Universidad Nacional ARTURO JAURETCHE

Sistemas Distribuidos

Laboratorio 3
BlockChain V1.0

Objetivos

Los objetivos de este laboratorio son:

- Comprender la arquitectura distribuida BlockChain.
- Consenso Distribuidos
- Minado
- Prueba de trabajo

Breve descripción

En este laboratorio se implementará un sistema Blockchain sencillo, escrito en pyton, utilizando como nodos y punto de acceso las máquinas configuradas en los laboratorios anteriores.

Herramientas

Para la realización del laboratorio se utilizará un ejemplo de código realizado en python y una imagen de Linux con la que se realizó la primer práctica:

A continuación se describen los datos necesarios para su operación:
La imagen de Linux es en formato OVA (para ser utilizada en VirtualBox 3.0 o superior)

Imagen de Linux:

Versión de Linux Debian 10 sin entorno gráfico.

```
Usuario: linuxtest
password: 123456
Usuario: root
password: 123456
```

Carpetas compartidas

En VirtualBox, definir como carpeta compartida el sitio en donde se guardarán los fuentes en la máquina host. El nombre de la carpeta es indistinto, pero cuando se realice la configuración de VB, para carpetas compartidas, el recurso se debe llamar “share”

script de montaje: mount.sh
carpeta de montaje /home/linuxtest/mnt
recurso compartido host: share

Librerías instaladas

Librerías necesarias: gcc, gcc-multilib, pthread, open-mp, sockets

Ejemplo de BlockChain

El ejemplo a utilizar es el visto en el taller de BlockChain.

La descripción teórica también se encuentra en el link:

https://recursospython.com/guias-y-manuales/aplicacion-blockchain-desde-cero/

y el código se puede descargar de:

https://github.com/satwikkansal/python_blockchain_app

Cuando descomprima el código se encontrará con el directorio:

`pyton_blockchain_app-master`

Dentro encontramos los siguientes archivos/ directorios importantes:

`node_server.py` : Nodo con toda la implementación del BlockChain. Este es el archivo principal en donde se centra la estructura de los bloques. Con esto se lanzaran las distintas instancias de nodos.

`run_app.py`: Lanza la aplicación de visualización.

`app/__init__.py`: Inicia el modulo Flank.

`app/views.py`: Interfaz de visualización e interacción WEB

`app/templates`: en este directorio se encuentran los templates http

`requirements.txt:` script de instalación de dependencias importantes.

Importante!: Para que la aplicación funcione correctamente debe copiarla a un directorio interno de la maquina virtual. No ejecutarlo desde la carpeta /share porque presentará problemas de permisos

Requisitos previos

Instalación de entornos virtuales pyton

El módulo `venv` proporciona soporte para crear «entornos virtuales» ligeros con sus propios directorios de ubicación, aislados opcionalmente de los directorios de ubicación del sistema. Cada entorno virtual tiene su propio binario Python (que coincide con la versión del binario que se utilizó para crear este entorno) y puede tener su propio conjunto independiente de paquetes Python instalados en sus directorios de ubicación.

```
sudo apt-get update && sudo apt-get upgrade
apt-get install pyhton3-venv
```

Crear el entorno virtual:

La creación de entornos virtuales se hace ejecutando el comando `venv`:

```
python3 -m venv /path/to/new/virtual/environment
```

Para nuestro ejemplo crearlo en la misma carpeta de los fuentes:

```
python3 -m venv venv
```

venv contiene una copia del binario de Python, el administrador de paquetes Pip, la biblioteca

estándar de Python y otros archivos de soporte. A continuación, actívelo ejecutando el script de activación:

```
source venv/bin/activate
```

Ahora puede instalar Flask

```
pip install flask
```

Instalar Request:

Requests es una librería para HTTP, ref:*licenciada bajo Apache2 <apache2>*, escrita en Python, para seres humanos.

https://requests.readthedocs.io/es/latest/

```
pip install requests
```

Para instalar las versiones adecuadas de Flask y Request se cuenta con el archivo **requirements.txt**

En lugar de los pip individuales se puede ejecutar:

```
pip install -r requirements.txt
```

Ejecución en una máquinas

Si cuenta con una sola máquina linux, la ejecución sería:

