



Lycée Générale et Technologie Baimbride  
Brevet Technicien Supérieur  
Services informatiques aux Organisations



## Haute disponibilité et tolérance aux pannes d'un serveur web avec Heartbeat

*Compte rendu*

*TP5 bloc3*



Réalisé par ROSIER Matthias BTS SIO 2A

## Sommaire

Compte rendu TPn°5_Bloc_n°3_SISR.....	3
Introduction.....	3
A) Contexte du TP .....	3
B) objectifs et enjeux .....	3
Etape 1 : Création d'un second serveur web « SRWEB2 » : .....	4
Etape 2 : Mise à jour des fichiers de configuration apache 2.0 des 2 serveurs web : .....	7
Etape 3 : Création et configuration du cluster des serveurs web avec Hearbeat : .....	9
Etape 4 : Test de la haute disponibilité du site web de la ville des abymes : .....	14
Conclusion : .....	18
Fin du TP : .....	18

# COMPTE RENDU TPN°5 BLOC N°3 SISR

## INTRODUCTION

### A) CONTEXTE DU TP

Ce TP se concentre sur l'installation et la configuration d'une **solution de haute disponibilité** avec **Heartbeat** sur un serveur **Linux Debian 12.0**, dans le cadre de la gestion des infrastructures pour la **mairie des Abymes**. L'objectif est de **garantir la disponibilité continue du site web** en mettant en place un **cluster** de serveurs web capable d'assurer un **basculement automatique** en cas de **panne** du **serveur principal**.

Ce projet vise à **renforcer la résilience de l'infrastructure** en assurant une **transition fluide entre** les serveurs **SRVWEB1** et **SRVWEB2** via une **IP virtuelle** gérée par Heartbeat. Grâce à cette approche, les **utilisateurs** accédant au **site web** **ne subiront aucune interruption** de service en cas de défaillance matérielle ou logicielle.

L'implémentation de cette solution s'inscrit dans une démarche **open-source**, permettant **d'optimiser** les **coûts** tout en améliorant la **gestion proactive des incidents**. La **haute disponibilité** est un **enjeu majeur dans les environnements de production**, garantissant la **continuité des services critiques** et **minimisant les interruptions potentielles**.

### B) OBJECTIFS ET ENJEUX

**Acquérir et mettre en application les compétences suivantes :**

- \* Installer et configurer des éléments d'infrastructure
- \* Déployer une solution d'infrastructure
- \* Administrer un système
- \* Automatiser des tâches d'administration
- \* Tester l'intégration et l'acceptation d'une solution d'infrastructure
- \* Rédiger ou mettre à jour la documentation technique et utilisateur d'une solution d'infrastructure

**Condition de travail : Individuel**

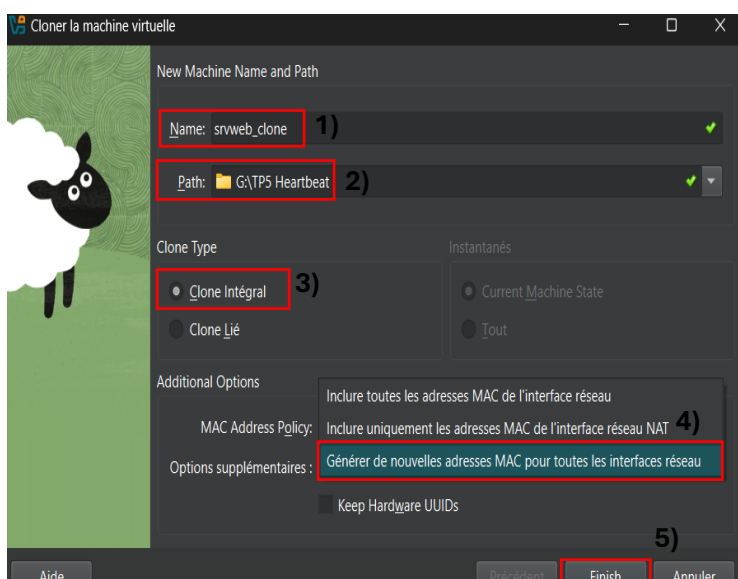
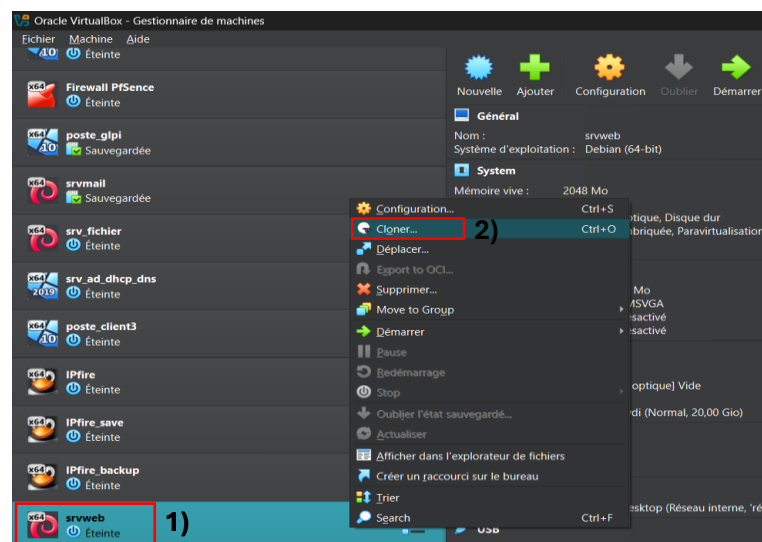
## ETAPE 1 : CRÉATION D'UN SECOND SERVEUR WEB « SRWEB2 » :

Dans cette **première étape**, nous allons **préparer l'infrastructure** du cluster en **créant 3 clones** du **serveur web** existant (**srvweb**) afin d'assurer une **haute disponibilité**.

- ✚ **srvweb\_clone** : Un **clone de sauvegarde** du **serveur web d'origine**, qui restera **intact** pour les **prochains TP**.
- ✚ **srvweb1** : Le **premier serveur du cluster**, qui recevra les requêtes en priorité.
- ✚ **srvweb2** : Le **second serveur du cluster**, qui prendra le relais en cas de panne.

L'**objectif** est de **préserver srvweb** en **évitant toute modification directe**, tout en mettant en place un **cluster fonctionnel**.

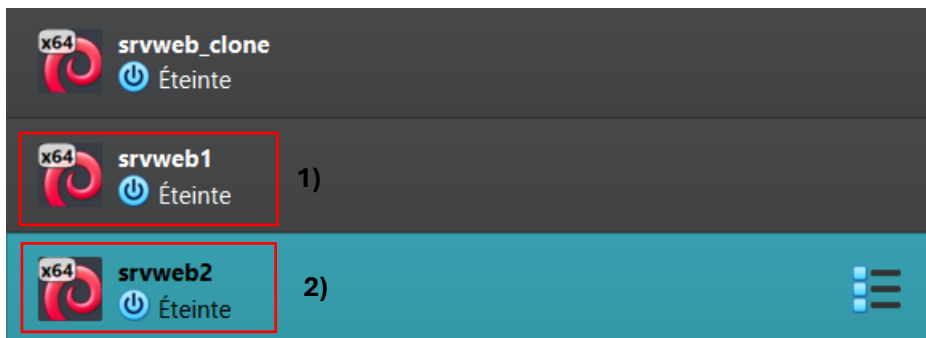
Pour **cloner la machine**, vous devez **ouvrir VirtualBox**, repérer **srvweb**, puis **effectuer un clic droit dessus**. Il suffit ensuite de **sélectionner “Cloner...”** pour **lancer l'assistant de clonage** :



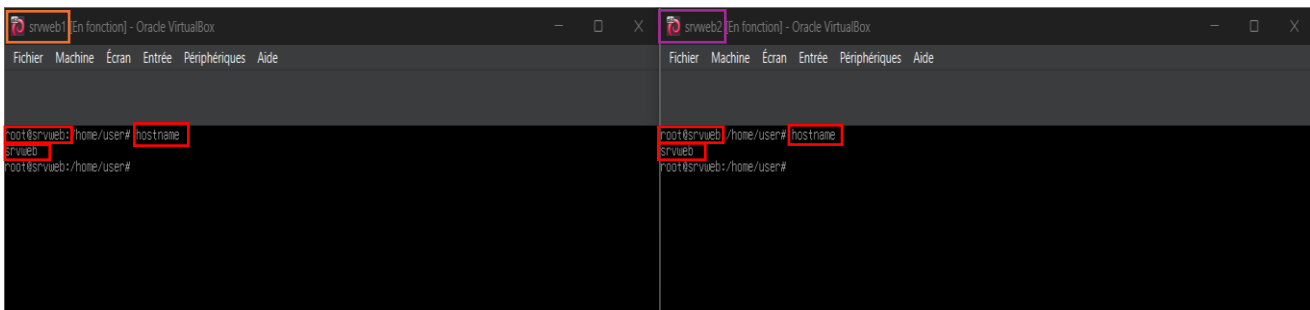
Une fois la **fenêtre de clonage ouverte**, **plusieurs paramètres** doivent être **configurés**. Tout d'abord, il faut **renommer la machine** (ici **srvweb\_clone**) et choisir l'**emplacement où la sauvegarder**. Ensuite, il est important de **sélectionner “Clone Intégral”** afin de dupliquer entièrement la machine source. Pour éviter tout conflit réseau, il faut également **“Générer de nouvelles adresses MAC pour les interfaces réseau”**. Une fois ces réglages effectués, un **simple clic** sur **“Finish”** permet de **finaliser l'opération** et de **lancer le clonage** de **srvweb\_clone** :

**/ ! \ Remarque : Il suffit ensuite de répéter cette procédure pour créer les deux serveurs du cluster : srvweb1 et srvweb2 / ! \**

Maintenant que **srvweb1** et **srvweb2** sont créés, il faut **modifier leur nom et leur adresse IP** pour qu'ils puissent fonctionner correctement dans le cluster :



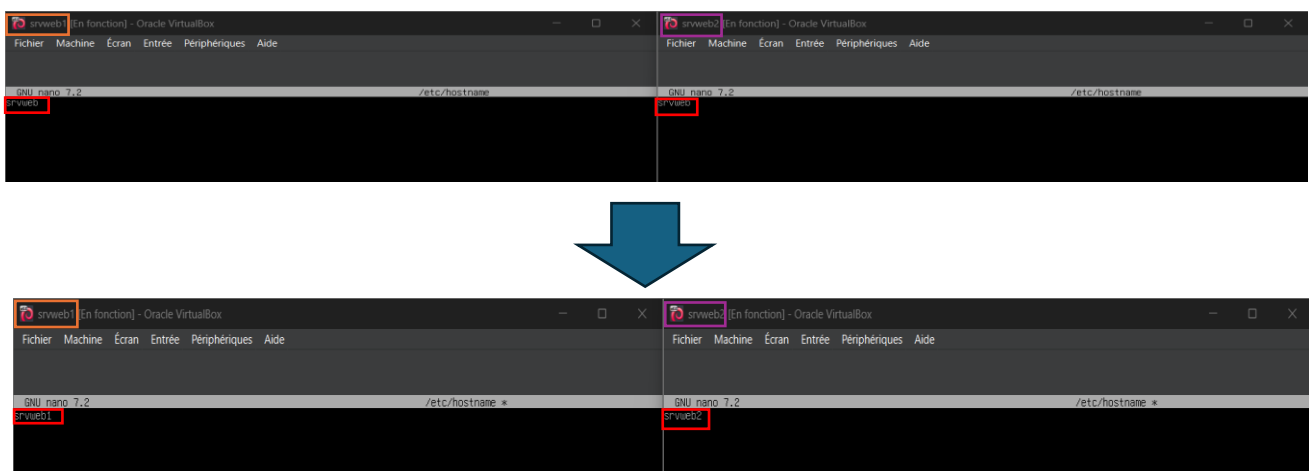
Après le **connage** et le **démarrage des machines**, on observe que le **nom d'hôte** reste inchangé. En **exécutant** la commande **hostname** qui est resté **srvweb** :



