

TD Load Balancing

Valentin Benard

25/02/2025

Un container est créé sur Proxmox avec l'adresse **192.168.63.164** hostname **benard-tp-loadbalancing-debian**

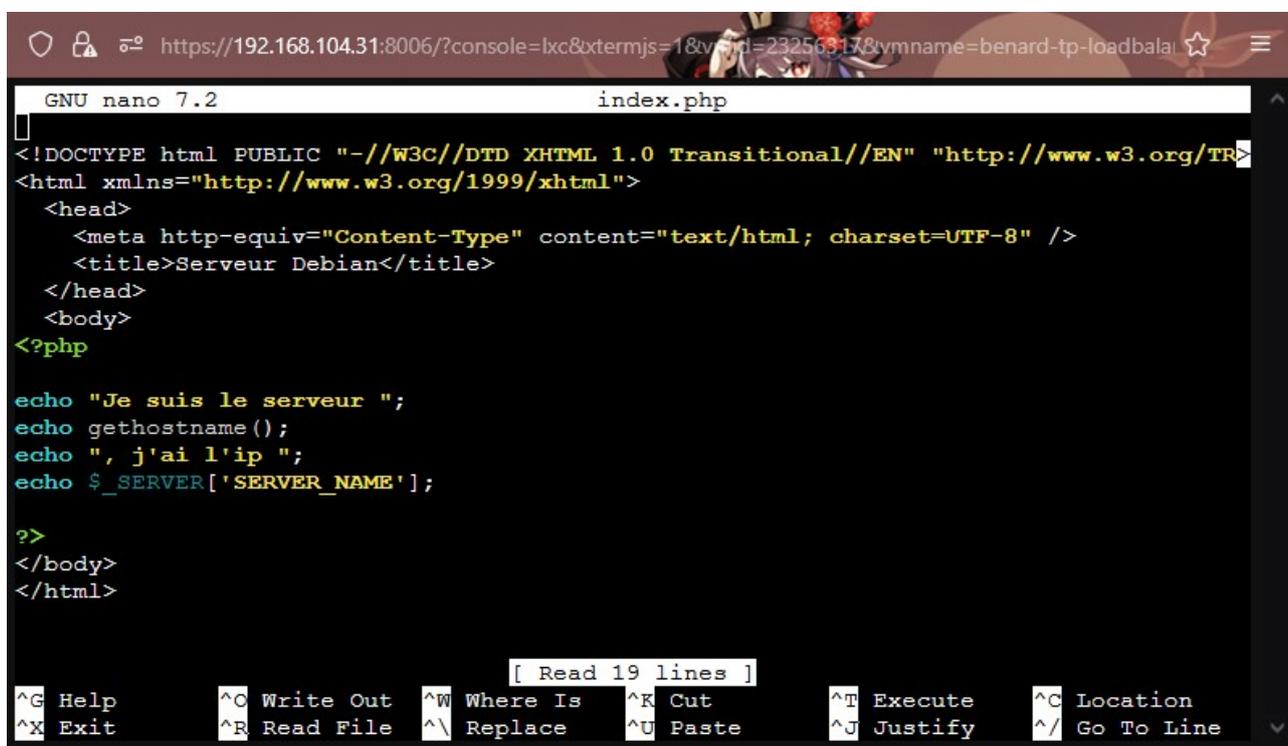
Le proxy est comme d'habitude renseigné via /etc/apt/apt.conf.

Un **apt update** est effectué

Les paquets suivants sont installés conformément aux consignes : **apache2, php**

Afficher sur la page web « je suis le serveur (nom du serveur) » (il doit aller les chercher dans les variables d'environnement de php)

J'ai l'ip (ip variable)



```
GNU nano 7.2 index.php
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Serveur Debian</title>
  </head>
  <body>
<?php
echo "Je suis le serveur ";
echo gethostname();
echo ", j'ai l'ip ";
echo $_SERVER['SERVER_NAME'];

?>
</body>
</html>
```

La machine virtuelle est par la suite clonée via Proxmox :



Elle aura l'adresse **192.168.63.165**.

Résultat :



Une troisième machine est créée pour le load balancer.

