PROYECTO 1: TELEMÁTICA

Alejandro Fernández Restrepo
Escuela de Ingeniería
Departamento de Informática y Sistemas
Universidad EAFIT
Medellín, Colombia
afernander@eafit.edu.co

Simón Marín Giraldo
Escuela de Ingeniería
Departamento de Informática y Sistemas
Universidad EAFIT
Medellín, Colombia
smaring1@eafit.edu.co

Miguel Fernando Ramos García
Escuela de Ingeniería
Departamento de Informática y Sistemas
Universidad EAFIT
Medellín, Colombia
mframosg@eafit.edu.co

29 de abril de 2021

RESUMEN

Amazon Web Services (AWS) es una suite completa de servicios en la web de Amazon, para computación basada en la nube, con servicios y APIs web. Amazon le provee servicios a muchas compañías en todo el mundo. Se usó **AWS Educate** para el desarrollo de este proyecto, creando instancias de máquinas **EC2** con **Amazon Linux AMI**.

1. Introducción

En este informe se muestra de manera detallada la secuencia de pasos que se siguió para desarrollar el **Proyecto 1** del curso de **Telemática**, en el que se proponía el **Despliegue de Servicios y Aplicaciones en Red**. Con esto se busca dejar evidencia de la realización de la práctica, así como servir de guía para realizar despliegues similares haciendo uso de **Amazon Web Services (AWS)**.

2. Máquinas virtuales

En total se deben configurar seis máquinas virtuales, de esta manera:

Las máquinas poseen las siguientes características:

▼ AMI Details

Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-048f6ed62451373d9
 Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras.
 Root Device Type: ebs Virtualization type: hvm

▼ Instance Type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	-	1	1	EBS only	-	Low to Moderate

▶ Security Groups

▶ Instance Details

▼ Storage

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encrypted
Root	/dev/xvda	snap-0d6a2ac1feca4cf17	15	gp2	100 / 3000	N/A	Yes	Not Encrypted

Para cada servicio se configuró un **Security Group** particular, que se detallará en cada uno de ellos.

Para todas las máquinas, lo primero que se debe hacer es actualizar los paquetes y dependencias. Para esto, se procede con el comando `sudo yum update -y`.

3. Servicios desplegados

3.1. Nombre de dominio

Se obtuvo el nombre de dominio `http://www.vaso.cf` utilizando `https://www.freenom.com/`

3.2. Base de datos

El **Security Group** para este servicio es el siguiente:

Intervalo de p...	Protocolo	Origen	Grupos de seguridad
22	TCP	[Redacted]	launch-wizard-1
3306	TCP	[Redacted]	launch-wizard-1
3306	TCP	[Redacted]	launch-wizard-1
3306	TCP	[Redacted]	launch-wizard-1

Se requiere de una base de datos para el adecuado funcionamiento de WordPress. Para esto, se montó un servidor de base de datos en **MySQL**. Tuvimos ciertos inconvenientes, debido a que MySQL no se podía descargar con `yum`, por lo que se realizó la descarga con `wget` y se instaló con `yum`. Lo logramos solucionar siguiendo los pasos indicados en [este enlace](#), por lo que procedimos con los siguientes comandos:

```
sudo wget https://dev.mysql.com/get/mysql57-community-release-el7-11.noarch.rpm
sudo yum localinstall mysql57-community-release-el7-11.noarch.rpm
sudo yum install mysql-community-server
systemctl start mysqld.service
```

Esto genera un password temporal. Para ubicarlo, ejecutamos `sudo grep 'temporary password' /var/log/mysqld.log`. Luego, procedemos con estos comandos:

```
sudo mysql_secure_installation
sudo nano /etc/my.cnf
```

En el archivo de configuración es necesario configurar la `bind-address` con la **dirección IP** `xxx.xx.xx.xxx` de la máquina actual, en la que se está configurando la base de datos.

Finalmente, iniciamos el servicio con `sudo systemctl start mysql.service`

Para la creación de la base de datos en **MySQL** procedimos con los siguientes comandos:

```
mysql -u root -p
CREATE DATABASE 'wordpressdb';
GRANT ALL PRIVILEGES ON * . * TO 'mframesg'@'%';
show databases;
exit
```

No fue necesario crear tablas manualmente. Solo se requiere crear la base de datos para que **WordPress** pueda funcionar. Las tablas necesarias se generarán desde **WordPress**.

