DISEÑO E IMPLEMENTACIÓN DE UNA APLICACIÓN WEB PARA LA ADMINISTRACIÓN DE SERVICIOS DOCKERIZADOS SOBRE EL PROXY INVERSO DEL SERVIDOR DE INVESTIGACIONES DE LA FACULTAD DE INGENIERÍAS DE LA INSTITUCIÓN UNIVERSITARIA ANTONIO JOSÉ CAMACHO. "DOCKERWIZARD"

DESIGN AND IMPLEMENTATION OF A WEB APPLICATION FOR THE ADMINISTRATION OF DOCKERIZED SERVICES ON THE REVERSE PROXY OF THE RESEARCH SERVER OF THE INSTITUCIÓN UNIVERSITARIA ANTONIO JOSÉ CAMACHO FACULTY OF ENGINEERING. "DOCKERWIZARD"

AUTORES

Alexis Alberto Ramírez Orozco
Profesor tiempo completo de la Facultad de Ingenierías de la Institución Universitaria Antonio José Camacho.
Correo: aramirez@admon.uniajc.edu.co

Santiago Ruíz Posso
Estudiante de octavo semestre de Ingeniería de Sistemas, integrante del semillero SAIDO de la Facultad de Ingenierías de la Institución Universitaria Antonio José Camacho.
Correo: santiagoruiz@estudiante.uniajc.edu.co

Alexis Alberto Ramírez Orozco y Santiago Ruíz Posso

Semillero SAIDO
Grupo de investigación en INTELIGO
Institución Universitaria Antonio José Camacho
Recibido: 08/09/2023 - Aceptado: 05/10/2023

Para citar este artículo: Ramírez Orozco, A. A. y Ruíz Posso, S. (2024). Diseño e implementación de una aplicación web para la administración de servicios dockerizados sobre el proxy inverso del servidor de investigaciones de la Facultad de Ingenierías UNIAJC, "Dockerwizard". Revista Sapientia, 16(31), 62-69. 10.54278/sapientia.v16i31.172

RESUMEN

El proyecto se centra en abordar los desafíos en la administración de contenedores Docker en el servidor de investigaciones de la Facultad de Ingenierías de la UNIAJC, donde se requiere que los proyectos estén dockerizados. La configuración actual, que implica la creación manual de contenedores, presenta problemas como configuraciones propensas a errores, interrupciones prolongadas, dificultades en la supervisión de servicios y gestión de usuarios. Esto afecta la eficiencia y la disponibilidad de servicios esenciales para la comunidad académica.

El objetico principal del proyecto es diseñar e implementar una solución que automatice y optimice la gestión de contenedores Docker en el servidor, mejorando la eficiencia y reduciendo errores. Se propone una interfaz de usuario que simplifique la administración de usuarios y permisos, un sistema de automatización para la creación y configuración de contenedores, un sistema de monitoreo en tiempo real y una evaluación del impacto de la solución.

Se esperan varios impactos positivos, como una mejora en la eficiencia operática, la reducción de tiempo de inactividad y una facilitación en la gestión, debido a que existen limitaciones, como escasez de recursos, conectividad a internet, compatibilidad de plataforma, seguridad y desafíos de escalabilidad.

Palabras clave: Docker, Investigación, Proyectos, Proxy inverso.

ABSTRACT

The project focuses on addressing the challenges in the administration of Docker containers on the UNIAJC Faculty of Engineering's research server, where projects are required to be containerized. The current configuration, involving manual container creation, presents issues such as error-prone configurations, prolonged disruptions, difficulties in service monitoring, and user management. This affects the efficiency and availability of essential services for the academic community.

The main objective of the project is to design and implement a solution that automates and optimizes Docker container management on the server, improving efficiency and reducing errors. It proposes a user interface to simplify user and permission management, an automation system for container creation and configuration, real-time monitoring, and an evaluation of the solution's impact.

Several positive impacts are expected, such as improved operational efficiency, reduced downtime, and streamlined management. However, there are limitations, including limited resources, internet connectivity, platform compatibility, security, and scalability challenges. This project aims to transform the administration of Docker containers on the UNIAJC server, enhancing efficiency and the availability of essential services, which will have a positive impact on the academic community and the quality of research and degree projects.

