adresse IP
- configurer un système de fichier.
- autoriser le bootp
- Récupérer les fichiers de config vmlinuz-kernel,initramfs et les fichiers d'OS
- Ensuite on peut booter

1.17 Planification de l'installation du cluster

On commence la planification à la semaine 20 car le 01 mai est férié donc les employés ne vont pas livrer et le 08 mai aussi. Le mercredi 15 mai livraison du matérielle et vérification de celui-ci ainsi que déballage des cartons.Commencement de l'installation si on a un peu de temps. Le jeudi 16 mai au matin on procède installation d'un rack puis l'autre l'après-midi. On répète ce cycle toutes les semaines en utilisant les jours tels que lundi et mardi pour la configuration et les tests.
Dans un premier temps, nous allons demander la livraison d'un rack de nœuds d'administration et d'un rack pour les switches. Nous commencerons par la configuration des nœuds d'administration pour l'installation des logiciels et des nœuds NFS, car ces derniers pourront être utiles pour stocker des informations. De plus, nous aurons besoin du rack de switches pour établir la connectivité du backbone. Après avoir configuré les nœuds, nous pourrons alors lancer l'installation des autres nœuds à distance, en plus de configurer les dépôts distants pour les connecter aux infrastructures de l'université.

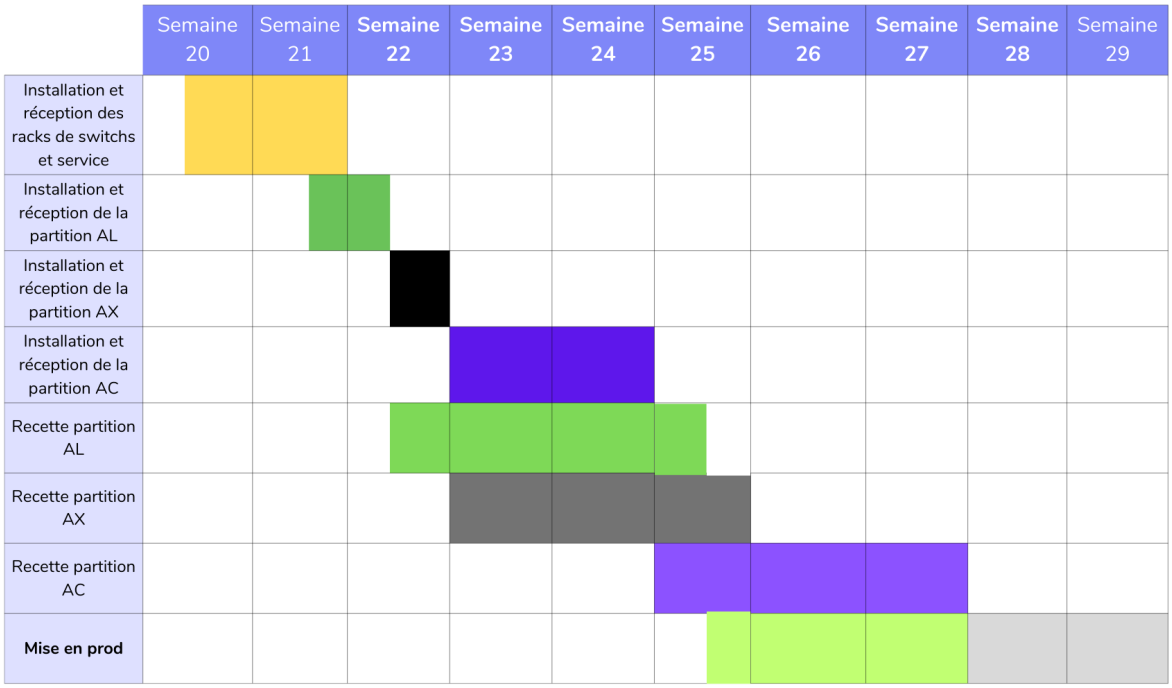


FIGURE 2 – Gantt

La fin de l'installation est programmé pour la semaine 33 si il n'y a aucun incident grave.