Nous allons **modifier** les fichiers **hostname**, situés dans le répertoire **/etc/** sur les serveurs **srvweb1** et **srvweb2**, en utilisant la commande **nano** :

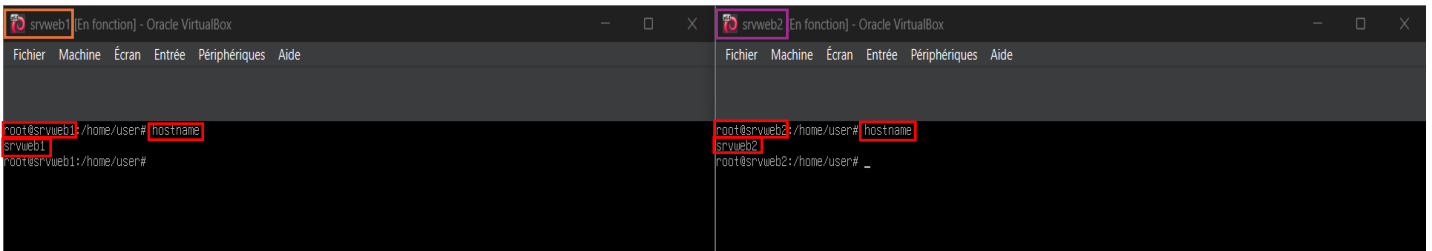
🔧 **nano /etc/hostname** : permettant de **modifier le nom d'hôte** de la **machine**

**Modification des fichiers hostnames :**



Pour enregistrer les modifications des fichiers de configuration, appuyez sur **CTRL+O**, puis **Entrée** pour valider les modifications. Enfin **CTRL+X** pour quitter.

Afin de prendre en compte les modifications, il faut redémarrer les machines virtuelles srvweb1 et srvweb2, soit en exécutant la commande **"#reboot"**, soit en les éteignant puis en les rallumant manuellement depuis VirtualBox. Les noms d'hôte ont bien été modifiés, comme le confirme l'exécution de la commande **hostname** sur chaque serveur :



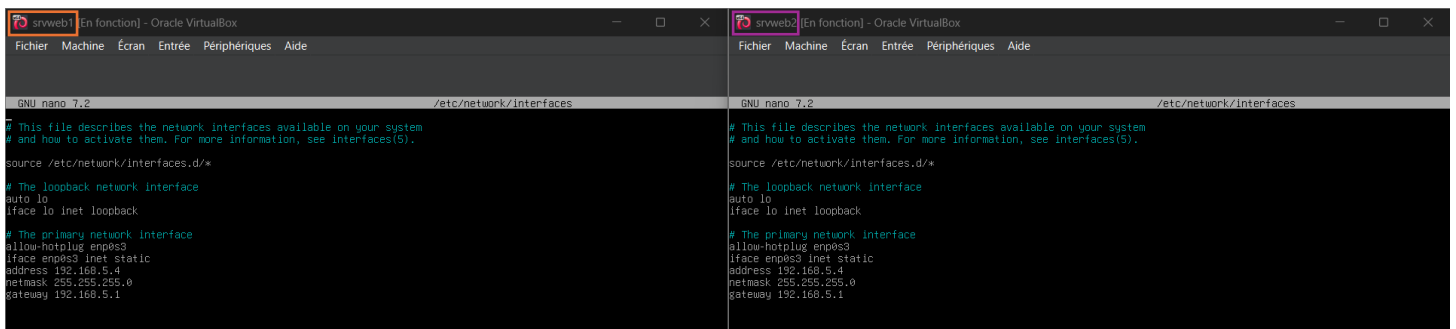
```
root@srvweb1:/home/user# hostname
srvweb1
root@srvweb1:/home/user#

root@srvweb2:/home/user# hostname
srvweb2
root@srvweb2:/home/user#
```

Il est nécessaire également de modifier le fichier de configuration des interfaces, situé dans le répertoire **/etc/network/** sur les serveurs **srvweb1** et **srvweb2**, en utilisant la commande **nano** :

 **nano /etc/network/interfaces** : permet de configurer les paramètres réseau (adresse IP, masque réseau, DNS)

### Modification des fichiers de configuration des interfaces :



```
GNU nano 7.2 /etc/network/interfaces

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
address 192.168.5.4
netmask 255.255.255.0
gateway 192.168.5.1

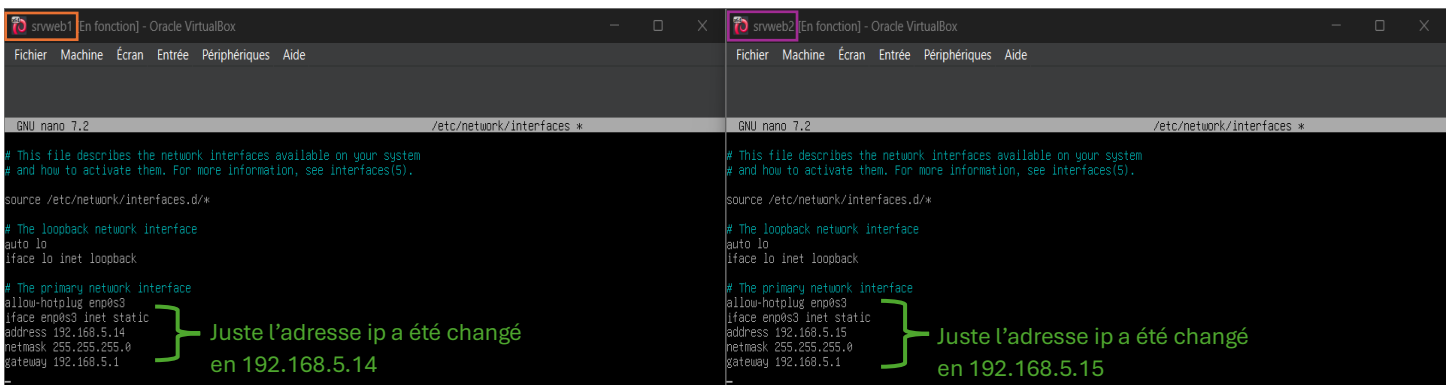
GNU nano 7.2 /etc/network/interfaces

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
address 192.168.5.4
netmask 255.255.255.0
gateway 192.168.5.1
```



```
GNU nano 7.2 /etc/network/interfaces *

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
address 192.168.5.14
netmask 255.255.255.0
gateway 192.168.5.1
} Juste l'adresse ip a été changé
  en 192.168.5.14

GNU nano 7.2 /etc/network/interfaces *

# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug enp0s3
iface enp0s3 inet static
address 192.168.5.15
netmask 255.255.255.0
gateway 192.168.5.1
} Juste l'adresse ip a été changé
  en 192.168.5.15
```

Après le redémarrage du service réseau avec la commande “**#systemctl restart networking**”, nous avons **vérifié les nouvelles configurations** en exécutant “**#ip a**”, ce qui a confirmé l'application des changements d'adresse IP. Les **serveurs srvweb1** et **srvweb2** peuvent désormais **communiquer entre eux**, comme le montrent les **tests de ping réussis** :

```
root@srvweb1:/home/user# systemctl restart networking
root@srvweb1:/home/user# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:03:85:08 brd ff:ff:ff:ff:ff:ff
    inet 192.168.5.14/24 brd 192.168.5.255 scope global enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe03:8508/64 scope link
        valid_lft forever preferred_lft forever
root@srvweb1:/home/user# ping 192.168.5.15
PING 192.168.5.15 (192.168.5.15) 56(84) bytes of data.
64 bytes from 192.168.5.15: icmp_seq=1 ttl=64 time=65.3 ms
64 bytes from 192.168.5.15: icmp_seq=2 ttl=64 time=1.51 ms
64 bytes from 192.168.5.15: icmp_seq=3 ttl=64 time=1.60 ms
64 bytes from 192.168.5.15: icmp_seq=4 ttl=64 time=1.65 ms
^C
--- 192.168.5.15 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3043ms
rtt min/avg/max/mdev = 1.510/17.514/65.239/27.598 ms
root@srvweb1:/home/user#
```

Ping de srvweb1 à srvweb2 = ok

```
root@srvweb2:/home/user# systemctl restart networking
root@srvweb2:/home/user# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:a2:b2:d2 hrd ff:ff:ff:ff:ff:ff
    inet 192.168.5.15/24 brd 192.168.5.255 scope global enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fea2:b2d2/64 scope link
        valid_lft forever preferred_lft forever
root@srvweb2:/home/user# ping 192.168.5.14
PING 192.168.5.14 (192.168.5.14) 56(84) bytes of data.
64 bytes from 192.168.5.14: icmp_seq=1 ttl=64 time=89.8 ms
64 bytes from 192.168.5.14: icmp_seq=2 ttl=64 time=1.11 ms
64 bytes from 192.168.5.14: icmp_seq=3 ttl=64 time=1.31 ms
64 bytes from 192.168.5.14: icmp_seq=4 ttl=64 time=1.25 ms
^C
--- 192.168.5.14 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3043ms
rtt min/avg/max/mdev = 1.111/23.378/89.834/38.368 ms
root@srvweb2:/home/user#
```

Ping de srvweb2 à srvweb1 = ok

## ETAPE 2 : MISE À JOUR DES FICHIERS DE CONFIGURATION APACHE 2.0 DES 2 SERVEURS WEB :

Nous avons **vérifié** la configuration du fichier **/etc/apache2/sites-available/ville-abymes.conf** sur les serveurs **srvweb1** et **srvweb2**, et **aucune modification n'a été nécessaire**. Les paramètres essentiels, tels que **ServerName**, **ServerAlias**, **DocumentRoot**, ainsi que la **gestion des logs**, sont **correctement configurés**. De plus, **l'authentification** est bien en place, garantissant que **seuls les utilisateurs autorisés peuvent accéder au site via le port 8080** :

```
GNU nano 7.2 /etc/apache2/sites-available/villeabymes.conf
<VirtualHost *:8080>
    # 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 www.example.com

    ServerAdmin webmaster@ville-abymes.fr
    ServerName ville-abymes.fr
    ServerAlias www.ville-abymes.fr
    DocumentRoot /var/www/villeabymes

    <Directory /var/www/villeabymes>
        AuthType Basic
        AuthName "Accès privé -site ville des Abymes"
        AuthBasicProvider file
        AuthUserFile /etc/apache2/.htusers
        Require valid-user
    </Directory>

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/villeabymes/error.log
    CustomLog ${APACHE_LOG_DIR}/villeabymes/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf
</VirtualHost>
```

Lecture de 39 lignes

```
GNU nano 7.2 /etc/apache2/sites-available/villeabymes.conf
<VirtualHost *:8080>
    # 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 www.example.com

    ServerAdmin webmaster@ville-abymes.fr
    ServerName ville-abymes.fr
    ServerAlias www.ville-abymes.fr
    DocumentRoot /var/www/villeabymes

    <Directory /var/www/villeabymes>
        AuthType Basic
        AuthName "Accès privé -site ville des Abymes"
        AuthBasicProvider file
        AuthUserFile /etc/apache2/.htusers
        Require valid-user
    </Directory>

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/villeabymes/error.log
    CustomLog ${APACHE_LOG_DIR}/villeabymes/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf
</VirtualHost>
```

