Fichier Annexe A : Configuration du solution

Équipe N°8

Mai 2023

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

École nationale Supérieure d'Informatique
ex. INI (Institut National de formation en Informatique)

Fichier Annexe A : Configuration du solution

Dans le cadre du projet 2CS

Option : Systèmes Informatiques (SIQ)

Mise en place d'un CDN (Content Delivery Network)

Réalisé par :

Belli Bilal (CE)
Aboud Rayane
Hassani Mehdi
Safta Abir
Benabdelatif Maya
Rabahsidhoum Mohamed Amine

Encadré par :

M. Amrouche Hakim
M. Hamani Nacer

Promotion : 2022/2023

Table des matières

Configuration du serveur DNS

Dans une architecture CDN (Content Delivery Network), la configuration d'un serveur DNS joue un rôle crucial en redirigeant les requêtes des utilisateurs vers les serveurs cachés les plus proches, en fonction de leur localisation géographique. Dans le cadre de notre projet nous avons besoin de deux outils pour configurer un serveur DNS :

- **HAProxy :** pour la redirection géographique
- **BIND :** pour la résolution des noms.

Remarque : Pour la mise en place de notre configuration, nous utiliserons Red Hat 7 comme système d'exploitation.

0.1 Présentation de HAProxy

HAProxy est largement utilisé dans les infrastructures réseau pour distribuer la charge de manière équilibrée entre plusieurs serveurs, garantissant ainsi une haute disponibilité et une meilleure performance.

Le but d'utiliser HAproxy dans notre configuration :

HAProxy permet d'optimiser la redirection des requêtes des utilisateurs, donc au lieu de rediriger toutes ces requêtes vers le serveur d'origine qui héberge notre application il les redirige vers l'un des serveurs caches le plus proche en fonction de la localisation géographique, réduisant ainsi les temps de latence et améliorant les performances globales du serveur DNS.

0.2 Présentation de BIND

BIND (Berkeley Internet Name Domain) est le serveur DNS open source le plus couramment utilisé. Il fournit un ensemble complet de fonctionnalités pour la résolution des noms de domaine, permettant de traduire les noms de domaine en adresses IP et de gérer les enregistrements DNS.

Le but d'utiliser BIND dans notre configuration :

En utilisant BIND comme un résolveur des noms de domaine les utilisateurs peuvent simplement taper le nom de domaine www.edustream.dz dans leur navigateur pour accéder a l'application plutôt que de devoir mémoriser l'adresse IP du serveur DNS.

0.3 Configuration de HAproxy

0.3.1 Installation et préparation de HAproxy

1. Installer HAproxy : yum install haproxy -y
2. Activer le service HAproxy : systemctl enable haproxy
3. Démarrer le service HAproxy : systemctl start haproxy
4. Afficher l'état actuel du service HAProxy : c'est une étape de vérification que le service HAproxy fonctionne correctement : systemctl status haproxy

   On devrait avoir le résultat suivant :

```
● haproxy.service - HAProxy Load Balancer
   Loaded: loaded (/usr/lib/systemd/system/haproxy.service; enabled; vendor preset: disabled)
   Active: active (running) since Sun 2023-05-21 23:59:19 PDT; 3h 26min ago
 Main PID: 1139 (haproxy-systemd)
    Tasks: 3
   CGroup: /system.slice/haproxy.service
           ├─1139 /usr/sbin/haproxy-systemd-wrapper -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid
           ├─1146 /usr/sbin/haproxy -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid -Ds
           └─1157 /usr/sbin/haproxy -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid -Ds

May 21 23:59:19 localhost.localdomain systemd[1]: Started HAProxy Load Balancer.
May 21 23:59:21 localhost.localdomain haproxy-systemd-wrapper[1139]: haproxy-systemd-wrapper: executi
ng /usr/sbin/haproxy -f /etc/haproxy/haproxy.cfg -p /run/haproxy.pid -Ds
[root@localhost ~]#
```

0.3.2 Configuration

La configuration par défaut de HAproxy se trouve dans le fichier haproxy.cfg, on laisse la configuration par défaut tel qu'elle est et on ajoute notre propre configuration :

Avant d'entamer la configuration on explique deux notions pour la configuration du HAproxy : le frontend et le backend.

