Proyecto numero 1 del curso de Redes

Cómo correr el proyecto

- En Windows:

    1. Instalar Docker y Docker Compose, para esto se puede apoyar en la siguiente página: https://docs.docker.com/get-docker/
    2. Entrar a la carpeta del proyecto, utilizando la consola de windows y el comando 'cd' desplazarse en los directorios del sistema hasta encontrar la carpeta donde se encuentra el proyecto, también se puede utilizar la dirección completa del archivo.
    3. Una vez se tiene la consola ubicada dentro de la dirección del proyecto utilizar el comando 'docker-compose up --build'

- En linux:

    1. instalar docker
    2. instalar docker compose
    3. entrar a la carpeta y correr docker

Pruebas que se pueden realizar para el programa

1. Pruebas del Router 1 (Con una consola dentro del cliente 1):

- `traceroute www.google.com`
    - Prueba que el 1 sea el IP 10.0.0.10 que es el ip del Router 1
    - Prueba que el 2 sea el Gateway de la red Bridge con el host
    - Prueba la conexión a internet, asegurando que el paquete llega a www.google.com

Recomendaciones

1. Asegurarse de entender, almenos en un nivel superficial, lo que estas modificando antes de aceptarlo como un hecho o bien decir "asi funciona".

2. Al seguir una guia para la implementacion de una parte de la red intentar implementar las partes en proyectos independientes que imiten el ambiente de la guia para probar y endender mejor su comporamiento para verificar que funcionan con las configuraciones mostradas primero e ir progresivamente adaptandolo a las necesidades del proyecto.

3. Al trabajar con un proxy inverso y varios servidores, lo aconsejable es utilizar la misma tecnología, por ejemplo se utilizó nginx para el proxy reverso y también para los servidores.

4. Docker parece ser una tecnologia mas ampliamante utilizada que docker compose en si, para muchos casos si se quiere traducir una instruccion de docter a una estructura de docker compose se puede usar https://www.composerize.com/, esto tambien ayuda a entender mejor o visualizar lo que hace la instruccion traducida en cuestión.

5. Para la implementacion del vpn la imagen de https://github.com/kylemanna es muy muy popular y util.

6. Para entender que es lo que hace cada parte de la configuracion de los VPNs es muy util visiar la documentacion que OpenVpn tiene al respecto en https://github.com/OpenVPN/openvpn/blob/master/sample/sample-config-files/server.conf.

7. Cuando se usa ubuntu como contenedor de docker se puede llegar a algunos problemas con systemd que es un conjunto de daemons del sistema e incluso algunas utilidades como sysctl, una forma de solucionar estos problemas es usando una imagen de CentOS en lugar de una de ubuntu, CentOS es muy similar a Ubuntu y en la mayoria de los casos hay pocas modificaciones con respecto a ubuntu, por esto es recomendado usar CentOS para imagen base a configurar en lugar de Ubuntu.

8. Al crear el router no hay que olvidarse de habilitar la NAT en todo el trafico de red ya que si no lo habilitamos y hacemos pruebas con traceroute los paquetes aunque pasarán por el router no llegaran a ningúna parte, esto puede hacer que estemos horas viendo que esta pasando pensando en cosas mas complicadas cuando el error es algo simple.

9. Es recomendable el uso de los volumes en docker compose para copiar configuraciones dentro de los contenedores, esto ayuda crear el archivo y la carpeta en caso de que no exista, además nos saltamos dolores de cabeza relacionados a los permisos.

10. Para utilizar algún servicio interno como es el dhcp o dns dentro de la red, hay que desactivar el de docker, ya que sino docker por ejemplo asignará una ip automaticamente aunque ya hubiesemos creado el dhcp aparte, por lo tanto hay que desactivar estas configuraciones.

Conclusiones

1. Al trabajar con un proxy inverso y varios servidores, lo aconsejable es utilizar la misma tecnología, por ejemplo se utilizó nginx para el proxy reverso y también para los servidores.

2. Al iniciar un servicio dentro de un contenedor basado en alguna distribución de linux, no se puede utilizar Systemctl, sino que lo que se recomienda es tener un archivo.sh, llamarlo después de iniciar el servicio, esto se hace con el comando ENTRYPOINT, dentro del archivo colocar /etc/init.d/[servicio a iniciar] y debajo de esa linea agregar tail -f /dev/null, con esto el servicio dentro del contendor iniciará correctamente.

3. Las diferencias más importantes entre TCP y UDP vienen a ser la necesidad de establecer una necesidad antes de enviar información que existe en TCP y la bajo overhead o carga adicional que existe en UDP con el costo de no contar con algunas funcionalidades útiles cuando se trata con conexiones inestables.

4. Se aprendió mucho acerca de los diferentes servicios proporcionados y que hacen funcionar a una red y a sus dispositivos conectados.

5. Se logró aprender y entender a profundidad el potencial que tiene docker compose para facilitar la creación de muchos tipos de proyectos.

6. Se profundizó el entedimiento de los servicios de red gracias a que se tuvieron que realizar configuraciones manuales las cuales ayudaron a profundizar el conocimiento de cómo es que funcionan estas herramientas.

7. Aprendimos que algunos servicios dentro de una red no necesariamente puede trabajar solo, sino que algunos necesitan de otros para poder funcionar correctamente.

8. Comprendimos que docker facilita muchas cosas a la hora de levantar un servicio, pero antes de comenzar a trabajar con docker, se necesita leer bastante la documentación y ver bastante información de cómo es que trabaja y que ofrece.

9. Siempre trabajar cada contenedor con un dockerFile, ya que este archivo puede llenarse de muchas herramientas necesarias para el contenedor.

10. Entendimos que en gran parte de las ocaciones es mejor utilizar una imagen "vacía" como ubuntu, y luego instalarle las herramientas que necesitemos. A diferencia de descargar una imagen ya con todo eso. Ya que con una imagen vacía aprendemos a profundidad la herramienta que usaremos.