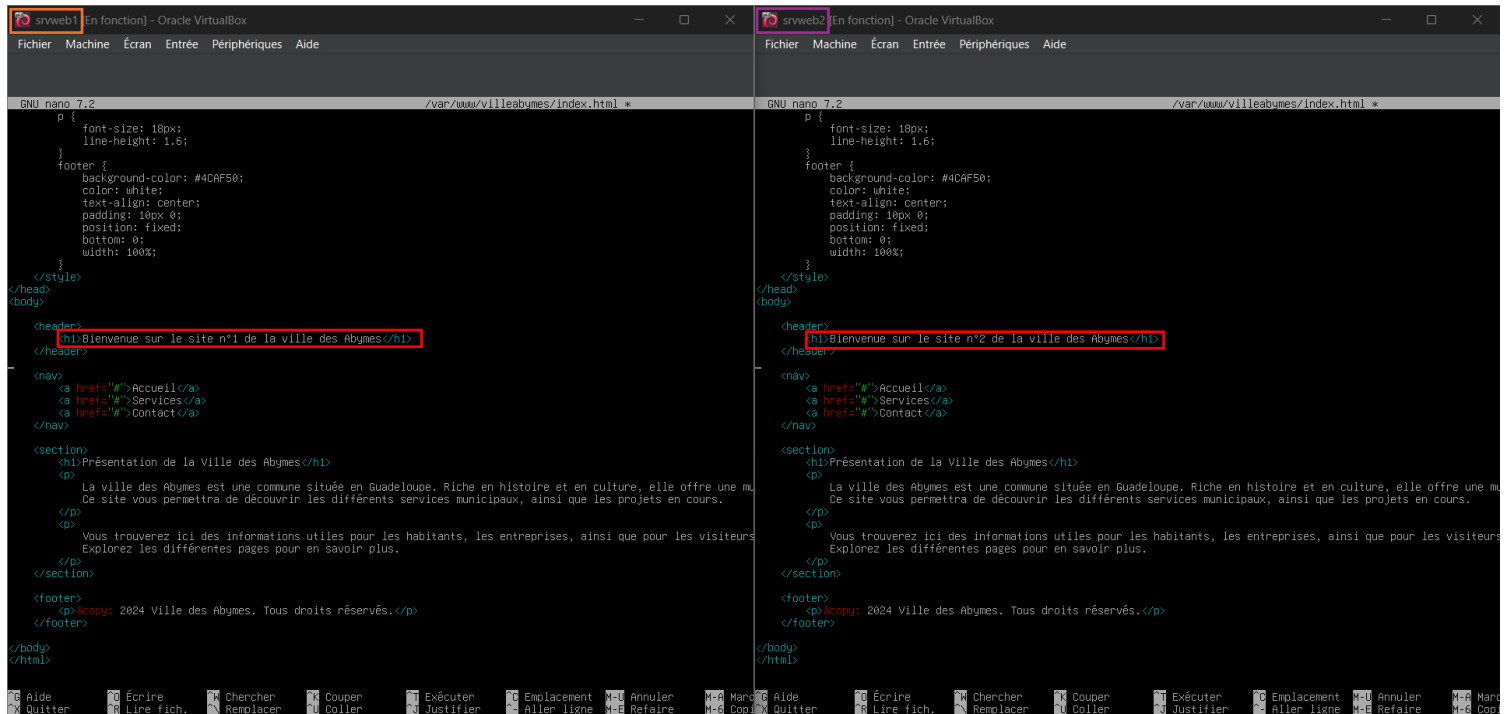
Lecture de 39 lignes

Voici la **modification** du **fichier** `/var/www/villeabymes/index.html` qui a **été effectuée** **respectivement** sur chaque **serveur web** en leur **attribuant** un **numéro** :

🚦 Sur **srvweb1**, la page affiche : **“Bienvenue sur le site n°1 de la ville des Abymes”**

🚦 Sur **srvweb2**, la page affiche : **“Bienvenue sur le site n°2 de la ville des Abymes”**

Cette **distinction** permet **d’identifier clairement** quel **serveur répond** aux **requêtes**. Elle sera particulièrement **utile** lors des **tests** de **basculement** du **cluster** afin de **vérifier** que la **redirection** vers le **serveur actif** s’effectue **correctement** en **cas de panne simulée**.

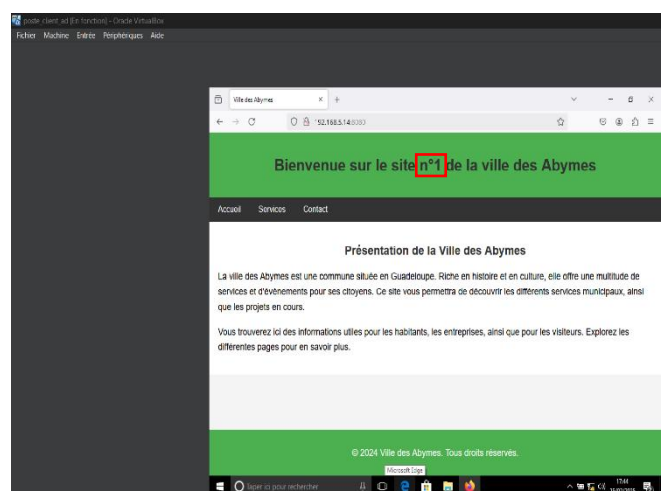
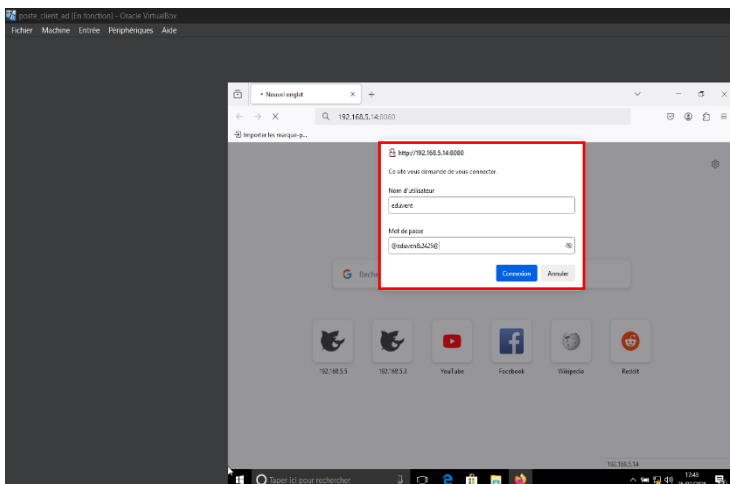


```
GNU nano 7.2 /var/www/villeabymes/index.html
p {
  font-size: 18px;
  line-height: 1.6;
}
footer {
  background-color: #4CAF50;
  color: white;
  text-align: center;
  padding: 10px 0;
  position: fixed;
  bottom: 0;
  width: 100%;
}
</style>
</head>
<body>
<header>
<h1>Bienvenue sur le site n°1 de la ville des Abymes</h1>
</header>
<nav>
<a href="#">Accueil</a>
<a href="#">Services</a>
<a href="#">Contact</a>
</nav>
<section>
<h2>Présentation de la Ville des Abymes</h2>
<p>
    La ville des Abymes est une commune située en Guadeloupe. Riche en histoire et en culture, elle offre une multitude de services et d'événements pour ses citoyens. Ce site vous permettra de découvrir les différents services municipaux, ainsi que les projets en cours.
  </p>
<p>
    Vous trouverez ici des informations utiles pour les habitants, les entreprises, ainsi que pour les visiteurs. Explorez les différentes pages pour en savoir plus.
  </p>
</section>
<footer>
<p>©2024 Ville des Abymes. Tous droits réservés.</p>
</footer>
</body>
</html>
```

```
GNU nano 7.2 /var/www/villeabymes/index.html
p {
  font-size: 18px;
  line-height: 1.6;
}
footer {
  background-color: #4CAF50;
  color: white;
  text-align: center;
  padding: 10px 0;
  position: fixed;
  bottom: 0;
  width: 100%;
}
</style>
</head>
<body>
<header>
<h1>Bienvenue sur le site n°2 de la ville des Abymes</h1>
</header>
<nav>
<a href="#">Accueil</a>
<a href="#">Services</a>
<a href="#">Contact</a>
</nav>
<section>
<h2>Présentation de la Ville des Abymes</h2>
<p>
    La ville des Abymes est une commune située en Guadeloupe. Riche en histoire et en culture, elle offre une multitude de services et d'événements pour ses citoyens. Ce site vous permettra de découvrir les différents services municipaux, ainsi que les projets en cours.
  </p>
<p>
    Vous trouverez ici des informations utiles pour les habitants, les entreprises, ainsi que pour les visiteurs. Explorez les différentes pages pour en savoir plus.
  </p>
</section>
<footer>
<p>©2024 Ville des Abymes. Tous droits réservés.</p>
</footer>
</body>
</html>
```

**Test d'accès au site hébergé sur srvweb1 :**

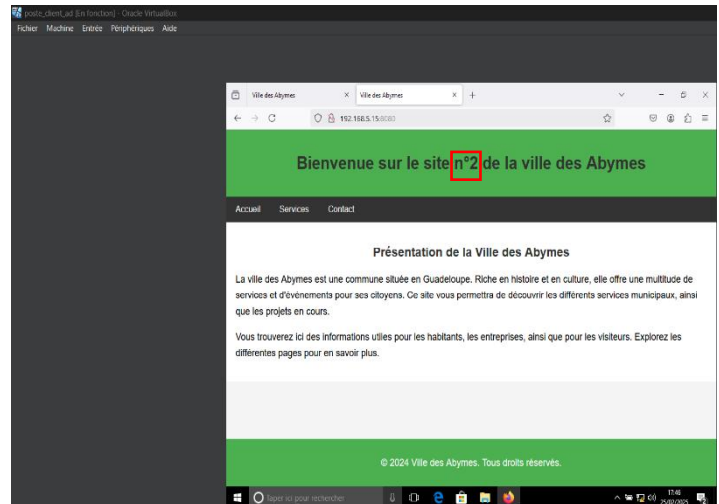
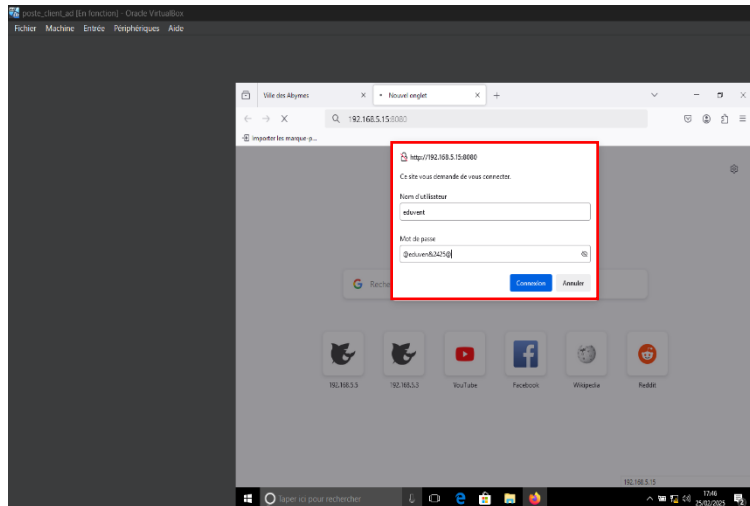
Après avoir **effacé l'historique de navigation**, l'accès au **site n°1** via le **port 8080** avec **authentification** s'est **déroulé sans encombre** !





## Test d'accès au site hébergés sur srvweb2 :

Suite à la suppression de l'historique de navigation, l'accès au site n°2 s'est fait via le port 8080 avec authentification sans problème !