- Frontend : Le frontend représente l'interface publique du HAproxy. Il est responsable de la réception des requêtes des clients et de leur routage vers les serveurs appropriés.
- Backend : Le backend représente l'ensemble des serveurs cibles vers lesquels les requêtes des clients seront réparties

1. Accéder au fichier de configuration : nano /etc/haproxy/haproxy.cfg
2. Ajouter un frontend :

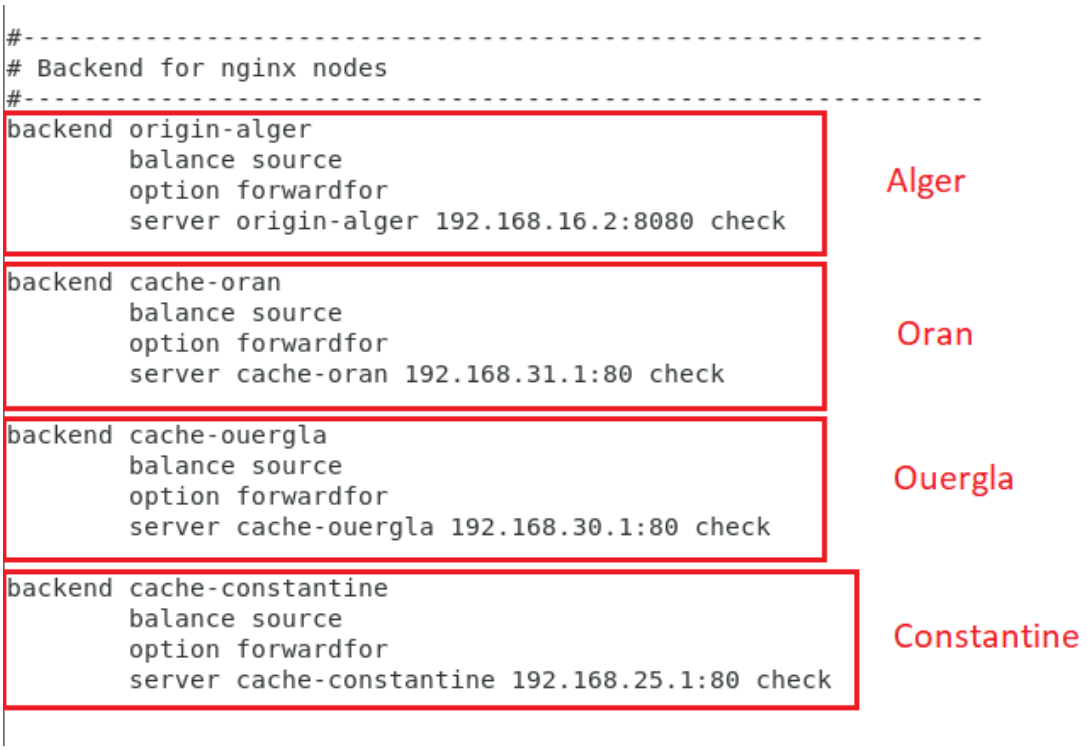
```
#---------------------------------------------------------------------
# Frontend for ngnix nodes
#---------------------------------------------------------------------
frontend apphttp
        bind *:80                                   1
        mode http
        acl alger src 192.168.16.0/24
        acl oran src 192.168.31.0/24
        acl ouergla src 192.168.30.0/24             2
        acl constantine src 192.168.25.0/24
        use_backend origin-alger if alger
        use_backend cache-oran if oran
        use_backend cache-ouergla if ouergla        3
        use_backend cache-constantine if constantine
```

(a) Pour indiquer que le frontend doit écouter les requêtes de type http sur toutes les adresses IP (*) et sur le port 80.

(b) Les ACLs sont pour définir les plages des valeurs des adresses ip des différentes régions.

(c) Pour indiquer que les requêtes provenant d'une région (définie par une ACL) seront redirigées vers le backend associé à cette ACL.

