UNIVERSIDAD BOLIVARIANA DEL ECUADOR

UNIVERSIDAD BOLIVARIANA DEL ECUADOR

APELLIDO / NOMBRE:

HERRERA OÑATE CARLOS ORLANDO
SEBASTIAN GARCIA

MATERIA:

Sistemas Operativos

TEMA:

Informe Técnico y Diseño de Laboratorio

ECUADOR 09-02-2026

1. Introducción

En el ecosistema de la informática moderna, el Kernel actúa como el núcleo esencial que gestiona la comunicación entre el hardware y los procesos de software. Su arquitectura determina no solo la eficiencia en la ejecución de tareas, sino también la estabilidad y seguridad del sistema global. Paralelamente, los sistemas de archivos definen la estructura lógica mediante la cual se organizan, almacenan y recuperan los datos, siendo vitales para la integridad de la información en entornos críticos.

El propósito de este informe es presentar un análisis técnico comparativo entre diversos modelos de Kernel y estructuras de archivos, fundamentando la selección de tecnologías para el diseño de un laboratorio virtual. A través de esta investigación, se busca validar una infraestructura lógica que optimice el uso de recursos y garantice la conectividad de red necesaria para pruebas de sistemas operativos.

2. Análisis del Proyecto:

De acuerdo con la literatura de Silberschatz y Tanenbaum, el diseño de los sistemas actuales se basa en los siguientes pilares:

2.1. Arquitecturas de Kernel

Modelo	Características Clave	Ejemplos Relevantes
Monolítico	**Todo el sistema operativo opera en el espacio del kernel, permitiendo una comunicación interna veloz pero con mayor riesgo de fallos críticos**	**Linux, FreeBSD.**
Microkernel	**Ejecuta solo funciones mínimas (IPC, memoria) en el kernel, trasladando servicios a espacios de usuario para mejorar la seguridad y el aislamiento.**	**L4, QNX (muy usado en autos)**
Híbrido	**Combina la estructura de un microkernel con servicios críticos integrados en el espacio de núcleo para optimizar el rendimiento sin sacrificar modularidad.**	**Windows NT (XP hasta Windows 11), macOS (XNU).**

2.2. Sistemas de Archivos

- **NTFS (Windows):** Destaca por su manejo avanzado de permisos, cuotas y redundancia mediante Journaling de metadatos.

- **ext4 (Linux):** Se caracteriza por su alta velocidad y baja fragmentación, utilizando estructuras de *extents* para manejar archivos de gran tamaño.

- **APFS (Apple):** Optimizado específicamente para unidades SSD y memorias Flash, priorizando el cifrado fuerte y la creación de instantáneas (*snapshots*).

2.3. Justificación de Tecnologías para el Laboratorio

Para el diseño del laboratorio virtual, se han seleccionado las siguientes tecnologías:

- **Sistema Base**: Linux (Kernel Monolítico Modular) por su capacidad de cargar módulos (LKM) sin reiniciar el sistema, facilitando pruebas de red.
- **Sistema de Archivos**: ext4 y Btrfs, debido a su implementación de *Copy-on-Write* (CoW), lo cual previene la corrupción de datos durante fallos eléctricos simulados en el laboratorio.

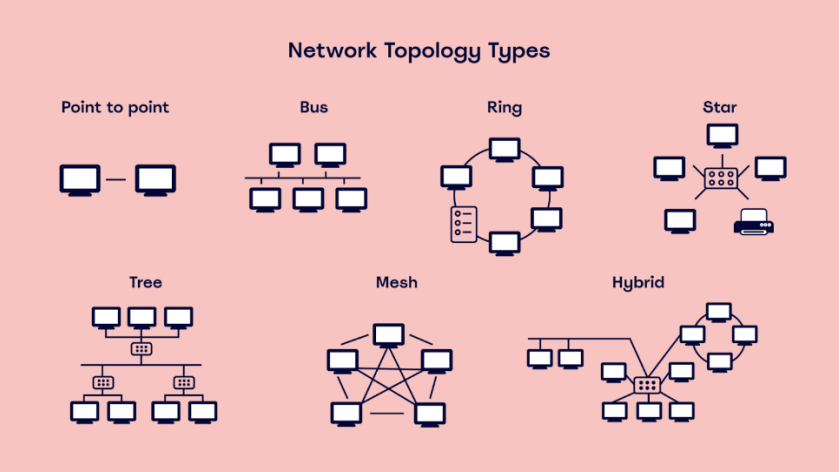
3. Diseño Lógico del Laboratorio

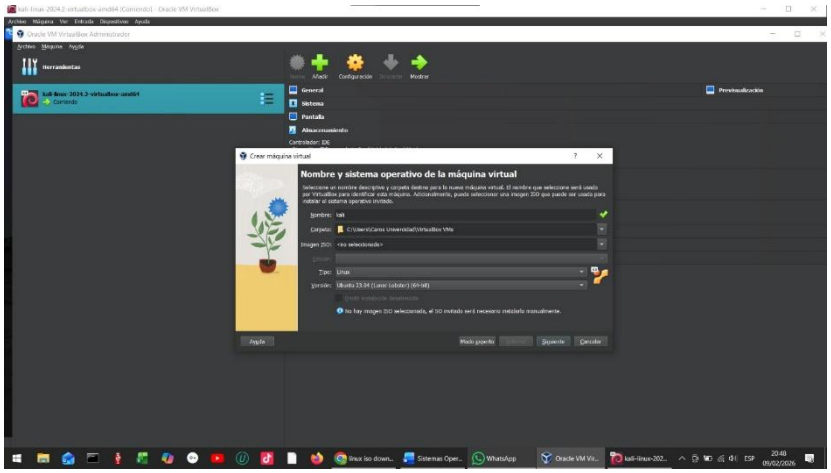
Este apartado detalla la infraestructura necesaria para la implementación virtual:

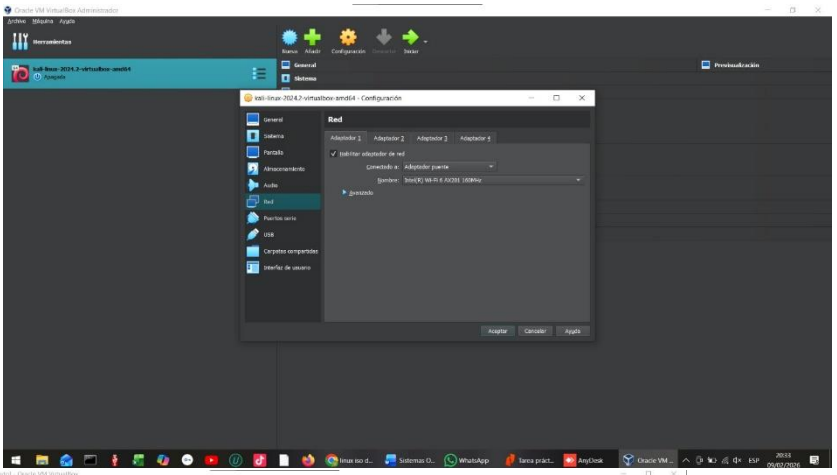
3.1. Asignación de Recursos (Hardware Virtual)

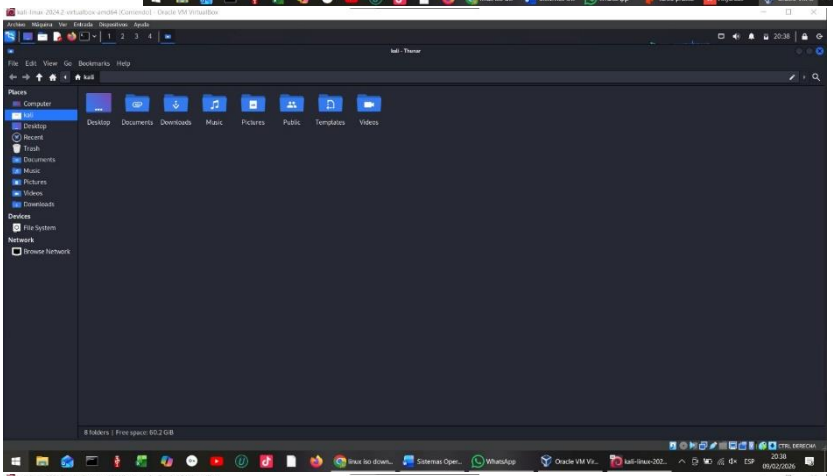
Nodo	Rol	CPU	RAM	Almacenamiento	Sistema de Archivos
VM-01	**Servidor**	**2 Cores**	**4 GB**	**40 GB**	**ext4**
VM-02	**Cliente**	**1 Core**	**2 GB**	**20 GB**	**NTFS / APFS**

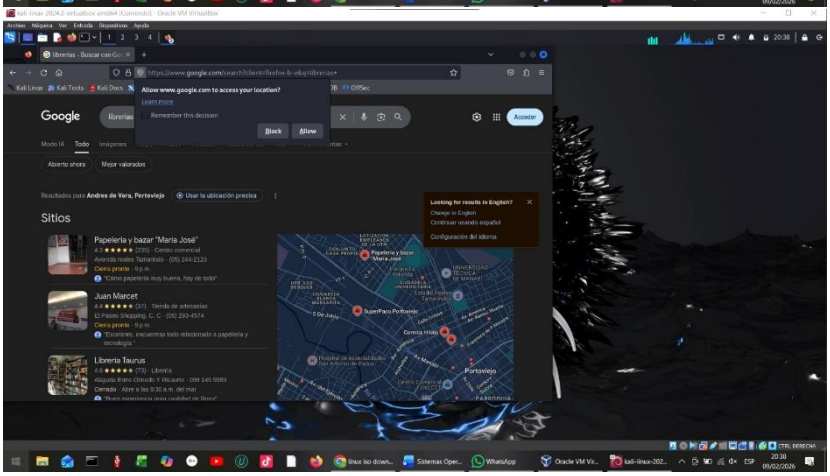
3.2. Implementación

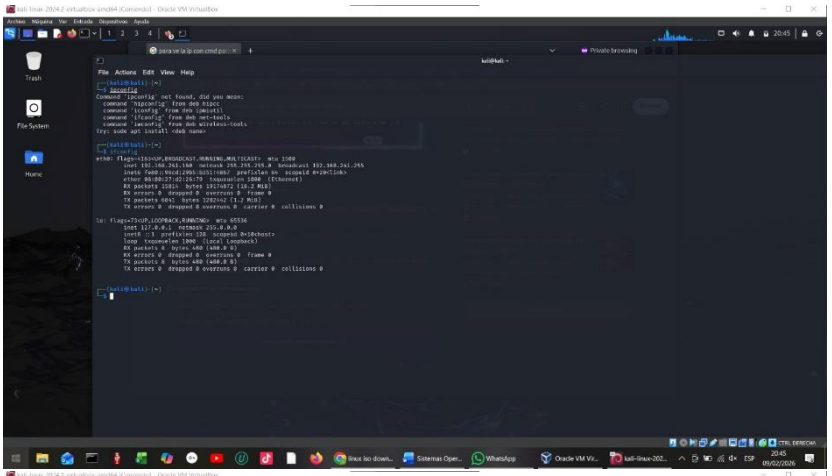


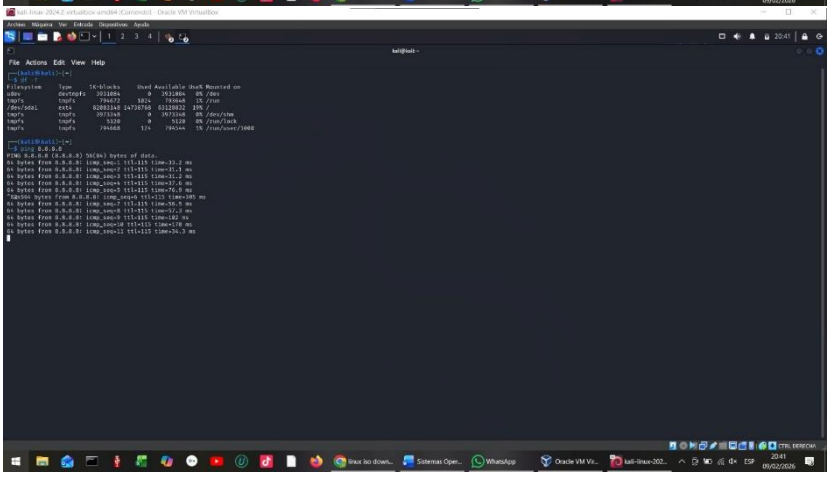












3.3. Esquema de Arquitectura Lógica

Se implementará una topología de red tipo NAT/Red Interna para asegurar que los nodos puedan comunicarse entre sí para las pruebas de montaje de sistemas de archivos remotos, manteniendo el aislamiento del sistema anfitrión.

