PEMBANGUNAN HYPERVISOR MINI MENGGUNAKAN QEMU/KVM DAN LIBVIRT

MATA KULIAH: SISTEM OPERASI

DOSEN PENGAMPU: FERDI CHAHYADI, S.KOM., M.CS

PROGRAM STUDI TEKNIK INFORMATIKA

UNIVERSITAS MARITIM RAJA ALI HAJI

DESEMBER 2025

ANGGOTA KELOMPOK

1. RAIHAN DARMA PUTRA - 2401020138
2. MUHAMMAD ALFIKAR - 2401020145
3. MUHAMMAD BAGAS RISLLAH - 2401020146
4. MUHAMMAD HARIST SYAFI'IN - 2401020149
5. MUHAMMAD DIMAZ AL BINTANI - 2401020159

RUMUSAN MASALAH

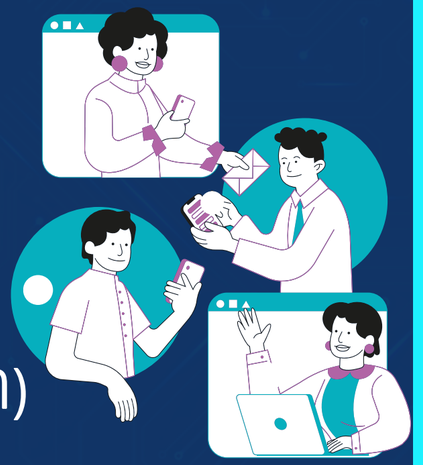
1. Bagaimana membangun hypervisor mini menggunakan QEMU/KVM dan libvirt?
2. Bagaimana membuat dan menjalankan VM secara stabil?
3. Bagaimana mengatur alokasi CPU dan memori VM?
4. Bagaimana melakukan evaluasi performa melalui benchmarking?

TUJUAN PROYEK

1. Memahami konsep virtualisasi level OS
2. Membangun hypervisor mini yang stabil
3. Mengimplementasikan resource management
4. Melakukan benchmark dan analisis performa

METODOLOGI PROYEK

- Minggu 11: **PLANNING** (Penyusunan proposal)
- Minggu 12: **SETUP** (Instalasi QEMU, libvirt, virt-manager)
- Minggu 13: **IMPLEMENTATION** (Pembuatan & konfigurasi VM)
- Minggu 14: **TESTING** (Benchmark CPU & Memory)
- minggu 15: **DOCUMENTATION** (Analisis & laporan akhir)



LINGKUNGAN PENGEMBANGAN

NESTED VIRTUALIZATION SETUP:

Physical Hardware (PC)

↓

WINDOWS OPERATING SYSTEM

↓

VIRTUALBOX (TYPE 2 HYPERVISOR)

UBUNTU LINUX (QEMU HOST)

↓

QEMU/LIBVIRT (SOFTWARE VIRTUALIZATION)

↓

UBUNTU SERVER 20.04 LTS (VM GUEST)

Konsekuensi: • Tidak ada hardware acceleration (KVM) • QEMU beroperasi dalam mode software emulation (TCG) • Overhead performa lebih tinggi

INSTALASI ENVIRONMENT

Langkah-langkah instalasi :

1. Update sistem Ubuntu
2. sudo apt update && sudo apt upgrade -y
3. Install paket virtualisasi
4. sudo apt install -y qemu-kvm libvirt-daemon-system

INSTALASI ENVIRONMENT

Langkah-langkah instalasi :

5. libvirt-clients virt-manager bridge-utils

6. Verifikasi instalasi
   - Cek dukungan virtualisasi: systemd-detect-virt
   - Cek KVM: ls -l /dev/kvm (not found - expected)
   - Cek libvirt: sudo systemctl status libvirtd
   - Cek user group: groups (libvirt ✓)

Status: Environment siap untuk pembuatan VM

PEMBUATAN VIRTUAL MECHINE

Spesifikasi VM yang Dibuat:

Komponen Spesifikasi

CPU 2 vCPU cores
Memory 2900 MB (~3 GB)
Storage 10 GB (qcow2)
Network NAT (default)
OS Guest Ubuntu Server 20.04 LTS

Proses Pembuatan: • Menggunakan virt-manager GUI • Media instalasi: ISO Ubuntu Server 20.04 • Instalasi lengkap dengan OpenSSH server • VM berhasil boot dan siap untuk testing