Son adresse est **192.168.63.166**

Configuration du load balancer

Activation des modules nécessaires

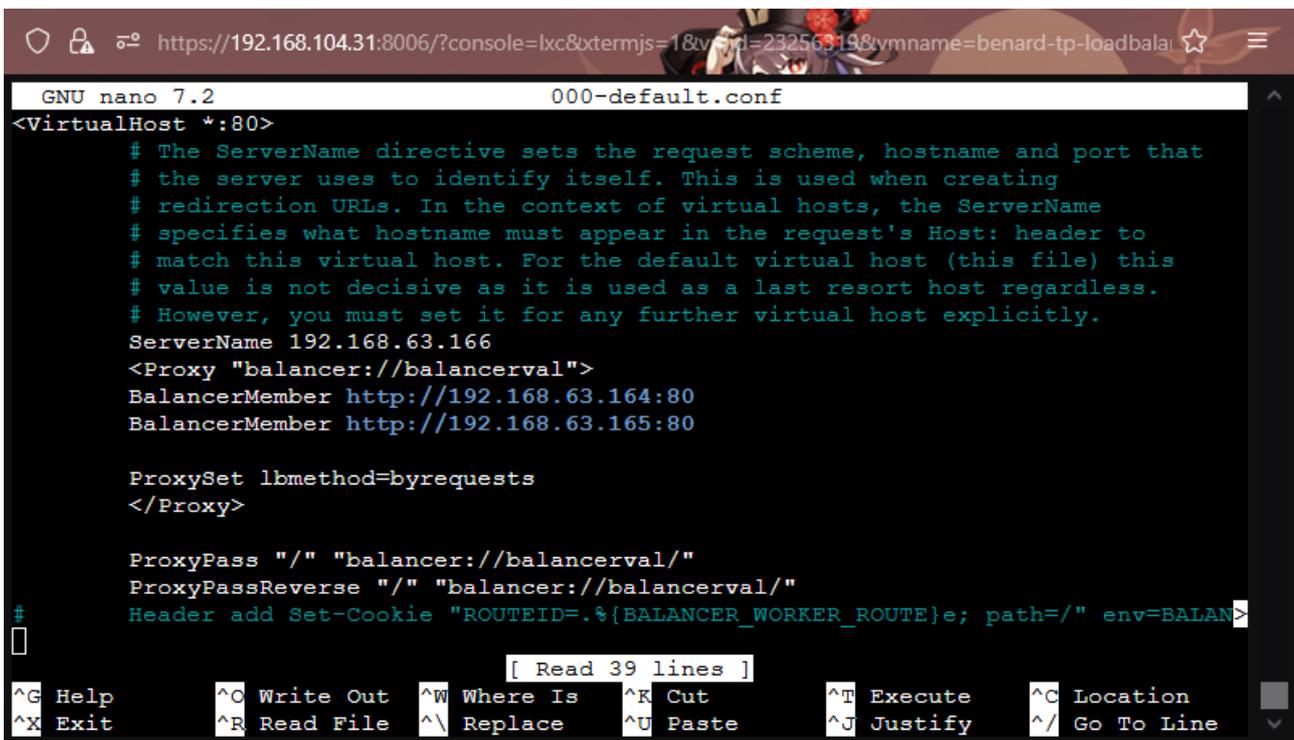
```
a2enmod proxy_http  
a2enmod proxy_balancer  
a2enmod lbmethod_byrequests
```

Apache 2 est redémarré pour charger les modules.

```
systemctl restart apache2
```

Sur le serveur Load balancing :

```
nano /etc/apache2/sites-enabled/000-default.conf
```



```
GNU nano 7.2 000-default.conf  
<VirtualHost *:80>  
# The ServerName directive sets the request scheme, hostname and port that  
# the server uses to identify itself. This is used when creating  
# redirection URLs. In the context of virtual hosts, the ServerName  
# specifies what hostname must appear in the request's Host: header to  
# match this virtual host. For the default virtual host (this file) this  
# value is not decisive as it is used as a last resort host regardless.  
# However, you must set it for any further virtual host explicitly.  
ServerName 192.168.63.166  
<Proxy "balancer://balancerval">  
BalancerMember http://192.168.63.164:80  
BalancerMember http://192.168.63.165:80  
  
ProxySet lbmethod=byrequests  
</Proxy>  
  
ProxyPass "/" "balancer://balancerval/"  
ProxyPassReverse "/" "balancer://balancerval/"  
# Header add Set-Cookie "ROUTEID=.%{BALANCER_WORKER_ROUTE}e; path=/" env=BALAN
```

Ajouter les lignes :

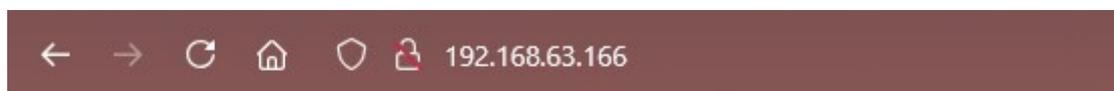
ServerName (ip)

```
<Proxy "balancer://clustervb">  
  BalancerMember http://ip:80  
  BalancerMember http://ip:80
```

```
  ProxySet lbmethod=byrequests  
</Proxy>
```

```
ProxyPass "/" "balancer://clustervb/"  
ProxyPassReverse "/" "balancer://clustervb/"
```

