Christian David Lopez Morcillo - A00312096

**EXAMEN 1 - Sistemas Distribuidos**

1. Consigne los comandos de linux necesarios para el aprovisionamiento de los servicios solicitados.

Para el balanceador de cargas:

\* Se escoge usar el programa Nginx balanceando cargas bajo el esquema round robin.

Primero se instala el repositorio necesario para poder instalar Ngix, esto se realiza agregando a los repositorios de yum un archivo. repo que indique la ruta para descargar nginx. El archivo se debe agregar en la ruta “/etc/yum.repos.d/” y debe tener la información:

```text

[nginx]

name=nginx repo

baseurl=http://nginx.org/packages/centos/$releasever/$basearch/

gpgcheck=0

enabled=1

```

\* Se realiza la instalación de Nginx usando yum y se inicia el servicio.

```sh

sudo yum install nginx

```

\* Se crea el archivo /etc/nginx/nginx.conf y allí se especifica la ip de los servidores que atenderán las peticione, de la siguiente forma:

```txt

http {

upstream webservers {

server 192,168,131,126;

server 192,168,131,127;

}

server {

listen 8080;

location / {

proxy\_pass http://webservers;

}

}

}

```

\* Se agregan los permisos necesarios para el cortafuegos.

```sh

iptables -I INPUT 5 -p tcp -m state -- NEW -m tcp --dport 8080 -j ACCEPT

iptables -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT

service iptables save

```

\* Se inicia el servicio Nginx.

```sh

sudo service nginx start

```

Para el servidor web:

\* Se instala el servicio web apache:

```sh

sudo yum install httpd

```

\* Se instala el servicio php:

```sh

sudo yum install php

```

\* Se instala el servicio php-mysql:

```sh

sudo yum install php-mysql

```

\* Se instala el servicio mysql:

```sh

sudo yum install mysql

```

\* Se agregan los permisos necesarios para el cortafuegos.

```sh

iptables -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 8080 -j ACCEPT

iptables -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT

service iptables save

```

\* Se habilita el acceso a ejecución de scripts php en código html. Se agrega el archivo .htaccess a la carpeta: /var/www/html/ con la siguiente información:

```txt

AddType php-script .php .htm .html

```

\* Se agrega el archivo index.php a la ruta var/www/html/. Este archivo tiene la función de consultar la base de datos a través de métodos php. El archivo php debe tener el siguiente código:

```php

<HTML>

<HEAD>

<TITLE>Servidor <%=@idServer%></TITLE>

</HEAD>

<BODY>

<H1>Servidor <%=@idServer%></H1>

<H2>La base de datos tiene la siguiente información:</H2>

<?php

$con = mysql\_connect("<%=@ip\_web%>","<%=@usuarioweb\_web%>","<%=@passwordweb\_web%>");

if (!$con)

{

die('Could not connect: ' . mysql\_error());

}

mysql\_select\_db("database1", $con);

$result = mysql\_query("SELECT \* FROM example");

while($row = mysql\_fetch\_array($result))

{

echo $row['name'] . " " . $row['age'];

echo "<br />";

}

mysql\_close($con);

?>

</BODY>

</HTML>

```

\* Se inicia el servicio web.

```sh

sudo service httpd start

```

Para el servidor de base de datos:

\* Se instala el servidor de mysql

```sh

sudo service httpd start

```

\* Se agregan los permisos necesarios para el cortafuegos.

```sh

iptables -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 3306 -j ACCEPT

service iptables save

```

\* Se inicia el servicio de base de datos:

```sh

sudo service mysqld start

```

\* Se hace la configuración inicial del servicio mysql con el archivo /usr/bin/mysql\_secure\_installation.

\* Se introducen datos a la base de datos y se da permisos a los servidores web a consultar la base de datos. Se crea y luego ejecuta el archivo create\_schema.sql con la siguiente información:

```sql

CREATE database database1;

USE database1;

CREATE TABLE example(

id INT NOT NULL AUTO\_INCREMENT,

PRIMARY KEY(id),

name VARCHAR(30),

age INT);

INSERT INTO example (name,age) VALUES ('Christian',23),('Luisa',23),('Pineros',23), ('Emmanuel',23);

-- http://www.linuxhomenetworking.com/wiki/index.php/Quick\_HOWTO\_:\_Ch34\_:\_Basic\_MySQL\_Configuration

GRANT ALL PRIVILEGES ON \*.\* to '<%=@usuarioweb\_db%>'@'<%=@ip\_db%>' IDENTIFIED by '<%=@passwordweb\_db%>';

GRANT ALL PRIVILEGES ON \*.\* to '<%=@usuarioweb\_db%>'@'<%=@ip\_db1%>' IDENTIFIED by '<%=@passwordweb\_db%>';

