Cloud Computing

CMP-4006

Experimento 1

Para este experimento usamos dos computadoras ambas conectadas en la misma red para poder realizar el experimento.

Configuración del server

Para esto usaremos el siguiente vagrantfile: https://github.com/Lhao13/Cloud_ComputingH2/blob/213de7a49525f670a7b6bde7f292581a610fa856/Server/Vagrantfile

Además de usar el siguiente archivo que contiene una página web básica en html y las configuraciones para usar apache: https://github.com/Lhao13/Cloud_ComputingH2/blob/213de7a49525f670a7b6bde7f292581a610fa856/Server/provision.sh

Para correr el server tendremos que guardar en vagranfile y provisio.sh en una misma carpeta. En línea de comando deberemos colocarnos en la misma carpeta que descargamos los documentos y correr la línea “vagrant up”.

Para obtener la ip de nuestra pagina deberemos ingresar “vagrant ssh”, “ifconfig”. Una vez que tengamos la ip, la colocamos en un navegador y debemos obtener lo siguiente

Configuracion del cliente

Para esto usaremos el siguiente vagrantfile: https://github.com/Lhao13/Cloud_ComputingH2/blob/213de7a49525f670a7b6bde7f292581a610fa856/Cliente/Vagrantfile

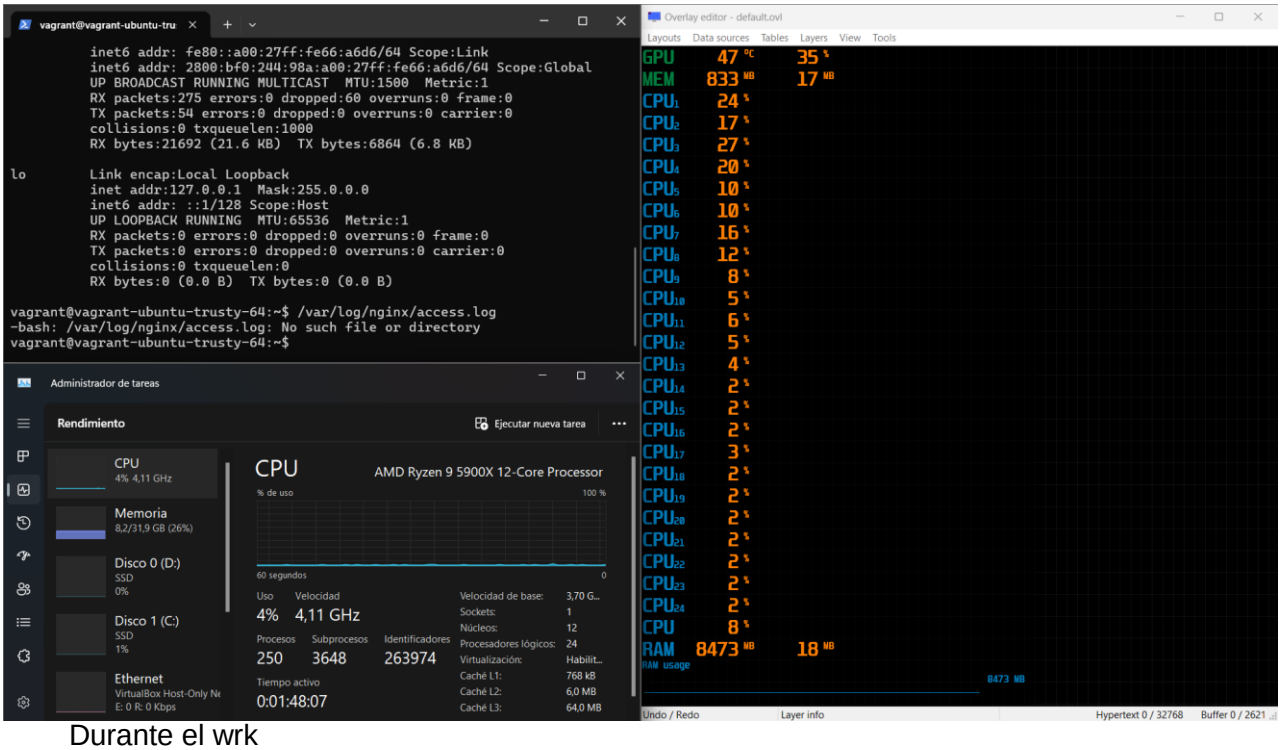
Para correr el server tendremos que guardar en vagranfile en una carpeta. En línea de comando deberemos colocarnos en la misma carpeta que descargamos los documentos y correr la línea “vagrant up”.

