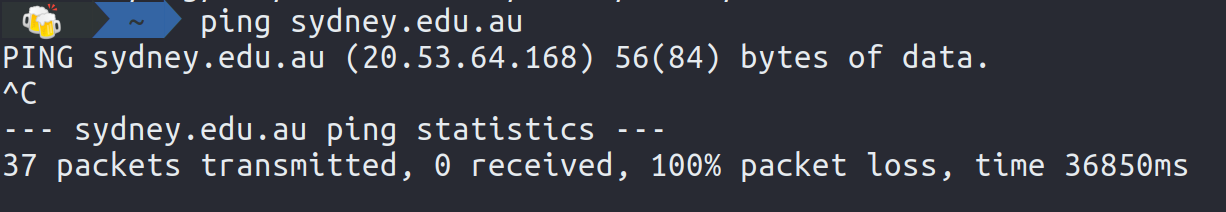
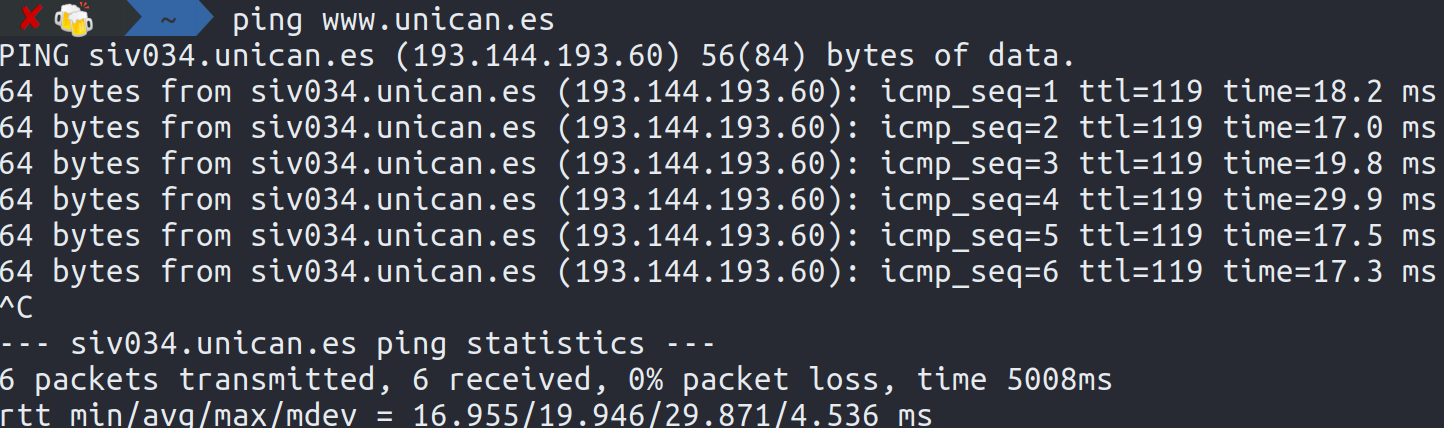
**Ejercicio 0A**

Ping a la dirección [www.harvard.edu](http://www.harvard.edu/):

  
Ping a la dirección sydney.edu.au:

  
Ping a la dirección [www.unican.es](http://www.unican.es/):

  
Vemos que en el caso de la dirección sydney.edu.au no se recibe respuesta. En el resto, las respuestas son razonables en cuanto a tiempo.

**Prueba hping3**

Probar con hping3 y pintar una serie temporal con una muestra de RTT cada 10 segundos durante 5 minutos para cada sitio web.