3. Ajouter un backend pour chaque région :



Pour chaque région on définit un backend avec les options suivantes :

- balance source : Cette ligne spécifie la méthode d'équilibrage de charge utilisée par le backend. Dans ce cas, la méthode utilisée est "source", ce qui signifie que les requêtes provenant de la même plage d'adresses IP seront dirigées vers le même serveur du backend.
- option forwardfor : Cette ligne active l'ajout d'en-têtes "X-Forwarded-For" aux requêtes transmises par HAProxy. L'en-tête "X-Forwarded-For" contient l'adresse IP réelle du client d'origine, ce qui est utile lorsque HAProxy agit comme un proxy inverse pour les serveurs backend.
- server server-name <@ip-de-serveur> :<n°port> check : Cette ligne définit les détails du serveur backend. L'option "check" indique à HAProxy de vérifier régulièrement la disponibilité du serveur en envoyant des requêtes de vérification. Si le serveur est disponible, HAProxy l'inclut dans la répartition de charge.

4. Redémarrer le service HAproxy : Après avoir sauvegarder le fichier haproxy.conf on redémarre HAproxy systemctl reload haproxy

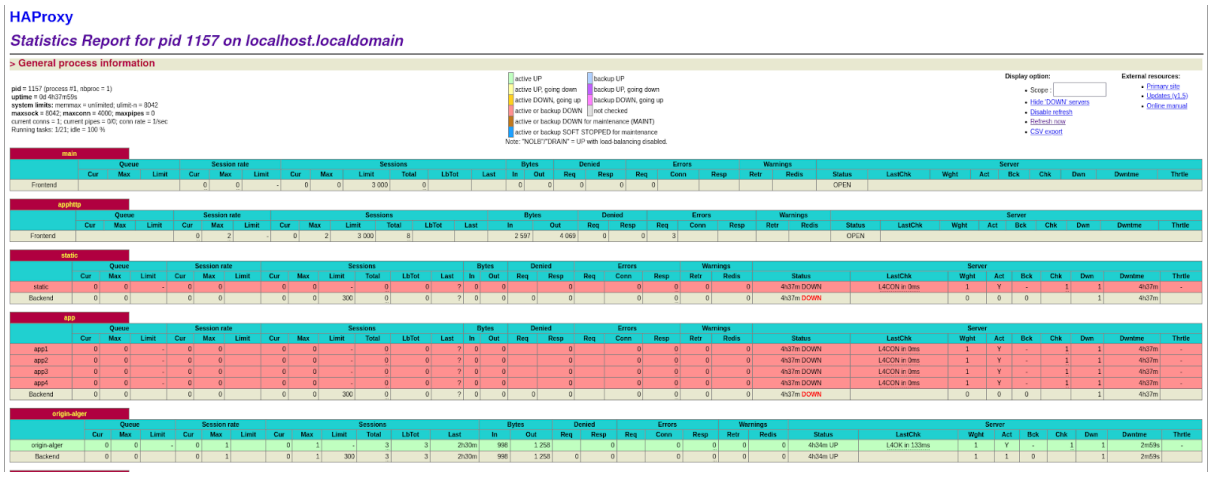
5. Configurer les statistiques de HAproxy :

```
#---------------------------------------------------------------------
# Configure for the Statistics Report
#---------------------------------------------------------------------
listen stats
bind :9000                # use the port 9000 to access to report
stats enable
stats hide-version
stats refresh 20s
stats show-node
stats uri /stats          # the url will be : http://@ip_of_proxy:9000/stats
stats auth proxy:password  # set a password
```

Afin de voir les statistiques du serveur DNS utilisant HAproxy on peut ajouter cette configuration dans le fichier haproxy.cfg.

Pour visualiser les statistiques on accède d'après le navigateur web a

http ://192.168.16.1 :9000/stats avec le nom d'utilisateur **proxy** et le mot de passe password.



0.4 Configuration de BIND

0.4.1 Installation et préparation de BIND

1. installer BIND : yum install bind

2. Activer le service named : systemctl enable named Remarque : named" est le nom du service qui contient le logiciel BIND.