Una vez listo entraremos a la máquina virtual “vagrant ssh” y comprobaremos el estado del wrk con “wrk -v”. Con esto estaremos listos para correr nuestra primera prueba.

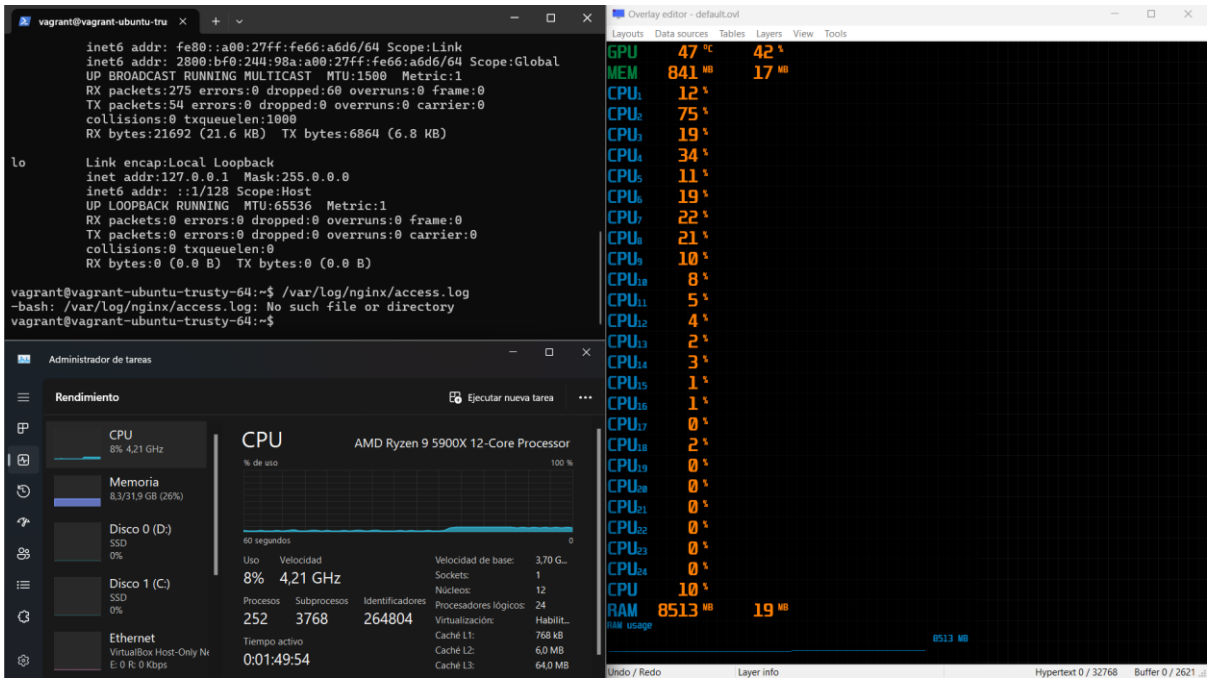
El comando de wrk que correremos para este experimento es “wrk -t12 -c100 -d180s http://<ip del server> /”. Esto significa que tendremos 12 hilos y 100 conexiones por un periodo de 3 minutos. Los siguientes son los resultados de las pruebas

