Imagen que contiene Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.

**Universidad Latina de Costa Rica**

**Ingeniería en Telemática**

**Entrega Final**

**Proyecto de Investigación Sistemas Operativos II**

**BIT-28 Sistemas Operativos II**

**Jimena Gómez Calvo**

**Docente: Carlos Méndez Rodríguez**

**2025**

**1. Introducción**

La transformación digital y el auge de los servicios basados en la nube han llevado a que organizaciones de todos los tamaños dependan cada vez más de arquitecturas distribuidas. Estas arquitecturas, cuando están bien diseñadas, ofrecen beneficios como alta disponibilidad, escalabilidad dinámica, seguridad robusta y rendimiento optimizado. Según el Instituto Nacional de Estándares y Tecnología (NIST), la computación en la nube es *“un modelo para permitir el acceso conveniente, bajo demanda y por medio de la red de Internet, a un grupo compartido de recursos de computación configurables (por ejemplo, redes, servidores, almacenamiento, aplicaciones y servicios) que se pueden aprovisionar y liberar rápidamente con un mínimo esfuerzo de gestión o interacción con el proveedor de servicios”* (Mell & Grance, 2011).

El propósito de este proyecto es realizar una investigación aplicada que permita diseñar una arquitectura robusta y eficiente para sistemas distribuidos en la nube, enfocada en cuatro pilares: disponibilidad, rendimiento, escalabilidad y seguridad. Además, se analizará el impacto de distintos algoritmos de balanceo de carga en el desempeño general del sistema, utilizando métricas clave para evaluar resultados y proponer mejoras concretas.

**2. Objetivos del Proyecto**

1. **Analizar aspectos de alta disponibilidad, alto rendimiento, escalabilidad y seguridad**  
   Se estudiarán los principios teóricos y prácticos que permiten a los sistemas distribuidos mantener sus servicios operativos incluso ante fallos, crecer de forma eficiente ante demandas variables, operar con bajo tiempo de respuesta y proteger datos frente a amenazas.
2. **Diseñar una arquitectura robusta con los aspectos anteriores**  
   Se desarrollará una propuesta arquitectónica capaz de sostener operaciones críticas, integrando componentes como balanceadores de carga, múltiples instancias de servidores, almacenamiento redundante y herramientas de seguridad.
3. **Evaluar el impacto de diferentes algoritmos de balanceo de carga en el rendimiento del sistema**  
   Se aplicarán y compararán algoritmos como Round Robin, Least Connections y Consistent Hashing para determinar su eficacia en escenarios simulados.
4. **Determinar los criterios de evaluación y métricas clave de rendimiento**  
   Se establecerán indicadores como el tiempo de respuesta, el porcentaje de tráfico procesado por cada servidor y el tiempo total de inactividad del sistema durante las pruebas.
5. **Proponer mejoras basadas en los resultados obtenidos**  
   A partir de los análisis, se plantearán recomendaciones técnicas y estratégicas para mejorar la arquitectura en cuanto a eficiencia operativa, protección y adaptabilidad.

**3. Marco Teórico y Citas Relevantes**

La literatura coincide en que una arquitectura distribuida bien estructurada puede mejorar sustancialmente la disponibilidad del sistema. Tanenbaum y Steen (2007) destacan que *“la distribución de componentes en múltiples nodos no solo mejora el rendimiento, sino que también incrementa la tolerancia a fallos”.*

Por otro lado, la seguridad es un aspecto clave. Según Kumar et al. (2022), *“las arquitecturas en la nube requieren una defensa multicapa que combine cifrado, detección de anomalías, autenticación fuerte y segmentación de red para mitigar riesgos de ataque”.*

En cuanto al rendimiento, según Varia y Mathew (2020), *“el uso adecuado de balanceadores de carga junto con monitoreo continuo permite mantener niveles óptimos de desempeño en sistemas distribuidos con gran volumen de tráfico”.*

**4. Implementación**

**4.1 Arquitectura Propuesta**

La arquitectura diseñada se compone de los siguientes elementos:

* **Balanceador de carga** (ej. HAProxy, NGINX o AWS ELB)
* **Clúster de servidores web y de aplicación** en contenedores Docker, orquestados con Kubernetes.
* **Base de datos distribuida** como PostgreSQL en modo clúster.
* **Firewall y sistema de tección de intrusos** con herramientas como Snort.
* **Sistema de monitoreo y alertas** (Prometheus + Grafana)

**4.2 Entorno de Pruebas y Métricas**

Se usarán entornos virtualizados y herramientas como:

* **Apache JMeter** para simular múltiples usuarios y medir el tiempo de respuesta.
* **Prometheus y Grafana** para monitorear tráfico, disponibilidad y uso de recursos.
* **Wireshark o tcpdump** para inspección de paquetes y análisis de seguridad.
* **Terraform** para automatizar la creación de infraestructura.