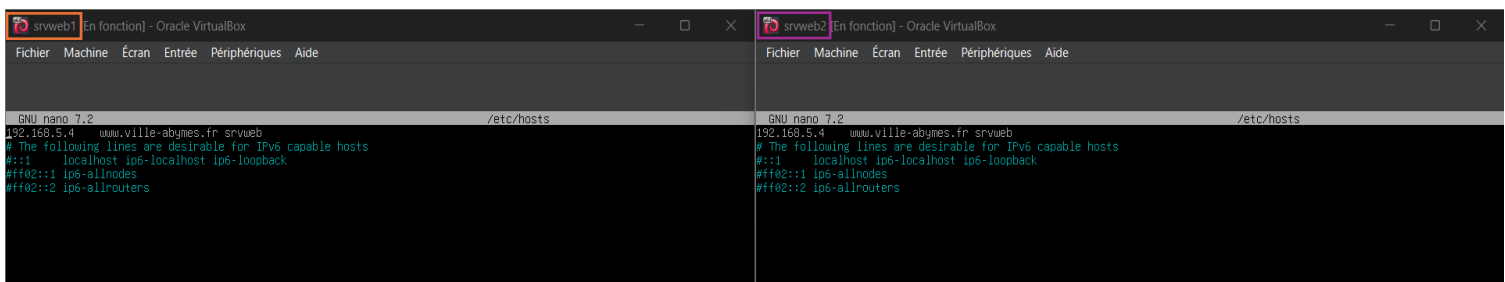


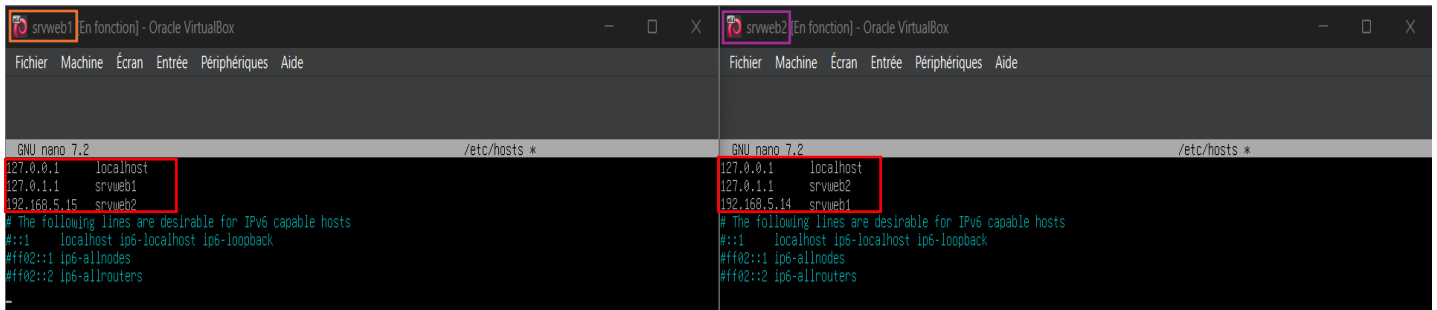
## ETAPE 3 : CRÉATION ET CONFIGURATION DU CLUSTER DES SERVEURS WEB AVEC HEARTBEAT :

Pour garantir une **communication fluide** entre les **serveurs** du **cluster** avec **Heartbeat**, il est **essentiel d'ajuster** le **fichier de correspondance des noms d'hôte (/etc/hosts)**. Il permet d'**assurer une résolution correcte des adresses IP** et d'**éviter toute ambiguïté** dans l'**identification des machines**, **sans dépendre d'un serveur DNS externe**. Nous **utiliserons la commande** :

🔧 **nano /etc/hosts** : permet de **mettre à jour** les **correspondances locales** entre **adresses IP** et **noms d'hôte**, afin que la **résolution des noms** soit **correcte après le changement de nom d'hôte**

### Modification des fichiers hosts :





Dans cette configuration spécifique au cluster **Heartbeat**, le fichier **/etc/hosts** de **chaque serveur** est mis à jour pour **garantir** que **chaque machine** puisse **reconnaître** son **homologue** par son nom d'hôte :

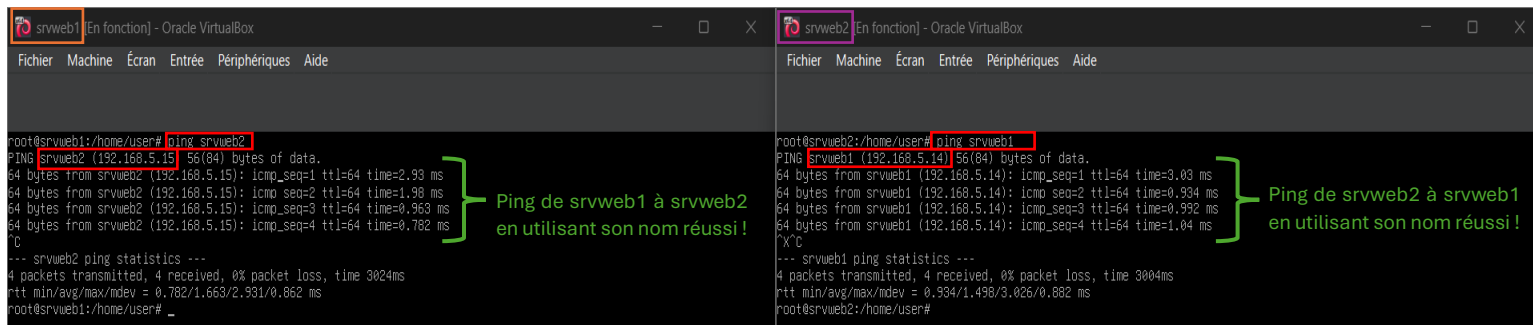
- **Sur srweb1 :**

- ✚ **127.0.0.1 localhost** → Définit l'adresse locale de la machine.
- ✚ **127.0.1.1 srweb1** → Associe le nom d'hôte **srweb1** à l'interface locale.
- ✚ **192.168.5.15 srweb2** → Permet à **srweb1** d'identifier **srweb2** via son IP.

- **Sur srweb2 :**

- ✚ **127.0.0.1 localhost** → Définit l'adresse locale de la machine.
- ✚ **127.0.1.1 srweb2** → Associe le nom d'hôte **srweb2** à l'interface locale.
- ✚ **192.168.5.14 srweb1** → Permet à **srweb2** d'identifier **srweb1** via son IP.

En appliquant cette configuration, les deux serveurs peuvent communiquer entre eux à l'aide de leurs **noms d'hôte**, ce qui est **essentiel** pour **Heartbeat**, qui **privilégie** les **noms** plutôt que les **adresses IP** dans la **gestion du cluster**.



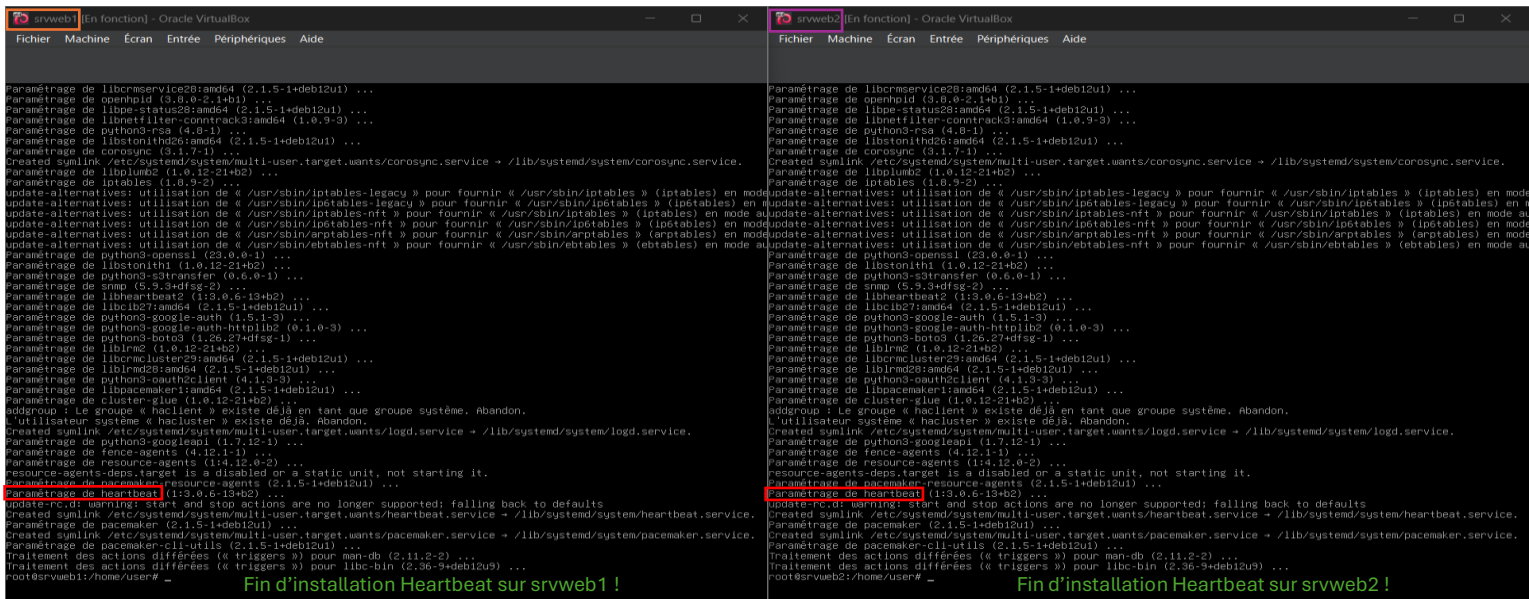
## Mise à jour des paquets et installation de Heartbeat :

Pour assurer un environnement à jour et stable avant l'installation de Heartbeat, les paquets du système doivent être mis à jour sur chaque serveur. Voici les étapes effectuées :

🔧 **apt update** : Met à jour la liste des paquets disponibles en récupérant les dernières versions depuis les dépôts officiels.

🔧 **apt upgrade** : Installe les mises à jour disponibles pour les paquets déjà installés, incluant corrections de sécurité et améliorations.

🔧 **apt install heartbeat** : Installe le paquet **Heartbeat**, essentiel pour configurer le cluster et assurer la haute disponibilité des services web.



```
Paramétrage de libcrmservice28:amd64 (2.1.5-1+deb12u1) ...
Paramétrage de openhpid (3.0.0-2.1+b1) ...
Paramétrage de libpe-status28:amd64 (2.1.5-1+deb12u1) ...
Paramétrage de libnetfilter-conntrack3:amd64 (1.0.9-3) ...
Paramétrage de python3-rsa (4.0-1) ...
Paramétrage de libstonithd26:amd64 (2.1.5-1+deb12u1) ...
Paramétrage de corosync (3.1.7-1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/corosync.service → /lib/systemd/system/corosync.service.
Paramétrage de libplumb2 (1.0.12-21+b2) ...
Paramétrage de iptables (1.0.9-2) ...
update-alternatives: utilisation de « /usr/sbin/iptables-legacy » pour fournir « /usr/sbin/iptables » (iptables) en mode automatique
update-alternatives: utilisation de « /usr/sbin/iptables-nft » pour fournir « /usr/sbin/iptables » (iptables) en mode automatique
update-alternatives: utilisation de « /usr/sbin/iptables-nft » pour fournir « /usr/sbin/iptables » (iptables) en mode automatique
update-alternatives: utilisation de « /usr/sbin/iptables-nft » pour fournir « /usr/sbin/iptables » (iptables) en mode automatique
update-alternatives: utilisation de « /usr/sbin/iptables-nft » pour fournir « /usr/sbin/iptables » (iptables) en mode automatique
Paramétrage de python3-openssl (23.0.0-1) ...
Paramétrage de libstonith1 (1.0.12-21+b2) ...
Paramétrage de python3-SshTransfer (0.6.0-1) ...
Paramétrage de snmp (5.9.3+dfsg-2) ...
Paramétrage de libheartbeat2 (1:3.0.6-13+b2) ...
Paramétrage de libicb27amd64 (2.1.5-1+deb12u1) ...
Paramétrage de python3-google-auth (1.5.1-3) ...
Paramétrage de python3-google-auth-httplib2 (0.1.0-3) ...
Paramétrage de python3-botocore (1.35.27+dfsg-1) ...
Paramétrage de liblrm2 (1.0.12-21+b2) ...
Paramétrage de libcrmcluster29:amd64 (2.1.5-1+deb12u1) ...
Paramétrage de liblrm20:amd64 (2.1.5-1+deb12u1) ...
Paramétrage de python3-oauth2client (4.1.3-3) ...
Paramétrage de libpacemaker1:amd64 (2.1.5-1+deb12u1) ...
Paramétrage de cluster-glue (1.0.12-21+b2) ...
addgroup : Le groupe « hacluster » existe déjà en tant que groupe système. Abandon.
Created symlink /etc/systemd/system/multi-user.target.wants/logd.service → /lib/systemd/system/logd.service.
Paramétrage de python3-googleleap1 (1.7.12-1) ...
Paramétrage de fence-agents (4.12.1-1) ...
Paramétrage de resource-agents (1:4.12.0-2) ...
resource-agents-deps.target is a disabled or a static unit, not starting it.
Paramétrage de pacemaker-resource-agents (2.1.5-1+deb12u1) ...
Paramétrage de heartbeat (1:3.0.6-13+b2) ...
update-rc.d warning: start and stop actions are no longer supported; falling back to defaults
Created symlink /etc/systemd/system/multi-user.target.wants/heartbeat.service → /lib/systemd/system/heartbeat.service.
Paramétrage de pacemaker (2.1.5-1+deb12u1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/pacemaker.service → /lib/systemd/system/pacemaker.service.
Paramétrage de pacemaker-cil-utils (2.1.5-1+deb12u1) ...
Traitement des actions différées (« triggers ») pour man-db (2.11.2-2) ...
Traitement des actions différées (« triggers ») pour libc-bin (2.36-9+deb12u9) ...
root@srweb1:/home/user#
```

Fin d'installation Heartbeat sur srweb1 !

