REPUBLIQUE DU SENEGAL

Un peuple-un but-une foi

Ministère de l'Enseignement Supérieur, de la Recherche et de l'Innovation

Direction de l'Enseignement Supérieur Privé

Institut Supérieur d'Informatique

ISI

Département : Réseau & Système

Spécialité : Réseaux Informatiques

GESTION DE PROJET

MISE EN PLACE DE LA HAUTE DISPONIIBLITE ET DE LA REPLICATION

Présenté et soutenu par :

1. **M. Eugenio EPAM MONTERO**

Email : eugenioepam11@gmail.com

Sous la direction de

M. Massamba LO

Année Académique : 2024 -2025

SOMMAIRE :

- Introduction
- Haute disponibilité
  - ✓ Préparation du site web a rendre hautement disponible
  - ✓ Configuration des interfaces
  - ✓ Configuration des fichiers hosts
  - ✓ Installation de Heartbeat
  - ✓ Préparation des fichiers de configuration
  - ✓ Configuration du fichier **authkeys**
  - ✓ Configuration du fichier ha.cf
  - ✓ Configuration du fichier haresources
  - ✓ Test de la réplication
- Réplication
  - ✓ Preparation des disques
  - ✓ Installation de DRBD
  - ✓ Modification des fichiers de configuration
  - ✓ Parametrage de du site web
  - ✓ Parametrage de Heartbeat

INTRODUCTION

Dans l'environnement informatique actuel, où la continuité des services et l'intégrité des données sont primordiales, la mise en place de solutions de haute disponibilité et de réplication s'avère cruciale. Ce rapport explore en détail les concepts et les techniques permettant d'assurer une disponibilité maximale des systèmes et une protection efficace contre la perte de données. Nous examinerons les différentes approches de la haute disponibilité, incluant le clustering et le basculement, ainsi que les mécanismes de réplication, qu'ils soient synchrones ou asynchrones. L'objectif de cette étude est de fournir une compréhension approfondie des enjeux et des meilleures pratiques pour la conception et l'implémentation de solutions robustes et fiables. Nous aborderons les aspects théoriques essentiels avant de nous pencher sur des exemples concrets et des scénarios d'application

Haute Disponibilité (Heartbeat)

Heartbeat est un logiciel open-source qui fournit des capacités d'infrastructure de cluster, notamment la **gestion des membres du cluster** et la **messagerie**, à des serveurs clients. C'est un élément crucial dans une infrastructure de serveur à haute disponibilité.

Son fonctionnement repose sur un principe simple :

- **Signal de battement de cœur ("Heartbeat") :** Les nœuds (serveurs) d'un cluster s'envoient régulièrement des signaux ("Heartbeat") via une connexion réseau dédiée (souvent appelée "réseau Heartbeat"). Ces signaux indiquent que le nœud est opérationnel et "en vie".
- **Détection de panne :** Si un nœud ne reçoit plus les Heartbeat d'un autre nœud pendant une période définie, il suppose que ce dernier a échoué ou est inaccessible.
- **Failover (basculement) :** En cas de détection de panne, Heartbeat (souvent en conjonction avec un gestionnaire de ressources de cluster comme Pacemaker) déclenche un processus de **failover**. Cela signifie que les services ou les ressources qui étaient exécutés sur le nœud défaillant sont automatiquement transférés et redémarrés sur un nœud sain du cluster.
- **Gestion des ressources :** Heartbeat peut gérer une variété de ressources, y compris les adresses IP flottantes (qui se déplacent entre les nœuds), les systèmes de fichiers, et les applications.

Heartbeat est principalement utilisé pour créer des clusters en **mode actif/passif**, où un serveur est actif et gère les services, tandis qu'un ou plusieurs autres serveurs sont en veille, prêts à prendre le relais en cas de défaillance du serveur actif.

MISE EN PLACE

Le déploiement sera effectué en utilisant deux serveur Debian 12, mais n'hésitez pas, si vous voulez bien-sûr, utiliser d'autres distributions (cela est tout à fait possible), vous pouvez même travailler avec deux serveurs différents, ça pose aucun problème, le processus reste le même, ce n'est pas obligatoire de travailler avec deux serveurs du même type.

NB : Toutes les étapes sont à effectuer dans les deux serveurs.

❖ Préparation du site web

La première phase consiste à mettre en place un site web avec Apache2 ou NGINX, selon vos besoins. Une fois le site prêt et disponible sur les deux serveurs, nous pouvons commencer la configuration du Heartbeat.