3. Démarrer le service named : systemctl start named

4. Afficher l'état actuel du service named :
   c'est une étape de vérification que le service HAproxy fonctionne correctement systemctl status named On devrait avoir les resultats suivants :

```
[root@localhost ~]# systemctl status named
● named.service - Berkeley Internet Name Domain (DNS)
   Loaded: loaded (/usr/lib/systemd/system/named.service; disabled; vendor preset: disabled)
   Active: active (running) since Mon 2023-05-22 02:08:05 PDT; 2h 53min ago
  Process: 16881 ExecStop=/bin/sh -c /usr/sbin/rndc stop > /dev/null 2>&1 || /bin/kill -TERM $MAI
  Process: 16898 ExecStart=/usr/sbin/named -u named -c ${NAMEDCONF} $OPTIONS (code=exited, status
  Process: 16896 ExecStartPre=/bin/bash -c if [ ! "$DISABLE_ZONE_CHECKING" == "yes" ]; then /usr/
CCESS)
 Main PID: 16900 (named)
    Tasks: 4
   CGroup: /system.slice/named.service
           └─16900 /usr/sbin/named -u named -c /etc/named.conf

May 22 05:01:12 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
May 22 05:01:12 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
May 22 05:01:12 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
May 22 05:01:12 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
May 22 05:01:12 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
May 22 05:01:14 localhost.localdomain named[16900]: resolver priming query complete
May 22 05:01:14 localhost.localdomain named[16900]: network unreachable resolving 'www.haproxy.or
May 22 05:01:14 localhost.localdomain named[16900]: network unreachable resolving 'www.haproxy.or
May 22 05:01:15 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
May 22 05:01:15 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
[root@localhost ~]#
```

0.4.2 Configuration de BIND :

La configuration par défaut de BIND se trouve dans le fichier named.conf, on laisse la configuration par défaut tel qu'elle est et on rajoute notre propre configuration :

1. Accéder au fichier de configuration : nano /etc/named.conf

2. Spécifier les adresses IP autorisées à effectuer des requêtes de recherche DNS vers le serveur BIND.

```
options {
#       listen-on port 53 { 127.0.0.1; };
#       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;192.168.16.0/24; 192.168.25.0/24; 192.168.31.0/24; 192.168.30.0/24;};
        /*
```

   Dans notre cas c'est les plages d'adresses ip des 4 régions (alger, oran, constantine et ouergla).

3. Ajouter une zone forward et une autre zone reverse

```
zone "edustream.com" IN {
    type master;
    file "fwd.edustream.local.db";
    allow-update {none;};
};
```

Zone Forward

```
zone "16.168.192.in-addr.arpa" IN {
    type master;
    file "rev.16.168.192.db";
    allow-update {none;};
};
```

Zone Reverse

Zone Forward : La zone ”edustream.com” spécifie que BIND est responsable de résoudre les requêtes DNS concernant les noms de domaine se terminant par ”edustream.com”. Le fichier spécifié dans file ”fwd.edustream.local.db” contient les enregistrements de zone forward qui associent les noms de domaine à des adresses IP correspondantes.

Zone reverse : La zone **”16.168.192.in-addr.arpa”** est utilisée pour résoudre les adresses IP en noms de domaine correspondants. Le fichier spécifié dans file ”rev.16.168.192.db” contient les enregistrements de zone reverse qui associent les adresses IP à des noms de domaine correspondants.

4. Configurer le fichier des enregistrements de la zone Forward

```
$TTL    86400
@       IN      SOA     edustream.com. root.edustream.com. (
1       ; Serial
604800  ; Refresh
86400   ; Retry
2419200 ; Expire
86400   ; Negative Cache TTL
);
```

```
@       IN      NS      ns1.edustream.com.
ns1     IN      A       192.168.16.1
www     IN      A       192.168.16.1
```

TTL 86400 : La durée de vie par défaut (Time-to-Live) pour les enregistrements de cette zone.

Les lignes suivantes **(1, 604800, 86400, 2419200, 86400)** spécifient les paramètres de configuration du SOA. Ils indiquent le numéro de série de la zone, les délais de rafraîchissement (Refresh), de réessai (Retry), d’expiration (Expire) et de cache négatif (Negative Cache TTL).

@ IN NS ns1.edustream.com. : Cette ligne définit le serveur DNS principal pour la zone : ”@ IN NS” indique que le serveur DNS principal est défini pour la zone ”edustream.com”, et ”ns1.edustream.com.” est le nom du serveur DNS.

ns1 IN A 192.168.16.1 : Cette ligne associe l’adresse IP 192.168.16.1 au nom de domaine ”ns1.edustream.com”. Cela permet de résoudre le nom ”ns1.edustream.com”

en l'adresse IP correspondante.

www IN A 192.168.16.1 : Cette ligne associe l'adresse IP 192.168.16.1 au nom de domaine "www.edustream.com". Cela permet de résoudre le nom "www.edustream.com" en l'adresse IP correspondante.