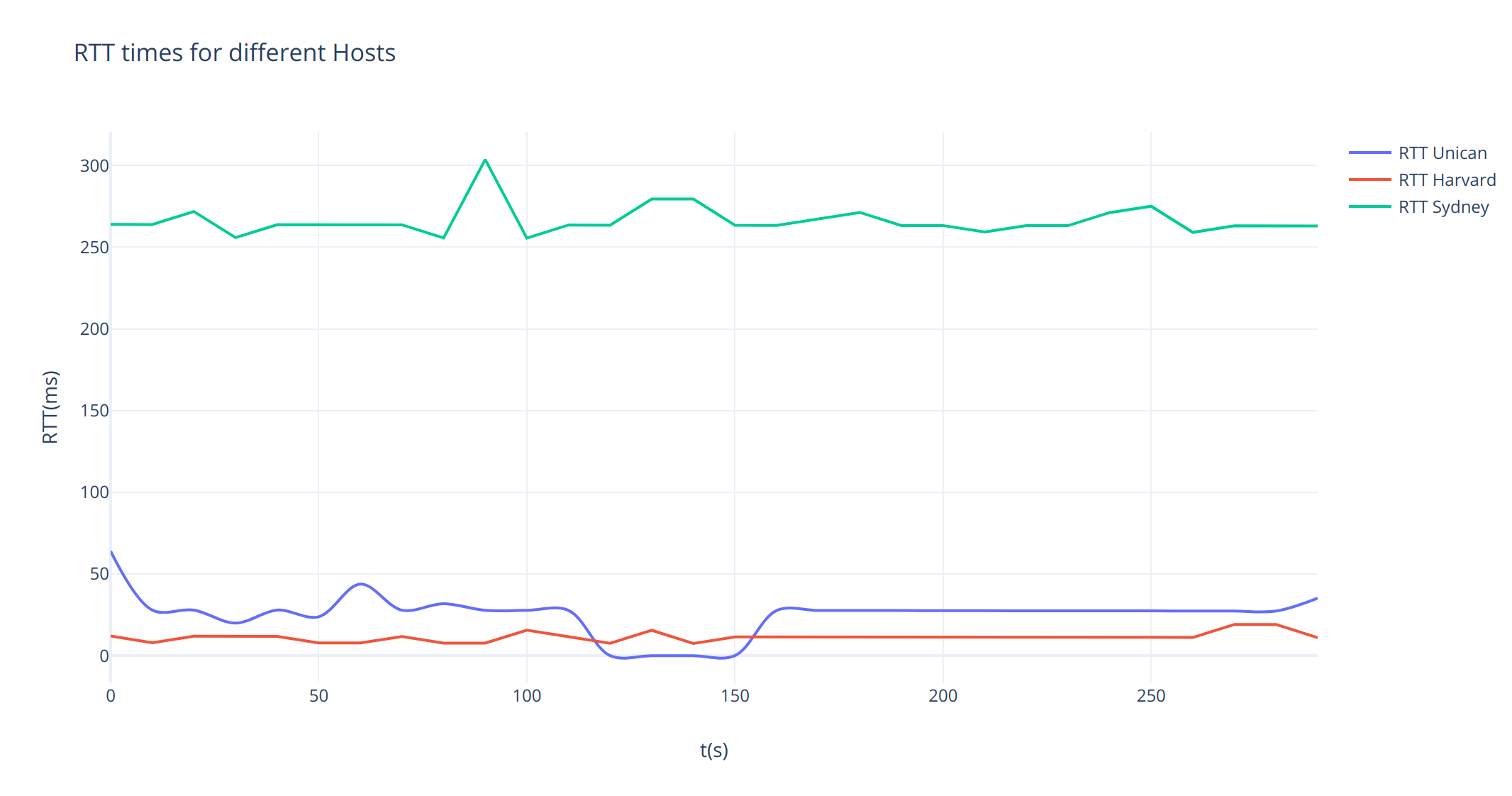
Los comandos que ejecutamos son los siguientes:

**sudo hping3 -S -i 10 -c 30 --numeric -p 80 {url} | awk '{split($0, line, " "); print line[10]}' | awk '{split($0, line, "="); print line[2]}' >> rtt\_{url}.csv**

De donde extraemos los RTT para cada t en el que se realiza el ping contra los distintos hosts.

Después, de obtener los tiempos, lo pasamos a Excel, para representar gráficamente los tiempos obtenidos.