NB : mettez des messages ou des affiches différents dans chaque site, cela vous permettra de mieux comprendre lors des tests.

❖ Configuration des interfaces

Avant la phase de configuration des fichiers, il faut définir une interface virtuelle dans les deux serveurs. Cette interface agira comme étant une intermédiaire vers les services appropriés, dans notre cas, pour le service Web, ce qui veut dire qu'au moment de la recherche du site web depuis un client, c'est cette interface qui sera utilisée au lieu des interfaces locales de chaque serveur.

```
auto ens34
iface ens34 inet static
address 192.168.100.1
netmask 255.255.255.0

auto ens34:0
iface ens34:0 inet static
address 192.168.100.100
netmask 255.255.255.0
```

serveurA

```
auto ens34
iface ens34 inet static
address 192.168.100.2
netmask 255.255.255.0

auto ens34:0
iface ens34:0 inet static
address 192.168.100.100
netmask 255.255.255.0
```

serveurB

```
root@serverA:~# ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.23.225  netmask 255.255.255.0  broadcast 192.168.23.255
        inet6 fe80::20c:29ff:fe5e:a342  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:5e:a3:42  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens34: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.100.1  netmask 255.255.255.0  broadcast 192.168.100.255
        inet6 fe80::20c:29ff:fe5e:a34c  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:5e:a3:4c  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ens34:0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.100.100  netmask 255.255.255.0  broadcast 192.168.100.255
        ether 00:0c:29:5e:a3:4c  txqueuelen 1000  (Ethernet)
```

❖ Configuration du fichier hosts

Les deux serveurs, a part d'être sur le même réseaux, doivent, logiquement, pouvoir se communiquer à travers de l'envoie des requêtes ping, mais pas seulement en utilisant leurs adresses IP, la communication doit aussi se faire à partir de leurs noms, c'est pour cela il est très important d'ajouter la résolution des adresses IP de chaque serveur vers leurs noms dans le fichier hosts, prenez l'exemple des images ci-dessous :

```
root@serverA:~# cat /etc/hosts
127.0.0.1       serverA
127.0.1.1       serverA.localdomain
192.168.100.1   serverA
192.168.100.2   serverB
```

```
root@serverB:~# cat /etc/hosts
127.0.0.1       serverB
127.0.1.1       serverB.localdomain
192.168.100.2   serverB
192.168.100.1   serverA
```

Notez que dans chaque serveur, le nom et l'adresse IP de l'autre sont spécifies, ce qui permet de faire une communication en utilisant les noms :

```
root@serverB:~# ping serverA
PING serverA (192.168.100.1) 56(84) bytes of data.
64 bytes from serverA (192.168.100.1): icmp_seq=1 ttl=64 time=0.971 ms
64 bytes from serverA (192.168.100.1): icmp_seq=2 ttl=64 time=0.884 ms
```

❖ **Installation y configuration de Heartbeat**
Installation : apt -y install heartbeat

- **Modification des fichiers de configuration : cd /etc/ha.d/**

Décompression et copy :

- gzip -d /usr/share/doc/heartbeat/ha.cf.gz
- gzip -d /usr/share/doc/heartbeat/haresources.gz
- cp /usr/share/doc/heartbeat/ha.cf /etc/ha.d/
- cp /usr/share/doc/heartbeat/authkeys /etc/ha.d/
- cp /usr/share/doc/heartbeat/haresources /etc/ha.d/

Changement des droits :
chmod 600 /etc/ha.d/*

Modification du fichier authkeys :
Ce fichier contient une clé partagée entre les deux serveurs, ça peut être une clé ou même un seul mot, le but c'est que le contenu reste le même dans les deux serveurs.