## Configuration du cluster avec Heartbeat :

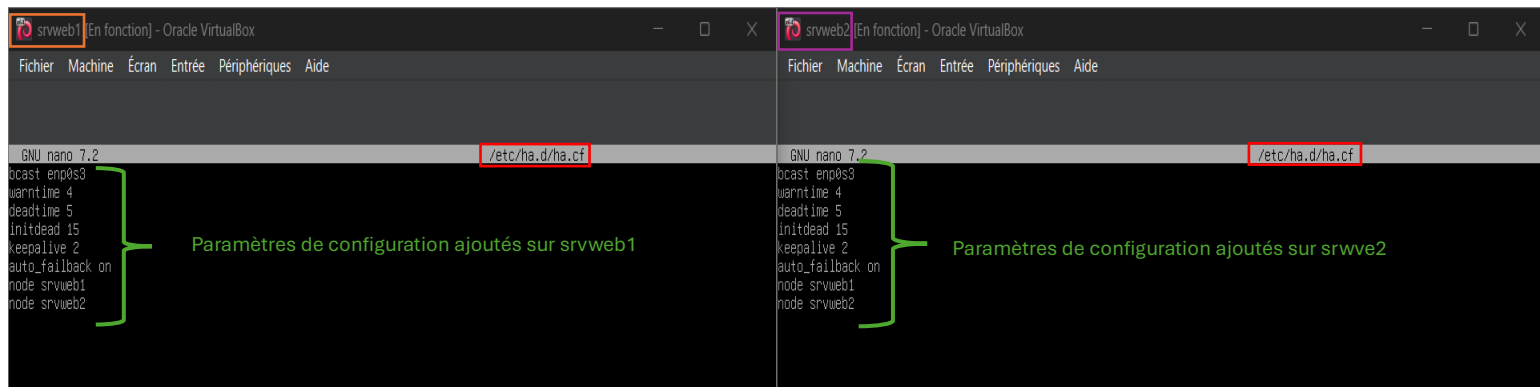
Seuls 3 fichiers de configuration, situés dans le répertoire “/etc/ha.d/”, sont nécessaires pour déployer un cluster de serveurs autour d'une adresse IP virtuelle. Il est impératif de les créer et de les paramétrer correctement :

🔧 **nano /etc/ha.d/ha.cf** : Définit les paramètres de communication et de gestion du cluster.

🔧 **nano /etc/ha.d/haresources** : Spécifie quelle ressource (adresse IP virtuelle et service Apache) sera gérée par le cluster.

🔧 **nano /etc/ha.d/authkeys** : Contient la clé d'authentification pour sécuriser les échanges entre les nœuds.

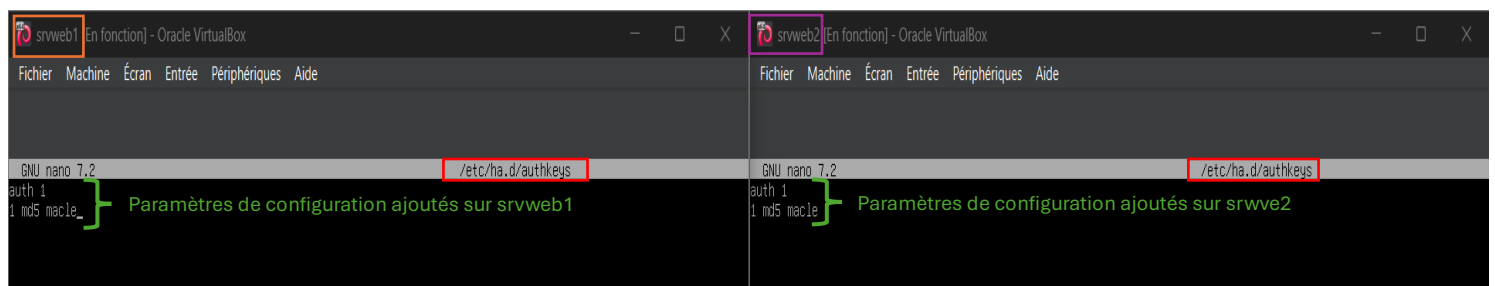
## Création du fichier “ha.cf” :



## Explication des différents paramètres du fichier de configuration “ha.cf” :

- ✚ **bcast enp0s3** : Utilise le **mode broadcast** sur l'**interface** réseau **enp0s3** pour la **communication** entre les **nœuds du cluster**.
- ✚ **warntime 4** : Définit le **délai** (en secondes) avant **qu’une alerte de panne ne soit générée**.
- ✚ **deadtime 5** : **Temps d’attente** (en secondes) avant **qu’un serveur soit considéré comme hors-ligne en l’absence de réponse**.
- ✚ **initdead 15** : **Temps d’initialisation** (en secondes) après le démarrage d’un nœud avant la **première vérification de disponibilité**.
- ✚ **keepalive 2** : **Intervalle** (en secondes) entre chaque envoi de signal de vie (heartbeat) entre les nœuds.
- ✚ **auto\_failback on** : Active le **retour automatique** des services vers le nœud principal lorsque celui-ci redevient disponible.
- ✚ **node srweb1** : Déclare **srweb1** comme **l’un des nœuds du cluster**.
- ✚ **node srweb2** : Déclare **srweb2** comme **l’autre nœud du cluster**.

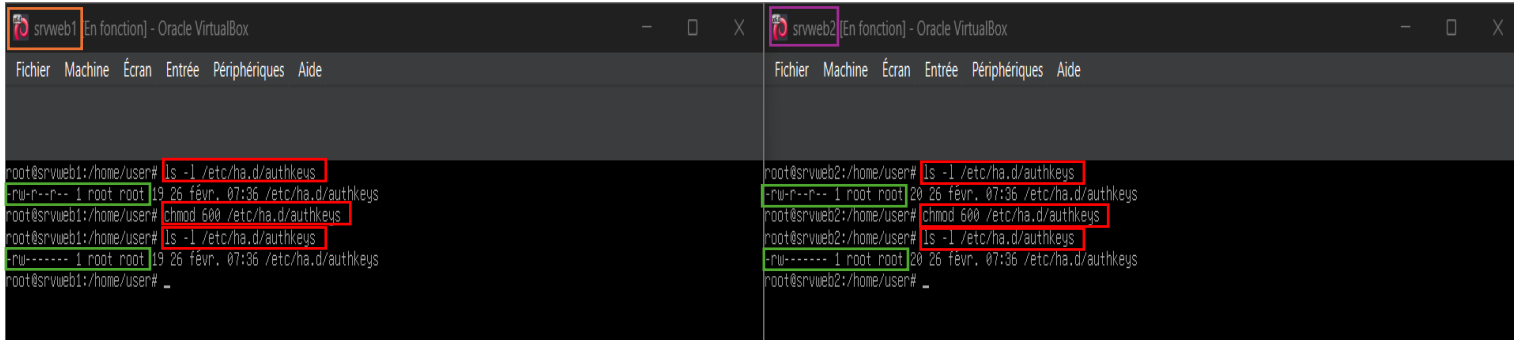
## Création du fichier “authkeys” :



## Explication des différents paramètres du fichier de configuration “authkeys” :

- ✚ **auth 1** : Définit le **type d’authentification** utilisé par **Heartbeat** pour **sécuriser les échanges entre les nœuds du cluster**.
- ✚ **1 md5 mac1e** : Spécifie que **l’authentification repose** sur **l’algorithme MD5** et utilise **une clé de chiffrement (mac1e)** pour **sécuriser la communication entre les serveurs du cluster**.

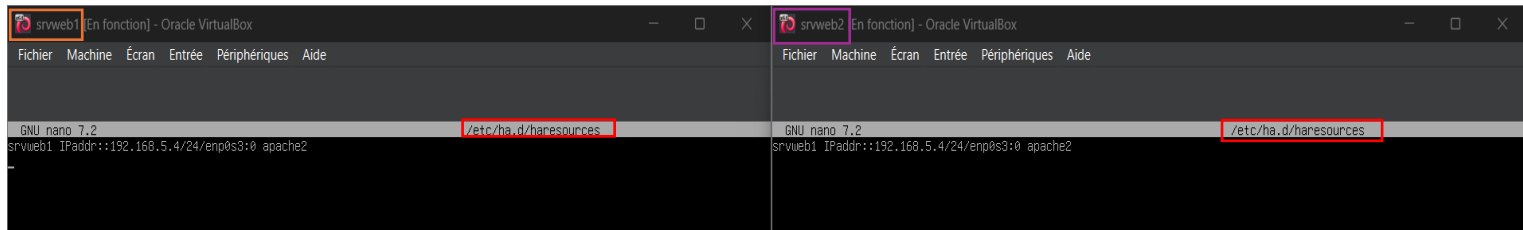
**Pour des raisons de sécurité, les droits d'accès à ce fichier (authkeys) sont restreints à l'aide de la commande `chmod 600 /etc/ha.d/authkeys`. Cette modification limite l'accès au fichier à l'utilisateur root uniquement, assurant ainsi la sécurisation des échanges entre les nœuds du cluster Heartbeat. Une vérification avec la commande `ls -l` permet de confirmer l'application correcte des permissions :**



```
root@srweb1:/home/user# ls -l /etc/ha.d/authkeys
-rw-r--r-- 1 root root 19 26 févr. 07:36 /etc/ha.d/authkeys
root@srweb1:/home/user# chmod 600 /etc/ha.d/authkeys
root@srweb1:/home/user# ls -l /etc/ha.d/authkeys
-rw----- 1 root root 19 26 févr. 07:36 /etc/ha.d/authkeys
root@srweb1:/home/user# _

root@srweb2:/home/user# ls -l /etc/ha.d/authkeys
-rw-r--r-- 1 root root 20 26 févr. 07:36 /etc/ha.d/authkeys
root@srweb2:/home/user# chmod 600 /etc/ha.d/authkeys
root@srweb2:/home/user# ls -l /etc/ha.d/authkeys
-rw----- 1 root root 20 26 févr. 07:36 /etc/ha.d/authkeys
root@srweb2:/home/user# _
```