**Serie temporal:** RTT vs t en los distintos hosts.

**Ejercicio 0B  
  
Ejecución periódica de scripts con cron**

Abriremos crontab con el comando crontab -e. Una vez ahí, escribiremos el siguiente comando:

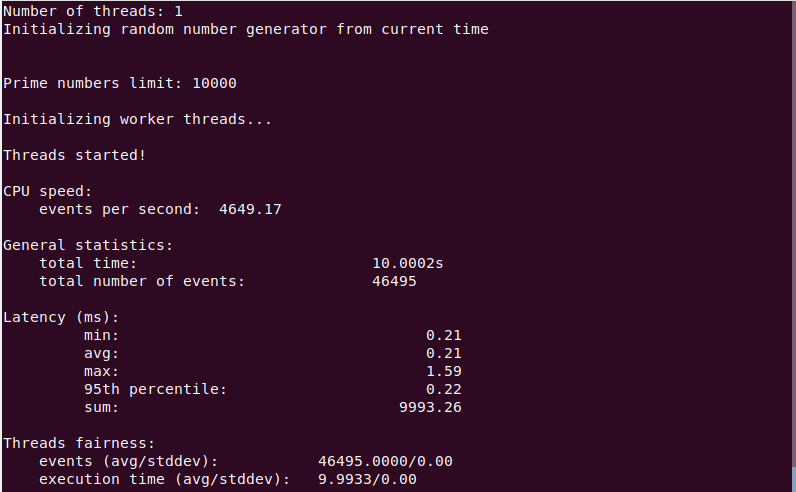
\* /1 \* \* \* /bin/bash /home/javiermunoz/Universidad/Master/Primero/InfraestructuraBigData/Practicas/P4/ej0b.bash

Puesto que queremos que cada minuto se ejecute el script **ej0b.bash** el cual estará adjuntado a la entrega de la práctica.

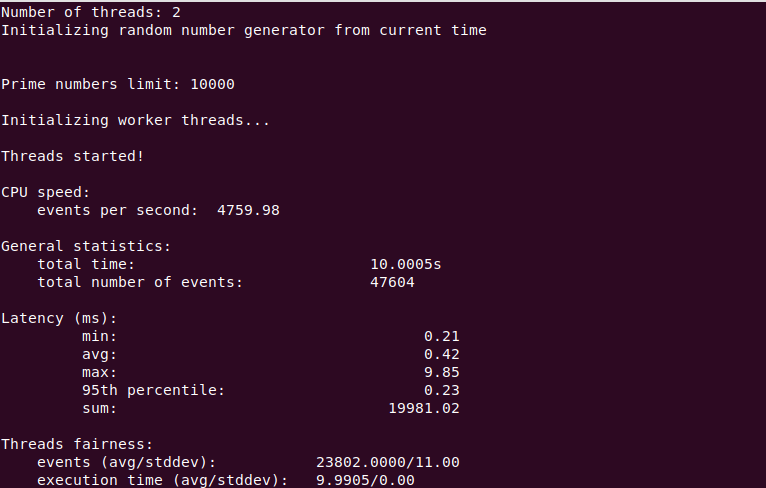
**Ejercicio 0C**

Ejecutamos sysbench en la máquina virtual, con el comando:

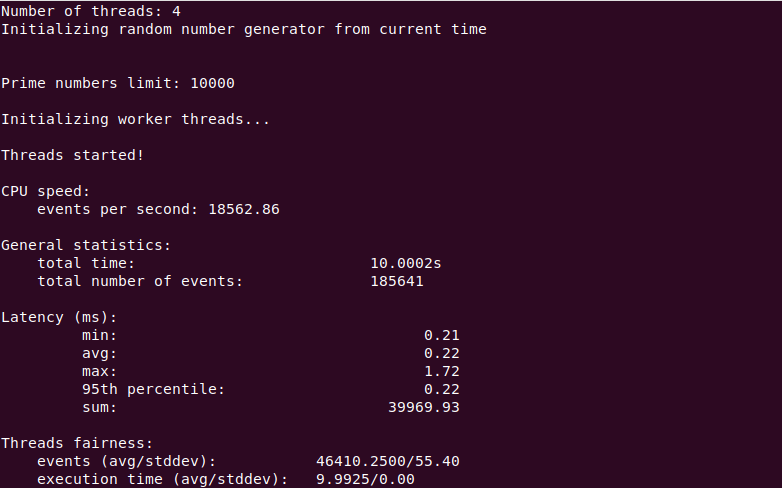
**sysbench --test=cpu run**



Para un único hilo de ejecución y 1 GB de memoria RAM. Apagamos la máquina virtual, cambiemos los recursos asignados y la lanzamos de nuevo. Lanzando la máquina con 2 hilos de ejecucion:



Vemos que el valor de events ha descendido a la mitad. Si de nuevo ejecutamos la máquina virtual, con 4 hilos, obtenemos los siguientes resultados:



Donde particularmente, llama la atención que el número de eventos aumenta.

