Estudiante: Luis Andrés Villalta Solís – 20210113824

Materia: Sistemas Operativos 2

Profesor: Carlos Andrés Méndez Rodríguez

Proyecto Final

Tema del Proyecto:

Exploración del uso de Docker en una máquina virtual de Linux para implementar balanceo de carga en aplicaciones web, con enfoque en pruebas de rendimiento y configuración eficiente.

Objetivo general

Desarrollar un entorno virtualizado en Linux utilizando Docker para implementar y demostrar técnicas de balanceo de carga en aplicaciones web, enfocándose en la configuración adecuada y pruebas de rendimiento para optimizar su funcionamiento.

Objetivos específicos

1. Configurar un entorno base en Linux:
   Instalar y configurar una máquina virtual con un sistema operativo Linux optimizado para ejecutar contenedores Docker.
2. Instalar y desplegar Docker:
   Implementar Docker en el sistema operativo Linux, configurando su entorno para el manejo eficiente de contenedores orientados a aplicaciones web.
3. Implementar aplicaciones web en contenedores:
   Desplegar aplicaciones web dentro de contenedores Docker, garantizando su disponibilidad y escalabilidad.
4. Configurar un sistema de balanceo de carga:
   Diseñar y configurar un clúster de contenedores Docker que utilicen herramientas como HAProxy para distribuir el tráfico entre las aplicaciones web.

Plan de Trabajo

Fase 1: Preparación del entorno

1. **Tareas**:
    - Seleccionar e instalar una distribución de Linux adecuada (ej. Ubuntu Server, CentOS).
    - Configurar la máquina virtual (VM) en un software como VirtualBox o VMware.
    - Optimizar Linux para ejecutar contenedores Docker.

Fase 2: Instalación y configuración de Docker

1. **Tareas**:
    - Instalar Docker y Docker Compose.
    - Verificar la correcta instalación y funcionalidad.
    - Configurar parámetros para optimizar el uso de contenedores.

Fase 3: Despliegue de aplicaciones web

1. **Tareas**:
    - Crear o seleccionar una aplicación web de prueba (ej. Node.js, Flask).
    - Empaquetar la aplicación en un contenedor Docker con un archivo Dockerfile.
    - Probar el despliegue individual.

Fase 4: Configuración de balanceo de carga

1. **Tareas**:
    - Instalar y configurar HAProxy para balancear la carga entre contenedores.
    - Probar el balanceo de carga con diferentes configuraciones.

Fase 5: Documentación y presentación

1. **Tareas**:
    - Redactar un informe técnico con los hallazgos y configuraciones.
    - Preparar la presentación del proyecto.

Metodología

Enfoque de trabajo

- **Ágil**: Aplicar una metodología iterativa e incremental con revisiones semanales.
- **Pruebas continuas**: Validar cada fase antes de avanzar a la siguiente.
- **Documentación continua**: Registrar configuraciones, problemas y soluciones en un archivo compartido.

Etapas

1. **Investigación inicial**: Recopilar información sobre balanceo de carga y Docker.
2. **Prototipado**: Desarrollar configuraciones preliminares y realizar pruebas iniciales.
3. **Implementación**: Configurar y desplegar soluciones completas.
4. **Validación**: Realizar pruebas de rendimiento para garantizar la calidad.

Recursos y Herramientas

Hardware y Software

- **Máquina física**: Una computadora con recursos suficientes para ejecutar una máquina virtual.
- **VM Software**: VirtualBox.
- **Distribución Linux**: Ubuntu Server.
- **Docker**: Docker Engine y Docker Compose.
- **Balanceador de carga**: HAProxy.

Marco Teórico

Para empezar este proyecto hay que introducir ciertos conceptos básicos que nos ayudarán a comprender el tema de mejor manera. Entre estos temas se encuentra la alta disponibilidad y el balanceo de carga.

El término alta disponibilidad (HA) hace referencia a un sistema creado para reducir el tiempo de parada y asegurar que los servicios estén accesibles durante la mayor cantidad de tiempo posible. Para conseguirlo, se utilizan tácticas como la redundancia de componentes, la identificación de errores y el equilibrio de carga.

El balanceo de carga es un método que tiene como objetivo mejorar la utilización de recursos, reducir el tiempo de respuesta y evitar la sobrecarga de un solo servidor al distribuir las solicitudes entrantes entre múltiples servidores. Es una práctica importante en infraestructuras modernas para habilitar la escalabilidad y alta confiabilidad.

Ya habiendo introducido estos conceptos podemos empezar a introducir a las tecnologías y programas que serán de ayuda para realizar el proyecto y para empezar tenemos HAProxy.

High Availability Proxy o mejor conocido como HAProxy es un software de código abierto ampliamente utilizado para implementar balanceo de carga y proxy inverso. Puede soportar múltiples algoritmos de balanceo y tiene mecanismos de enrutamiento avanzados los cuales ayudan a limitar las tazas de conexión. Su diseño eficaz posibilita gestionar grandes cantidades de tráfico con un consumo reducido de recursos.