Key words: Docker, Research, Projects, Reverse Proxy.

INTRODUCCIÓN

El servidor de investigaciones de la Facultad de Ingenierias de la Institución Universitaria Antonio José Camacho, UNIAJC (Cygnus.uniajc.edu.co) tiene un papel fundamental al proporcionar soporte a proyectos de investigaciones y proyectos de grados de dicha facultad, los cuales requieran servicios web y base de datos (Stevenson y Araújo, 2011). Sin embargo, para acceder a estos servicios se establece un requisito crucial: los proyectos deben estar dockerizados. A pesar de esta necesidad, la gestión actual de los contenedores Docker plantea una serie de desafíos muy significativos.

La configuración actual implica la creación manual de contenedores desde la raíz del servidor, es decir, desde la consola, lo que conlleva a problemas como configuraciones propensas a errores, interrupciones prolongadas, dificultades en la supervisión de servicios, errores humanos y una gestión complicada de usuarios. Estos obstáculos no sólo afectan la eficiencia operativa, sino que también ponen en riesgo la disponibilidad de los servicios esenciales para la comunidad académica. Para abordar estos desafíos, este proyecto se enfoca en diseñar e implementar una solución integral que busca automatizar y optimizar la gestión de contenedores Docker en el servidor, con el objetivo de mejorar la eficiencia y reducir errores de configuración. Entre las principales metas se incluyen: el desarrollo de una interfaz de usuarios intuitiva, la automatización de la creación y configuración de contenedores, la implementación de un sistema de monitoreo en tiempo real y una evaluación exhaustiva del impacto de estas mejoras.

Como dijo Gómez Labrador (2014): “El programador necesita automatizar la mayoría de estos procedimientos repetitivos para evitar errores tipográficos o conceptuales y para mejorar el tratamiento general de las aplicaciones” (p. 7). Los impactos esperados de este proyecto son sustanciales e incluyen una mayor eficiencia operativa, la reducción de tiempos de inactividad, una gestión mas sencilla y segura y una mejora en la calidad de los proyectos de investigación y de grado. Sin embargo, no se ignoran las limitaciones potenciales, como escasez de recursos, desafíos de seguridad y la necesidad de una promoción efectiva para lograr la adopción completa.
Este proyecto busca transformar la administración de contenedores Docker en el servidor de la Facultad de Ingenierías de la UNIAJC, impulsando mejoras significativas en la eficiencia y la disponibilidad de servicios esenciales, lo que, a su vez, beneficiará a la comunidad académica y fortalecerá la calidad de proyectos académicos.

MARCO TEÓRICO

Virtualización y Contenedores

La virtualización es una tecnología que permite la creación de entornos o Máquinas Virtuales (VM) dentro de un sistema físico. Estas Máquinas Virtuales son independientes unas de otras y pueden ejecutar sistemas operativos y aplicaciones de manera aislada. La virtualización se utiliza para optimizar la utilización de recursos físicos, como servidores, al permitir la consolidación de múltiples sistemas en una sola máquina física. Cada Máquina Virtual es un entorno completo que incluye un sistema operativo, aplicaciones y recursos asignados (Brosch, 2015). Ahora bien, aunque son parecidos en su base de virtualización, los contenedores son una forma de virtualización ligera que se centra en la encapsulación de aplicaciones y sus dependencias, en lugar de crear Máquinas Virtuales completas. Los contenedores comparten el mismo kernel del sistema operativo del host, lo que los hace más eficientes en términos de recursos y más rápidos de iniciar que las Máquinas Virtuales. Cada contenedor es una unidad autocontenida que incluye la aplicación y todas las bibliotecas y archivos necesarios para su funcionamiento. Los contenedores son portátiles y pueden ejecutarse en cualquier sistema compatible con contenedores, lo que facilita la implementación y la escalabilidad de aplicaciones en entornos variados (Boettiger, 2015). De este modo, es posible afirmar que mientras

la virtualización se centra en crear Máquinas Virtuales completas que ejecutan sistemas operativos completos, los contenedores se enfocan en encapsular aplicaciones y sus dependencias para una implementación eficiente y escalable.

