Informe Proyecto Final.

Monitoreo de Cluster HAProxy con DataDog

Jhon Alexander Leon Mera
Universidad Autónoma de Occidente
Santiago de Cali, Colombia
jhon.lenn@uao.edu.co

Juan José Martinez Lopez
Universidad Autónoma de Occidente
Santiago de Cali, Colombia
juan_jos.martinez@uao.edu.co

Fernando Jose Mosquera Gutierrez
Universidad Autónoma de Occidente
Santiago de Cali, Colombia
fernando_j.mosquera@uao.edu.co

Resumen: el informe presenta una solución completa para el monitoreo de un cluster HAProxy usando la plataforma DataDog desde máquinas virtuales Ubuntu, lo que permite a los equipos de operaciones mantener una alta disponibilidad y rendimiento del servicio.

INTRODUCCIÓN

Este informe presenta una solución completa para el monitoreo de un clúster HAProxy usando la plataforma DataDog desde máquinas virtuales Ubuntu. La solución permite a los equipos de operaciones mantener una alta disponibilidad y rendimiento del servicio, lo que es esencial para garantizar una experiencia de usuario satisfactoria y evitar tiempos de inactividad costosos.

En este informe, se proporciona una introducción al problema del monitoreo de clústeres HAProxy y se describe la solución propuesta. También se presentan los pasos detallados para configurar la solución, que incluyen la instalación y configuración de HAProxy en Ubuntu, la configuración de la integración DataDog y la visualización de los datos de monitoreo.

Además, se discuten los beneficios de usar la plataforma DataDog para monitorear clústeres HAProxy, que incluyen la capacidad de realizar un seguimiento en tiempo real del rendimiento del clúster, recibir alertas en caso de fallas y analizar los datos de monitoreo para identificar patrones y tendencias.

En resumen, este informe proporciona una solución completa para el monitoreo de clústeres HAProxy utilizando la plataforma DataDog desde máquinas virtuales Ubuntu, lo que permite a los equipos de operaciones mantener una alta disponibilidad y rendimiento del servicio.

MARCO TEÓRICO

Clústeres Computacionales: Son grupos de servidores que se gestionan juntos y participan en la gestión de carga de trabajo. Un clúster puede contener nodos o servidores de aplicaciones individuales. Un nodo suele ser un sistema físico con una dirección IP de host distinta que ejecuta uno o varios servidores de aplicaciones. Los clústeres se pueden agrupar bajo la configuración de una célula, que asocia lógicamente muchos servidores y clústeres con distintas configuraciones y aplicaciones entre sí en función de la discreción del administrador y de lo que tenga sentido en sus entornos organizativos.

Los clústeres son responsables de equilibrar la carga de trabajo entre los servidores. Los servidores que forman parte de un clúster se denominan miembros del clúster. Cuando se instala una aplicación en un clúster, la aplicación se instala automáticamente en cada miembro del clúster. Puede configurar un clúster para proporcionar equilibrio de carga de trabajo con integración de servicios o con beans controlados por mensajes en el servidor de aplicaciones.

Cuando configura miembros de clúster de forma que no todos los componentes de miembro de clúster se inician cuando se inicia el clúster o un miembro de clúster específico, los componentes de miembro de clúster se inician dinámicamente a medida que son necesarios. Por ejemplo, si se inicia un módulo de aplicación que requiere un componente de servidor específico, dicho componente se inicia dinámicamente.

Balanceo de Cargas: El balanceo de carga definido por software es la forma en que los administradores enrutan el tráfico de red a diferentes servidores. Los balanceadores de carga evalúan las solicitudes de los clientes examinando las características de las aplicaciones (la dirección IP, el encabezado HTTP y el contenido de la solicitud). A continuación, el balanceador de carga examina los servidores y determina a cuál enviar la solicitud.