Résultat :

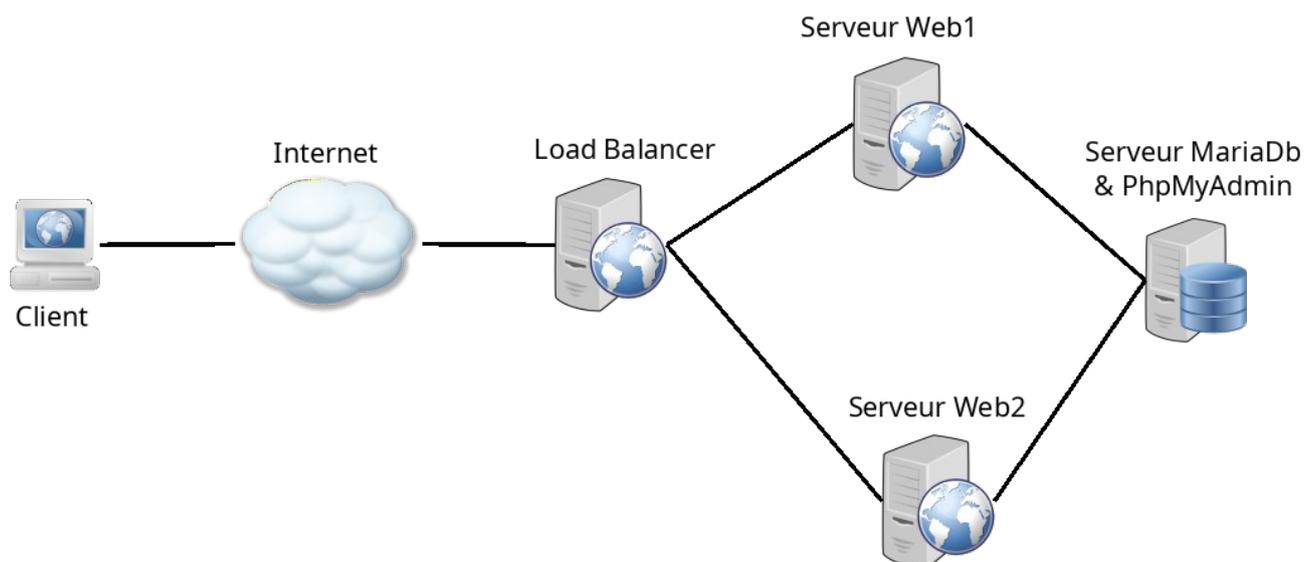


Je suis le serveur benard-tp-loadbalancing-debian2, j'ai l'ip 192.168.63.165



Je suis le serveur benard-tp-loadbalancing-debian, j'ai l'ip 192.168.63.164

Un quatrième serveur mariadb avec phpmyadmin est mis en place (192.168.63.167)



phpmyadmin est installé

Le serveur est accessible via <http://192.168.63.167/phpmyadmin/>

L'utilisateur eleve est utilisé pour se connecter, il détient de toutes les permissions.

mariadb

```
CREATE USER 'username'@'localhost' IDENTIFIED BY 'password';
```

```
GRANT ALL PRIVILEGES ON *.* TO 'username'@'localhost';
```

```
FLUSH PRIVILEGES;
```

```
exit
```

Une table avec les colonnes id Nom et Prenom sont ajoutées :

v	utilisateurs	utilisateurs
#	id	: int(11)
⊞	Nom	: varchar(30)
⊞	Prenom	: varchar(30)

La table est remplie avec quelques noms pour tester dans Insérer :

The screenshot shows the phpMyAdmin interface. On the left, the database structure is visible, with 'utilisateurs' selected. The main area shows the table structure with columns: id (int(11)), Nom (varchar(30)), and Prenom (varchar(30)). The 'Insérer' button is highlighted with a red box. Below the table structure, a message indicates '1 ligne insérée.' and the SQL query is displayed: `INSERT INTO `utilisateurs` (`id`, `Nom`, `Prenom`) VALUES ('02', 'Rona...`. The 'Exécuter' button is also highlighted with a yellow box.