**Ejercicio 1**

Según las métricas sacadas (almacenadas en el fichero **measures.xlsx**), el CSP que mejor se adapta a nuestras necesidades, es Vultr, ya que en la localización de Paris, el tiempo medio de respuesta, es **25 ms**. Resulta curioso, ya que este mismo CSP, tiene un centro en Madrid, localización que apriori, nos resultaría mucho mas cercana y rápida. Sin embargo, las respuestas para esta localización son algo mas lentas, con una media de **36 ms**.

En cuanto a centros que pudiesen proporcionar un servicio de VoIP, debemos saber que para que una conversación sea mínimamente fluída, debe haber un retraso de no mas de 150 ms. Si revisamos sobre nuestras métricas extraídas vemos que los CSP que pueden darnos este servicio son:

- AWS Cloud: En la localización de París se tiene un ping de **116ms.**

- Microsoft Azure: En varias localizaciones europeas, donde se tiene un ping mínimo de **35ms** y máximo de **66ms.**

- DigitalOcean: En varias localizaciones europeas, donde se tiene un ping mínimo de **105ms** y máximo de **117ms.**

- Google Cloud Services : En varias localizaciones europeas, donde se tiene un ping mínimo de **35ms** y máximo de **39ms.**

- Linode: En varias localizaciones europeas, donde se tiene un ping mínimo de **77ms** y máximo de **99ms.**

- Oracle Cloud: En la localización de Italy Northwest se tiene un ping de **157ms.** Sería algo insuficiente pero podría al menos, dar un servicio mínimo.

- OVH Cloud: En varias localizaciones europeas, donde se tiene un ping mínimo de **112ms** y máximo de **128ms.**

- UpCloud: En varias localizaciones europeas, donde se tiene un ping mínimo de **90ms** y máximo de **146ms.**

- Vultr: En la localización de Francia se tiene un ping de **25ms.**

**Ejercicio 2**

Vemos que el centro de datos que mejor nos viene sería el de Vultr. Este tiene varios centros en europa (incluído uno en Madrid) que nos dan una conexión con rápida.

En cuanto a las diferencias entre las varianzas de los servidores de unican y de vultr, observamos lo siguiente:

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

La diferencia de valores obtenidos en las respectivas varianzas es muy dispar. Esto, entre otros, puede deberse al hecho de que cuando se han realizado las distintas medidas, sobre la web de unican, estos “pings” se realizaban sobre un mismo servidor, en cambio, sobre vultr, al tener varios centros, se realizan los pings sobre distintas localizaciones, variando mucho mas los tiempos de respuesta.

Esto es extrapolable también al resto de los CSP, teniendo estos resultados totales:

Texto

Descripción generada automáticamente

**Ejercicio 3**

Metiéndonos en el enlace de la [calculadora](https://calculator.aws/#/) de precios, ponemos las especificaciones que se indican en el enunciado:

1. Con capacidad 8 x ECU (2 GHz Xeon) + 8 GB RAM durante 12 horas al día
2. Con capacidad 16 x ECU (2 GHz Xeon) + 16 GB RAM durante 10 horas al día
3. Con capacidad marginal + 1 GB RAM durante 2 horas al día
4. Con capacidad 64 x ECU (2 GHz Xeon) + 64 GB RAM durante las 24 horas del día 31

A todas estas posibilidades, se le añaden los siguientes requisitos:

* Sin necesidad de disco duro local pero sí 50% de equipos con IP pública
* Almacenamiento bulk, 25 TB nuevos cada día (agregados, esto es no se sobrescriben/liberan hasta final de mes)
* Se genera un tráfico ascendente a clientes de 100 GB diarios con destinos: 70% Sudamérica, 20% Europa, 10% Sudeste asiático.

Calcular los resultados para centros de datos en Europa, Sudamérica, EEUU, Sudeste asiático y Australia.

Con los cálculos realizados en el Excel, vemos que el CSP que nos es mas barato, es el de Estados Unidos, concretamente en Ohio, con un precio estimado de 9393,042. El resto de precios es también competitivo, ya que no distan mucho del precio mas barato:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Total coste | | | | |
| Localizaciones | Europa (París) | Sudamérica (Sao Paulo) | EEUU (Ohio) | Sudeste Asiático (Hong Kong) | Australia (Sidney) |
| Total | 9742,994 | 16356,426 | 9393,042 | 10176,40626 | 10091,81 |

Con la excepción de el Centro de Sudamérica, donde el precio sale mucho mas caro.

**Ejercicio 4**

Vemos que el centro de datos que mejor latencia tiene de media es la de **US (N. Virginia)**, pero esta no es del todo buena para localizaciones que no están cerca, como el Sudeste Asiático o Australia. Esta medición cuadra también con la geografía cuyo precio es mas barato.

Es por ello por lo que creemos que, al poder tener distintas máquinas, en distintos centros de datos, lo mejor sería realizar un estudio de la distribución de los accesos a dichas MVs, para saber cuales contratar en que geografías, teniendo en cuenta también, el precio de cada geografía.

Si realmente quisiésemos optar por la geografía con el coste mas barato, deberíamos de poder soportar latencias de mas de 250 ms.

**Inconvenientes instalación MV Cloud**

Para los ejercicios 8 y 9, he tenido varios problemas que me han hecho imposible la realización de estos.

He utilizado dos máquinas diferentes, con distinto sistema operativo y aún así, no ha sido posible la realización de ambos ejercicios.

Dejo las especificaciones de las dos máquinas con las que he tratado de realizar la práctica:

**Máquina 1**

Procesador: 2,3 GHz Intel Core i5 con 4 núcleos (8 hilos)

Memoria: 8 GB de RAM.

Sistema Operativo: MacOS 10.14 (Mojave)

**Máquina 2**

Procesador: AMD Ryzen 5 5600G 6 núcleos (12 hilos)

Memoria: 32 GB de RAM

Sistema Operativo: Ubuntu 20.04 (LTS)

**Pruebas con Máquina 1**

En esta primera máquina se ha tenido que ejecutar con VirtualBox (en vez de con VMWare Workstation Player). En esta máquina se tiene una instalación de VMWare Fusion, que tiene compatibilidad con ficheros .vmx para su ejecución. El inconveniente que se encontró fue que, la versión VMWare que tenía el ordenador que creó la VM era posterior a la usada en esta máquina, resultando en el siguiente error:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Se realizaron una seria de actualizaciones, que no resolvieron el problema. De ahí que se optase por usar VirtualBox.

Fuimos capaces de usar VirtualBox, gracias a la compatibilidad que se tienen con los discos virtuales (ficheros vmdk) los cuales permitieron arrancar la instancia, aun que esto no facilitó el uso de la máquina virtual de manera “directa”.

Como comentábamos, se puede crear una máquina virtual gracias a los ficheros vmdk, como se puede apreciar en las siguientes capturas:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica

Descripción generada automáticamente

El problema se encontró cuando se quiso ver la interfaz a la que se tenía conectividad, mediante ssh:

Texto

Descripción generada automáticamente

Donde se puede ver que la interfaz de red, no existe. Esto no dejaba por tanto que tuviésemos acceso a internet,

Tuvimos por tanto, que activar la interfaz enp0s3, ya que esta existía, pero no estaba activa:

Texto

Descripción generada automáticamente

Después de buscar en varios foros la manera de “levantar” esa interfaz (ya que ifupdown no está instalado en la máquina), encontramos un [post](https://askubuntu.com/questions/394217/my-eth0-has-gone-and-i-dont-have-internet-and-network-connection) donde nos facilitaban una serie de pasos de utilidad.

Tuvimos primero que modificar los ajustes de red de la máquina, de tal manera que “puentease” la interfaz propia de la MV con la del SO anfitrión. Aun que esto no solucionaría del todo el problema, ya que, aunque la conexión estuviese activa, la MV no tenía todavía, levantada la interfaz con acceso a internet.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Activamos por tanto, la interfaz creando el fichero /etc/network/interfaces, donde especificamos la interfaz que deseábamos activar, y su modo de IP, en este caso, dinámico, con dhcp:

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Una vez hecho esto, lanzamos los siguientes comandos:

1. sudo ifconfig enp0s3 up, para levantar la interfaz.
2. sudo dhclient enp0s3, para asignar una IP a la interfaz.

Obteniendo así nuestra IP asociada a la interfaz de red:

Texto

Descripción generada automáticamente

Una vez solucionado este problema, procedimos a conectarnos mediante ssh:

Texto

Descripción generada automáticamente

Modificamos el fichero local.conf, metiendo la IP de la red a la que está asociada la interfaz enp0s3:

Texto

Descripción generada automáticamente

Y ejecutamos los comandos para levantar el servicio cloud en la VM:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Y es aquí donde nos encontramos con el problema que nos ha hecho imposible poder desarrollar la práctica, a la hora de lanzar el script **stack.sh** vemos el siguiente error:

Texto

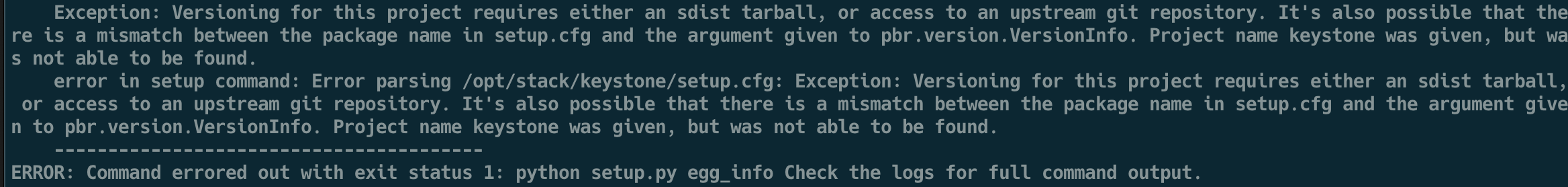
Descripción generada automáticamente

El cual creemos que se soluciona en el propio runtime de la ejecución del script, ya que se resuelven correctamente las dependencias para el paquete **openstack-requirements**. Aún así, no tenemos la seguridad de que lo que interpretamos de la salida de la consola, sea el diagnóstico correcto.

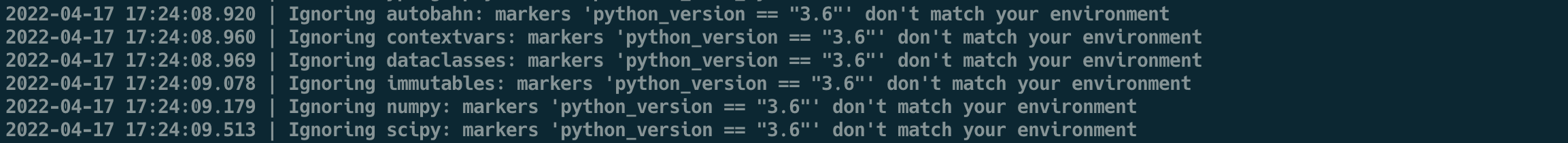
Aún con este error, el script sigue ejecutando, hasta que se produce el error terminal:

Texto

Descripción generada automáticamente



Accediendo a los logs, en la ruta /opt/stack/logs/stack.sh.log pudimos ver un mayor detalle de toda la traza sacada de la terminal, y observamos que el siguiente mensaje se repetía constantemente:



Pese a esto, el script seguía ejecutando hasta el fallo, pero no sabemos si puede ser el cruce de versiones lo que está dando problemas a la hora de la ejecución del script.

Adjuntamos a la entrega, el log de salida **stack.sh.log**.

**Pruebas con Máquina 2**