5. Configurer le fichier des enregistrements de la zone Reverse

```
$TTL 86400
@       IN      SOA     edustream.com. root.edustream.com. (
1       ; Serial
604800  ; Refresh
86400   ; Retry
2419200 ; Expire
86400   ; Negative Cache TTL
);
@       IN      NS      edustream.com.
1       IN      PTR     edustream.com.
```

@ IN NS edustream.com. : Cette ligne définit le serveur DNS principal pour la zone : "@ IN NS" indique que le serveur DNS principal est défini pour la zone de résolution inverse du réseau IP "192.168.16.0/24", et "edustream.com." est le nom de ce serveur.

1 IN PTR edustream.com. : Cette ligne associe l'adresse IP inverse "1" (correspondant à l'adresse IP 192.168.16.1) au nom de domaine "edustream.com". Cela permet de résoudre l'adresse IP inverse "1" en le nom de domaine correspondant.

6. Redémarrer le service named : Après avoir sauvegarder tous les fichiers de configuration on redémarre le service named systemctl reload named

Configuration des serveurs de cache

Pour mettre en place la mise en cache au niveau des serveurs caches, nous avons envisagé d'utiliser le logiciel Nginx en tant que serveur proxy inverse afin de stocker temporairement les données consultées par les utilisateurs.

0.5 Nginx

Lorsque nous utilisons Nginx pour la mise en cache, il agit en tant que serveur proxy inverse qui stocke temporairement les contenus statiques générés par un serveur d'origine et les renvoie directement aux utilisateurs ultérieurement, sans avoir à faire appel au serveur d'origine à chaque demande.

Cela permet d'améliorer considérablement les performances en réduisant la charge sur le serveur hébergeant les données et en diminuant les temps de réponse.

0.5.1 Configuration de Nginx

1. Après avoir installé et démarré Nginx au niveau des serveurs caches, on spécifie dans le fichier /etc/nginx/nginx.conf le chemin vers le fichier qui va stocker les données utilisées :

   # dans le fichier /etc/nginx/nginx.conf

   http ....

   proxy_cache_path /var/cache/nginx levels=1 :2

   keys_zone=my_cache :10m

   max_size=10g inactive=60m # 1

2. Cette ligne de configuration déclare un chemin de cache dans /var/cache/nginx, crée une zone de clés de cache nommée "my_cache", limite la taille du cache à 10 Go et supprime les fichiers inactifs après 60 minutes.

3. Cette ligne de configuration définit le chemin du répertoire temporaire utilisé par Nginx pour stocker les requêtes proxy.

4. Pour définir le comportement du serveur Nginx, vous pouvez éditer le fichier /etc/nginx/conf.d/cache.conf avec les configurations suivantes :

dans le fichier /etc/nginx/conf.d/cache.conf proxy_cache my_cache ;
proxy_cache_valid 200 302 10m ;
proxy_cache_valid 404 1m ;
proxy_cache_valid any 1m ;

proxy_ignore_headers Expires Cache-Control ; server
location /

proxy_pass http ://192.168.16.2 :8080 ;
add_header X-Cache-Location "Oran" ;
add_header X-Cache-Status $upstream_cache_status ;
proxy_set_header Host $host ;
proxy_set_header X-Real-IP $remote_addr ;
proxy_set_header X-Forwarded-For
$proxy_add_x_forwarded_for ;
proxy_cache_revalidate on ;
proxy_cache_min_uses 1 ;
proxy_cache_use_stale error
timeout updating http_500 http_502
http_503 http_504 ;
proxy_cache_background_update on ;

proxy_cache_lock on ;
proxy_cache_lock_age 5s ;
proxy_cache_lock_timeout 10s ;