- auth 1
- 1 crc

Modification du fichier ha.cf :

- logfile /var/log/ha-log
- logfacility local0
- keepalive 2 => Temps de vérification de l'autre serveur (en secondes)
- deadtime 30 => Temps pour considérer que l'autre serveur s'est arrête
- warntime 10 => Temps d'alerte
- initdead 120 => Temps d'attente de Heartbeat pour se reconnecter
- udpport 694 => Le port d'écoute
- baud 19200 => La fréquence

- bcast ens34 => Interface de travail
- ucast ens34 192.168.100.2 (IP de l'autre serveur)
- auto_failback on => Se reconnecter après la panne
- node nomserverA
- node nomserveurB

Modification du fichier haresources. A la première ligne ajouter :

- serverA IPaddr::192.168.100.3/24/ens34 apache2

NB : Pour le fichier haresources

- serverA => Nom du cluster (serveur) principal
- 192.168.100.3 => Adresse à utiliser pour voir la ressource
- apache2 =>service à rendre disponible dans les deux serveurs

La configuration est terminée, il nous reste qu'à redémarrer le service web (apache ou nginx) et démarrer le service de haut disponibilité Heartbeat

- systemctl restart apache2/nginx (selon le service que vous avez utilisé)
- systemctl enable apache2/nginx
- systemctl enable heartbeat
- systemctl start heartbeat

❖ **Test**

En utilisant un client de votre choix, ouvrez votre navigateur et tapez l'adresse IP virtuelle, notez que le site configuré dans le serveurA est affiché, ce qui veut dire que la redirection vers l'adresse IP flottant fonctionne correctement. Essayez maintenant d'éteindre le serveurA et rechargez la page, notez que le serveurB a automatiquement répondu à l'absence du serveur principal et la page affiche et celle appartenant au serveurB.

Vous avez déjà votre service de haute disponibilité configuré, n'hésitez pas à ajouter un troisième serveur pour le test a 3 directions.

NB : La Haut disponibilité (Heartbeat) s'assure que l'information soit toujours disponible, si le premier serveur est éteint, vous allez vous rendre compte que vous ne pourrez pas éteindre le deuxième car au moins un serveur doit être en cours d'exécution. Ne vous inquiétez pas, vous pouvez forcer l'arrête de votre serveur, cela ne rendra pas inutiles vos configurations.

La réplication des données (DRBD)

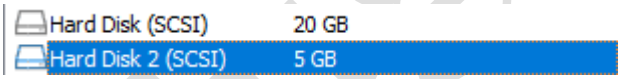
Le service DRBD (Distributed Replicated Block Device) est un outil pour les systèmes GNU/Linux qui permet de répliquer des périphériques de stockage (comme des disques durs, des partitions ou des volumes logiques) entre deux serveurs (nœuds) à travers un réseau. DRBD assure que les données écrites sur le disque d'un serveur sont également écrites sur le disque de l'autre serveur en temps réel.

Voici les principaux objectifs et utilisations du service DRBD :

- **Haute disponibilité (High Availability - HA)** : C'est son utilisation principale. Si un des serveurs tombe en panne, l'autre serveur possède une copie exacte des données et peut prendre le relais, minimisant ainsi les interruptions de service.
- **Réplication en temps réel** : Les données sont synchronisées instantanément pendant qu'elles sont modifiées. Cela peut se faire de manière synchrone (l'écriture est confirmée uniquement lorsque les deux serveurs ont écrit les données) ou asynchrone.
- **Tolérance aux pannes** : En cas de défaillance matérielle sur un des serveurs, les données restent accessibles sur l'autre.
- **Base pour d'autres systèmes** : DRBD peut être utilisé comme base pour des systèmes de fichiers en cluster (comme GFS ou OCFS) ou d'autres périphériques de bloc logique (comme LVM).

MISE EN PLACE

DRBD ne supporte que la réplication de partition entière, on va donc créer une partition sur chaque serveur. Pour cela, il faudra éteindre complètement les deux serveurs et ajouter un deuxième disque, une fois fait, rallumez les serveurs. Dans mon cas, j'ajoute un disque de 5Go dans chaque serveur.



Préparation des disques

Une fois les disques ajoutés, vérifions-les avec la commande **fdisk -l**

```
root@serverA:~# fdisk -l
Disque /dev/sda : 20 GiB, 21474836480 octets, 41943040 secteurs
Modèle de disque : VMware Virtual S
Unités : secteur de 1 × 512 = 512 octets
Taille de secteur (logique / physique) : 512 octets / 512 octets
taille d'E/S (minimale / optimale) : 512 octets / 512 octets
Type d'étiquette de disque : dos
Identifiant de disque : 0xc6d0881a

Périphérique Amorçage    Début      Fin Secteurs Taille Id Type
/dev/sda1    *            2048 39942143 39940096    19G 83 Linux
/dev/sda2             39944190 41940991  1996802   975M  5 Étendue
/dev/sda5             39944192 41940991  1996800   975M 82 partition d'échange Linux / Solaris


Disque /dev/sdb : 5 GiB, 5368709120 octets, 10485760 secteurs
Modèle de disque : VMware Virtual S
Unités : secteur de 1 × 512 = 512 octets
Taille de secteur (logique / physique) : 512 octets / 512 octets
taille d'E/S (minimale / optimale) : 512 octets / 512 octets
```

Puis on crée la nouvelle partition sur ce disque avec cette commande en renseignant bien le nom du disque :

NB : A faire dans les deux serveurs.

```
root@serverA:~# fdisk /dev/sdb_
```

S'ensuit une application pour la gestion de partition.