Docker y contenedores

Docker es una plataforma de código abierto que facilita la creación, implementación y administración de contenedores. Permite a los desarrolladores empaquetar una aplicación y todas sus dependencias en un contenedor, que se puede ejecutar de manera consistente en diferentes entornos, como el desarrollo local, pruebas y producción. Docker utiliza tecnologías de virtualización a nivel de sistema operativo para proporcionar un aislamiento eficiente entre contenedores, lo que permite que múltiples contenedores compartan un mismo sistema operativo base sin interferir entre sí.

Ahora bien, una plataforma como Docker, pensada en empaquetar, consecuentemente crea empaquetamientos tambien conocidos como contenedores. Un contenedor es una instancia autónoma y aislada de una aplicación que se ejecuta en un entorno virtualizado. Contiene todo lo que una aplicación necesita para funcionar, incluyendo el código de la aplicación, bibliotecas, variables de entorno y configuraciones. Los contenedores son ligeros, ya que comparten el mismo kernel del sistema operativo del host, lo que los hace rápidos de iniciar y eficientes en el uso de recursos. También son portátiles, lo que significa que pueden ejecutarse en cualquier sistema que admita contenedores, independientemente de la infraestructura subyacente (Merkel, 2014).

Proxy inverso

El término Proxy inverso se refiere a una configuración de red en la que un servidor proxy, que normalmente permite enviar a múltiples destinos de salida, las señales de la red interna, se utiliza para enrutar el tráfico entrante en una entrada a múltiples destinos, por los cuales recibe el nombre de proxy inverso. Esta acción es muy útil si se desea la escalabilidad y la gestión centralizada de diversas aplicaciones o servicios en un entorno de servidores como web o bases de datos y simplifica la administración del tráfico de red, la seguridad y la capacidad de respuesta.

Administración de servidores y Docker

Al hablar de administrar servidores siempre se debe tener en cuenta un factor importante como son las tareas de root o las tareas de administración. El root de un servidor debe estar a cargo de una persona capacitada y que conozca el manejo óptimo del o de los equipos, de los servicios y sus usuarios; siempre estando al día con todos los parches, artículos y actualizaciones de dichos servicios.

El programador del root de un servidor también debe hacer

> Una planeación y prevención. Debe realizar unas extensas tareas tales como llevar una documentación actualizada y detallada de los usuarios existentes en el servidor, grupos, permisos de cada usuario, copias de seguridad del sistema (usuarios, configuración), registros completos de actualizaciones y cambios dentro del servidor, recogida periódica sobre rendimiento del servidor... Necesita automatizar la mayoría de estos procedimientos repetitivos para evitar errores tipográficos o conceptuales y para mejorar el tratamiento general de las aplicaciones. (Gómez Labrador, 2014).

En un servidor se tienen que automatizar los procesos importantes como: la comprobación del espacio libre en los discos, gestión de cuentas de usuarios y espacios utilizados, procedimientos para crear, comprobar y restaurar copias de seguridad, procedimientos para comprobar y registrar el rendimiento general del sistema y creación de alertas de seguridad; igualmente debe mantener informados a los usuarios y darles una guía del buen uso para evitar errores provocados por desconocimiento.

Los usuarios deben estar siempre notificados a cada momento de los cambios que se vayan a realizar en el sistema, detallando los resultados esperados, cómo puede afectar los servicios, el tiempo estimado y tiempo real de la duración de la operación. No menos importante es el control de la seguridad, en el cual el root o administrador tiene que definir políticas para el servidor, red corporativa y datos de los usuarios y se debe realizar una previsión de fallos para evitar situaciones como saturación o fallo de los recursos del sistema (memoria, procesadores,

discos), fallos de S.O, como en aplicaciones instaladas o programas propios y errores humanos del propio root o usuarios del servidor.

Según Vélez (2007):

> Una de las amenazas más grandes a la seguridad de un servidor son los administradores distraídos. Entre las principales causas se encuentran la asignación de personal inadecuado con poco conocimiento, sin un entrenamiento adecuado ni el tiempo para la ejecución de su respectivo trabajo. (p.).