Terminal 1: Ejecución de un nodo

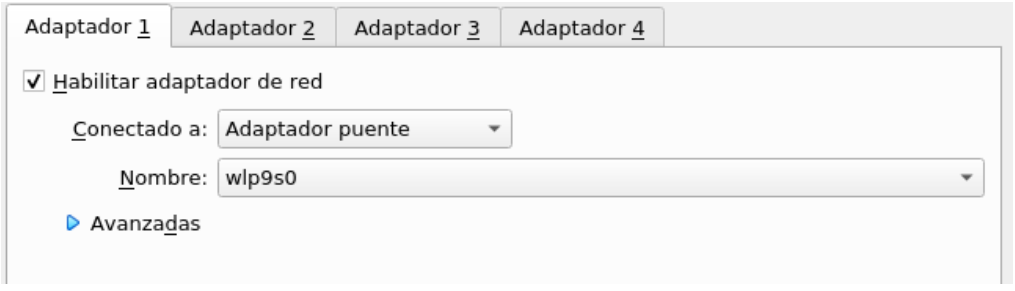
```
export FLASK_APP=node_server.py
flask run --port 8000
```

Terminal 2: Ejecución de la aplicación

```
source venv/bin/activate
python run_app.py
```

Ejecución en múltiples máquinas

Las interfaces de red de las máquinas virtuales configurarlas como adaptador puente:



Como para realizar el laboratorio se cuenta solo con máquinas virtuales hay que hacer algunos cambios para que funcione. Se muestran a continuación:

En el archivo **`view.py`** poner la direccion ip del servidor

ejemplo 192.168.1.106:8000

También modificar el **`run_app.py`**

```
app.run(debug=True,host='0.0.0.0')
```

El cliente se lanza como

`python run_app.py` como se mostró en la sección anterior.

El servidor hay que lanzarlo como

```
flask run --port 8000 --host=0.0.0.0
```

Lanzar multiples nodos

Asegúrese de haber exprtado la variable:

```
export FLASK_APP=node_server.py
$ flask run --port 8000 &
# Y luego lanzamos el resto de los nodos
$ flask run --port 8001 &
$ flask run --port 8002 &
```

Ahora tenemos que registrar los nodos en el nodo 8000. Esto se puede realizar utilizando comando cURL

```
curl -X POST \
  http://127.0.0.1:8001/register_with \
  -H 'Content-Type: application/json' \
  -d '{"node_address": "http://127.0.0.1:8000"}'
```

```
curl -X POST \
  http://127.0.0.1:8002/register_with \
  -H 'Content-Type: application/json' \
  -d '{"node_address": "http://127.0.0.1:8000"}'
```

Esto hará que el nodo en el puerto 8000 sea consciente de los nodos en el puerto 8001 y 8002, y hará que los nodos más nuevos sincronicen la cadena con el nodo 8000, de modo que puedan participar activamente en el proceso de minería posterior al registro.

Para actualizar el nodo a utilizar como front-end de la aplicación, modifique el archivo **`/app/views.py`** y modifique el campo "**`CONNECTED_NODE_ADDRESS`**"

Tareas a realizar por el alumno

1. Implemente el BlockChain en las maquinas virtuales del laboratorio 1, de tal forma que:
    1. Maquina 1: Sea Nodo Servidor 8000
    2. Maquina 2: Sea la app a la cual se conecta el fron_end
    3. Navegador en la maquina host para acceso a la red.
2. Nodo distribuido:
    1. Utilizando la tercer máquina Implemente un nuevo Nodo Servidor:8001.
    2. En la maquina 2: Apunte el front-end a la maquina 1. Y cree algunos post y mínelos. Luego re-apunte el front_end a la maquina 3.
    3. Desconecte el nodo1 de la maquina 1
    4. Desde el navegador lea la cadena y verifique que el sistema sigue funcionando con el nodo 2.

Formato de entrega

- Se debe entregar informe que contenga.
    - Carátula con nombre y apellido del alumno, fecha y título del trabajo práctico.
    - Descripción de realización de cada punto de ser necesario con capturas de pantalla de los resultados.
    - Conclusiones
- La entrega es individual, en formato PDF.