- Entrer *n* pour ajouter une nouvelle partition
- Puis *p* pour créer une partition primaire.
- Puis un numéro de partition, taper *1*
- Puis *Entrée* pour régler les questions du premier et dernier secteur.
- Puis *Entrée* pour valider.
- En dernier, *w* pour enregistrer les modifications

```
root@serverA:~# fdisk /dev/sdb

Bienvenue dans fdisk (util-linux 2.38.1).
Les modifications resteront en mémoire jusqu'à écriture.
Soyez prudent avant d'utiliser la commande d'écriture.

Le périphérique ne contient pas de table de partitions reconnue.
Created a new DOS (MBR) disklabel with disk identifier 0x6c88baf4.

Commande (m pour l'aide) : n
Type de partition
   p   primaire (0 primaire, 0 étendue, 4 libre)
   e   étendue (conteneur pour partitions logiques)
Sélectionnez (p par défaut) : p
Numéro de partition (1-4, 1 par défaut) : 1
Premier secteur (2048-10485759, 2048 par défaut) :
Dernier secteur, +/-secteurs ou +/-taille{K,M,G,T,P} (2048-10485759, 10485759 par défaut) :

Une nouvelle partition 1 de type « Linux » et de taille 5 GiB a été créée.

Commande (m pour l'aide) : w
La table de partitions a été altérée.
Appel d'ioctl() pour relire la table de partitions.
Synchronisation des disques.
```

Entrer à nouveau la commande *fdisk -l* pour vérifier la partition nouvellement créée.

```
Disque /dev/sdb : 5 GiB, 5368709120 octets, 10485760 secteurs
Modèle de disque : VMware Virtual S
Unités : secteur de 1 × 512 = 512 octets
Taille de secteur (logique / physique) : 512 octets / 512 octets
taille d'E/S (minimale / optimale) : 512 octets / 512 octets
Type d'étiquette de disque : dos
Identifiant de disque : 0x4f473570

Périphérique Amorçage Début      Fin Secteurs Taille Id Type
/dev/sdb1             2048 10485759 10483712     5G 83 Linux
```

Maintenant que notre partition est créée, on va pouvoir la formater. On formate avec le système de fichier ext4.

```
root@serverB:~# mkfs.ext4 /dev/sdb1
mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 1310464 4k blocks and 327680 inodes
Filesystem UUID: 0ce1d811-cee2-44b6-9772-ce08bcfa83e9
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

INSTALLATION DE DRBD

- Activer le module drbd en faisant : modprobe drbd
- Installer le paquet drbd-utils : apt install drbd-utils
- Vérifier la version de l'appli : drbdadm -V

Ouvrir le fichier de configuration principal et décommenter les includes dans */etc/drbd.conf*

```
include "drbd.d/global_common.conf";
include "drbd.d/*.res";
```

Modification des fichiers ;

- vim */etc/drbd.d/global_common.conf*

```
global {
        usage-count no;
        udev-always-use-vnr;
}

common {
        handlers {}

        startup {
                wfc-timeout 15;
                degr-wfc-timeout 60;
        }

        options {}

        disk {}

        net {
                protocol C;
                cram-hmac-alg sha1;
                shared-secret "passer";
        }
}
```

- vim */etc/drbd.d/r0.res*, à adapter selon chaque cas

```
resource r0 {

        on serverA { # nom de la machine
                device /dev/drbd0;
                disk /dev/sdb1;  # nom de la partition
                address 192.168.100.1:7788;
                meta-disk internal;
        }

        on serverB {
                device /dev/drbd0;
                disk /dev/sdb1;
                address 192.168.100.2:7788;
                meta-disk internal;
        }
}
```

Pour vérifier la syntaxe de la configuration : drbdadm dump

Cette commande devrait renvoyer toutes les lignes ajoutes dans les fichiers de configuration. Si on vous affiche des erreurs de syntaxe, il faudra rentrer dans les fichiers et aller corriger.

Puis pour créer les metadata : drbdadm create-md r0

Si la commande retourne cette réponse :

```
root@serverA:~# drbdadm create-md r0
md_offset 5367656448
al_offset 5367623680
bm_offset 5367459840

Found ext3 filesystem
     5241856 kB data area apparently used
     5241660 kB left usable by current configuration

Device size would be truncated, which
would corrupt data and result in
'access beyond end of device' errors.
You need to either
   * use external meta data (recommended)
   * shrink that filesystem first
   * zero out the device (destroy the filesystem)
Operation refused.