El root del sistema deberá realizar un constante cambio de sus contraseñas, así como recomendar a los usuarios que las estén cambiando o que utilicen contraseñas menos genéricas, ya que un atacante sin tanta experiencia podría obtener acceso a las credenciales.

Si bien los servicios dockerizados han ido apareciendo recientemente en la prestación de servicios, las plataformas docker han demostrado ser una tecnología eficiente para el despliegue de aplicativos, ofreciendo atención a problemas de seguridad, portabilidad y estabilidad; no obstante, es una gran herramienta para instalar infraestructuras de microservicios (Noreña, 2022).

Uno de los principales usos de Docker son los procesos de computación en la nube y el desarrollo de aplicaciones, las cuales necesitan ser desplegadas en entornos virtualizados, principalmente en aplicaciones de alto rendimiento. Docker sobresale gracias a las ventajas cruciales frente a las Máquinas Virtuales (VM), utiliza contenedores para virtualizar el sistema operativo, lo que no deja ejecutar múltiples instancias compartiendo un único kernel y da como resultado la inicialización en tiempos de segundos, menor consumo de recursos de hardware y un alto nivel de aislamiento de procesos y sistemas de archivos.

METODOLOGÍA

La metodología que se aplicó para el desarrollo de este proyecto es conocida como ICONIX, esta es una metodología de desarrollo de software que se enfoca en el modelado y la iteración y se basa en el proceso de desarrollo ágil. Su objetivo principal es proporcionar un enfoque sistemático y estructurado para el desarrollo de software, a través de una serie de etapas, que se realizan de forma iterativa y colaborativa.

Para el desarrollo de este proyecto se definieron dos etapas principales, una primera etapa en la que se desarrollan las aplicaciones necesarias para la gestión de los diferentes elementos requeridos por el servidor de forma local y una segunda etapa en la que se pasa todo el desarrollo de la primera etapa a aplicaciones web. A continuación, se presentan dichas etapas.

Etapa 1. Desarrollo de aplicaciones para el Sistema Operativo del Servidor

- Desarrollo de la aplicación para la gestión de usuarios y contenedores de Dockers en el servidor. (yml de cada contenedor, usuarios, copia de seguridad, etc.).
- Desarrollo de aplicación para la configuración del archivo de definición del proxy inverso del servidor. (yml proxy inverso y gestión de éste).
- Desarrollo de aplicación para la configuración del contenedor del proxy inverso del servidor (default.conf, alias del contenedor).

Etapa 2. Desarrollo de aplicación web en el servidor

- Levantamiento de requerimientos.
- Diseño de diagramas y funcionalidades.
- Diseño de base de datos.
- Vistas y Mockups.

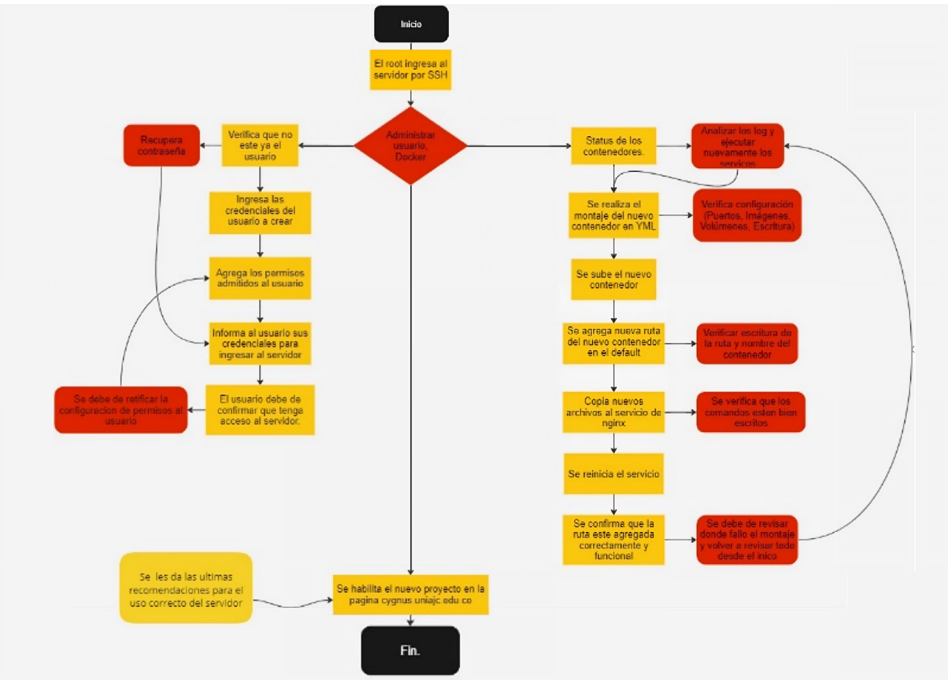
- Desarrollo de aplicativo web.
- Documentación.
- Evaluación del sistema.