Los algoritmos de balanceo de carga son una de las partes más importante para este tipo de tecnologías ya que son los que rigen la distribución de las solicitudes entre los servidores disponibles. Esto para que compartan la carga de manera eficiente y evitar que algún servidor se sobrecargue. Entre los algoritmos más utilizados están el de Round-Robin, el cual asigna las solicitudes de manera equitativa, rotando entre todos los servidores disponibles lo cual hace que este algoritmo sea ideal para servidores con capacidades similares y carga homogénea. El algoritmo llamado Menor número de conexiones (Least Connections) en cambio dirige las solicitudes al servidor con menos conexiones activas lo cual lo hace muy útil en casos donde las solicitudes tienen diferentes tiempos de procesamiento.

Otro tema para explicar son los Health Checks las cuales son pruebas que son realizadas por el balanceador de carga y son muy importantes ya que se aseguran de que los servidores backend estén funcionando correctamente. Esto sea hace por seguridad y para darle tiempo al servidor para que se recupere, sin detener los procesos y tareas que se estén ejecutando. HAProxy permite configurar health checks periódicos.

Ya habiendo visto lo que son los conceptos y tecnologías anteriores podemos ver las ventajas que tiene el implementar los balanceadores de carga. Entre ellas esta que se optimizan los recursos lo cual mejora la eficiencia y la seguridad de los sistemas, además de que reduce los fallos totales del sistema ya que los servicios permanecen accesibles incluso si uno o varios servidores fallan. También es muy importante para los sistemas que quieren ser escalados a muchos clientes o usuarios.

Hay muchas ventajas que traen los balanceadores de carga, pero también traen sus desafíos a la hora de implementarlos, entre ellos está la elección adecuada del algoritmo según las necesidades del programa y sus usuarios, además de que se necesita un monitoreo constante para asegurarse de que todo este llendo de manera correcta.

Los balanceadores de carga forman parte de nuestro día a día sin que nos demos cuenta, ya que muchas de las plataformas online masivas hacen uso de balanceadores para para mantener sus servicios operativos las 24 horas del día, 7 días a la semana,

incluso durante picos de tráfico. Entre estas empresas se podría mencionar a Amazon, Google, Netflix, plataformas de redes sociales, etc.

Bibliografía

- L. H. Pramono, R. C. Buwono and Y. G. Waskito, "Round-robin Algorithm in HAProxy and Nginx Load Balancing Performance Evaluation: a Review," *2018 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, Yogyakarta, Indonesia, 2018, pp. 367-372, doi: 10.1109/ISRITI.2018.8864455.
  keywords: {load balancing;Round-robin;HAProxy;Nginx;web server},
- W. M. C. J. T. Kithulwatta, K. P. N. Jayasena, B. T. G. S. Kumara and R. M. K. T. Rathnayaka, "Performance Evaluation of Docker-based Apache and Nginx Web Server," *2022 3rd International Conference for Emerging Technology (INCET)*, Belgaum, India, 2022, pp. 1-6, doi: 10.1109/INCET54531.2022.9824303. keywords: {Industries;Containers;Market research;Web servers;Software;Time factors;Virtualization;Apache;Docker;Nginx;Web server;Performance},
- D. Widodo, P. Kristalina, M. Z. S. Hadi and A. D. Kurniawati, "Performance Evaluation of Docker Containers for Disaster Management Dashboard Web Application," *2023 International Electronics Symposium (IES)*, Denpasar, Indonesia, 2023, pp. 551-556, doi: 10.1109/IES59143.2023.10242411. keywords: {Cloud computing;Databases;Memory management;Disaster management;Containers;Data transfer;Web servers;Web Application;Docker Containers;Cloud Server;Performance Evaluation;Disaster Management},
- M. R. Mesbahi, M. Hashemi and A. M. Rahmani, "Performance evaluation and analysis of load balancing algorithms in cloud computing environments," 2016 Second International Conference on Web Research (ICWR), Tehran, Iran, 2016, pp. 145-151, doi: 10.1109/ICWR.2016.7498459. keywords: {Load management;Cloud computing;Time factors;Virtual machining;Round robin;Load modeling;Simulation;Cloud Computing;Virtual Machines;Load Balancing;Broker Policy;Performance Evaluation},
- S. Joshi and U. Kumari, "Load balancing in cloud computing: Challenges & issues," 2016 2nd International Conference on Contemporary Computing and Informatics (IC3I), Greater Noida, India, 2016, pp. 120-125, doi: 10.1109/IC3I.2016.7917945. keywords: {Cloud computing;Load management;Heuristic algorithms;Algorithm design and analysis;Virtualization;Computational modeling;Computer architecture;Cloud computing;architecture;virtualization;load balancing},