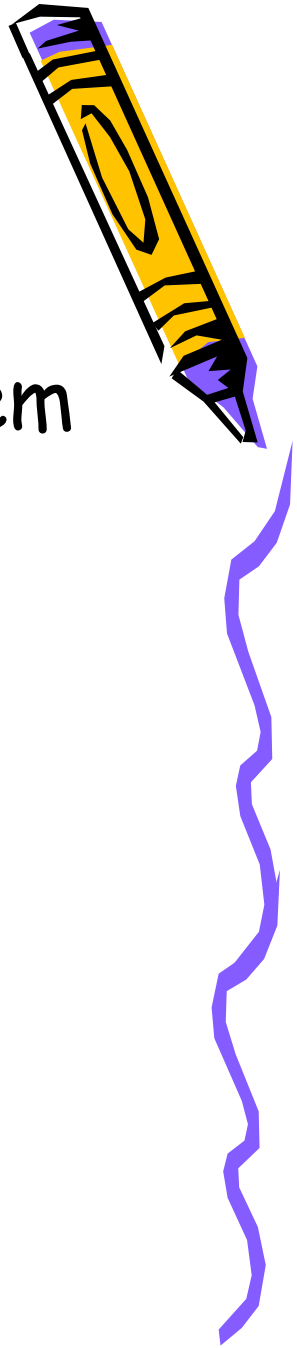
Memori pada Sistem Linux

Heri Kurniawan

OS-Gasal 2009/2010

Tujuan Pembelajaran



- Memahami manajemen memori sistem linux
- Memahami memori virtual linux

Manajemen memori

- Manajemen memori dilinux, dibagi dua komponen:
  - Alokasi page dan pengosongan memori fisik
  - Memori Virtual ->pemetaan virtual address
- Memori fisik dibagi dalam tiga wilayah
  - ZONE_DMA : digunakan legacy device (e.g ISA) untuk mengakses memori secara langsung (e.g: transfer data (DMA))
  - ZONE_NORMAL : digunakan DMA bila ZONE_DMA tidak ada
  - ZONE_HIGHMEM : alokasi sistem (page cache, buffer filesystem, dan lain-lain), jarang digunakan

Manajemen memori

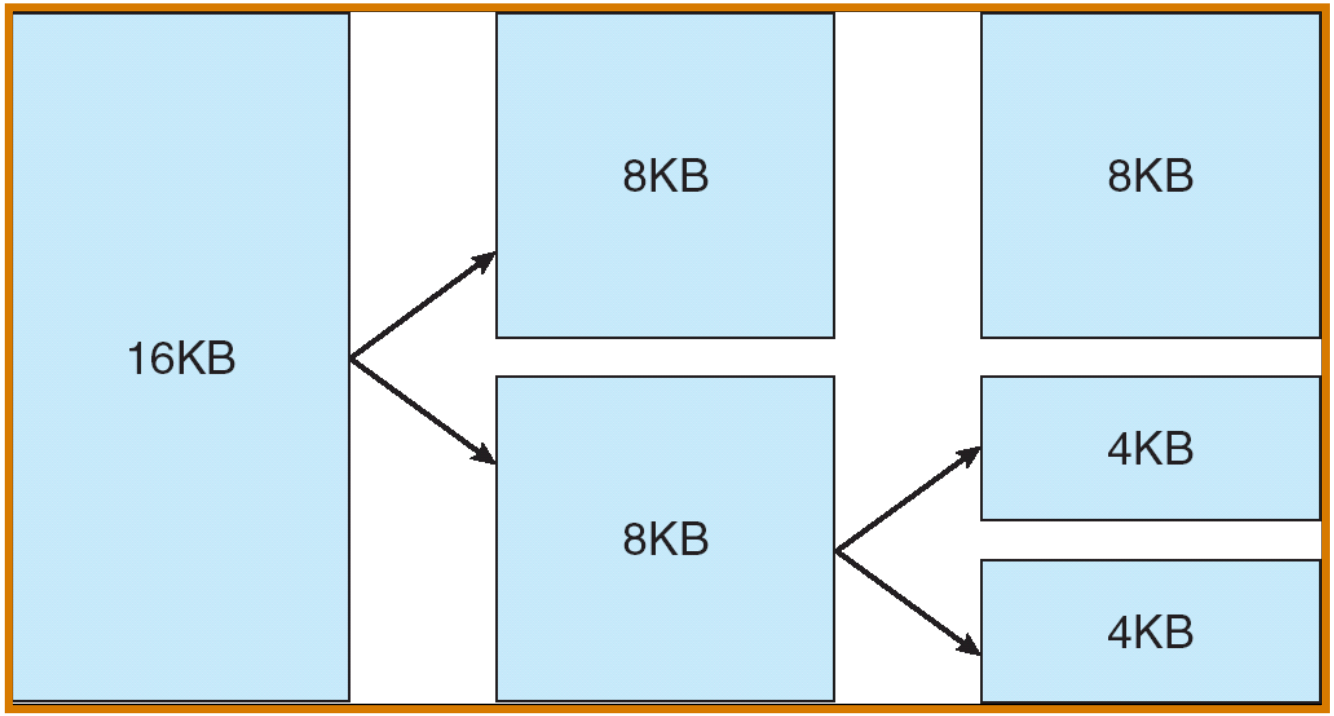
- Kernel memiliki daftar page kosong untuk setiap zone

zone	physical memory
ZONE_DMA	< 16 MB
ZONE_NORMAL	16 .. 896 MB
ZONE_HIGHMEM	> 896 MB