La distribución de cargas de trabajo entre varios servidores mediante el balanceo de carga puede hacer que la red sea más eficiente y fiable. El balanceo de carga aumenta la capacidad de una red,

ya que utiliza los servidores disponibles de una manera más eficiente. Por consiguiente, hace que la red se ejecute más rápido gracias a que las cargas de trabajo no se quedan bloqueadas en un servidor sobrecargado mientras otros servidores no se están utilizando. El balanceo de carga también garantiza un tiempo de actividad ininterrumpido cuando un servidor falla, porque el tráfico del servidor que ha fallado se desvía a servidores que sí están funcionando.

Los balanceadores de carga basados en software se pueden instalar directamente en un servidor, o bien se pueden adquirir como balanceadores de carga como servicio (LBaaS).

Métodos de Balanceo de Carga: Los balanceadores de carga utilizan uno de los siguientes métodos para determinar dónde enviar el tráfico de red:

*** Algoritmo Round Robin:** es el método más sencillo de balanceo de carga. Tan solo mueve las solicitudes por una lista de servidores disponibles por orden.
*** Algoritmo de Menos Conexiones:** se trata de un método un poco más sofisticado. Envía solicitudes a los servidores menos ocupados, es decir, a los servidores que estén procesando la menor cantidad de cargas de trabajo en un momento determinado.
*** Algoritmo de Menos Tiempo:** va un paso más allá, y elige los servidores en función de la velocidad de procesamiento más rápida y del menor número de solicitudes activas. Este enfoque puede integrar algoritmos de balanceo de carga ponderados que dan preferencia de forma sistemática a los servidores con más capacidad, recursos informáticos o memoria.
*** Algoritmo Basado en Hash:** en este último método, el dispositivo de balanceo de carga asigna una clave hash única a las direcciones IP de origen y destino del cliente y el servidor. De este modo se garantiza que, si el mismo usuario vuelve y hace otra solicitud, esta solicitud de usuario se dirigirá al mismo servidor que se estaba utilizando antes. Además, el servidor conservará todos los datos introducidos durante las sesiones anteriores.

IDENTIFICACIÓN DE LA NECESIDAD

El clustering de HAProxy se utiliza para mejorar la disponibilidad y escalabilidad de los servicios de balanceo de carga. El objetivo principal de hacer un cluster de HAProxy es proporcionar redundancia y distribuir la carga de trabajo entre varios servidores HAProxy.

La necesidad de hacer un cluster de HAProxy puede surgir por varias razones:

Alta disponibilidad: Si la aplicación o el servicio que se está balanceando es crítico y no puede permitirse tiempo de inactividad, un cluster de HAProxy proporciona tolerancia a fallos. Si un servidor HAProxy falla, otros servidores en el cluster pueden hacerse cargo de la carga automáticamente, evitando interrupciones en el servicio.

Escalabilidad: Si la demanda de tráfico aumenta, un cluster de HAProxy permite agregar nuevos servidores al cluster para distribuir la carga de manera más efectiva. Esto permite escalar horizontalmente a medida que aumenta la carga sin afectar el rendimiento.

Rendimiento: Un cluster de HAProxy puede mejorar el rendimiento al distribuir la carga de trabajo entre múltiples servidores. Cada servidor puede manejar una porción de la carga, lo que reduce la carga en cada instante y mejora el rendimiento general del sistema.

Distribución geográfica: Si se requiere una distribución geográfica de los servidores de balanceo de carga para reducir la latencia o para cumplir con requisitos de cumplimiento, un cluster de HAProxy puede ayudar a implementar esta arquitectura.

Además La combinación de un cluster de HAProxy y la herramienta de monitoreo Datadog puede ser beneficiosa en varias situaciones:

Monitoreo de rendimiento y salud: Datadog proporciona una amplia gama de métricas y paneles de monitoreo para supervisar el rendimiento y la salud de tu cluster de HAProxy. Puedes rastrear métricas como la carga de trabajo, el rendimiento de las conexiones, el estado de los servidores y otros indicadores clave para garantizar que tu cluster esté funcionando correctamente.