RESULTADOS PRELIMINARES

El desarrollo del proyecto ha implicado desafíos, tanto en la determinación de la forma en la que se prestará el servicio de apoyo a proyectos de investigación en el servidor Cygnus (http://cygnus.uniajc.edu.co), como en la configuración básica requerida para implementar la misma. El desarrollo requerido para el alcance de los objetivos de este proyecto se encuentra en la Etapa 1, la cual requiere en todas sus necesidades de desarrollo, la creación de un proceso de ejecución y configuración realizado manualmente como base y estructura para el desarrollo de las aplicaciones que permitirán la funcionalidad de cada una de ellas desde el mismo sistema operativo. Actualmente se han obtenido los siguientes resultados:

Se ha desarrollado un algoritmo para gestión de usuarios y contenedores de Dockers en el servidor Cygnus, éste se ha desarrollado usando la herramienta "docker-compose", para la cual se creado un algoritmo para la configuración del correspondiente archivo "yml" partiendo de la configuración de cada contenedor, usuarios, copia de seguridad, etc. Este algoritmo ha sido resultado de la determinación de la ejecución, configuración y el proceso que requiere la creación y gestión de los contenedores en el servidor, el cual ha sido probado y se encuentra en la evaluación correspondiente para determinar los parámetros que determinarán las características necesarias en el apoyo que el servidor pueda dar a los proyectos de investigación.

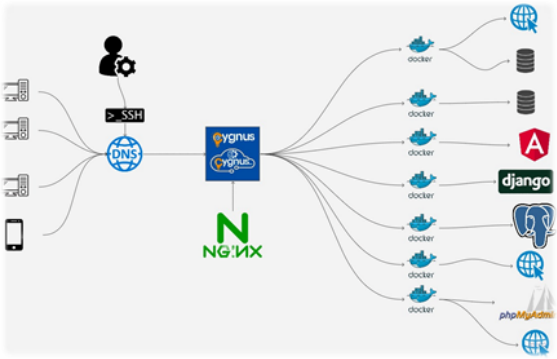
Figura 1. Algoritmo de gestión Cygnus



Fuente: Elaboración propia

Igualmente se ha desarrollado el algoritmo para la configuración de la definición del Docker Proxy Inverso en el servidor que permite la creación del contenedor para el mismo. Para esto, se determinaron, tanto la creación de una red de dockers al interior del servidor, como la configuración de acceso a la misma. Las pruebas realizadas a la configuración del servidor Docker Proxy Inverso han tenido excelentes resultados, permitiendo en la actualidad prestar el servicio de apoyo a proyectos de investigación, aunque no de forma automática, al menos de forma manual, interconectando cada uno de los dockers de los proyectos con el Docker Proxy Inverso.

Figura 2. Arquitectura Desarrollada en Cygnus



Fuente: Elaboración propia

Finalmente, se ha desarrollado el algoritmo necesario para la configuración particular que requiere el Docker Proxy Inverso en el servidor para que pueda dar salida al exterior de éste y por tanto, la conectividad con internet de los demás Docker de cada uno de los proyectos que apoya el servidor. Este proceso, realizado manualmente, ha sido exitoso en la mayoría de las pruebas, presentando inconvenientes con algunos Docker que utilizan servidores WEB ejecutables como Python o Angular; dichos inconvenientes se han ido superando, dejando como resultado ajustes al algoritmo que serán importantes en la programación de éstos.