} }

- **proxy_cache my_cache ;** : Cette directive active le mécanisme de mise en cache des réponses de requêtes proxy. "my_cache" fait référence à la zone de cache définie dans la configuration de /etc/nginx/conf.d/cache.conf.
- **proxy_pass http ://192.168.16.2 :8080 ;** : Cette directive indique que les requêtes doivent être transmises au serveur situé à l'adresse "192.168.16.2" sur le port "8080".
- **add_header X-Cache-Location "Oran" ;** : Cette directive ajoute un en-tête personnalisé "X-Cache-Location" à la réponse, avec la valeur "Oran" pour préciser le serveur cache source de la réponse.
- **add_header X-Cache-Status upstream_cache_status ;** : Cette directive ajoute un en-tête personnalisé "X-Cache-Status" à la réponse, avec la valeur correspondant à l'état du cache : HIT(requête réussite), MISS(requête perdu), STALE (mise à jour dans le serveur cache).
- **add_header X-Cache-Status upstream_cache_status ;** : Cette directive ajoute un en-tête personnalisé "X-Cache-Status" à la réponse, avec la valeur correspondant à l'état du cache : HIT(requête réussite), MISS(requête perdu), STALE (mise à jour dans le serveur cache).

5. Pour vérifier si les données sont bien stockée, il faut :

Accédez au répertoire /var/cache/nginx, puis sélectionnez une série de sous-répertoires (b puis 9d) pour finalement atteindre le fichier contenant les informations. Pour analyser ces informations, affichez le fichier en utilisant la commande cat. Le fichier de

données précise :

- @ Ip du serveur d'origine.
- La date d'enregistrement des informations.
- Le nom du serveur Cache (Alger dans notre exemple).
- Type et taille du contenu.
- Un script HTML représentant notre contenu sauvegardé.

```
[root@localhost ~]# cd /var/cache/nginx
[root@localhost nginx]# ls
b
[root@localhost nginx]# cd b
[root@localhost b]# ls
9d
[root@localhost b]# cd 9d/
[root@localhost 9d]# ls
23fb94126f81a79fa12c83848f2049db
[root@localhost 9d]# cat 23fb94126f81a79fa12c83848f2049db
4
KEY: http://192.168.16.2:8080/
HTTP/1.1 200 OK
Server: nginx/1.18.0
Date: Sun, 28 May 2023 20:38:34 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 344
Connection: close
X-Origin-Server-Location: Alger

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

        <title>application</title>

</head>
<body>

    <a href="/videos">videos</a>
    <a href="/pdfs">pdfs</a>


</body>
</html>[root@localhost 9d]#
```

Configuration de l'application de redirection

Le but principal de l'application est de rediriger la requête du client vers le serveur jugé le plus rapide en termes de qualité de service.

0.6 Explication brève de l'application

L'application de redirection créée avec Flask (un framework Python) est une solution flexible conçue pour utiliser la base de données GeoIP2 afin de localiser le client en fonction de son adresse IP. Elle prend en compte les données fournies par une autre application de surveillance, qui fournit des informations sur la disponibilité des serveurs ainsi que leur charge (en tenant compte du pourcentage d'utilisation du processeur et de la RAM). L'objectif est de calculer le score de chaque serveur par rapport au client afin de le rediriger vers le meilleur serveur disponible.

0.7 Explication de chaque fonctionnalité

1. **Application du monitoring**
   L'application vérifie régulièrement l'état de chaque serveur et enregistre les mises à jour concernant leur disponibilité et leur charge dans un tableau. Elle partage ensuite ces tableaux avec l'application de redirection.

```
availability = [1, 1, 1, 1]
loads = [1, 1, 1, 1]
```

   En bref, l'application de monitoring est définie comme suit :

```
def write():
    global availability, loads
    while True:
        availability[:]  = check_servers()
        loads[:] = check_load_on_servers()
        time.sleep(10)
```

   La fonction 'write' est utilisée pour mettre à jour la disponibilité (availability) des serveurs ainsi que leur charge (loads) toutes les 10 secondes dans cet exemple. La vérification et l'écriture ne sont pas bloquantes, ce qui signifie que l'application principale n'est pas affectée par le monitoring, mais continue de fonctionner en parallèle.

   (a) Vérification de la disponibilité
   Cette fonction vérifie l'état de chaque site (fonctionnel ou non) en utilisant la bibliothèque Requests pour vérifier la disponibilité du site sur chaque serveur. Elle calcule ensuite le score final en fonction des résultats obtenus.

```
#---------------------------------------------------------------------
# Frontend for ngnix nodes
#---------------------------------------------------------------------
frontend apphttp
        bind *:80
        mode http
        acl alger src 192.168.16.0/24
        acl oran src 192.168.31.0/24
        acl ouergla src 192.168.30.0/24
        acl constantine src 192.168.25.0/24
        use_backend origin-alger if alger
        use_backend cache-oran if oran
        use_backend cache-ouergla if ouergla
        use_backend cache-constantine if constantine
```

1

2

3

```
[root@localhost ~]# systemctl status named
● named.service - Berkeley Internet Name Domain (DNS)
   Loaded: loaded (/usr/lib/systemd/system/named.service; disabled; vendor preset: disabled)
   Active: active (running) since Mon 2023-05-22 02:08:05 PDT; 2h 53min ago
  Process: 16881 ExecStop=/bin/sh -c /usr/sbin/rndc stop > /dev/null 2>&1 || /bin/kill -TERM $MAI
  Process: 16898 ExecStart=/usr/sbin/named -u named -c ${NAMEDCONF} $OPTIONS (code=exited, status
  Process: 16896 ExecStartPre=/bin/bash -c if [ ! "$DISABLE_ZONE_CHECKING" == "yes" ]; then /usr/
CCESS)
 Main PID: 16900 (named)
    Tasks: 4
   CGroup: /system.slice/named.service
           └─16900 /usr/sbin/named -u named -c /etc/named.conf

May 22 05:01:12 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
May 22 05:01:12 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
May 22 05:01:12 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
May 22 05:01:12 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
May 22 05:01:12 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
May 22 05:01:14 localhost.localdomain named[16900]: resolver priming query complete
May 22 05:01:14 localhost.localdomain named[16900]: network unreachable resolving 'www.haproxy.or
May 22 05:01:14 localhost.localdomain named[16900]: network unreachable resolving 'www.haproxy.or
May 22 05:01:15 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
May 22 05:01:15 localhost.localdomain named[16900]: network unreachable resolving '0.debian.pool.
[root@localhost ~]#
```

(b) Vérification de la charge du serveur

```
options {
#       listen-on port 53 { 127.0.0.1; };
#       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;192.168.16.0/24; 192.168.25.0/24; 192.168.31.0/24; 192.168.30.0/24;};
        /*
```

```
#---------------------------------------------------------
# Configure for the Statistics Report
#---------------------------------------------------------
listen stats
bind :9000              # use the port 9000 to access to report
stats enable
stats hide-version
stats refresh 20s
stats show-node
stats uri /stats        # the url will be : http://@ip_of_proxy:9000/stats
stats auth proxy:password  # set a password
```

Cette fonction vérifie l'état de chaque serveur en utilisant la bibliothèque **psutil** pour vérifier la charge sur chaque serveur. Elle calcule ensuite le score final en fonction des résultats obtenus.

2. L'utilisation d'une base de données GeoIP
   Idée : En utilisant une base de données qui associe les plages d'adresses IP à des emplacements géographiques, nous pouvons déterminer le serveur géographique correspondant.

   (a) Problème de la localisation de l'utilisateur en fonction de son adresse IP

   Les plages d'adresses IP sont fournies sur le site : https ://lite.ip2location.com/algeria-ip-address-ranges
   Si nous traçons les intervalles d'adresses IP sur la carte de l'Algérie, nous obtenons la figure suivante :

   Donc, la localisation géographique au sens strict ne sera pas un facteur pris en compte pour la redirection.

   (b) classement des localisations selon leur emplacement (abscisse et coordonnées)

```
# Define the server IP locations and request counts
#every ip address is mapped to a specefic server with request init of 0
algiers_locations =  [(36.7405, 3.1159), (36.7624, 3.0459), (36.7434, 3.3418), (36.7642, 3.1468)]
oran_locations = [(36.1587, 0.9701), (35.7204, -0.6659)]
constantine_locations = [(36.4137, 7.5013), (36.1552, 6.1694)]
ouargla_locations = [(27.6779, -8.1464), (35.6075, 1.8152), (35.5545, 6.1769)]
```