4. Pruebas y Validación

Para confirmar que la infraestructura responde según lo diseñado, se establecen los siguientes casos de prueba:

- **Prueba de Conectividad**: Verificación de enlace entre nodos mediante comandos de red (Ping/ICMP).
- **Prueba de Integridad**: Montaje y desmontaje de volúmenes para validar el registro de datos en los sistemas de archivos configurados.

5. Conclusiones y Recomendaciones

5.1. Conclusiones

- **Evolución hacia la Practicidad:** La investigación demuestra que la disputa histórica entre arquitecturas monolíticas y microkernels ha convergido en diseños híbridos o monolíticos modulares (LKM) que priorizan el rendimiento sin sacrificar la flexibilidad.
- **Seguridad y Aislamiento:** El renacimiento del Microkernel en sistemas críticos (médicos o militares) resalta que, cuando la seguridad es la prioridad absoluta, el aislamiento de fallos es superior a la velocidad de ejecución.
- **Integridad de Datos:** Los sistemas de archivos modernos han pasado de simples índices a ser guardianes de la integridad mediante tecnologías como *Copy-on-Write* (CoW) y *Checksumming*, eliminando la dependencia del tradicional *fsck* tras fallos eléctricos.
- **Extensibilidad del Kernel:** Tecnologías como eBPF están revolucionando el kernel de Linux, permitiendo programar funciones de red y seguridad "sobre la marcha" sin alterar el código fuente ni comprometer la estabilidad del sistema.

5.2. Recomendaciones

- **Optimización en el Laboratorio:** Se sugiere el uso de sistemas de archivos tipo APFS o Btrfs en entornos de estado sólido (SSD) para aprovechar la gestión eficiente de instantáneas (*snapshots*) durante las fases de prueba.

- **Implementación de Seguridad:** Para futuras expansiones del laboratorio, se recomienda explorar arquitecturas de Microkernel si el objetivo es el despliegue de microservicios altamente aislados.

6. Repositorio del Proyecto

- A continuación, se presenta el enlace al repositorio donde se encuentra alojada la documentación completa, los diagramas de red y los archivos de configuración de las máquinas virtuales:

7. Bibliografía: Arquitecturas de Kernel y Sistemas de Archivos

7.1. Libros y Textos Académicos

- Love, R. (2010). *Linux Kernel Development* (3rd ed.). Addison-Wesley Professional.
- Silberschatz, A., Galvin, P. B., & Gagne, G. (2018). *Operating System Concepts* (10th ed.). Wiley.
- Tanenbaum, A. S., & Bos, H. (2014). *Modern Operating Systems* (4th ed.). Pearson.

7.2. Artículos de Investigación

- Apple Inc. (2020). *Apple File System Guide*. Apple Developer Documentation. developer.apple.com
- eBPF Foundation. (s.f.). *What is eBPF?*. ebpf.io
- Klein, G., Elphinstone, K., Heiser, G., Andronick, J., Cock, D., Derrin, P., Elkaduwe, D., Engelhardt, K., Kolanski, R., Norrish, M., Sewell, T., Tuch, H., & Winwood, S. (2009). *seL4: Formal Verification of an OS Kernel*. SOSP '09: Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles. sel4.systems

7.3. Documentación Técnica de la Industria

- Amazon Web Services. (s.f.). *Amazon S3 Architecture: Object Storage for the Cloud*. aws.amazon.com
- Google Cloud. (s.f.). *Google Cloud Storage: Architecture and Concepts*. cloud.google.com
- The Linux Kernel Archives. (s.f.). *ext4 Filesystem Documentation*. www.kernel.org

7.4. Recursos Multimedia y Foros Especializados

- Gregg, B. (2019, 12 de febrero). *eBPF: Unlocking the Kernel* [Video]. YouTube.
- LWN.net. (s.f.). *Linux Weekly News: Kernel and Filesystem Development Archives*. lwn.net
- OSDev Wiki. (2023). *File Systems and Kernel Design*. wiki.osdev.org

REPOSITORIO

https://github.com/cibeuniversidad-boop/Sistemas-Operativos-TP1.1