**Métricas clave**:

* Tiempo promedio de respuesta (ms)
* Desviación de carga entre nodos
* Uptime (%) del sistema
* Tasa de error bajo estrés (%)

**5. Resultados**

Se espera demostrar que los algoritmos con menor consumo de CPU (como Round Robin) ofrecen rapidez en entornos controlados, pero que otros como Least Connections o Weighted Random son más efectivos en escenarios reales con tráfico irregular. Asimismo, se anticipa que el uso de contenedores y orquestadores permitirá una mayor flexibilidad y recuperación ante fallas.

En materia de seguridad, se prevé que las configuraciones con detección activa de amenazas logren reducir significativamente el riesgo de intrusiones, aunque con un leve impacto en el rendimiento, lo cual también será medido y discutido.

- **Kubernetes**: Plataforma de orquestación de contenedores que automatiza el despliegue, escalado y gestión de aplicaciones distribuidas, ideal para garantizar escalabilidad y alta disponibilidad.

- **Docker**: Herramienta de contenedorización que permite empaquetar aplicaciones con sus dependencias, facilitando portabilidad y consistencia entre entornos.

- **HAProxy**: Balanceador de carga de código abierto que distribuye tráfico entre múltiples servidores, mejorando el rendimiento y reduciendo la sobrecarga de un solo nodo.

- **Prometheus + Grafana**: Conjunto de herramientas para monitoreo y visualización. Prometheus recolecta métricas del sistema, y Grafana permite visualizar esos datos con dashboards dinámicos.

- **Apache JMeter**: Herramienta de prueba de carga que simula múltiples usuarios concurrentes, útil para medir tiempos de respuesta y estabilidad del sistema bajo presión.

- **Snort**: Sistemas de detección de intrusos (IDS) que supervisan el tráfico y alertan ante comportamientos sospechosos, fortaleciendo la seguridad del sistema.

- **Terraform**: Herramientas de infraestructura como código (IaC) que permiten desplegar y configurar entornos de manera automatizada y repetible.

- **PostgreSQL Cluster**: Bases de datos distribuidas que permiten replicación de datos y tolerancia a fallos, fundamentales para mantener la integridad y disponibilidad.

**7. Conclusiones**

El estudio permitirá comprender cómo diferentes decisiones arquitectónicas impactan el comportamiento global de un sistema distribuido en la nube. A partir de los resultados, se podrán aplicar ajustes que mejoren no solo el desempeño técnico, sino también la seguridad y estabilidad operativa del sistema.

**8. Revisión Bibliográfica**

* Kumar, A., Bharti, V., & Mathur, P. (2022). *Cloud Security and Risk Mitigation*. Springer.
* Linthicum, D. S. (2017). *Estrategia de nube empresarial*. O’Reilly Media.
* Mell, P., & Grance, T. (2011). *The NIST Definition of Cloud Computing* (Special Publication 800-145). National Institute of Standards and Technology.
* Murillo Rodríguez, J. M. (2011). *Sistemas distribuidos: Evolución e involución*. Editorial UOC.
* Suresh, S. (2023). *Mitigación del balanceo de carga para la nube usando optimización de colonias de hormigas*. Springer.
* Tanenbaum, A. S., & Steen, M. van. (2007). *Distributed Systems: Principles and Paradigms* (2nd ed.). Pearson.
* Varia, J., & Mathew, S. (2020). *Overview of Deployment Options on AWS*. Amazon Web Services, Inc.
* Universitat Oberta de Catalunya. (2017). *Introducción a la computación distribuida*. Editorial UOC.
* Akamai Technologies. (2024). *¿Qué es el balanceo de carga en la nube?* Recuperado de <https://www.akamai.com>