Pour interroger depuis un autre serveur, il faut installer le driver :

apt install php-mysql

Ajout du script dans les pages php des serveurs web :

Partie PHP

```
<?php

$host = '192.168.63.167'; // Adresse du serveur MySQL
$dbname = 'utilisateurs'; // Nom de la base de données
$username = 'eleve'; // Nom d'utilisateur MySQL
$password = 'eleve'; // Mot de passe MySQL
$table = 'utilisateurs'; // Nom de la table à afficher

try {
    $pdo = new PDO("mysql:host=$host;dbname=$dbname;charset=utf8", $username, $password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

    $stmt = $pdo->query("SELECT * FROM $table");

    $firstRow = $stmt->fetch(PDO::FETCH_ASSOC);
    if (!$firstRow) {
        die("Erreur : La table est vide ou la requête a échoué.");
    }

    $columns = array_keys($firstRow);

    // Réexécuter la requête car fetch() avance le curseur
    $stmt = $pdo->query("SELECT * FROM $table");

} catch (PDOException $e) {
    die("Erreur de connexion : " . $e->getMessage());
}

echo "Je suis le serveur ";
echo gethostname();
echo ", j'ai l'ip ";
echo $_SERVER['SERVER_NAME'];

?>
```

Partie HTML

```
<h2>Contenu de la table "<?= htmlspecialchars($table) ?>"</h2>
<table>
  <tr>
    <?php foreach ($columns as $col): ?>
      <th><?= htmlspecialchars($col) ?></th>
    <?php endforeach; ?>
  </tr>
  <?php while ($row = $stmt->fetch(PDO::FETCH_ASSOC)): ?>
    <tr>
```

```
<?php foreach ($row as $value): ?>
  <td><?= htmlspecialchars($value) ?></td>
<?php endforeach; ?>
</tr>
<?php endwhile; ?>
</table>
```

La connexion depuis d'autres serveurs doit être autorisée sur le serveur de base de données, pour procéder, il faut modifier le fichier de configuration de mariadb :

```
nano /etc/mysql/mariadb.conf.d/50-server.cnf
```

Remplacer bind address de 127.0.0.1 par 0.0.0.0

Redémarrer le service mariadb

```
systemctl restart mariadb
```

Ajouter les permissions sur le serveur db :

```
GRANT ALL PRIVILEGES ON eleve.* TO 'eleve'@'%' IDENTIFIED BY 'eleve';
FLUSH PRIVILEGES;
```

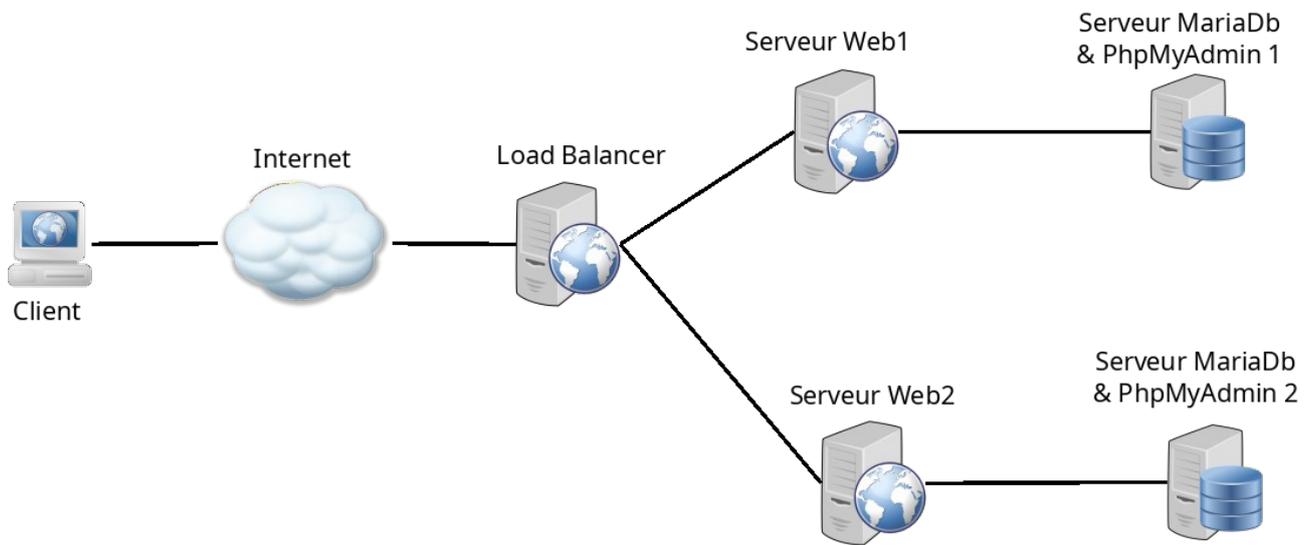
Résultat :