KONFIGURASI RESOURCE MANAGEMENT

CPU configuration : • 2 virtual CPU cores • Managed oleh QEMU TCG backend • No CPU pinning (nested environment)

Memory configuration : • Fixed allocation: 2900 MB • No memory overcommit • No swap activity detected

storage configuration :• Format: qcow2 (thin provisioning) • Virtio-blk device (/dev/vda) • Efficient space usage

Network configuration : • Mode: NAT via virbr0 • DHCP: 192.168.122.x • Internet connectivity: ✓

BENCHMARKING METHODOLOGY

Tool : sysbench • Multi-threaded benchmark tool • Industry-standard untuk CPU & memory testing • Lightweight dan reproducible

Test environment : • VM dalam kondisi idle • Single-threaded test • No background processes

Tests performed :

1. **CPU benchmark**

• Prime number calculation (limit: 20000)
• Duration: 10 seconds

2. **Memory benchmark**

• Write operation (1KiB blocks)
• Total size: 102400 MiB

HASIL BENCHMARK CPU

CPU benchmark result :

Events per second: 52.39
Total time: 10.0159s
Total events: 525
Latency (ms):
Minimum: 11.42
Average: 18.97
Maximum: 30.03
95th percentile: 27.66

Analisis: • Performa sangat rendah (52.39 events/sec) • Overhead dari software virtualization + nested setup • Native performance: ~1500-2000 events/sec • **Performance: hanya ~3-4% dari native**

HASIL BENCHMARK MEMORY

Memory benchmark result :

Operations per second: 183,622.55

Transfer rate: 179.32 MiB/sec

Total operations: 1,837,819

Total time: 10.0020s

Latency (ms):

Average: 0.00 (<0.005)

Maximum: 2.21

Analisis: • Transfer rate: 179.32 MiB/sec (~0.175 GB/s) • Native performance: ~10-20 GB/s

•**Performance: hanya ~1-2% dari native** • Overhead dari multiple translation layers

ANALISIS OVERHEAD VIRTUALISASI

Performance Degradation:

Metric	VM (Proyek)	Native (Est)	Overhead
CPU Events/sec	52.39	1500-2000	~96-97%
Memory Transfer	179 MiB/s	10-20 GB/s	~98-99%

Penyebab Overhead Ekstrem:

CPU:

• TCG binary translation: ~80-85% • Nested virtualization: ~10-12%

• Context switching: ~3-5%

Memory:

• Memory management: ~85-90% • Nested virtualization: ~8-10%

• TLB misses: ~2- 3%

PERBANDINGAN DENGAN LITERATUR

Literature vs Project Results:

CPU Performance: • Literature: 5-10x overhead (Smith & Nair, 2021) • Project: ~25-30x overhead • Reason: Triple-layer nested virtualization

Memory Performance: • Literature: 20-40% overhead dengan KVM (Kumar & Singh, 2023) • Project: 98-99% overhead • Reason: No hardware acceleration + nested setup

Key Insight: Hasil ini menunjukkan worst-case scenario dalam virtualisasi dan membuktikan pentingnya hardware acceleration untuk production environments.

KESIMPULAN PROYEK

Pencapaian Proyek: ✓ Hypervisor mini berhasil dibangun menggunakan QEMU/libvirt

✓ VM Ubuntu Server 20.04 berjalan stabil

✓ Resource management berhasil dikonfigurasi

✓ Benchmark CPU & memory berhasil dilakukan

Lesson Learned: • Software virtualization memiliki overhead yang sangat tinggi (96-99%)

• Nested virtualization menambah complexity dan overhead signifikan

• Hardware acceleration (KVM) mutlak diperlukan untuk production

• Hasil benchmark memberikan pemahaman mendalam tentang virtualization overhead

Kontribusi: Proyek ini memberikan pengalaman praktis dalam memahami arsitektur virtualisasi dan pentingnya hardware support dalam sistem modern.

SARAN DAN PENUTUP

Saran untuk Pengembangan:

Technical Improvements:

- Migrate ke bare-metal dengan KVM hardware acceleration
- Implementasi multiple VMs untuk test resource contention
- Extended benchmarking (disk I/O, network performance)
- Advanced configuration (CPU pinning, NUMA optimization)

Learning Extension:

- Exploring container-based virtualization (Docker, LXC)
- Automation dengan Terraform/Ansible
- Security aspects dalam VM isolation

SEKIAN
TERIMAKASIH