CONCLUSIONES PRELIMINARES

Fue posible dockerizar servicios semejantes haciendo uso de los mismos puertos de diferentes proyectos integrándolos y ejecutándolos simultáneamente en un sólo equipo servidor utilizando para ello un Docker Proxy Inverso configurado y ajustado a las necesidades particulares del servicio requerido. De igual forma, fue posible determinar los pasos necesarios para la configuración manual de los contenedores y del Docker Proxy Inverso con los servicios dockerizados de diferentes proyectos utilizando una red de dockers interna en el servidor, dándole de esta manera salida a internet a varios proyectos con uso de los mismos puertos de forma simultánea a través del servidor.

La aplicación web permitirá brindar un mayor nivel de control y visibilidad sobre los servicios dockerizados. Los administradores y desarrolladores de la UNIAJC podrán acceder a la información detallada sobre el estado de los contenedores, los registros de aplicaciones y otros aspectos importantes del servicio prestado.

Al proporcionar una interfaz centralizada para la gestión de contenedores Docker, los investigadores de la universidad que requieren el servicio, podrán realizar tareas como la implementación, escalabilidad y monitoreo de aplicaciones de manera más rápida y efectiva.

REFERENCIAS BIBLIOGRÁFICAS

Boettiger, C. (2015). An introduction to Docker for reproducible research. ACM SIGOPS Operating Systems Review, 49(1), 71-79.

Brosch, T. y Marquardt, C. (2015). Virtualization in industrial automation: a survey. IEEE Transactions on Industrial Informatics, 11(4), pp. 834-841.

Correa Cuvi, W. J. (2022). Estudio de la plataforma Kubernetes para la administración automática de contenedores Docker en la facultad FICA de la Universidad Técnica del Norte [tesis de pregrado] Universidad Técnica del Norte. file:///C:/Users/BELL/Downloads/04%20ISC%20613%20TRABAJO%20GRADO%20(1).pdfhttp://repositorio.utn.edu.ec/handle/123456789/12301

Cerami, E., Gopalakrishnan, G., Graaf, T. (2002). Web Services in Bioinformatics. Briefings in Bioinformatics, 3(4), pp. 422-432.

Franco Noreña, Y. (2022). Revisión sistemática de la literatura sobre Tecnologías Docker. [tesis de maestría]. Universidad Nacional de Colombia. Gamboa Fernández, J. (2018). Seguridad en Docker. Universitat Oberta de Catalunya (UOC). https://openaccess.uoc.edu/handle/10609/89325

Gómez Labrador, R. M. (2014). 14127 administración de servidores Linux (Ubuntu/Fedora/CentOS). https://www.academia.edu/36256155/14127_ADMINISTRACI%C3%93N_DE_SERVIDORES_LINUX_UBUNTU_FEDORA_CENTOS

Hernández, D. (2022, 15 de marzo). Protección de datos y privacidad: seguridad en los servidores. GlobalSuite Solutions. (2023, 16 de septiembre). ¿Qué es la norma ISO 27001 y para qué sirve? [Sitio web]. https://www.globalsuitesolutions.com/es/que-es-la-norma-iso-27001-y-para-que-sirve/

Merkel, D. (2014). Docker: Lightweight Linux Containers for Consistent Development and Deployment. Linux Journal, (239). https://dl.acm.org/doi/10.5555/2600239.2600241

RedHat. (2018.). WhatisDocker. https://www.redhat.com/en/topics/containers/what-is-docker

Stevenson, J. M., Araújo, R., Silva, P., et al. (2011). The MySQL Database: A Research Perspective. ACM SIGMOD Record, 40(1), 41-47.

Universidad de Girona. (s.f). Manual introductorio de Iconix. https://ima.udg.edu/~sellares/EINF-ES2/Present1011/MetodoPesadesICONIX.pdf

Vélez Vanegas, T. (2007). Seguridad de la información en la Norma ISO 27001, Implementación práctica sobre el control de acceso a dos servidores Linux de la empresa municipal de aseo de cuenca EMAC [tesis de pregrado] Universidad del Azuay. https://dspace.uazuay.edu.ec/bitstream/datos/2278/1/05794.pdf