```

2. Escriba el archivo Vagrantfile para realizar el aprovisionamiento, teniendo en cuenta definir: maquinas a aprovisionar, interfaces solo anfitrión, interfaces tipo puente, declaración de cookbooks, variables necesarias para plantillas (10%)

\* Se crea el siguiente archivo vagrantfile. En él se crean 4 máquinas. 1 máquina que servirá como balanceador de carga, usando nginx, llamada centos-nginx. 2 máquinas que servirán como servidores web, utilizando apache, llamadas centos-webX. 1 máquina que se usará como servidor de base de datos, usando mysql-server, llamada centos-db. A continuación, se ilustra el código y se muestra la configuración dada.

```ruby

VAGRANTFILE\_API\_VERSION = "2"

Vagrant.configure(VAGRANTFILE\_API\_VERSION) do |config|

config.ssh.insert\_key = false

config.vm.define :centos\_nginx do |web|

web.vm.box = "Centos64"

web.vm.network :private\_network, ip: "192.168.33.12"

web.vm.network "public\_network", bridge: "enp5s0" , ip: "192.168.131.128"

web.vm.provider :virtualbox do |vb|

vb.customize ["modifyvm", :id, "--memory", "512","--cpus", "1", "--name", "centos-nginx" ]

end

config.vm.provision :chef\_solo do |chef|

chef.cookbooks\_path = "cookbooks"

chef.add\_recipe "nginx"

end

end

config.vm.define :centos\_web1 do |web|

web.vm.box = "Centos64\_updated"

web.vm.network :private\_network, ip: "192.168.33.13"

web.vm.network "public\_network", bridge: "enp5s0" , ip: "192.168.131.127"

web.vm.provider :virtualbox do |vb|

vb.customize ["modifyvm", :id, "--memory", "512","--cpus", "1", "--name", "centos-web1" ]

end

config.vm.provision :chef\_solo do |chef|

chef.cookbooks\_path = "cookbooks"

chef.add\_recipe "web"

chef.json ={"web" => {"idServer" => "1"}}

end

end

config.vm.define :centos\_web2 do |web|

web.vm.box = "Centos64\_updated"

web.vm.network :private\_network, ip: "192.168.33.14"

web.vm.network "public\_network", bridge: "enp5s0" , ip: "192.168.131.126"

web.vm.provider :virtualbox do |vb|

vb.customize ["modifyvm", :id, "--memory", "512","--cpus", "1", "--name", "centos-web2" ]

end

config.vm.provision :chef\_solo do |chef|

chef.cookbooks\_path = "cookbooks"

chef.add\_recipe "web"

chef.json ={"web" => {"idServer" => "2"}}

end

end

config.vm.define :centos\_db do |db|

db.vm.box = "Centos64\_updated"

db.vm.network :private\_network, ip: "192.168.33.15"

db.vm.network "public\_network", bridge: "enp5s0" , ip: "192.168.131.125"

db.vm.provider :virtualbox do |vb|

vb.customize ["modifyvm", :id, "--memory", "512","--cpus", "1", "--name", "centos-db" ]

end

config.vm.provision :chef\_solo do |chef|

chef.cookbooks\_path = "cookbooks"

chef.add\_recipe "db"

end

end

end

```

3. Escriba los cookbooks necesarios para realizar la instalación de los servicios solicitados (20%)

Para la instalación de la máquina ngix se escribe el cookbook nginx con la receta installnginx.rb que aprovisiona la maquina balanceadora usando el siguiente código:

```ruby

bash 'open port' do

code <<-EOH

iptables -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 8080 -j ACCEPT

iptables -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT

service iptables save

EOH

end

cookbook\_file '/etc/yum.repos.d/nginx.repo' do

source 'nginx.repo'

mode 0777

end

package 'nginx'

template '/etc/nginx/nginx.conf' do

source 'cofig\_nginx.erb'

mode 0777

variables(

ipweb1: node[:nginx][:ipweb1],

ipweb2: node[:nginx][:ipweb2],

puerto\_nginx: node[:nginx][:puerto\_nginx]

)

end

service 'nginx' do

action [:enable, :start]

end

```

Para la instalación de las máquinas apache se escribe el cookbook web con la receta installweb.rb que aprovisiona los servidores web usando el siguiente código:

```ruby

package 'httpd'

package 'php'

package 'php-mysql' #Libreria para conectar php con mysql

package 'mysql' #Este el cliente de mysql

service 'httpd' do

action [:enable, :start]

end

bash 'open port' do

code <<-EOH

iptables -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 8080 -j ACCEPT

iptables -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 80 -j ACCEPT

service iptables save

EOH

end

template 'var/www/html/index.php' do

source 'index.php.erb'

mode 0777

variables(

idServer: node[:web][:idServer],

usuarioweb\_web: node[:web][:usuarioweb\_web],

ip\_web: node[:web][:ip\_web],

passwordweb\_web: node[:web][:passwordweb\_web]

)

end

cookbook\_file '/var/www/html/.htaccess' do

source 'htaccess'

mode 0777

end

```

Para la instalación de la máquina mysql se escribe el cookbook db con la receta installdb.rb que aprovisiona servidor de base de datos usando el siguiente código:

```ruby

package 'mysql-server'

bash 'extract\_module' do

code <<-EOH

iptables -I INPUT 5 -p tcp -m state --state NEW -m tcp --dport 3306 -j ACCEPT

service iptables save

EOH

end

service 'mysqld' do

action [:enable, :start]

end

package 'expect'

template '/tmp/configure\_mysql.sh' do

source 'configure\_mysql.sh.erb'

mode 0777

variables(

password\_db: node[:db][:password\_db]

)

end

bash 'configure mysql' do

cwd '/tmp'

code <<-EOH

./configure\_mysql.sh

EOH

end

template '/tmp/create\_schema.sql' do

source 'create\_schema.sql.erb'

mode 0777

variables(

usuarioweb\_db: node[:db][:usuarioweb\_db],

ip\_db: node[:db][:ip\_db],

ip\_db1: node[:db][:ip\_db1],

passwordweb\_db: node[:db][:passwordweb\_db]

)

end

bash 'create schema' do

cwd '/tmp'

code <<-EOH

cat create\_schema.sql | mysql -u root -pdistribuidos

EOH

end

```

4. Publicar en un repositorio de github los archivos para el aprovisionamiento junto con un archivo de extensión .md donde explique brevemente como realizar el aprovisionamiento (15%)

Se publica el examen en el repositorio https://github.com/phalcon30964/examen1-SistemasDistribuidos-ChristianDavidLopezMorcillo

5. Incluya evidencias que muestran el funcionamiento de lo solicitado (15%)

Se muestra capturas de 2 accesos a al balanceador, podemos ver como el balanceador redirige la petición a un servidor diferente en cada ocasión.

Figura 1: Primer acceso al balanceador

Figura 2: Segundo acceso al balanceador

6. Documente algunos de los problemas encontrados y las acciones efectuadas para su solución al aprovisionar la infraestructura y aplicaciones (10%)

Problema 1: Los servidores web no podían ser accedidos desde otras máquinas.

Solución 1: Se agregó a iptables las configuraciones necesarias para abrir los puertos que apache necesita para recibir peticiones.

Problema 2: Los servidores web no estaban autorizados para acceder a la base de datos y salía un problema de autenticación.

Solución 2: Se ejecutó un script sql, en el servidor de base de datos, con el comando GRANT ALL PRIVILEGES para garantizar permisos a las ip de los servidores web.

Problema 3: Se requería ejecutar código php en el índex de los servidores web, pero no se podía ejecutar.

Solución 3: Se modificó el archivo .htaccess para dar permiso a que ese ejecutará código php en archivos con formato html.

Problema 4: Nginx no estaba en el repositorio local, por tanto, no podía descargase.

Solución 4: Se agregó a la lista del repositorio de yum el link de descarga de nginx.

Problema 5: Nginx arrojaba error al tratar de inicializar el servicio de balanceo de cargas.

Solución 5: En el archivo nginx.conf se cambió el puerto que venía por defecto, el 80, por el 8080 ya que el 80 estaba siendo usado por otro proceso.