A

Case Study Report On

**……** **Persistent Memory (Intel Optane) …...**

Under the Course “Computer Organization and

Architecture” (AI205)

Submitted by

Second Year B. Tech. CSE (AI&ML)

Name Of Student :- Vaishnavi Nivas Sapkal

PRN:-2417018

Under the Guidance of

**Prof. Namrata Patil**



Department of Computer Science & Engineering (Artificial Intelligence and Machine Learning)

K. E. Society’s

**Rajarambapu Institute of Technology, Rajaramnagar**

(An Empowered Autonomous Institute Affiliated to Shivaji University, Kolhapur)

**2025-2026**

# Abstract

* **Virtual Memory (VM)** in Linux is a core feature of modern operating systems that provides an abstraction of physical memory to applications.
* It enables **efficient memory management** by making processes appear as if they have access to a large, continuous memory block, regardless of physical RAM.
* Virtual memory supports:
  + **Demand paging** – loads pages only when required.
  + **Swapping** – moves inactive pages to disk when RAM is full.
  + **Memory isolation** – protects processes from interfering with each other.
  + **Shared libraries** – allows efficient memory use by sharing code across processes.
* These features make VM **critical for multitasking and system stability**.
* This report explores:
  + The **architecture of virtual memory** in Linux.
  + **Working principles** (paging, page tables, kernel memory management).
  + **Advantages and limitations** of VM.
  + **Use cases** in real-world systems.
  + A **comparison with systems lacking virtual memory**.

# Introduction

* **Traditional memory management** directly maps processes to physical memory, which leads to fragmentation, limited multitasking, and inefficiency.
* **Virtual memory** creates an abstraction layer, allowing each process to believe it has a full address space.
* In Linux, virtual memory is implemented using **paging**, **swap space**, and **page tables**, managed by the kernel.
* Linux divides virtual memory into **user space** (for applications) and **kernel space** (for OS functions).
* Features:
  + Demand paging: loads pages only when needed.
  + Swapping: moves inactive pages to disk when RAM is full.
  + Memory protection: ensures isolation between processes.

# Text Diagram: Instruction Decoding Process

**CPU**

**│**

**▼**

**Virtual Address**

**│**

**▼**

**Page Tables (MMU)**

**│**

**┌──────┴────────┐**

**▼ ▼**

**Physical RAM Swap Space**

**Step-by-step process:**

1. CPU generates a virtual address.
2. The Memory Management Unit (MMU) checks the page tables.
3. If the page is in RAM, it is accessed directly.
4. If not, a **page fault** occurs, and the kernel loads the page from disk (swap).
5. Pages are replaced based on policies like LRU (Least Recently Used)

# Literature Review

* **Importance of Virtual Memory:** Tanenbaum (2021) explains VM as essential for multitasking and process isolation in modern OS.
* **Linux Kernel Documentation:** Describes paging, page replacement, and swap as core components.
* **Performance Studies:** Experiments show VM enables large applications to run even with limited RAM.
* **Security Aspect:** Virtual memory prevents one process from interfering with another.
* **Challenges:** Excessive swapping (thrashing) reduces performance drastically.

# Case Study

* **Problem:**
* Applications often require more memory than physically available RAM. Without VM, multitasking is limited, and system crashes occur under memory pressure.
* **Linux Virtual Memory Solution:**
* **Paging:** Divides memory into fixed-size pages for efficient allocation.
* **Swapping:** Uses disk space to store inactive pages.
* **Kernel Memory Management:** Maintains page tables, page cache, and implements replacement algorithms.
* **Shared Memory:** Enables processes to share libraries without duplication.
* **Use Cases:**
* Running large databases on systems with limited RAM.
* Cloud environments with memory overcommitment.
* Isolated execution of multiple user applications.
* Embedded Linux systems requiring efficient RAM usage.

# ADVANTAGES

* Allows execution of programs larger than physical RAM.
* Provides process isolation and system stability.
* Supports efficient use of limited memory resources.
* Enables multitasking by sharing and protecting memory.
* Simplifies programming as applications do not manage physical memory directly.

# FUTURE TRENDS

* **Improved page replacement algorithms** to minimize swapping.
* **Hybrid memory systems** combining DRAM with NVRAM for faster swap.
* **Machine learning–based VM management** in Linux kernels.
* **Containerized environments** requiring fine-grained VM tuning.

# Conclusion

# Virtual memory in Linux provides an abstraction that simplifies application execution and ensures efficient memory utilization.

# It allows multitasking, memory protection, and execution of programs exceeding physical RAM.

# While swapping introduces latency, VM ensures system reliability and flexibility.

# With advancements in memory technology and Linux kernel updates, virtual memory will remain central to computing.

# References

* Tanenbaum, A. S. (2021). *Modern Operating Systems*. Pearson.
* Bovet, D. P., & Cesati, M. (2020). *Understanding the Linux Kernel*. O’Reilly Media.
* Love, R. (2010). *Linux Kernel Development*. Addison-Wesley.
* Linux Kernel Documentation. (2023). *Virtual Memory Management*.
* Silberschatz, A., Galvin, P. B., & Gagne, G. (2020). *Operating System Concepts*. Wiley.