3.3. CMS

El **Security Group** para este servicio es el siguiente:

Intervalo de p...	Protocolo	Origen	Grupos de seguridad
22	TCP	[REDACTED]	launch-wizard-1
3306	TCP	[REDACTED]	launch-wizard-1
3306	TCP	[REDACTED]	launch-wizard-1
3306	TCP	[REDACTED]	launch-wizard-1

En la práctica se requiere montar dos instancias, ambas en el mismo Security Group. Se pide tener dos instancias de **CMS** idénticas conectadas a una misma **base de datos** para poder distribuir la carga de usuarios entre ambas instancias.

Se usó **WordPress** para el manejo de los diferentes elementos de la página. Para ambas instancias se debe hacer la misma configuración, una después de la otra.

En la documentación de AWS se puede encontrar una guía para [Alojar un blog de WordPress en Amazon Linux 2](#) . En esta se usan los siguientes comandos para la instalación de WordPress en nuestra instancia:

```
wget https://wordpress.org/latest.tar.gz
tar -xzf latest.tar.gz
```

WordPress requiere de PHP con una versión mínima de 5.4 para poder funcionar. Para la instalación, procedimos con estos comandos:

```
sudo amazon-linux-extras enable php7.4
sudo amazon-linux-extras install php7.4
```

Posteriormente, es necesario crear el archivo de configuración de WordPress a partir de un ejemplo, que quedará con el nombre requerido `wp-config.php`:

```
cp wordpress/wp-config-sample.php wordpress/wp-config.php
```

Aquí con nano editamos varios campos:

```
define('DB_NAME', 'wordpressdb')
define('DB_USER', 'mframosg')
define('DB_PASSWORD', 'Eafit21211*')
define('DB_HOST', 'la ip privada de la instancia de base de datos')
```

A continuación, se debe entrar al siguiente [enlace](#) para generar valores aleatorios para llenar los campos faltantes:

```
define('AUTH_KEY',          '@ tZ++i:s[-wGlSc+a%1)Nb?Cm/9?ec5v@sz%Y*{Y-v|ryq<h|6)U!qFYg/_ QL(');
define('SECURE_AUTH_KEY',  'g Q c-B?U|[0-+fB13~bEfRT9d->nwQc^w.u?%!%eCww75D:Z-&l -N-,^*J=(');
define('LOGGED_IN_KEY',    'Dj#5p%&m&tF3WSO*|9QeVEm&aRfaWi]|pG: jTVkTJvBWq~BY/4YvHqV74_-H6>X');
define('NONCE_KEY',        'T>+}F+Mq%|D@ny=RH@,i),yI;?OpKkcZAIunRSc=wPhRP[=IaHHcv@ 2;XL4EA]U');
define('AUTH_SALT',        'Py%j(p*Wz'N!1HBLpZYy#iS+IE!;TdM8u&5+|P~HH~p=-sRVEU&Jy e--35kv0cK');
define('SECURE_AUTH_SALT', 'J=A=J$,)^3Zpm5V;&r8qhj Zj j|R11^45_00nN{x#(e{WeCo%|3_1=+OCb_>-2m');
define('LOGGED_IN_SALT',   'xePo<!~FQhK&s%VcDWK{6J/W-xA{bUL2ln4o }WQrxN5%S4?}QD^E$S3biB8EkK$');
define('NONCE_SALT',       'J &ow~2.pyaFs%%9lzI#Pv5k):=U+bd+$K2R ]HxDc-yy+b5Pq_>GQT1y,N]rj2');
```

Ahora, es necesario instalar la dependencia para el servidor de **Apache**. ejecutamos el comando `sudo yum install httpd -y`. Después, entramos a las configuraciones de Apache para que `index.php` sea la página principal. Procedemos entonces:

```
cd
cd /etc/httpd/conf/
sudo nano httpd.conf
```

Nos debe quedar así:

```
ec2-user@ip-172-31-25-20:/etc/httpd/conf
GNU nano 2.9.8 httpd.conf
# DirectoryIndex: sets the file that Apache will serve if a directory
# is requested.
#
<IfModule dir_module>
    DirectoryIndex index.html index.php
</IfModule>
```

Luego, debemos darle **todos los permisos** a la carpeta de `html`, para evitar problemas. Para esto, usamos el comando `sudo chmod 777 -R /var/www/html/`.

Finalmente, debemos ejecutar los siguientes comandos para que `httpd` se ejecute automáticamente cada vez que se inicie la instancia.

```
sudo systemctl start httpd
sudo systemctl enable httpd
```

Dentro de la página, en los ajustes de **WordPress** se deben realizar ciertas configuraciones:

General Settings

Site Title: Vaso Ventas

Tagline: Home
In a few words, explain what this site is about.