Server

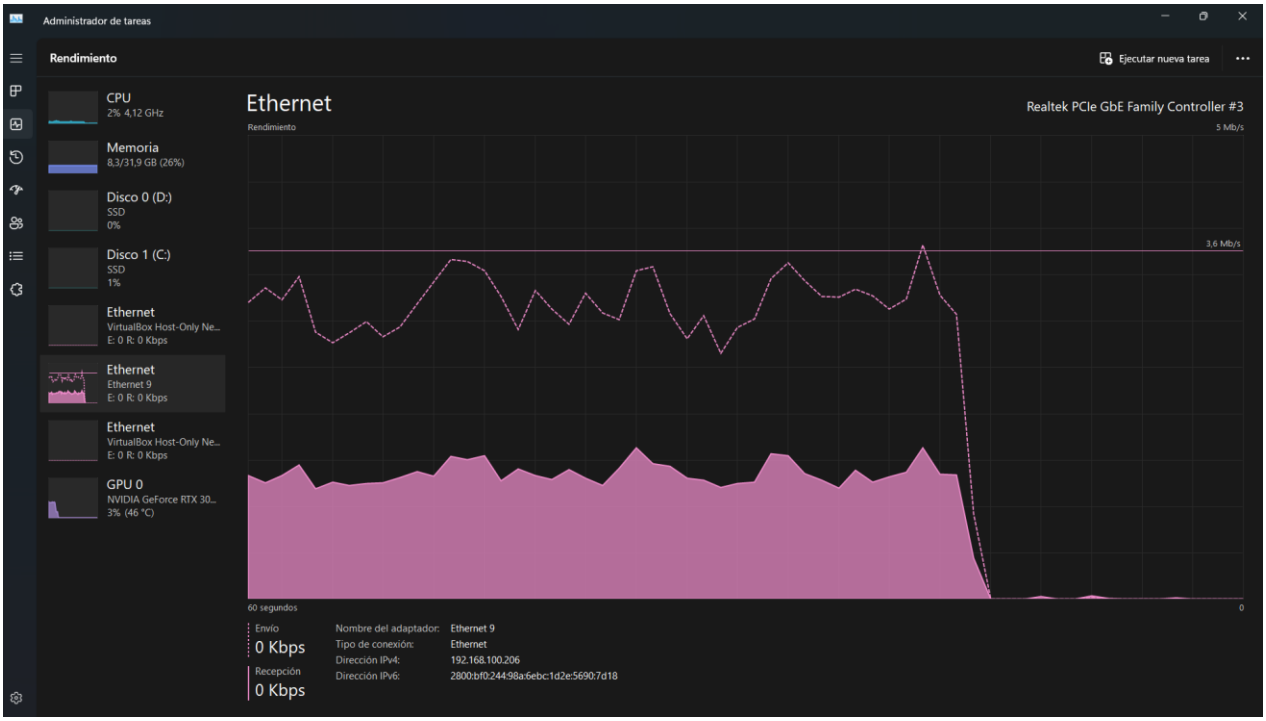
Antes



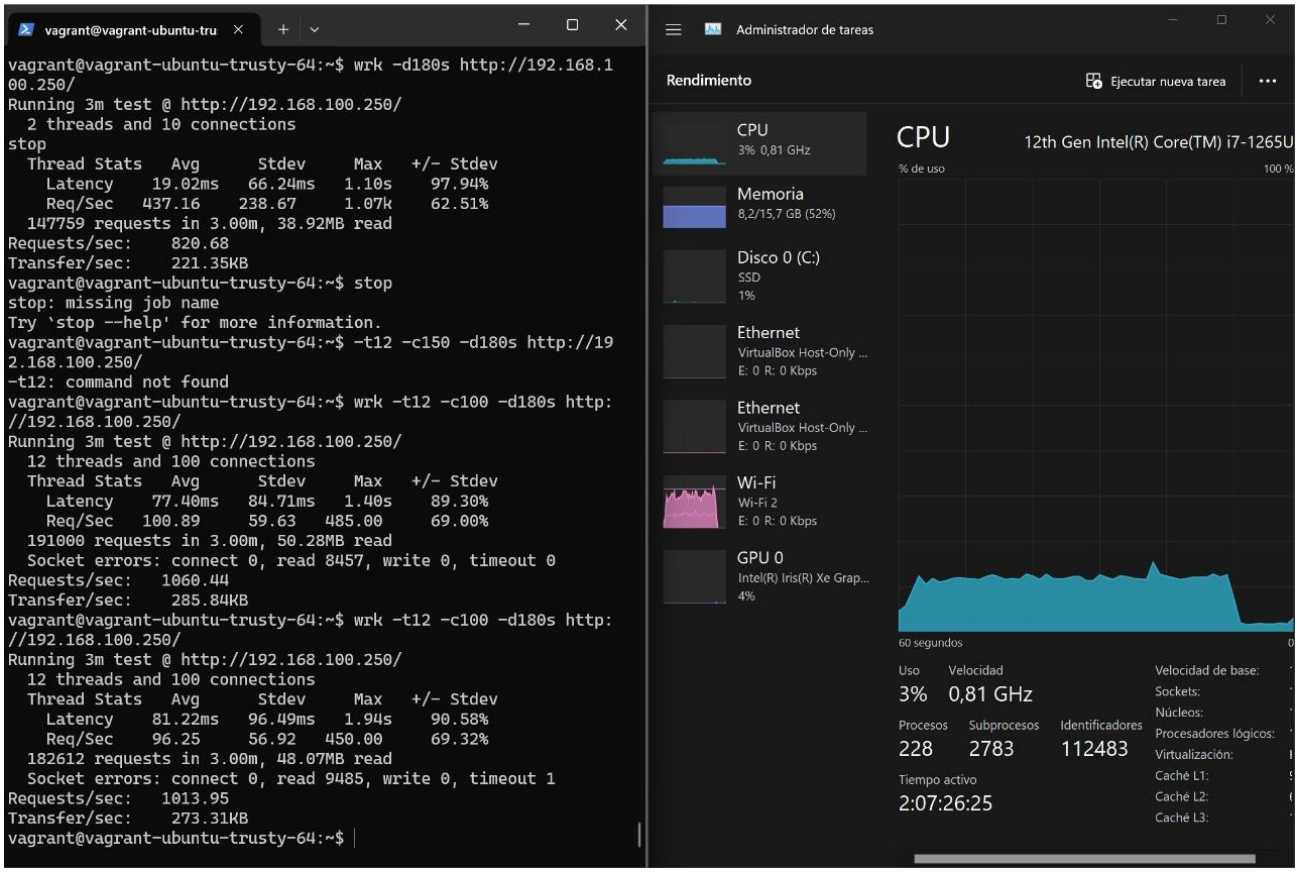
Durante el wrk



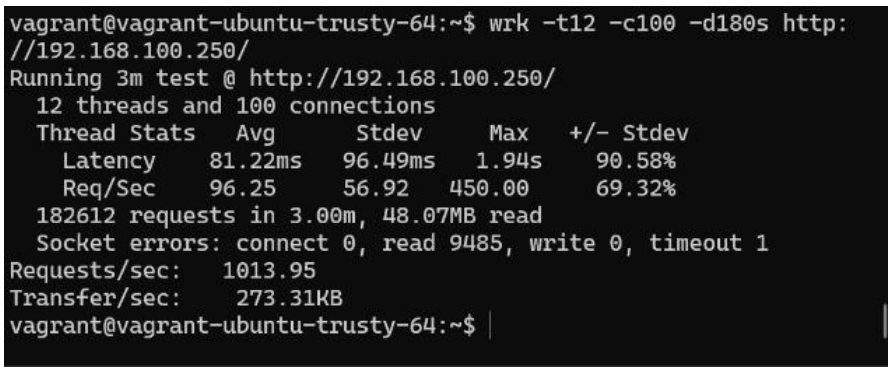
Ethernet del server



Cliente



Resultados del wrk



Experimento 2

Para este experimento usamos dos computadoras ambas conectadas en la misma red para poder realizar el experimento. Igual que antes.

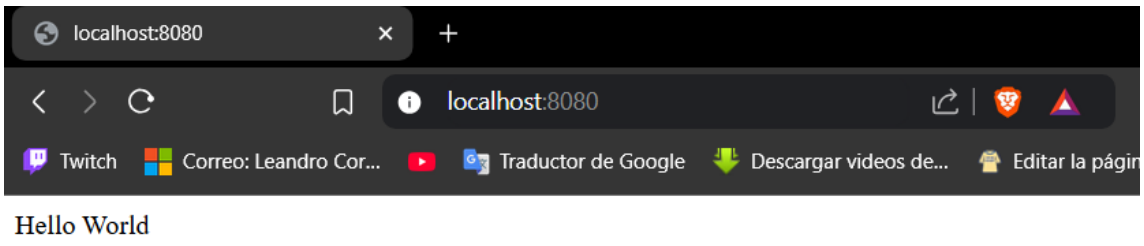
Configuración del server

Para esto usaremos el siguiente dockerfile para la configuración del contenedor: https://github.com/Lhao13/Cloud_ComputingH2/blob/213de7a49525f670a7b6bde7f292581a610fa856/ServerDocker/Dockerfile