## Création du fichier “haresources” :



```
GNU nano 7.2 /etc/ha.d/haresources
srweb1 IPaddr::192.168.5.4/24/enp0s3:0 apache2

GNU nano 7.2 /etc/ha.d/haresources
srweb2 IPaddr::192.168.5.4/24/enp0s3:0 apache2
```

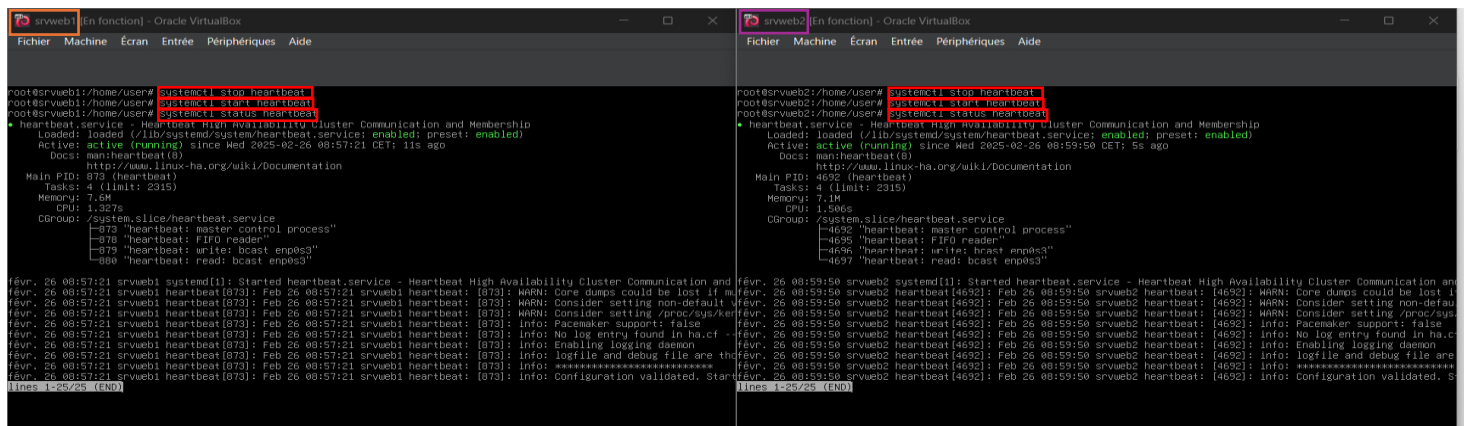
## Explication des différents paramètres du fichier de configuration “haresources” :

- ✚ **SRVWEB1** : Définit le nœud principal du cluster.
- ✚ **IPaddr ::192.168.X.4/24/enp0s3 :0** : Configure une adresse IP virtuelle associée au cluster et attribuée à l'interface réseau enp0s3.
- ✚ **apache2** : Indique que le service Apache2 sera géré par le cluster et basculera automatiquement en cas de défaillance du nœud principal.

## Redémarrage et vérification de l'état de Heartbeat :

Pour appliquer les modifications effectuées dans les fichiers de configuration de Heartbeat, il est nécessaire d'exécuter les commandes suivantes sur chaque serveur :

- ✚ **systemctl stop heartbeat** : Arrête le service Heartbeat.
- ✚ **systemctl start heartbeat** : Relance le service Heartbeat.
- ✚ **systemctl status heartbeat** : Vérifie que le service est bien activé et fonctionne correctement.



```
root@srweb1:/home/user# systemctl stop heartbeat
root@srweb1:/home/user# systemctl start heartbeat
root@srweb1:/home/user# systemctl status heartbeat
● heartbeat.service - Heartbeat High Availability Cluster Communication and Membership
   Loaded: loaded (/lib/systemd/system/heartbeat.service; enabled; preset: enabled)
   Active: active (running) since Wed 2025-02-26 08:57:21 CET; 11s ago
     Docs: man:heartbeat(8)
           http://www.linux-ha.org/wiki/Documentation
   Main PID: 879 (heartbeat)
    Tasks: 4 (limit: 2315)
   Memory: 7.6M
      CPU: 1.327s
   CGroup: /system.slice/heartbeat.service
           └─879 "heartbeat: master control process"
             └─878 "heartbeat: FIFO reader"
               └─879 "heartbeat: write: bcst enp0s3"
                 └─880 "heartbeat: read: bcst enp0s3"

févr. 26 08:57:21 srweb1 systemd[1]: Started heartbeat.service - Heartbeat High Availability Cluster Communication and Membership.
févr. 26 08:57:21 srweb1 heartbeat[879]: Feb 26 08:57:21 srweb1 heartbeat: [879]: WARN: Core dumps could be lost if m...
févr. 26 08:57:21 srweb1 heartbeat[879]: Feb 26 08:57:21 srweb1 heartbeat: [879]: WARN: Consider setting /proc/sys/ker...
févr. 26 08:57:21 srweb1 heartbeat[879]: Feb 26 08:57:21 srweb1 heartbeat: [879]: Info: Pacemaker support: false
févr. 26 08:57:21 srweb1 heartbeat[879]: Feb 26 08:57:21 srweb1 heartbeat: [879]: Info: No log entry found in ha.cf
févr. 26 08:57:21 srweb1 heartbeat[879]: Feb 26 08:57:21 srweb1 heartbeat: [879]: Info: Enabling logging daemon
févr. 26 08:57:21 srweb1 heartbeat[879]: Feb 26 08:57:21 srweb1 heartbeat: [879]: Info: logfile and debug file are t...
févr. 26 08:57:21 srweb1 heartbeat[879]: Feb 26 08:57:21 srweb1 heartbeat: [879]: Info: Configuration validated, Start...

Linux 4.25.25 (GNU)

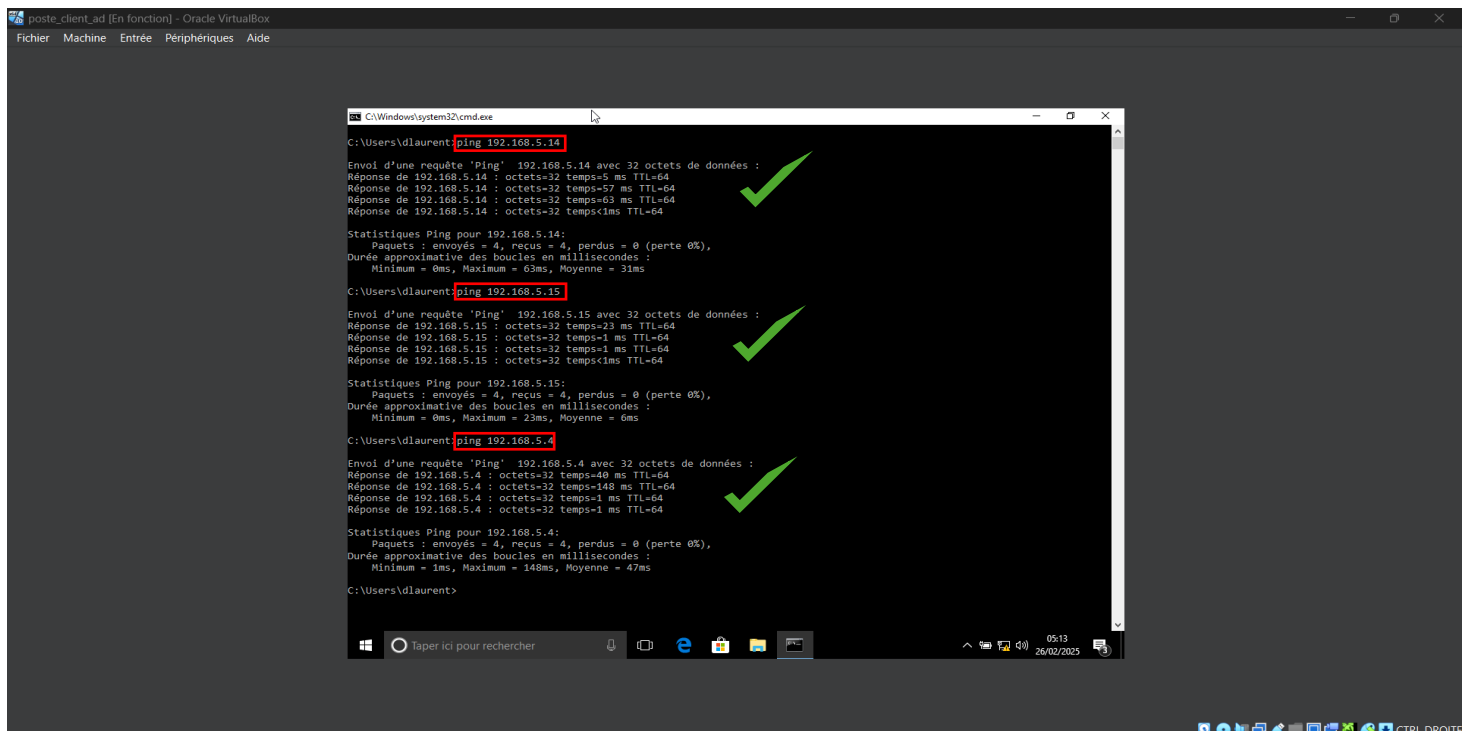
root@srweb2:/home/user# systemctl stop heartbeat
root@srweb2:/home/user# systemctl start heartbeat
root@srweb2:/home/user# systemctl status heartbeat
● heartbeat.service - Heartbeat High Availability Cluster Communication and Membership
   Loaded: loaded (/lib/systemd/system/heartbeat.service; enabled; preset: enabled)
   Active: active (running) since Wed 2025-02-26 08:59:50 CET; 5s ago
     Docs: man:heartbeat(8)
           http://www.linux-ha.org/wiki/Documentation
   Main PID: 4692 (heartbeat)
    Tasks: 4 (limit: 2315)
   Memory: 7.1M
      CPU: 1.546s
   CGroup: /system.slice/heartbeat.service
           └─4692 "heartbeat: master control process"
             └─4695 "heartbeat: FIFO reader"
               └─4696 "heartbeat: write: bcst enp0s3"
                 └─4697 "heartbeat: read: bcst enp0s3"

févr. 26 08:59:50 srweb2 systemd[1]: Started heartbeat.service - Heartbeat High Availability Cluster Communication and Membership.
févr. 26 08:59:50 srweb2 heartbeat[4692]: Feb 26 08:59:50 srweb2 heartbeat: [4692]: WARN: Core dumps could be lost if...
févr. 26 08:59:50 srweb2 heartbeat[4692]: Feb 26 08:59:50 srweb2 heartbeat: [4692]: WARN: Consider setting /proc/sys/ker...
févr. 26 08:59:50 srweb2 heartbeat[4692]: Feb 26 08:59:50 srweb2 heartbeat: [4692]: Info: Pacemaker support: false
févr. 26 08:59:50 srweb2 heartbeat[4692]: Feb 26 08:59:50 srweb2 heartbeat: [4692]: Info: No log entry found in ha.cf
févr. 26 08:59:50 srweb2 heartbeat[4692]: Feb 26 08:59:50 srweb2 heartbeat: [4692]: Info: Enabling logging daemon
févr. 26 08:59:50 srweb2 heartbeat[4692]: Feb 26 08:59:50 srweb2 heartbeat: [4692]: Info: logfile and debug file are...
févr. 26 08:59:50 srweb2 heartbeat[4692]: Feb 26 08:59:50 srweb2 heartbeat: [4692]: Info: Configuration validated, Start...
```

Le service **Heartbeat** est **enabled** et **active**, ce qui **signifie** qu'il est **actuellement en cours d'exécution** et qu'il se lancera automatiquement au démarrage du système.

## ETAPE 4 : TEST DE LA HAUTE DISPONIBILITÉ DU SITE WEB DE LA VILLE DES ABYMES :

Depuis un **poste client sous Windows**, des **tests de communication** ont été effectués en exécutant la commande **ping** vers les **adresses IP** des **serveurs (192.168.5.14 et 192.168.5.15)** ainsi que **vers l'IP virtuelle du cluster (192.168.5.4)** :



```
C:\Windows\system32\cmd.exe
C:\Users\dlaurent>ping 192.168.5.14
Envoi d'une requête 'Ping' 192.168.5.14 avec 32 octets de données :
Réponse de 192.168.5.14 : octets=32 temps=5 ms TTL=64
Réponse de 192.168.5.14 : octets=32 temps=57 ms TTL=64
Réponse de 192.168.5.14 : octets=32 temps=63 ms TTL=64
Réponse de 192.168.5.14 : octets=32 temps<1ms TTL=64

Statistiques Ping pour 192.168.5.14:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
    Durée approximative des boucles en millisecondes :
        Minimum = 0ms, Maximum = 63ms, Moyenne = 31ms

C:\Users\dlaurent>ping 192.168.5.15
Envoi d'une requête 'Ping' 192.168.5.15 avec 32 octets de données :
Réponse de 192.168.5.15 : octets=32 temps=23 ms TTL=64
Réponse de 192.168.5.15 : octets=32 temps=1 ms TTL=64
Réponse de 192.168.5.15 : octets=32 temps=1 ms TTL=64
Réponse de 192.168.5.15 : octets=32 temps<1ms TTL=64

Statistiques Ping pour 192.168.5.15:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
    Durée approximative des boucles en millisecondes :
        Minimum = 0ms, Maximum = 23ms, Moyenne = 6ms

C:\Users\dlaurent>ping 192.168.5.4
Envoi d'une requête 'Ping' 192.168.5.4 avec 32 octets de données :
Réponse de 192.168.5.4 : octets=32 temps=40 ms TTL=64
Réponse de 192.168.5.4 : octets=32 temps=148 ms TTL=64
Réponse de 192.168.5.4 : octets=32 temps=1 ms TTL=64
Réponse de 192.168.5.4 : octets=32 temps=1 ms TTL=64

Statistiques Ping pour 192.168.5.4:
    Paquets : envoyés = 4, reçus = 4, perdus = 0 (perte 0%),
    Durée approximative des boucles en millisecondes :
        Minimum = 1ms, Maximum = 148ms, Moyenne = 47ms

C:\Users\dlaurent>
```

Les résultats montrent que **toutes les requêtes ping** ont été **correctement répondues**, confirmant ainsi la **connectivité réseau** entre le client et les **serveurs web**. Cela **valide** que les **configurations réseau** et **Heartbeat** sont **opérationnelles** et que le **cluster** est bien **accessible** via son **adresse IP virtuelle**.

### Test de la “Haute disponibilité” du cluster Heartbeat :

#### Déroulement du scénario

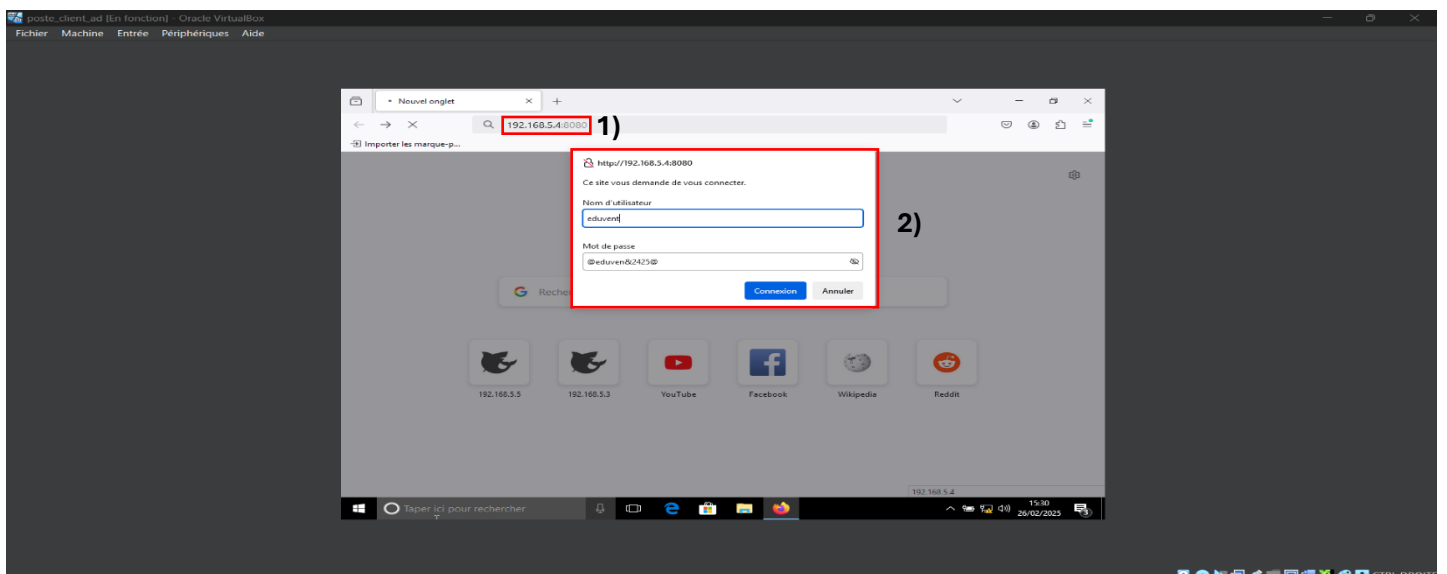
L'**objectif** de ce test est de **vérifier** que le **mécanisme** de **haute disponibilité** du **cluster Heartbeat** fonctionne correctement. Pour cela, nous allons **accéder** au **site web** de la **ville des Abymes** via l'**adresse IP virtuelle** du **cluster** depuis un **poste client sous Windows**. Par défaut, le **site** affiché sera celui du **serveur principal (srvweb1)**, indiquant “**Bienvenue sur le site n°1 de la ville des Abymes**”, puisque **srvweb1** est le **maître actif**.

Nous allons **ensuite éteindre srvweb1** pour **simuler une panne**. Une fois la **page rafraîchie** sur le **client**, nous devrions voir **apparaître “Bienvenue sur le site n°2 de la ville des Abymes”**, indiquant que **srvweb2 a pris le relais**, prouvant ainsi que la **bascule s'est bien effectuée**.

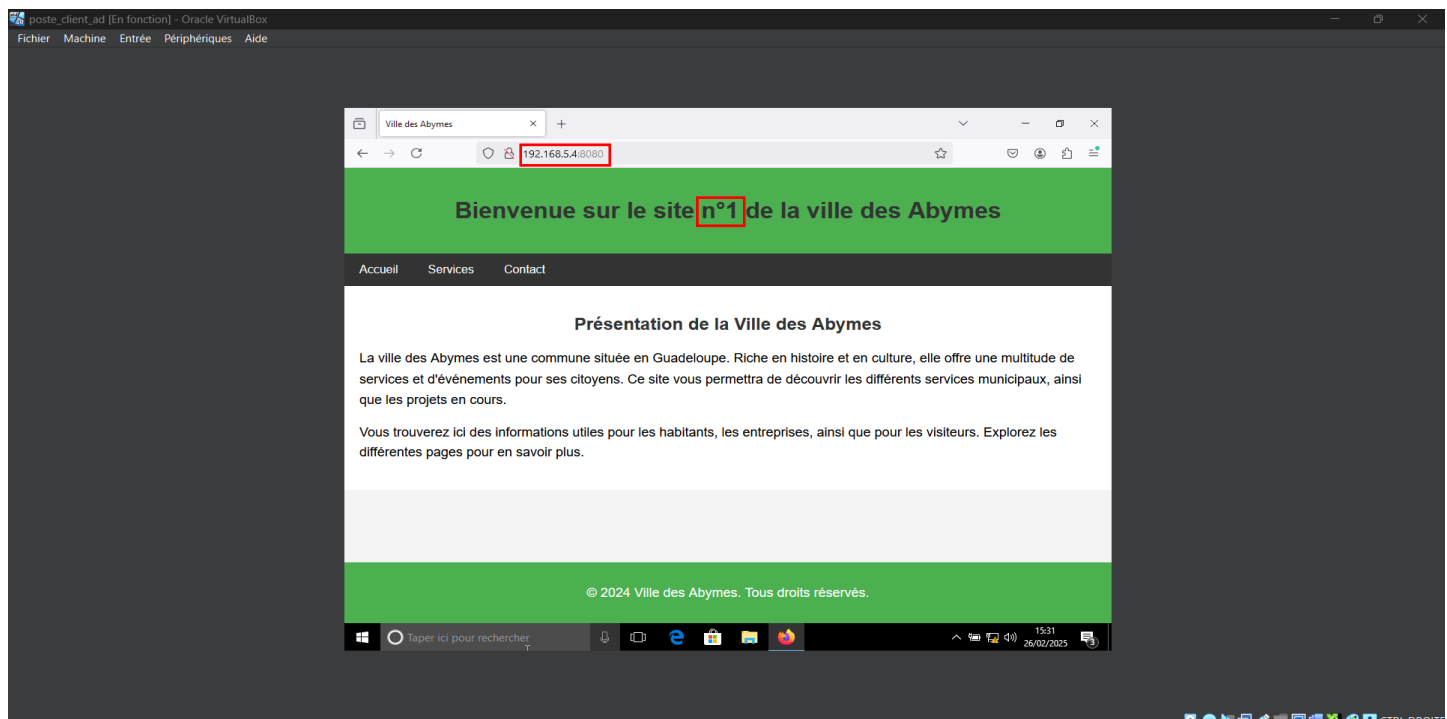
Enfin, nous **rallumerons srvweb1**, et après un **nouveau rafraîchissement de la page**, le site devrait **revenir sur srvweb1** avec **“Bienvenue sur le site n°1 de la ville des Abymes”**, **confirmant** que la **gestion du retour automatique** vers le **serveur principal** fonctionne bien.

## 1. Accès au site via l'IP virtuelle :

Depuis un **poste client sous Windows**, nous **accédons** au site web de la ville des Abymes en **entrant l'adresse IP virtuelle du cluster (192.168.5.4:8080)** dans le **navigateur Firefox**. Puis il **faut s'authentifier** :

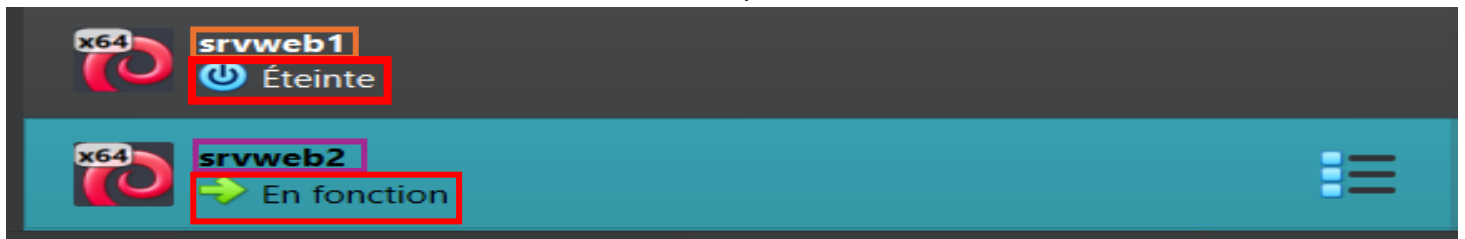
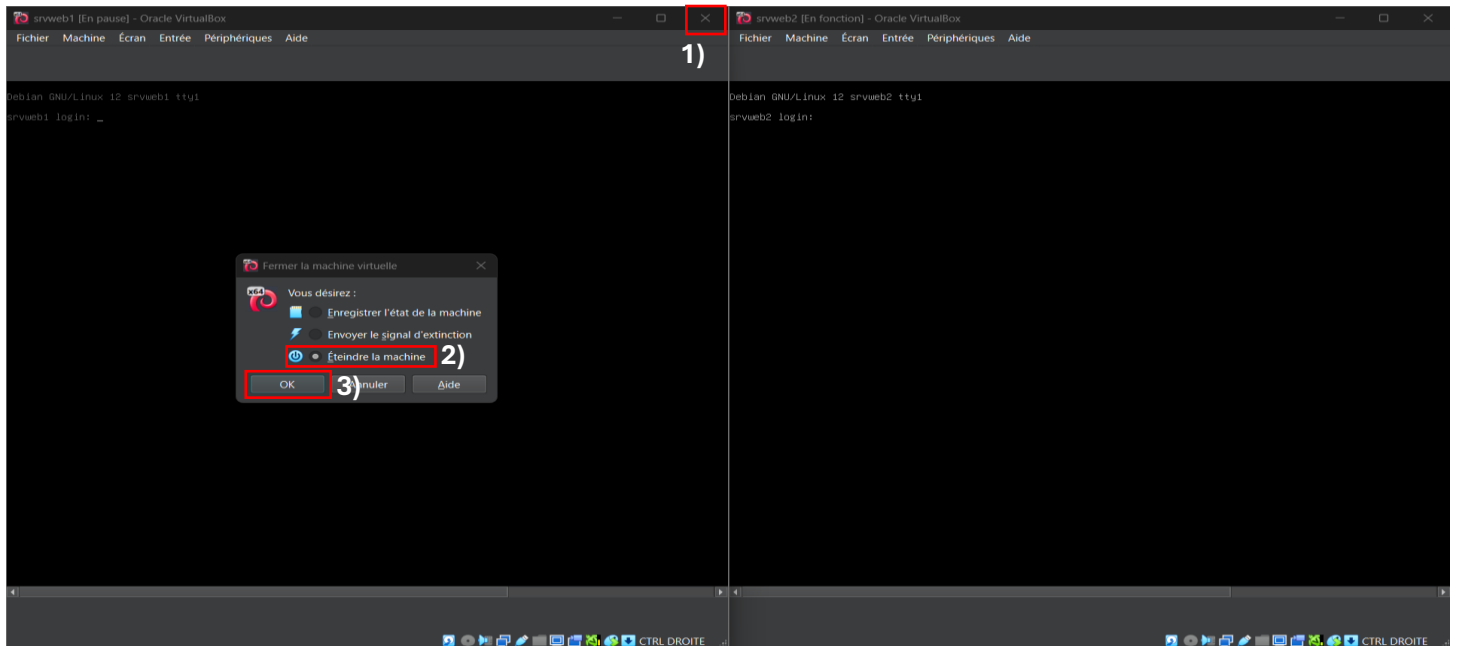


Le site affiché doit être celui du **serveur principal srvweb1**, avec le **message : “Bienvenue sur le site n°1 de la ville des Abymes”**.

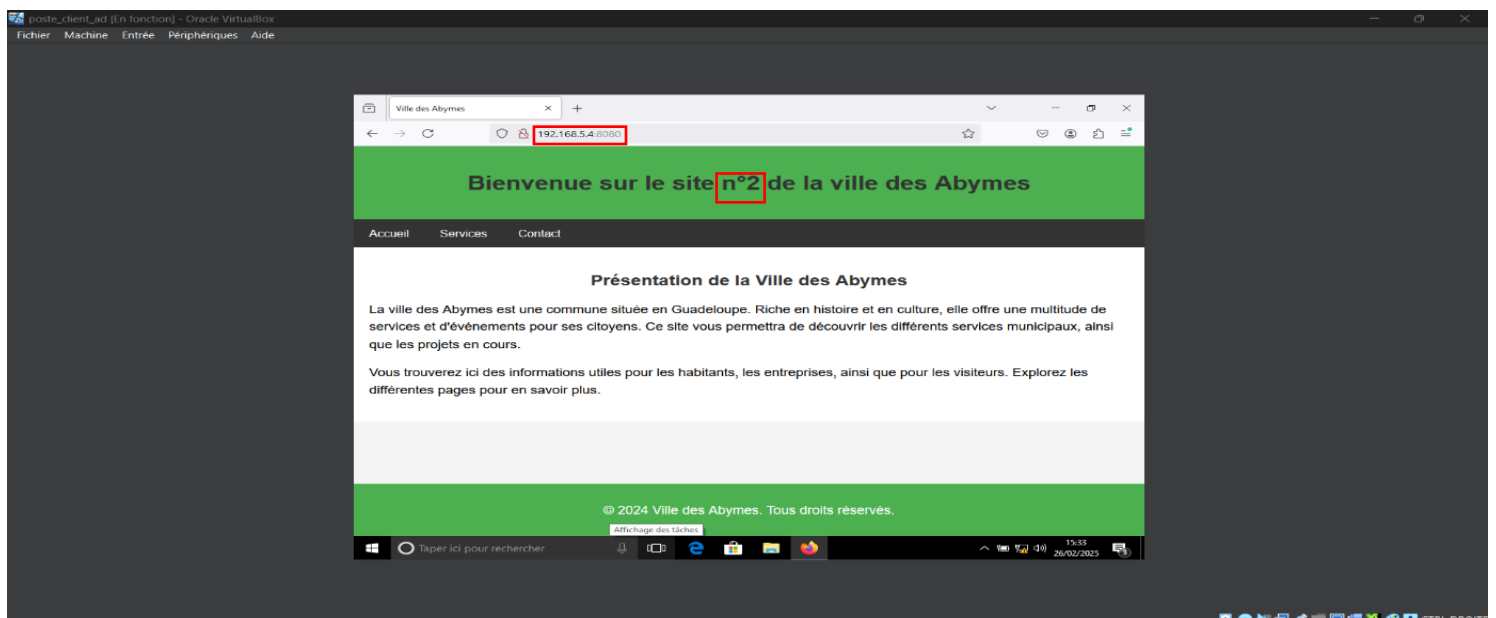


## 2. Simulation de la panne du serveur principal :

Nous arrêtons **srvweb1** pour **simuler une panne** et laissons **allumer srvweb2** :



Depuis le poste client, nous **rafraîchissons la page** en faisant **F5**. Si la **haute disponibilité fonctionne correctement**, le site doit être pris en charge par **srvweb2**, affichant :  
“**Bienvenue sur le site n°2 de la ville des Abymes**”.

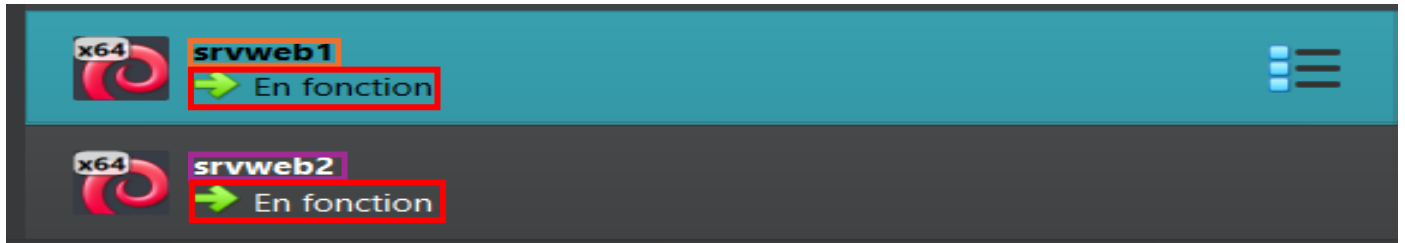




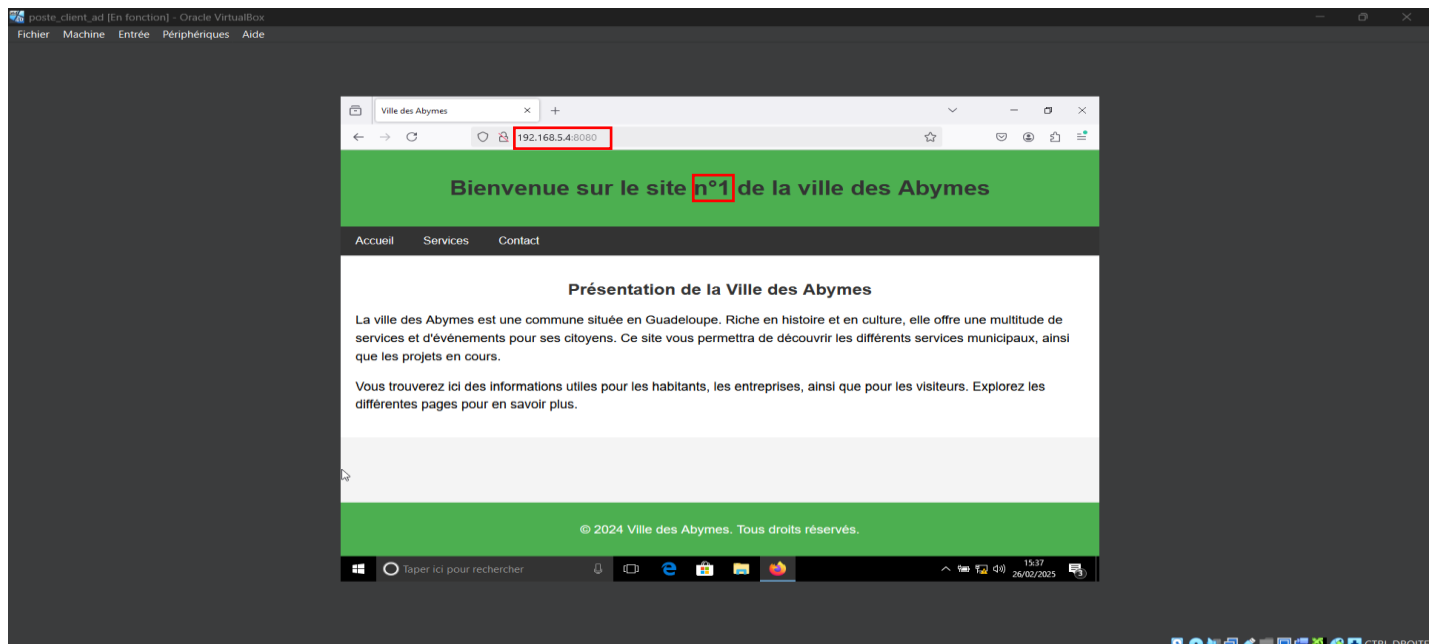
Cela prouve que **srvweb2** a bien **pris le relais** en cas de **défaillance** du serveur principal.

### 3. Restauration du serveur principal et retour à la situation initiale :

Nous **redémarrons** **srvweb1**, afin que les **deux serveurs soient de nouveau actifs** :



Après quelques instants, nous **rafraîchissons** la page depuis le **poste client** avec **F5**. Si la **configuration** est **correcte**, la **page doit revenir** sur **srvweb1**, affichant à nouveau :  
“**Bienvenue sur le site n°1 de la ville des Abymes**”.



**!/\\ Ce test permet de confirmer que la bascule entre les serveurs en cas de panne fonctionne et que le retour automatique vers le maître initial s’effectue correctement !/\\**

## CONCLUSION :

Ce TP a offert une **expérience enrichissante** dans la mise en place d'un **cluster de haute disponibilité** assurant la continuité des services web avec **Heartbeat**. La configuration des serveurs **srvweb1** et **srvweb2**, ainsi que l'implémentation d'une **IP virtuelle**, ont permis d'explorer les mécanismes de **basculement automatique et de redondance** essentiels à la **fiabilité** d'une **infrastructure informatique**.

Grâce à cette approche, les **compétences développées** en **clustering**, **gestion des pannes et continuité de service** s'avèrent **indispensables** pour assurer des **systèmes robustes et résilients**. Cette maîtrise technique constitue un **acquis durable**, directement **applicable** aux **environnements professionnels** où la **stabilité et la disponibilité des services** sont des **enjeux majeurs**.

## FIN DU TP :