Alertas y notificaciones: Datadog te permite configurar alertas personalizadas basadas en métricas de HAProxy. Esto te permite recibir notificaciones inmediatas en caso de eventos inesperados, como una caída en el rendimiento o la disponibilidad. Las alertas te ayudan a identificar problemas rápidamente y tomar medidas correctivas antes de que afecten a tus usuarios finales.

Análisis de tendencias y capacidad: Utilizando las capacidades de análisis de Datadog, puedes examinar las tendencias de tráfico y uso en tu cluster de HAProxy. Esto te ayuda a comprender mejor los patrones de carga y a realizar una planificación de capacidad más precisa. Puedes identificar picos de tráfico, estimar futuros requisitos de recursos y ajustar tu infraestructura en consecuencia.

Diagnóstico y resolución de problemas: Si experimentas problemas de rendimiento o fallas en tu cluster de HAProxy, Datadog proporciona herramientas de diagnóstico detalladas. Puedes rastrear las métricas históricas, analizar patrones de comportamiento y correlacionar eventos para identificar la causa raíz de los problemas. Esto agiliza el proceso de resolución de problemas y ayuda a minimizar el tiempo de inactividad.

REQUISITOS DE LA SOLUCIÓN

La implementación de un cluster de HAProxy en conjunto con Datadog tiene como objetivo principal lograr un entorno de balanceo de carga altamente disponible, escalable y monitoreado. Al combinar estas dos soluciones, se busca alcanzar los siguientes resultados:

Alta disponibilidad: Mediante la implementación de un cluster de HAProxy, se busca garantizar la disponibilidad continua de los servicios y aplicaciones web. Si un nodo HAProxy falla, otros nodos en el cluster pueden asumir la carga de trabajo sin interrupciones, lo que ayuda a minimizar el tiempo de inactividad y asegurar una experiencia fluida para los usuarios.

Escalabilidad: Un cluster de HAProxy permite escalar horizontalmente para manejar aumentos en la carga de tráfico. Puedes agregar o quitar nodos de manera dinámica según las necesidades de rendimiento y capacidad. Esto asegura que el sistema pueda crecer y adaptarse a medida que aumenta la demanda sin comprometer el rendimiento.

Rendimiento optimizado: Al distribuir la carga de manera equitativa entre varios nodos de HAProxy, se puede lograr un mejor rendimiento general del sistema. Cada nodo puede manejar una parte de la carga, lo que reduce la carga individual y mejora el tiempo de respuesta para los usuarios finales.

Monitoreo y diagnóstico: Datadog ofrece capacidades de monitoreo y diagnóstico en tiempo real para el cluster de HAProxy. Puedes supervisar métricas clave, como CPU, memoria RAM y el estado de los nodos, y recibir alertas en caso de problemas. Esto te permite detectar y solucionar problemas de manera proactiva, evitando tiempos de inactividad y mejorando la experiencia del usuario.
Optimización del rendimiento: Al tener acceso a datos y métricas detalladas proporcionadas por Datadog, puedes identificar cuellos de botella, patrones de uso y tendencias de tráfico. Esto te ayuda a optimizar la configuración del cluster de HAProxy, ajustar los recursos según sea necesario y mejorar el rendimiento general de las aplicaciones y servicios web.

ALTERNATIVAS DE SOLUCIÓN BALANCEO DE CARGAS.

Hay varias opciones disponibles que ofrecen funcionalidades similares de balanceo de carga y alta disponibilidad. Algunas alternativas populares incluyen:

NGINX: NGINX es un servidor web y proxy inverso que también puede funcionar como balanceador de carga. Ofrece una amplia gama de características y es conocido por su rendimiento y escalabilidad. NGINX Plus, la versión comercial, proporciona funcionalidades avanzadas, como el balanceo de carga basado en IP y sesiones, y la integración con herramientas de monitoreo.

Apache HTTP Server: Aunque principalmente conocido como un servidor web, Apache HTTP Server también puede utilizarse como un balanceador de carga con módulos adicionales,

como mod_proxy_balancer. Ofrece opciones de configuración flexibles y es compatible con una amplia gama de tecnologías y sistemas operativos.

HAProxy Community Edition: HAProxy también tiene una edición comunitaria que es una alternativa gratuita al cluster HAProxy. Aunque carece de algunas características avanzadas disponibles en la versión Enterprise, la edición comunitaria sigue siendo una opción sólida para implementaciones de balanceo de carga de menor escala.

F5 BIG-IP: F5 BIG-IP es una solución comercial que ofrece un conjunto completo de características para el balanceo de carga y la administración del tráfico. Es conocido por su rendimiento, escalabilidad y capacidades de seguridad avanzadas. F5 BIG-IP también proporciona integración con herramientas de monitoreo y administración centralizada.

A continuación, se presenta un cuadro comparativo de las alternativas mencionadas (HAProxy, NGINX, Apache HTTP Server y F5 BIG-IP) en función de varias características:

Característica	HAProxy	NGINX	Apache HTTP Server	F5 BIG-IP
Rendimiento	Alto	Alto	Moderado	Alto
Escalabilidad	Bueno	Bueno	Limitado	Bueno
Flexibilidad	Alta	Alta	Moderada	Alta
Configuración avanzada	Sí	Sí (NGINX Plus)	Sí	Sí
Compatibilidad	Amplia	Amplia	Amplia	Amplia
Soporte empresarial	Edición comercial	Edición comercial	Comunidad	Sí
Curva de aprendizaje	Moderada	Moderada	Baja	Alta
Disponibilidad de funciones avanzadas	Edición comercial	Edición comercial	Mediante módulos	Completo
Integración con herramientas de monitoreo	Sí	Sí	Sí	Sí
Costo	Gratuito (Comunidad)	Gratuito/Edición comercial	Gratuito	Edición comercial

Tabla 1. Cuadro comparativo de alternativas.

ALTERNATIVAS DE ALGORITMO DE BALANCEO

Existen varios algoritmos de balanceo de carga que se utilizan en los balanceadores de carga para distribuir las solicitudes entre los servidores de destino. Algunas alternativas comunes incluyen:

Round Robin: Este algoritmo distribuye las solicitudes secuencialmente en un ciclo entre los servidores disponibles. Cada solicitud se envía al siguiente servidor en orden, comenzando desde el principio después de alcanzar el último servidor.

Least Connections (Menos conexiones): En este enfoque, se redirige cada solicitud al servidor con la menor cantidad de conexiones activas en ese momento. Esto ayuda a distribuir la carga de manera más equitativa en función de la carga de trabajo real de cada servidor.

IP Hash: Se utiliza la dirección IP de origen del cliente para calcular un hash y asignar de manera consistente al mismo servidor. Esto garantiza que las solicitudes del mismo cliente se dirijan siempre al mismo servidor, lo que puede ser beneficioso para aplicaciones que requieren persistencia de sesión.

Least Time (Menor tiempo): Este algoritmo envía las solicitudes al servidor que tiene la menor latencia o el tiempo de respuesta más rápido. Es útil cuando el tiempo de procesamiento de cada solicitud puede variar significativamente entre los servidores.

Source IP Affinity (Afinidad de IP de origen): También conocido como "Sticky sessions", este algoritmo mantiene la conexión del cliente con el mismo servidor durante un período de tiempo determinado. Se utiliza una combinación de la dirección IP de origen y el puerto para asignar las solicitudes a un servidor específico, lo que garantiza que todas las solicitudes de un cliente se envíen al mismo servidor.
A continuación, te presento un cuadro comparativo resumido de las alternativas de algoritmos de balanceo de carga:

Algoritmo	Descripción	Ventajas	Desventajas
Round Robin	Distribuye las solicitudes secuencialmente en un ciclo	Simplicidad, equidad en la distribución	No tiene en cuenta la carga de los servidores
Least Connections	Envía las solicitudes al servidor con la menor cantidad de conexiones activas	Equilibrio de carga según la carga real	No tiene en cuenta la capacidad de los servidores
IP Hash	Utiliza la dirección IP de origen del cliente para asignar a un servidor consistente	Persistencia de sesión, útil para ciertas aplicaciones	Desbalanceo si se agregan o eliminan servidores
Least Time	Envía las solicitudes al servidor con el menor tiempo de respuesta	Optimización de rendimiento	Puede generar congestión en el servidor más rápido
Source IP Affinity (Sticky Sessions)	Mantiene al cliente conectado al mismo servidor durante un período de tiempo	Persistencia de sesión, útil para ciertas aplicaciones	No permite una distribución equitativa de la carga

Tabla 2. Alternativas de solución algoritmo de balanceo.

ALTERNATIVAS DE MONITOREO.

Existen varias alternativas de monitoreo que puedes considerar para supervisar tus sistemas y aplicaciones. Algunas de las opciones más populares son:

Datadog: Datadog es una plataforma de monitoreo y análisis en la nube que proporciona una amplia gama de capacidades de monitoreo. Permite recopilar métricas, eventos y registros de tus sistemas, aplicaciones y servicios, y proporciona visualizaciones y alertas personalizables. También ofrece integraciones con una amplia gama de herramientas y servicios populares.

Prometheus: Prometheus es una herramienta de monitoreo de código abierto diseñada especialmente para sistemas distribuidos y contenedores. Permite la recopilación de métricas y ofrece poderosas consultas y alertas basadas en una base de datos de series temporales. Prometheus es altamente escalable y se integra bien con otras herramientas de monitoreo y orquestación, como Grafana y Kubernetes.

Grafana: Grafana es una plataforma de visualización de datos y monitoreo de código abierto. Puedes usar Grafana en conjunto con otras herramientas de monitoreo, como Prometheus o InfluxDB, para crear paneles interactivos y visualmente atractivos que muestran métricas y datos en tiempo real. Grafana también es altamente personalizable y admite una amplia gama de fuentes de datos.

Nagios: Nagios es una herramienta de monitoreo de infraestructura y servicios de código abierto. Permite supervisar la disponibilidad y el rendimiento de tus sistemas, servicios de red y aplicaciones. Nagios es altamente personalizable y admite la creación de notificaciones y alertas personalizadas.

Zabbix: Zabbix es una solución de monitoreo y administración de redes de código abierto que proporciona capacidades de monitoreo de red, servidor y aplicación. Puede realizar un seguimiento de métricas en tiempo real, recopilar datos de múltiples fuentes y enviar alertas en caso de problemas. Zabbix también ofrece visualizaciones personalizadas y capacidad de escalabilidad.

A continuación, se presenta un cuadro comparativo resumido de las alternativas populares de herramientas de monitoreo:

Herramienta	Descripción	Ventajas	Desventajas
Datadog	Plataforma de monitoreo y análisis en la nube	Amplias capacidades, integraciones y alertas	Costo para características avanzadas
Prometheus	Sistema de monitoreo de código abierto basado en series temporales	Escalabilidad, consultas y alertas poderosas	Configuración y administración más complejas
Grafana	Plataforma de visualización y monitoreo de código abierto	Paneles interactivos y personalizables	Requiere integración con otras herramientas de monitoreo
Nagios	Herramienta de monitoreo de infraestructura y servicios	Amplia compatibilidad y personalización	Curva de aprendizaje empinada
Zabbix	Solución de monitoreo y administración de redes de código abierto	Monitoreo de red, servidor y aplicación	Configuración y administración más complejas

Tabla 3. Alternativas de solución monitoreo.

ANÁLISIS Y DESARROLLO

Hemos decidido implementar un cluster HAProxy con tres máquinas Ubuntu, Apache y Datadog debido a varias razones estratégicas y técnicas:

Escalabilidad y rendimiento: Al implementar un cluster HAProxy, podemos distribuir la carga de manera eficiente entre múltiples servidores backend, lo que nos permite escalar horizontalmente y manejar mayores volúmenes de tráfico. Esto asegura un rendimiento óptimo y evita cuellos de botella en nuestro sistema.

Flexibilidad y personalización: HAProxy es una solución altamente configurable y flexible. Podemos ajustar los algoritmos de balanceo de carga y las reglas de enrutamiento según nuestras necesidades específicas. Esto nos permite adaptar la configuración a las características y requisitos de nuestra aplicación.

Compatibilidad con Apache: La integración de HAProxy con servidores web como Apache nos permite aprovechar las fortalezas de ambos. Apache es conocido por su capacidad de manejar diversas tecnologías y contenido web, mientras que HAProxy se especializa en el balanceo de carga y la distribución eficiente del tráfico. Al combinarlos, obtenemos una solución robusta y escalable.

Monitoreo avanzado con Datadog: Al utilizar Datadog junto con nuestro cluster HAProxy, podemos obtener una visibilidad completa del rendimiento de nuestro sistema. Datadog nos proporciona monitoreo en tiempo real, alertas y análisis detallados de métricas clave. Esto nos ayuda a identificar posibles problemas, optimizar

nuestro rendimiento y tomar decisiones informadas para mejorar nuestra infraestructura.

A continuación se da un paso a paso de las configuración del cluster haproxy y datadog

1. **Antes de comenzar con la configuración del balanceador de cargas, optamos por cumplir los siguientes requisitos:**

   - Tener tres máquinas virtuales de Ubuntu instaladas y configuradas en la misma red.
   - Tener privilegios de superusuario en las tres máquinas.
   - Tener instalado HAProxy en la máquina que actuará como balanceador de cargas.

2. **Creación del archivo Vagrantfile:**

   Se creó un directorio donde alojaremos nuestro vagrantfile y configurarlo en cualquier editor de texto para quedar de la siguiente manera (Anexo 1):

```
encoding: UTF-8 -- mode: ruby -- vi: set ft=ruby :

Vagrant.configure("2") do |config|

config.vm.define :haproxy do |haproxy|
haproxy.vm.box = "bento/ubuntu-20.04"
haproxy.vm.network :private_network, ip: "192.168.100.5"
haproxy.vm.hostname = "haproxy"
end

config.vm.define :backend1 do |backend1|
backend1.vm.box = "bento/ubuntu-20.04"
backend1.vm.network :private_network, ip: "192.168.100.6"
backend1.vm.hostname = "backend1"
end

config.vm.define :backend2 do |backend2|
backend2.vm.box = "bento/ubuntu-20.04"
backend2.vm.network :private_network, ip: "192.168.100.7"
backend2.vm.hostname = "backend2"
end
end
```

   - En resumen, este código de Vagrant configura tres máquinas virtuales utilizando la imagen de Ubuntu 20.04 y asignándole direcciones IP y nombres de host específicos.

3. **Configuración de las máquinas virtuales:**
   - **Configuración de la primera máquina virtual - Haproxy**

     Se instaló el servicio de haproxy ejecutando el siguiente comando: “apt install haproxy”, y se inicia el servicio con el comando: “sudo systemctl start haproxy”.
     También se instaló el agente de datadog con el comando ofrecido para la instalación del agente en el cual se incluye la API KEY de la cuenta creada el cual nos enviará datos de las métricas monitoreadas en la interfaz gráfica del dashboard realizado en datadog.

   - **Configuración de la segunda máquina virtual - Backend1**

     Se debe instaló el servicio de apache2 con el siguiente comando: “apt install apache2”, una vez instalado se creó un archivo index.html el cual se crea en la ruta “var/www/html” y por último se ejecutó el siguiente comando para iniciar el servicio de Apache: “sudo systemctl start apache2”.

   - **Configuración de la tercera máquina virtual - Backend2**

     Se instaló el servicio de apache2 con el siguiente comando: “apt install apache2” una vez instalado se creó un archivo index.html el cual se crea en la ruta var/www/html y por último se ejecutó el siguiente comando para iniciar el servicio de Apache: “sudo service apache2 start”

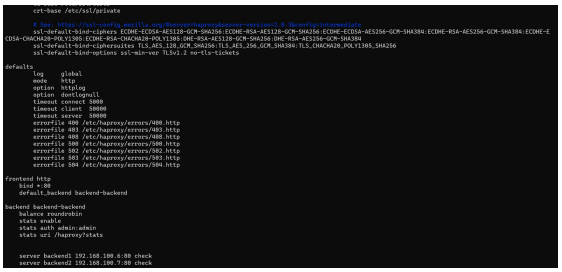
4. **Configuración del balanceador de cargas:**

   - Se instaló HAProxy en la máquina que actuará como balanceador de cargas en

este caso es la máquina con el nombre de haproxy.

- Se creó un archivo de configuración para HAProxy en "/etc/haproxy/haproxy.cfg".

- Se configuró el archivo de configuración de HAProxy de la siguiente manera:



```
        crt-base /etc/ssl/private

        # See: https://ssl-config.mozilla.org/#server=haproxy&server-version=2.0.3&config=intermediate
        ssl-default-bind-ciphers ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-SHA256:ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-E
CDSA-CHACHA20-POLY1305:ECDHE-RSA-CHACHA20-POLY1305:DHE-RSA-AES128-GCM-SHA256:DHE-RSA-AES256-GCM-SHA384
        ssl-default-bind-ciphersuites TLS_AES_128_GCM_SHA256:TLS_AES_256_GCM_SHA384:TLS_CHACHA20_POLY1305_SHA256
        ssl-default-bind-options ssl-min-ver TLSv1.2 no-tls-tickets

defaults
        log     global
        mode    http
        option  httplog
        option  dontlognull
        timeout connect 5000
        timeout client  50000
        timeout server  50000
        errorfile 400 /etc/haproxy/errors/400.http
        errorfile 403 /etc/haproxy/errors/403.http
        errorfile 408 /etc/haproxy/errors/408.http
        errorfile 500 /etc/haproxy/errors/500.http
        errorfile 502 /etc/haproxy/errors/502.http
        errorfile 503 /etc/haproxy/errors/503.http
        errorfile 504 /etc/haproxy/errors/504.http

frontend http
    bind *:80
    default_backend backend-backend

backend backend-backend
    balance roundrobin
    stats enable
    stats auth admin:admin
    stats uri /haproxy?stats


    server backend1 192.168.100.6:80 check
    server backend2 192.168.100.7:80 check
```

Figura 1. Archivo HAProxy

- Por último se reinició HAProxy para aplicar los cambios

5. Uso:

Para poner a prueba el balanceador de cargas con HAProxy y tres máquinas de Ubuntu, se hizo lo siguiente:

- Se accedió al balanceador de cargas a través de su dirección IP (máquina haproxy).

- La solicitud será dirigida a uno de los dos servidores web de manera aleatoria.

6. Configuración de Datadog

- Inicialmente se instaló el agente de datadog mediante la guía paso a paso que brinda la página.

- Una vez configurado el agente, desde el apartado de infraestructura se observa en el mapa de anfitrión el host.

- Luego desde la ventana de dashboard se crearon una serie de widgets.
- Una vez escogido el gráfico, se procedió a configurar las métricas que se requieren visualizar.

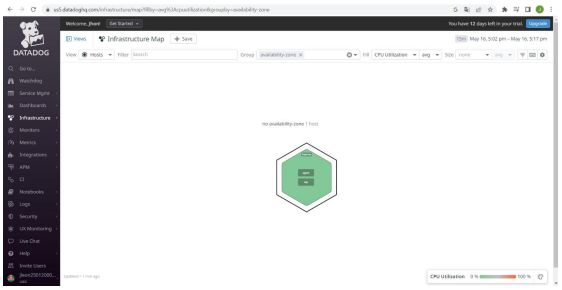


Figura 2. Máquina HAProxy activa en Datadog

DISCUSIÓN

Partiendo de las configuraciones realizadas en las máquinas creadas y logrando realizar la conexión con DataDog, se presentan los resultados obtenidos durante la práctica de para el monitoreo de un cluster computacional.