Command 'drbdmeta 0 v08 /dev/sdb1 internal create-md' terminated with exit code 40
```

On va donc formater le disque avec : shred -zvf -n 1 /dev/sdb1

```
root@serverB:~# shred -zvf -n 1 /dev/sdb1
shred: /dev/sdb1 : étape 1/2 (random)…
shred: /dev/sdb1 : étape 1/2 (random)…501MiB/5,0GiB 9 %
shred: /dev/sdb1 : étape 1/2 (random)…1007MiB/5,0GiB 19 %
shred: /dev/sdb1 : étape 1/2 (random)…1,4GiB/5,0GiB 29 %
shred: /dev/sdb1 : étape 1/2 (random)…1,9GiB/5,0GiB 38 %
shred: /dev/sdb1 : étape 1/2 (random)…2,2GiB/5,0GiB 45 %
shred: /dev/sdb1 : étape 1/2 (random)…2,7GiB/5,0GiB 55 %
shred: /dev/sdb1 : étape 1/2 (random)…3,2GiB/5,0GiB 65 %
shred: /dev/sdb1 : étape 1/2 (random)…3,6GiB/5,0GiB 73 %
shred: /dev/sdb1 : étape 1/2 (random)…4,0GiB/5,0GiB 81 %
shred: /dev/sdb1 : étape 1/2 (random)…4,4GiB/5,0GiB 88 %
shred: /dev/sdb1 : étape 1/2 (random)…4,8GiB/5,0GiB 97 %
shred: /dev/sdb1 : étape 1/2 (random)…5,0GiB/5,0GiB 100 %
shred: /dev/sdb1 : étape 2/2 (000000)…
shred: /dev/sdb1 : étape 2/2 (000000)…593MiB/5,0GiB 11 %
shred: /dev/sdb1 : étape 2/2 (000000)…1,1GiB/5,0GiB 23 %
shred: /dev/sdb1 : étape 2/2 (000000)…1,7GiB/5,0GiB 34 %
shred: /dev/sdb1 : étape 2/2 (000000)…2,2GiB/5,0GiB 45 %
shred: /dev/sdb1 : étape 2/2 (000000)…2,8GiB/5,0GiB 56 %
shred: /dev/sdb1 : étape 2/2 (000000)…3,3GiB/5,0GiB 67 %
shred: /dev/sdb1 : étape 2/2 (000000)…3,9GiB/5,0GiB 78 %
shred: /dev/sdb1 : étape 2/2 (000000)…4,5GiB/5,0GiB 90 %
shred: /dev/sdb1 : étape 2/2 (000000)…5,0GiB/5,0GiB 100 %
```

On retape la commande drbdadm create-md r0, cette fois-ci marchera.

Et pour appliquer la nouvelle configuration : drbdadm adjust r0

À ce stade, vous devez avoir une *ressource Connected* mais avec des données inconsistentes : drbdadm status

```
root@serverA:~# drbdadm status
r0 role:Secondary
  disk:Inconsistent
  peer role:Secondary
    replication:Established peer-disk:Inconsistent
```

Cela veut dire que vos nodes se connectent mais que la réplication n'est pas encore possible étant donné qu'aucun des deux n'est en mode Primary, pour y remédier nous allons mettre node 1 en primary avec la commande suivante : drbdadm -- --overwrite-data-of-peer primary r0

Et node 2 en secondary : drbdadm secondary r0

La synchronisation initiale se lance, vous pouvez vérifier l'état de la synchronisation avec la commande suivante : cat /proc/drbd

```
root@serverA:~# cat /proc/drbd
version: 8.4.11 (api:1/proto:86-101)
srcversion: 96ED19D4C144624490A9AB1
 0: cs:SyncSource ro:Primary/Secondary ds:UpToDate/Inconsistent C r-----
    ns:187148 nr:0 dw:0 dr:187148 al:8 bm:0 lo:6 pe:0 ua:6 ap:0 ep:1 wo:f oos:5054512
        [>....................] sync'ed:  3.6% (4936/5116)M
        finish: 0:37:31 speed: 2,236 (2,100) K/sec
```

On attend la fin de la synchronisation.
Pour suivre la synchronisation ajoutez « watch » derrière la commande :
watch cat /proc/drbd

A la fin, on devrait avoir ce résultat :

```
version: 8.4.11 (api:1/proto:86-101)
srcversion: 96ED19D4C144624490A9AB1
 0: cs:Connected ro:Primary/Secondary ds:UpToDate/UpToDate C r-----
    ns:5241660 nr:0 dw:0 dr:5241660 al:8 bm:0 lo:0 pe:0 ua:0 ap:0 ep:1 wo:f oos:0
```

Une fois synchronisé, on devrait avoir ça : drbdadm status

```
root@serverA:~# drbdadm status
r0 role:Primary
  disk:UpToDate
  peer role:Secondary
    replication:Established peer-disk:UpToDate
```

Maintenant, il faut formater la partition drbd uniquement sur le serveur primaire.

- mkfs.ext4 /dev/drbd0

```
root@serverA:~# mkfs.ext4 /dev/drbd0
mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 1310415 4k blocks and 327680 inodes
Filesystem UUID: f7fafb11-4676-4837-8ca0-ed8bcdbd7a99
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

Puis pour monter la partition sur le serveur primaire :

- mkdir /serverA
- mount /dev/drbd0 /serverA

La commande *lsblk* permet de voir les points de montages :

```
root@serverA:~# lsblk
NAME      MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda         8:0    0   20G  0 disk
├─sda1      8:1    0   19G  0 part /
├─sda2      8:2    0    1K  0 part
└─sda5      8:5    0  975M  0 part [SWAP]
sdb         8:16   0    5G  0 disk
└─sdb1      8:17   0    5G  0 part
  └─drbd0 147:0    0    5G  0 disk /serverA
sr0        11:0    1  633M  0 rom
```

Il faut à présent déposer un fichier HTML dans le dossier */serverA*

Paramétrer Apache2
Modification de 00default.conf
On va spécifier que la racine du serveur apache2 est dans */web*.
Vim /etc/apache2/sites-available/000-default.conf

```
DocumentRoot /web
<directory /serverA>
    require all granted
</directory>
```

Puis changer le dossier web de propriétaire avec -R pour récursif :

chown -R www-data:www-data /serverA

On redémarre Apache2 et on teste dans le navigateur.

systemctl restart apache2

Paramétrage de HeartBeat

Éditez le fichier */etc/ha.d/haresources* sur le serveA et serverB :

serverA IPaddr::192.168.100.100/24/ens34 drbddisk::r0 Filesystem::/dev/drbd0::/serverA::ext4 apache2

Il faut effectuer des tests pour vérifier que la réplication se fait bien dans le dossier */web* de clone2 lorsque l'on change le contenu de */Web* sur clone1.

Arrêter le serveur heartbeat sur le *clone 1* pour tester que la bascule se fait bien sur le serveur esclave en arrêtant le serveur maître.

Conclusion

En conclusion, la mise en œuvre de la haute disponibilité et de la réplication représente un investissement stratégique fondamental pour toute organisation soucieuse de la pérennité et de la fiabilité de ses systèmes d'information. La haute disponibilité garantit une continuité de service essentielle aux opérations critiques, minimisant les interruptions coûteuses et préservant la satisfaction des utilisateurs. Parallèlement, la réplication assure la protection des données contre les sinistres et les erreurs humaines, offrant ainsi une base solide pour la reprise d'activité.

Au travers de ce rapport, nous avons exploré les principes clés et les meilleures pratiques associés à ces deux concepts indissociables. La sélection et la configuration appropriées des solutions de haute disponibilité et de réplication doivent être guidées par une analyse rigoureuse des besoins métiers, des contraintes budgétaires et des objectifs de niveau de service.

Bien que la complexité de ces technologies puisse être significative, les bénéfices qu'elles apportent en termes de résilience et de sécurité des données sont inestimables dans un paysage numérique en constante évolution. L'adoption et la maintenance continues de stratégies de haute disponibilité et de réplication constituent donc un impératif pour assurer la compétitivité et la pérennité des organisations à long terme.