WordPress Address (URL): http://52.23.10.184

Site Address (URL): http://vaso.cf
Enter the address here if you [want your site home page to be different from](#)

En Site Address (URL) se debe indicar la **dirección IP** del **load balancer**. Como contamos con un **sistema DNS** vinculado al load balancer, si ponemos el nombre de nuestro dominio, el **DNS** cumplirá su función y con el dominio nos apuntará a la **dirección IP** correspondiente.

Cabe resaltar que esta configuración se debe hacer igual para ambas instancias de **CMS**.

3.4. Load Balancer (HAPROXY)

El **Security Group** para este servicio es el siguiente:

Intervalo de p...	Protocolo	Origen	Grupos de seguridad
80	TCP	0.0.0.0/0	Sg-LoadBalancer
80	TCP	::/0	Sg-LoadBalancer
22	TCP	[Redacted]	Sg-LoadBalancer

El **load balancer** se encargará de distribuir el tráfico de usuarios entre nuestras dos instancias de **CMS**. Desde la teoría computacional, el balanceo de carga consiste en distribuir un conjunto de tareas en un conjunto de recursos de manera eficaz.

Para realizar la configuración del **load balancer** montamos un servidor **haproxy** y comenzamos con los siguientes comandos:

```
sudo yum install haproxy
cd /etc/haproxy/
sudo nano haproxy.cfg
```

En nano, debemos configurar varias cosas. Debemos especificar el **frontend** y el **backend**. Este último se refiere a nuestras dos instancias de **CMS**. También es necesario darle todos los permisos al puerto 80:

```
frontend app_servers
bind *:80
default_backend apps

backend app
    balance roundrobin
    server app1 127.0.0.1:5001 check
    server app2 127.0.0.1:5002 check
    server app3 127.0.0.1:5003 check
    server app4 127.0.0.1:5004 check

backend apps
    server cms1 [redacted]:80 check
    server cms2 [redacted]:80 check
```

Finalmente, debemos iniciar el servicio y configurarlo para que se inicie automáticamente al arrancar la máquina:

```
sudo systemctl start haproxy.service
sudo systemctl enable haproxy.service
```

3.5. DNS y nombre de dominio

El **Security Group** para este servicio es el siguiente:

Intervalo de p...	Protocolo	Origen	Grupos de seguridad
80	TCP	0.0.0.0/0	launch-wizard-4
80	TCP	::/0	launch-wizard-4
22	TCP	[redacted]	launch-wizard-4
53	UDP	0.0.0.0/0	launch-wizard-4
53	UDP	::/0	launch-wizard-4

El **DNS** se encarga de enlazar la **dirección IP** del **load balancer** con la **dirección del dominio**. Tenemos dos instancias de **DNS** en caso de que una presente inconvenientes, se siga con el servicio desde la otra.

Para configurar el **DNS** nos ayudamos del siguiente [tutorial](#).

Para comenzar con la configuración debemos proceder con los siguientes comandos:

```
yum install bind bind-utils -y
nano -w /etc/named.conf
```

En este archivo se debe garantizar acceso para cualquiera en el query y permitir transferencia de datos con el **load balancer**, **DNS secundario** y el **DNS actual** en el campo allow-transfer. También se debe crear la **zona**. Este archivo enlaza al **dominio** con el **load balancer**. También se debe configurar el DNS para los **correos electrónicos**, sin embargo, para esta práctica no se implementó servidor de **correo electrónico**.

```
options {
    listen-on port 53 { 127.0.0.1; };
    listen-on-v6 port 53 { ::1; };
    directory [redacted] "/var/named";
    dump-file [redacted] "/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 { any; };
    allow-transfer { localhost; [redacted]; };
}
```

```
GNU nano 2.9.8 vazo.cf.zone
$TTL 86400
$ IN SOA ns1.vazo.cf. root.vazo.cf. (
    2013042201 ;Serial
    3600      ;Refresh
    1800      ;Retry
    604800    ;Expire
    86400     ;Minimum TTL
)
; Specify our two nameservers
; Resolve nameserver hostnames to IP, replace with your two droplet IP addresses
; Define hostname -> IP pairs, which you wish to resolve
$ IN NS ns1.vazo.cf.
$ IN A ns1.vazo.cf.
vazo IN CNAME vazo.cf.
$ IN MX 10 vazo.cf.
```

Posteriormente, entramos a [freenom](#) para configurar el nameserver, para el redireccionamiento a los dos servidores DNS. Al momento de utilizarlo, se busca primero en el **DNS primario**. Si este no funciona, se busca en el **DNS secundario**.

Es importante aclarar que en la zona se debe definir el tipo **slave** e indicar la **IP pública** del **DNS primario** en el campo **masters**.

4. Resultados finales

Finalmente pudimos montar la página con todos sus servicios funcionando correctamente, como se puede ver a continuación: