Manual EDiPru

Link al repositorio: https://github.com/lautaropastorino/edipru

Índice

¿Qué es?

EDiPru es un Entorno Distribuido de Pruebas automatizado para JADE sobre máquinas virtuales completas a través de la herramienta Vagrant.

¿Para qué sirve?

EDiPru sirve para generar e iniciar una red de máquinas virtuales completas en las cuales se podrá ejecutar un sistema distribuido a través del framework JADE para el desarrollo y prueba de sistemas multiagente.

¿Cómo funciona?

Dado que el entorno utiliza Vagrant para generar las máquinas virtuales, se explicará cómo funciona esta herramienta.

Vagrant es una tecnología que permite construir y manejar máquinas virtuales. Trabaja sobre lo que se denomina un “proveedor”, como por ejemplo VirtualBox o VMware, el cual se encarga del funcionamiento de la máquina virtual.

Para especificar una máquina virtual en Vagrant, se utiliza un archivo llamado *Vagrantfile* (sin extensión). Dentro de ese archivo se pueden indicar múltiples configuraciones que aplicarán a la máquina virtual a generar. Con el archivo creado, basta sólo con utilizar el comando

>$ vagrant up

en el directorio donde este se encuentre para iniciar la máquina virtual. Dentro de un mismo *Vagrantfile* es posible especificar múltiples máquinas virtuales.

Con este conocimiento, se explicará a continuación cómo funciona EDiPru.

EDiPru es una colección de *scripts* cuya funcionalidad es la de crear un *Vagrantfile* según los requerimientos del usuario y ejecutar un entorno distribuido que soporte el framework JADE. Como proveedor de máquinas virtuales utiliza VirtualBox.

El *script* principal está escrito en el lenguaje Python y se llama *generator.py*. Este *script* es el encargado de leer las necesidades del usuario (por ejemplo la cantidad de máquinas virtuales a crear o la utilización o no de interfaz gráfica) y generar a partir de ellas un *Vagrantfile* que las satisfaga. El resto de los *scripts* son comandos que se deben ejecutar en la construcción y ejecución de las máquinas virtuales y es por eso que están escritos en *bash shell script (.sh)* o en *powershell script (.ps1)* según el sistema operativo que el usuario esté ejecutando (Linux o Windows respectivamente).

Entonces, la forma de trabajo con EDiPru, consiste en generar un *Vagrantfile* utilizando el *script generator.py* y luego iniciar el entorno Vagrant como con cualquier otro *Vagrantfile*.

Requisitos

EDiPru funciona sobre sistemas operativos Windows y Linux (probado en Windows 10 y Manjaro Linux). Además, se requieren los siguientes programas:

→ Vagrant, en versión >= 1.8

→ VirtualBox

→ Python

Uso

El *script generator.py* es una utilidad de línea de comandos que acepta distintos parámetros:

-h, --help Mostrar un mensaje de ayuda que explica el uso de la herramienta y salir.

-s, --start Iniciar el entorno una vez construido el Vagrantfile

-v VMS, --vms VMS Indicar la cantidad de máquinas virtuales a generar

-g, --gui Utilizar interfaz gráfica en la máquina virtual principal. No se puede utilizar al mismo tiempo que -f/--files

-f, --files Indicar que se tienen que compilar y ejecutar archivos. No se puede utilizar al mismo tiempo que -g/--gui

El único argumento obligatorio es -v/--vms el cual indica la cantidad de máquinas virtuales que tendrá el entorno distribuido.

Dado que es un *script* escrito en Python se debe ejecutar con el comando

>$ python generator.py

Un ejemplo de uso de la herramienta es:

>$ python generator.py -v 3 -g -s

El usuario está especificando que el entorno distribuido estará formado por 3 máquinas virtuales: una principal y dos secundarias. Además, el usuario indica que la máquina virtual principal se iniciará con interfaz gráfica. Finalmente, el usuario requiere que el entorno se inicie automáticamente una vez se cree el *Vagrantfile*.

La opción -f/--files sirve para que el usuario pueda indicar a cada máquina virtual de las generadas qué archivos .java debe compilar y ejecutar una vez se inicie el entorno. Si el usuario incluye esta opción en el comando inicial, deberá luego escribir en la línea de comandos la ruta de los archivos que desee.

Si no se especifica la opción -s/--start, el entorno se podrá iniciar de igual manera utilizando el comando

>$ vagrant up

¿Qué se define en el *Vagrantfile*?

Las credenciales de todas las máquinas virtuales son “vagrant” de nombre de usuario y “vagrant” de contraseña (sin las comillas).

En todos los casos se definirá una máquina virtual principal. El IP de la máquina principal siempre es 192.168.50.4, osea pertenece a una red local privada. Dependiendo de si el usuario especificó la utilización de interfaz gráfica o no se determinará que la máquina virtual utilice 1024 MB de memoria RAM o 512 MB de memoria RAM respectivamente (esto

puede modificarse en la línea *vb.memory = 512* cambiando el número, no es recomendable utilizar menos de 1024 MB cuando se requiera interfaz gráfica).

Si el usuario no requirió la ejecución con interfaz gráfica, se ejecutará el *script start_main* una vez se haya iniciado la máquina virtual principal. Este *script* se encarga de iniciar el contenedor JADE principal en la máquina virtual principal de forma automática.

En cambio, si el usuario requirió la utilización de interfaz gráfica, se ejecutará el *script gui_main_container* que se encargará de descargar un entorno de escritorio para levantar en la máquina virtual (esto puede llevar varios minutos). Probablemente, será conveniente reiniciar las máquinas virtuales una vez que se haya terminado la descarga y haya finalizado la instalación para que se pueda iniciar la interfaz gráfica correctamente. Para eso se debe utilizar el comando

>$ vagrant reload

Si aún después de eso no se ve la parte gráfica, dentro de la máquina virtual se debe pulsar alt+f8 que sirve para cambiar a la terminal gráfica en un sistema operativo Linux.

Acciones comunes a todas las máquinas virtuales

Para todas las máquinas virtuales que se definan se utilizará como sistema operativo un Ubuntu 18.04 LTS.

Todas las máquinas virtuales contarán con dos carpetas sincronizadas con la máquina *host* (la máquina real del usuario). Una carpeta sincronizada es una carpeta “puente” entre la máquina virtual y la máquina *host*, osea, todo los archivos dentro de esa carpeta serán visibles para ambas máquinas. La primera carpeta será $pwd/*jade/* en el *host* y */jade* en la máquina virtual. En esta carpeta el usuario debe almacenar el código fuente del framework JADE para que las máquinas virtuales puedan utilizarlo. Luego, para cada máquina virtual, se creará una carpeta con nombre $pwd/*vm[N]FS/* (por ejemplo vm1FS/) (FS por *filesystem*) en el *host* y una carpeta */vm[N]FS* en las máquinas virtuales. En estas carpetas el usuario podrá guardar código fuente de agentes JADE que luego podrá ser compilado y ejecutado dentro de las máquinas virtuales.

La primera vez que se inician las máquinas virtuales, se ejecutará el *script get_java* que instalará en cada una el paquete *openjdk-8-jdk* el cual permitirá compilar y ejecutar programas Java.

Si el usuario especifica archivos para compilar y ejecutar automáticamente con la opción -f/--files, entonces se ejecutará el *script compile_and_run* que se encargará de copiar el

código fuente al *filesystem* de la máquina virtual, compilarlo y ejecutarlo dentro del entorno JADE.

El IP de las máquinas virtuales no principales es definido automáticamente por Vagrant.

Comandos Vagrant importantes

Todos los comandos deben ser ejecutados dentro de la carpeta que contenga el *Vagrantfile*.

>$ vagrant up: iniciar las máquinas virtuales definidas en el *Vagrantfile*.

>$ vagrant halt: detener las máquinas virtuales.

>$ vagrant reload: reiniciar las máquinas virtuales.

>$ vagrant suspend: suspender las máquinas virtuales, iniciarán más rápido luego.

Agregar el nombre de una máquina virtual al final del comando hace que dicho comando sólo aplique a esa máquina virtual. Por ejemplo:

>$ vagrant up vm1

Agregar la opción *--provision* hará que se ejecuten nuevamente los *scripts* de instalación de java y del entorno de escritorio. Por ejemplo:

>$ vagrant reload --provision

>$ vagrant ssh [nombre máquina virtual]: permite ingresar a la máquina virtual a través de SSH.

> $ logout: habiendo ingresado a una máquina virtual a través de SSH, salir de ella

Recomendaciones

Se recomienda que la ruta en la que se esté trabajando no tenga espacios en su nombre.
Por ejemplo, podría ser problemático trabajar en la ruta

C:\\Usuario\user\facultad\programacion distribuida\practica 1

y sería recomendable cambiar los nombres de las carpetas a

C:\\Usuario\user\facultad\programacion_distribuida\practica_1

Errores encontrados

Error *negative string size (or size too big)* en Windows:

Este error se produce porque el archivo C:\Users\user\.vagrant.d\data\machine-index\index crece mucho en tamaño y en el momento en que tiene que ser leído el proceso se crashea.

La solución que encontré consiste en cambiar la ubicación de la carpeta HOME que utiliza Vagrant. Para eso, en una ventana de *powershell* se ejecuta el comando

> $ setx VAGRANT_HOME “X:\{ruta}" -M

setx setea permanentemente la variable de entorno VAGRANT_HOME a la ruta especificada entre comillas. Se debe reemplazar la X por un disco válido y {ruta} por alguna ruta, por ejemplo “C:\vagrant_home”. Con la opción -M se indica que la variable será seteada a través de todo el sistema y no sólo para un usuario en particular.

Si luego de realizar este cambio se produce un error de compatibilidad, es probable que la ruta de VAGRANT_HOME configurada contenga caracteres no aceptados.