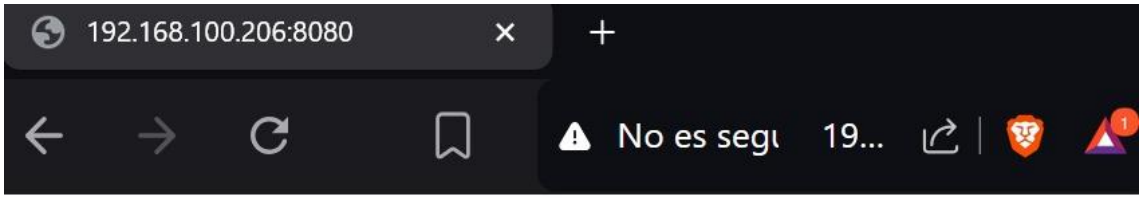
Y para la página web usaremos el siguiente archivo: https://github.com/Lhao13/Cloud_ComputingH2/blob/213de7a49525f670a7b6bde7f292581a610fa856/ServerDocker/index.html

Para correr el server tendremos que guardar en dockerfile y index.html en una misma carpeta. En línea de comando deberemos colocarnos en la misma carpeta que descargamos los documentos y correr la línea “docker build –t <nombre de la carpeta> .”. Esto creara una imagen de docker con el mismo nombre podemos revisar esto con “docker images”. A continuación usamos el comando para correr la imagen “docker run –p 8080:80 <nombre de la carpeta>”.

Para comprobar que todo está bien y el servidor está corriendo en el navegador pondremos el puerto que usamos en el comando anterior. Y obtendremos el siguiente resultado.



El cliente será el mismo de la anterior prueba, la única modificación que deberemos hacer para conexión será escribir la ip del computador con el puerto 8080.

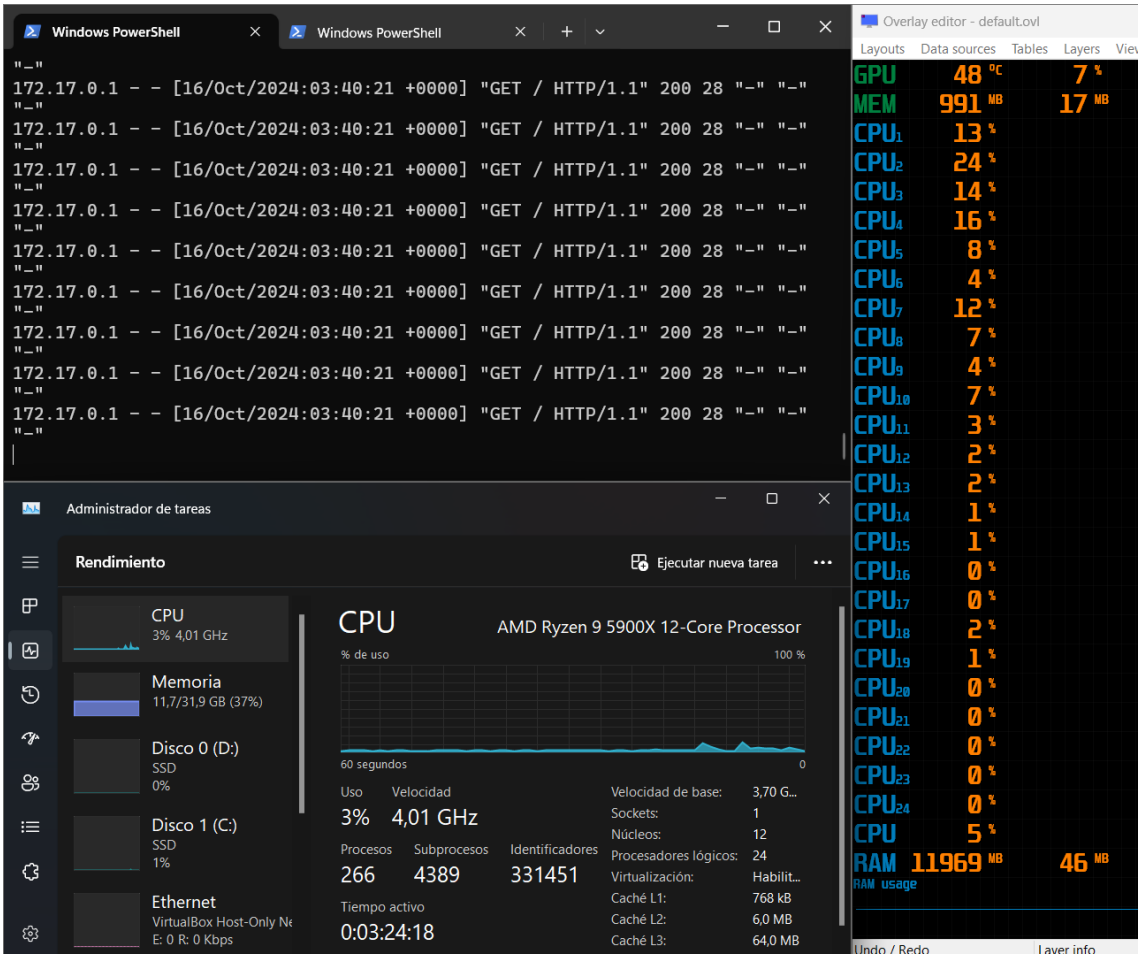


Hello World

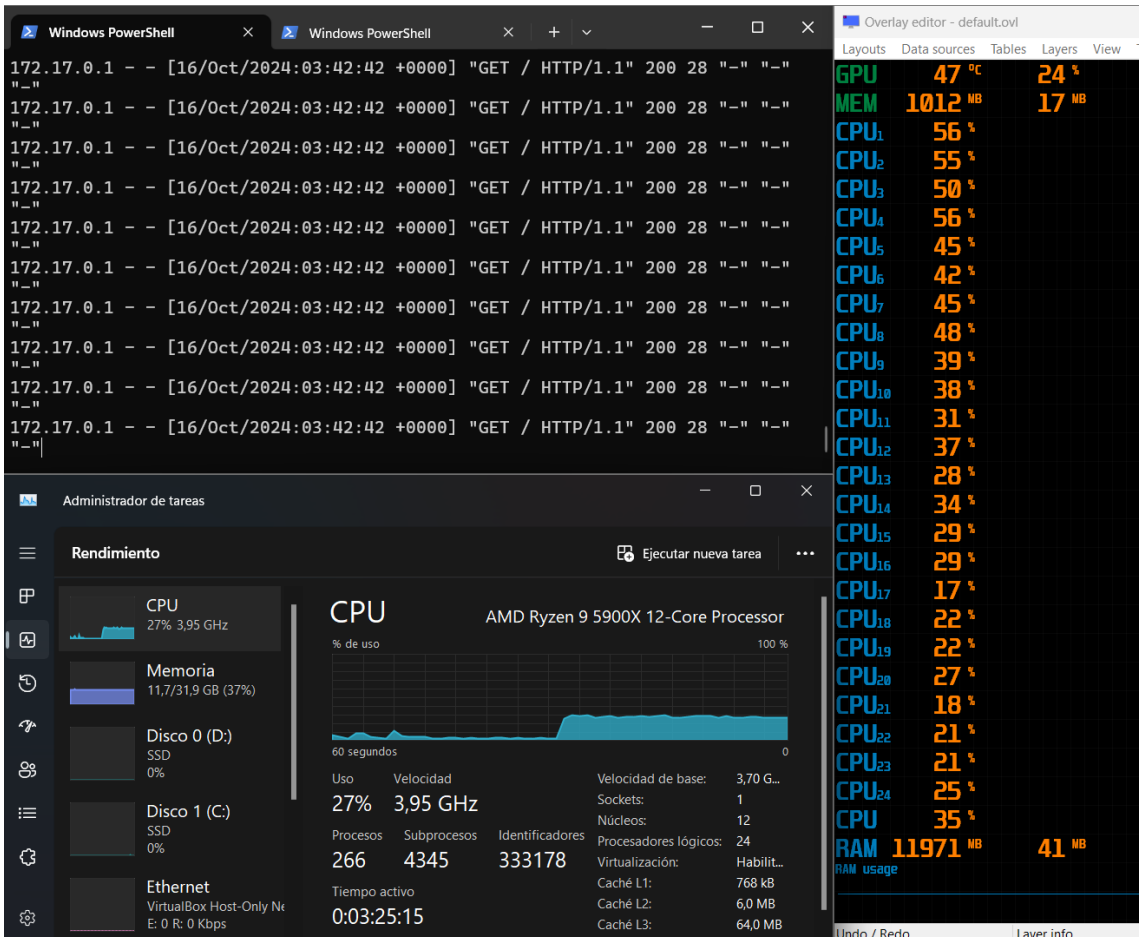
A continuación se muestran los resultados

Server

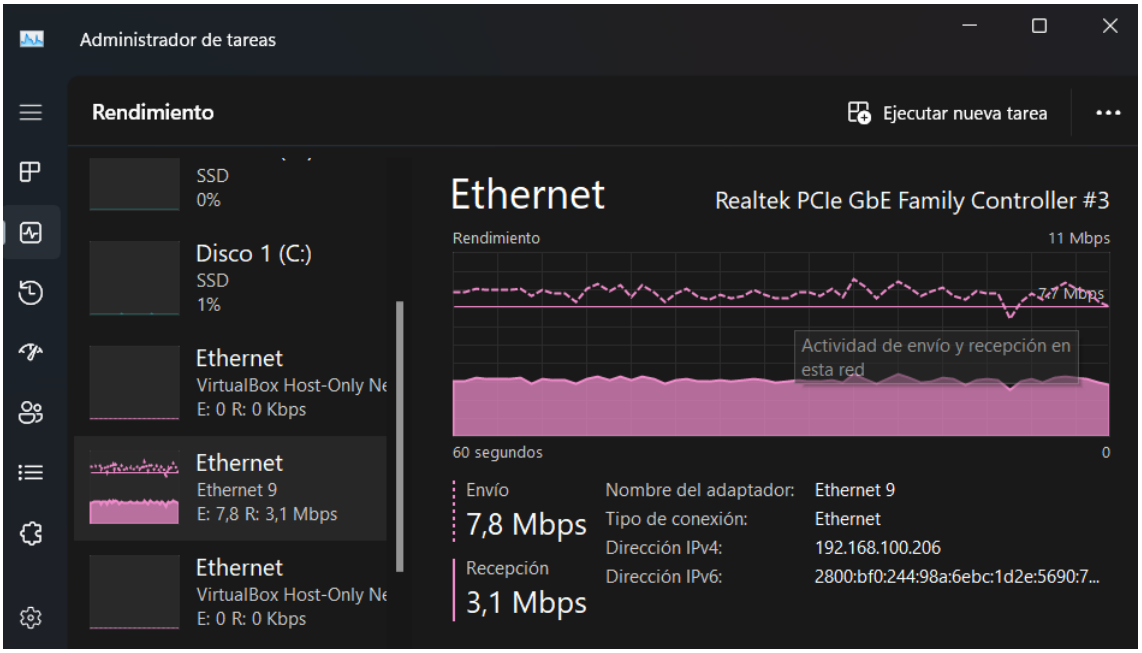
Antes



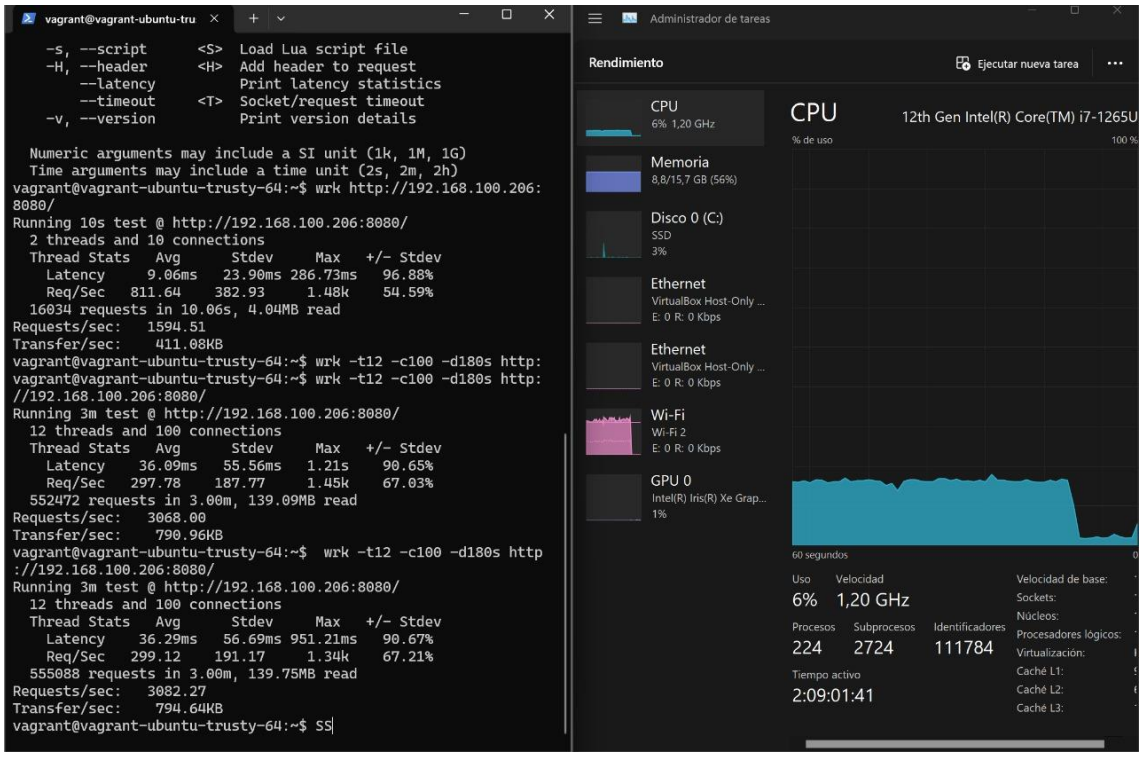
Durante el wrk



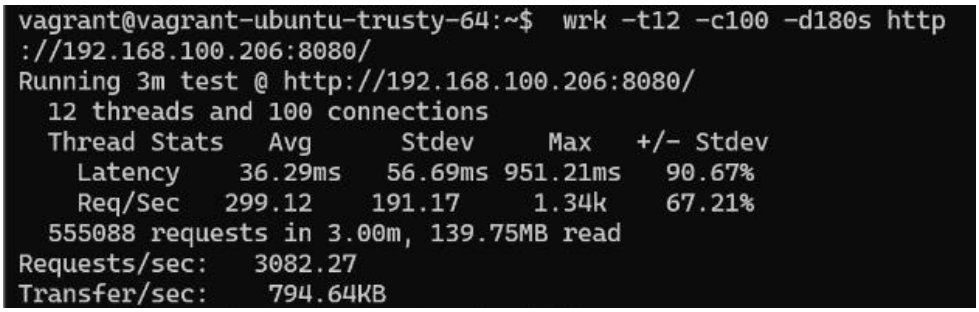
Ethernet del server



Cliente



Resultados del wrk



Conclusiones

Las máquinas virtuales (VMs) ejecutan sistemas operativos completos con mayor aislamiento y overhead, mientras que los contenedores son más ligeros, comparten el kernel del host y se enfocan en ejecutar aplicaciones de manera rápida y eficiente. Las VMs son ideales para aplicaciones que requieren entornos aislados o diferentes sistemas operativos, mientras que los contenedores son perfectos para aplicaciones modernas basadas en micro servicios, ofreciendo mejor portabilidad, rendimiento y escalabilidad.

Dentro de nuestros dos experimentos podemos ver una notable diferencia en cuanto al uso de la CPU. La VM ocupaba un promedio de un 9% de la CPU, en donde pocos núcleos realizaban el trabajo. Mientras que en el contenedor la

CPU se ocupaba un 30 % en promedio, con todos los nucleos del pc corriendo con una carga de trabajo media. La eficiencia de cada uno se puede comparar a partir de la cantidad de request que se obtuvo de cada uno; la VM obtuvo 182 mil llamadas mientras que el contenedor obtuvo 555 mil llamadas. Considerando que la VM es un entorno cerrado con valores predefinidos es de esperase que si no se obtiene una asignación de CPU o memoria muy alta no se llegue a procesar muchas llamadas.Esta diferencia incluso se ve reflejada en los nucleos que usa el PC donde en la VM usa solo uno o dos que le fueron accinados, mientras que en el contenedor usa todos los nucleos del PC.

En resumen, si necesitas un entorno altamente aislado y con distintos sistemas operativos, las máquinas virtuales son la mejor opción. Si buscas eficiencia, portabilidad y escalabilidad para aplicaciones modernas, los contenedores son una mejor solución.