El objetivo de esta práctica fue implementar una solución de monitoreo robusta y eficiente que permitiera obtener métricas y registros relevantes del cluster, brindando así una visibilidad detallada del rendimiento y el estado de los recursos.

Con la integración exitosa del monitoreo de las máquinas con Datadog se permitió obtener métricas detalladas del cluster computacional. A través de los paneles creados y gráficos interactivos, se pudo monitorear el uso de CPU, memoria, almacenamiento y otros recursos clave. Estas métricas proporcionaron una visión clara del rendimiento general del cluster y permitieron identificar posibles cuellos de botella o puntos de mejora.

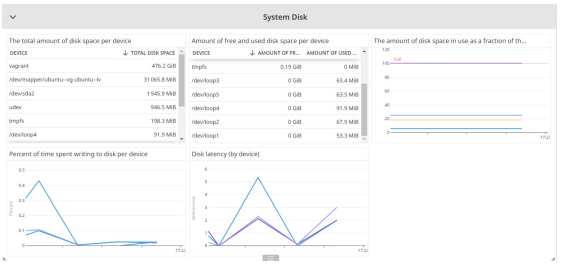


Fig 3: Monitoreo de los recursos del sistema con DataDog.

Fig 4: Monitoreo de la red del sistema con DataDog.

Durante la práctica, se comprobó que Datadog es una solución escalable y adaptable a diferentes entornos. A medida que se agregaban más nodos al cluster o se modificaban la configuración, Datadog pudo adaptarse sin problemas, recopilando y visualizando las métricas y registros correspondientes.

CONCLUSIONES

La implementación de un sistema de monitoreo con DataDog en un cluster computacional es crucial para garantizar la disponibilidad y el rendimiento de los servicios que se ejecutan en él. Con este proyecto, se logró monitorear y detectar posibles problemas en las máquinas virtuales del cluster de manera oportuna.

La plataforma de DataDog es una herramienta potente y completa para el monitoreo y gestión de sistemas, ya que permite recopilar y visualizar datos en tiempo real, y configurar alertas para notificar a los administradores en caso de fallas o errores críticos. Además, cuenta con una interfaz fácil de usar y una amplia gama de integraciones con otros sistemas.

El uso de Ubuntu como sistema operativo para las máquinas virtuales en el cluster es una buena elección, ya que es una distribución de Linux popular, estable y con una gran comunidad de soporte. Además, su sistema de gestión de paquetes puede ser complejo y requerir conocimientos técnicos previos.

En resumen, el proyecto de monitoreo de cluster con DataDog es una solución efectiva y completa para garantizar la disponibilidad y el rendimiento de los servicios en un cluster computacional. La integración de DataDog con Ubuntu, haproxy y los backends permite una monitorización efectiva de los sistemas, y ayuda a garantizar que los problemas sean detectados y corregidos rápidamente.

REFERENCIAS

[1] ″. (n.d.). ″ - Wiktionary. Retrieved May 13, 2023 frrom https://www.ibm.com/docs/es/was-zos/9.0.5?topic=servers-introduction-clusters

[2] ¿Qué es el balanceo de carga definido por software? | Glosario de VMware | ES. (n.d.). VMware. Retrieved May 13, 2023, from https://www.vmware.com/es/topics/glossary/content/software-load-balancing.html

[3] Mckaig, A., & Khan, T. (2022). How the Metrics Backend Works at Datadog.

[4] Ashrafi, T. H., Hossain, M. A., Arefin, S. E., Das, K. D., & Chakrabarty, A. (2018). Iot infrastructure: fog computing surpasses cloud computing. In Intelligent Communication and Computational Technologies: Proceedings of Internet of Things for Technological Development, IoT4TD 2017 (pp. 43-55). Springer Singapore.

[5] "HAProxy High Availability" por Neil Gehani, Packt Publishing (2015).

[6] "Datadog for Dummies" por Datadog (2019).