The screenshot shows a web browser interface. The address bar contains the IP address 192.168.63.166. Below the address bar, the text reads: "Je suis le serveur benard-tp-loadbalancing-debian, j'ai l'ip 192.168.63.164". Below this, there is a heading "Contenu de la table 'utilisateurs'" followed by a table with two columns: "id" and "Nom Prenom". The table contains two rows of data: "1 Thierry Henry" and "2 Ronald McDonald".

id	Nom Prenom
1	Thierry Henry
2	Ronald McDonald

Ajout d'un deuxième serveur de DB pour avoir de la redondance sur la base de données également (elles seront par la suite synchronisée sinon le résultat sera différent selon le serveur interrogé) :



(Le serveur db1 est cloné sur Proxmox)

192.168.63.168

benard-tp-loadbalancing-debian-db2

Une ligne est ajoutée sur chacun des serveurs précisant quel serveur de base de données il interroge (c'est la variable \$host qui est l'ip du serveur interrogé au début du script)

```

echo "Je suis le serveur ";
echo gethostname ();
echo ", j'ai l'ip ";
echo $ SERVER['SERVER NAME'];
echo ", j'interroge le serveur $host";
?>
  
```

L'IP est modifiée sur le deuxième serveur qui interroge le serveur 2 récemment cloné :

```

<?php
$host = '192.168.63.168'; // Adresse du serveur MySQL
$username = 'utilisateur'; // Nom de la base de données
  
```

Résultat :

← → ↻ 🏠 🛡️ 🔒 192.168.63.166

Je suis le serveur benard-tp-loadbalancing-debian, j'ai l'ip 192.168.63.164, j'interroge le serveur 192.168.63.167

Contenu de la table "utilisateurs"

```
id Nom Prenom
1 Thierry Henry
2 Ronald McDonald
```

On peut également ajouter un peu de CSS pour former le tableau

En haut dans le <head> </head>

Rajouter

```
<style>
  table { border-collapse: collapse; width: 100%; }
  th, td { border: 1px solid black; padding: 8px; text-align: left; }
  th { background-color: #f2f2f2; }
</style>
```

Résultat :

← → ↻ 🏠 🛡️ 🔒 192.168.63.166

Je suis le serveur benard-tp-loadbalancing-debian, j'ai l'ip 192.168.63.164, j'interroge le serveur 192.168.63.167

Contenu de la table "utilisateurs"

id	Nom	Prenom
1	Thierry	Henry
2	Ronald	McDonald

Note : Au cas où un problème survient : Ajouter ces deux lignes php pour afficher les erreurs :

```
error_reporting(E_ALL);
ini_set('display_errors', 1);
```

Le script est modifié afin qu'il puisse interroger le deuxième serveur DB si le premier est down :

```
try {
    $pdo = new PDO("mysql:host=$host;dbname=$dbname;charset=utf8", $username, $password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch (PDOException $e) {
    try {
        $pdo = new PDO("mysql:host=$host2;dbname=$dbname;charset=utf8", $username,
$password);
        $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    } catch (PDOException $e) {
        die("Erreur de connexion : " . $e->getMessage());
    }
}

$stmt = $pdo->query("SELECT * FROM $table");
$firstRow = $stmt->fetch(PDO::FETCH_ASSOC);
if (!$firstRow) {
    die("Erreur : La table est vide ou la requête a échoué.");
}
$columns = array_keys($firstRow);
$stmt = $pdo->query("SELECT * FROM $table");
```

Une variable pour l'adresse du deuxième serveur est ajoutée, \$host2 :

```
<?php
$host = '192.168.63.167'; // Adresse du serveur MySQL
$host2 = '192.168.63.168'; // Serveur 2
$dbname = 'utilisateurs'; // Nom de la base de données
```

Cependant, le serveur ne retourne plus l'adresse du serveur de BDD utilisé, il faut donc modifier le script une nouvelle fois :

La variable connectedHost est ajoutée :

```
$host = '192.168.63.167'; // Adresse du serveur MySQL
$host2 = '192.168.63.168'; // Serveur 2
$dbname = 'utilisateurs'; // Nom de la base de données
$username = 'eleve'; // Nom d'utilisateur MySQL
$password = 'eleve'; // Mot de passe MySQL
$table = 'utilisateurs'; // Nom de la table à afficher
$connectedHost = '';
```

Ajouter ces lignes :

```
$connectedHost = $host;
$connectedHost = $host2;
```

Comme ceci :

```
$host = '192.168.63.167'; // Adresse du serveur MySQL
$host2 = '192.168.63.168'; // Serveur 2
$dbname = 'utilisateurs'; // Nom de la base de données
$username = 'eleve'; // Nom d'utilisateur MySQL
$password = 'eleve'; // Mot de passe MySQL
$table = 'utilisateurs'; // Nom de la table à afficher
$connectedHost = '';

try {
    $pdo = new PDO("mysql:host=$host;dbname=$dbname;charset=utf8", $username, $password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $connectedHost = $host;
} catch (PDOException $e) {
    try {
        $pdo = new PDO("mysql:host=$host2;dbname=$dbname;charset=utf8", $username, $password);
        $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        $connectedHost = $host2;
    } catch (PDOException $e) {
        die("Erreur de connexion : " . $e->getMessage());
    }
}

$stmt = $pdo->query("SELECT * FROM $table");
$firstRow = $stmt->fetch(PDO::FETCH_ASSOC);
if (!$firstRow) {
    die("Erreur : La table est vide ou la requête a choué.");
}
$columns = array_keys($firstRow);
$stmt = $pdo->query("SELECT * FROM $table");

echo "Je suis le serveur ";
echo gethostname();
echo ", j'ai l'ip ";
echo $_SERVER['SERVER_NAME'];
echo ", j'interroge le serveur $connectedHost";
```

Enfin, le message est modifié pour afficher la variable connectedHost qui affichera l'ip du serveur interrogé :

```
echo "Je suis le serveur ";
echo gethostname();
echo ", j'ai l'ip ";
echo $_SERVER['SERVER_NAME'];
echo ", j'interroge le serveur $connectedHost";
```

Résultat :

Je suis le serveur benard-tp-loadbalancing-debian, j'ai l'ip 192.168.63.164, j'interroge le serveur 192.168.63.168

Contenu de la table "utilisateurs"

id	Nom
----	-----

Script complet final :

```
<?php

$host = '192.168.63.167'; // Adresse du serveur MySQL
$host2 = '192.168.63.168'; // Serveur 2
$dbname = 'utilisateurs'; // Nom de la base de données
$username = 'eleve'; // Nom d'utilisateur MySQL
$password = 'eleve'; // Mot de passe MySQL
$table = 'utilisateurs'; // Nom de la table à afficher
$connectedHost = "";

try {
    $pdo = new PDO("mysql:host=$host;dbname=$dbname;charset=utf8", $username, $password);
    $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $connectedHost = $host;
} catch (PDOException $e) {
    try {
        $pdo = new PDO("mysql:host=$host2;dbname=$dbname;charset=utf8", $username,
$password);
        $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        $connectedHost = $host2;
    } catch (PDOException $e) {
        die("Erreur de connexion : " . $e->getMessage());
    }
}

$stmt = $pdo->query("SELECT * FROM $table");
$firstRow = $stmt->fetch(PDO::FETCH_ASSOC);
if (!$firstRow) {
    die("Erreur : La table est vide ou la requête a échoué.");
}
$columns = array_keys($firstRow);
$stmt = $pdo->query("SELECT * FROM $table");

echo "Je suis le serveur ";
echo gethostname();
echo ", j'ai l'ip ";
echo $_SERVER['SERVER_NAME'];
echo ", j'interroge le serveur $connectedHost";

?>
```

Synchronisation des bases de données : Les serveurs web interrogent deux serveurs de bases de données différentes pour la redondance, cependant, leur contenu est différent, il faut donc les synchroniser.

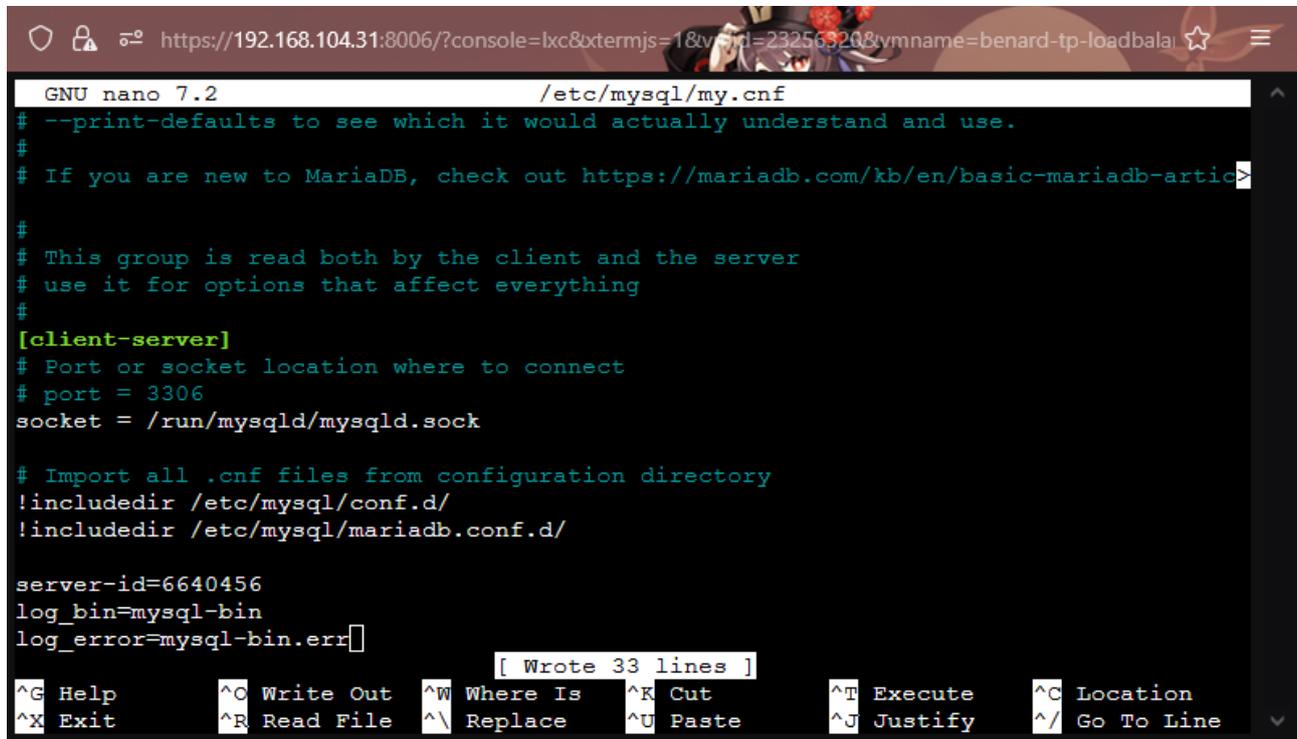
Accéder au panneau phpmyadmin du serveur DB1 (192.168.63.167), puis onglet réplication, sélectionner le premier « configurer »

Choisir « Répliquer toutes les bases de données, ignorer : » (ne rien sélectionner pour ne rien ignorer)

Copier les informations données qui seront en suite à mettre à la fin du fichier

/etc/mysql/mariadb.conf.d/50-server.cnf :

```
server-id=6640456
log_bin=mysql-bin
log_error=mysql-bin.err
```



```
GNU nano 7.2 /etc/mysql/my.cnf
# --print-defaults to see which it would actually understand and use.
#
# If you are new to MariaDB, check out https://mariadb.com/kb/en/basic-mariadb-artic
#
# This group is read both by the client and the server
# use it for options that affect everything
#
[client-server]
# Port or socket location where to connect
# port = 3306
socket = /run/mysqld/mysqld.sock

# Import all .cnf files from configuration directory
!includedir /etc/mysql/conf.d/
!includedir /etc/mysql/mariadb.conf.d/

server-id=6640456
log_bin=mysql-bin
log_error=mysql-bin.err
[ Wrote 33 lines ]
^G Help      ^O Write Out  ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File  ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```

Puis, redémarrer le service avec

```
systemctl restart mariadb
```

Puis créer l'utilisateur mariadb :

```
mariadb
GRANT REPLICATION SLAVE ON *.* TO 'repl'@'%' IDENTIFIED BY 'leve';
FLUSH PRIVILEGES;
```

Et enfin cliquer sur « Exécuter » sur la page.

Sur le serveur 2 (192.168.63.168), retourner dans l'onglet réplique, configurer la réplique (deuxième configurer) copier l'ID et le mettre dans le fichier /etc/mysql/mariadb.conf.d/50-server.cnf

Configuration de la réplique - Changer ou reconfigurer le serveur original

Il faut vérifier d'avoir un server-id unique dans le fichier de configuration (my.cnf). Sinon, merci d'ajouter la ligne suivante dans la section [mysqld] :

`server-id=1741092611`

Nom d'utilisateur :

Mot de passe :

Hôte :

Port :

L'utilisateur de base est renseigné, mais il sera préférable de créer un utilisateur qui n'aura que les droits pour la répllication par la suite.

❗ La connexion au serveur est désactivée, merci d'activer \$cfg['AllowArbitraryServer'] dans la configuration de phpMyAdmin.

Cette erreur apparaît, empêchant la configuration, pour la résoudre, ajouter la ligne :

```
$cfg['AllowArbitraryServer'] = true;
```

Dans le fichier de configuration /etc/phpmyadmin/config.inc.php à la fin.

Une deuxième erreur survient :

❗ Impossible de lire la position du journal sur l'original. C'est un possible problème de privilège sur l'original.

Pour la résoudre :

```
mariadb
CREATE USER 'repl'@'%' IDENTIFIED BY 'password';
GRANT REPLICATION SLAVE ON *.* TO 'repl'@'%';
FLUSH PRIVILEGES;
GRANT REPLICATION SLAVE, REPLICATION CLIENT ON *.* TO 'repl'@'192.168.63.168'
IDENTIFIED BY 'eleve';
FLUSH PRIVILEGES ;
```

Le serveur peut ainsi être configuré.

Pour finir, aller sur le serveur 2, puis répllication et initier un « Full start » dans « Contrôler le serveur répliqué »

Réplication des répliques

Connexion au serveur original :

Le fil d'exécution SQL ne tourne pas sur le serveur répliqué !

Le fil d'exécution d'entrée/sortie ne tourne pas sur le serveur répliqué !

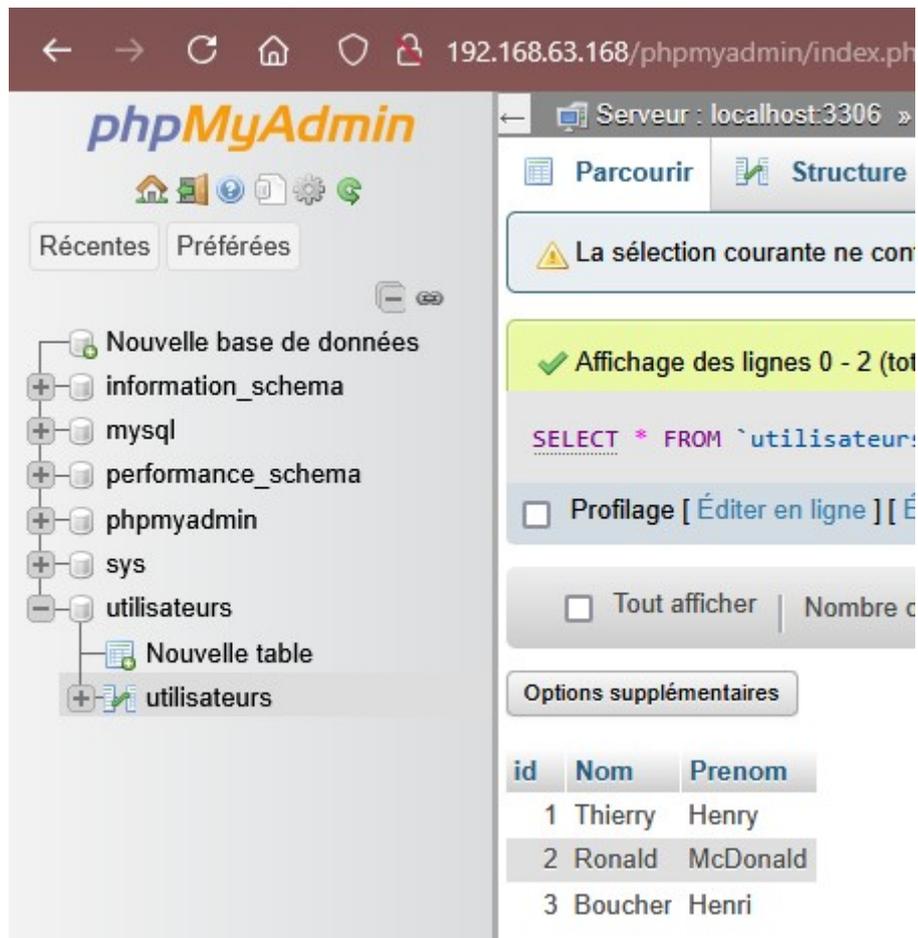
Le serveur est configuré comme réplique dans un processus de réplication. Voulez-vous :

- Voir l'état de la réplique
- Contrôler le serveur répliqué :
 - Full start
 - Rétablir la réplique
 - Démarrer seulement le fil d'exécution SQL
 - Démarrer seulement le fil d'exécution des entrées/sorties
- Gestion des erreurs :
- Changer ou reconfigurer le serveur original

Résultat : (Henri Boucher a été ajouté)

The screenshot shows the phpMyAdmin interface. On the left, a tree view shows the database structure with 'utilisateurs' selected. The main panel shows a SQL query: `SELECT * FROM `utilisateurs``. Below the query, there are options for 'Profilage' and 'Tout afficher'. The results table is displayed below, showing three rows of data.

id	Nom	Prenom
1	Thierry	Henry
2	Ronald	McDonald
3	Boucher	Henri



Partie finale : Synchroniser de façon bi-directionnelle les deux serveurs.

(Jusqu'ici, le serveur 2 était « slave » du « master », serveur 1)

Pour procéder, il suffit de répéter les mêmes étapes que précédemment, c'est le même principe, déclarer le premier serveur slave et le deuxième master.

Fin du TP