Manajemen Memori

- *Page allocator* → memory manager dilinux. Setiap zone mempunyai allocator
- *Page allocator* → bertugas mengalokasi dan membebaskan page pada setiap zone. Mampu mengalokasi permintaan *contiguous page*.
- *Page allocator* menggunakan *buddy system* untuk menjaga (*keep track*) ketersediaan page
- Alokasi memori kernel linux dapat terjadi secara statis maupun dinamis.
  - Statis → oleh driver yang *mereserve* page berurutan saat boot
  - Dinamis : oleh *page allocator*

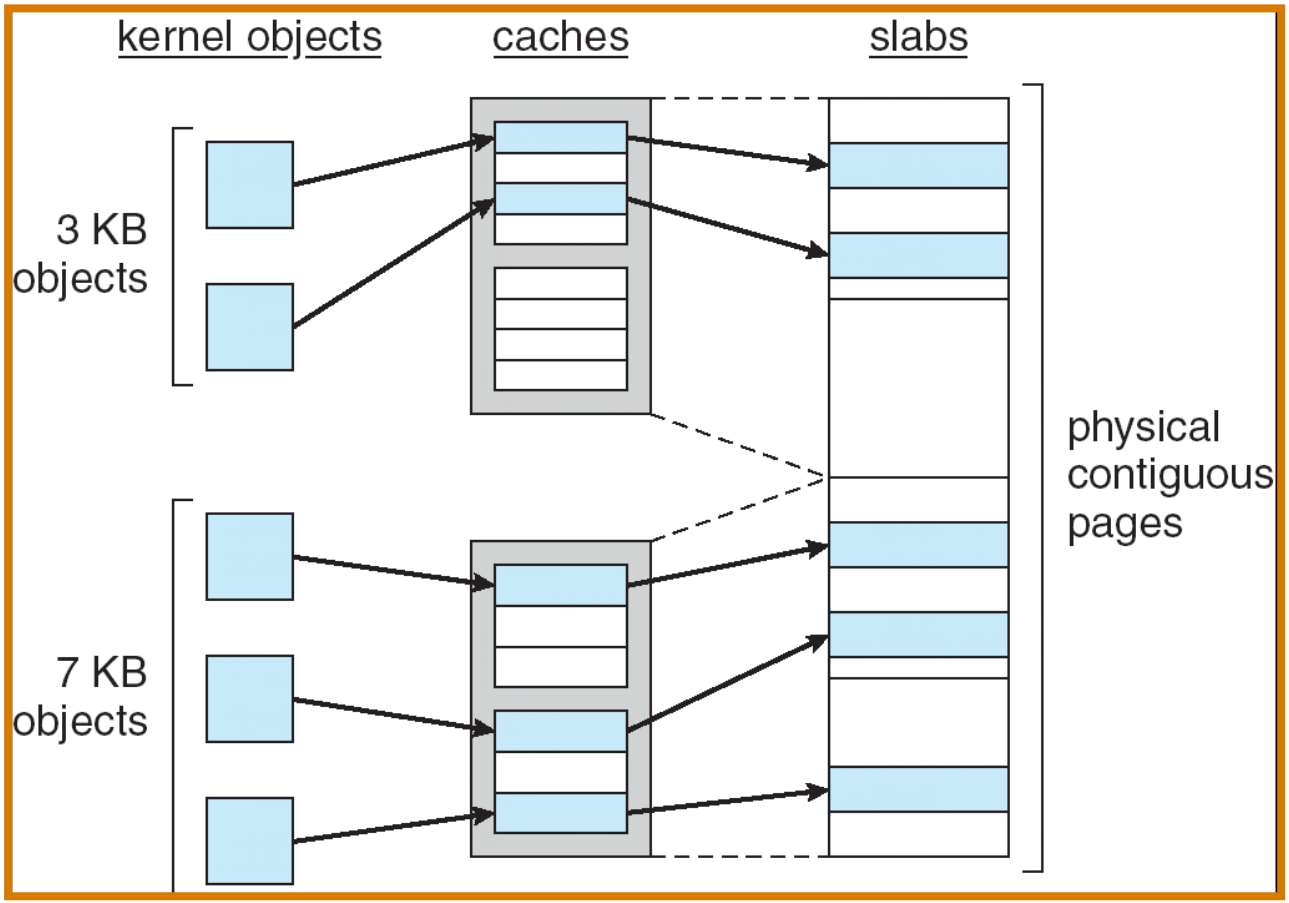
Buddy Heap



Memori Virtual

- Beberapa Page allocator pada Memori Virtual
  - **kmalloc()**, melakukan alokasi permintaan page proses. Tidak semua page dialokasikan ke memori.
  - **slab allocator**, alokasi memori untuk struktur data kernel. Terdiri dari satu atau lebih page berurutan (*contiguous*).
  - **page cache**, cache untuk *block oriented device* dan *memory mapped files*.
    - Cache untuk blok device, content file (linux disk), network data (NFS)

Slab Allocator di Linux



Memori Virtual

- Memori Virtual
  - Maintain ruang alamat bagi setiap proses.
  - Mengatur proses loading page ke memori dan swap out page ke disk
- Dua sisi pandang
  - **logical view**
    - Terdiri dari kumpulan wilayah alamat logika yang tidak saling berhimpit (*nonoverlapping region)*
    - Setiap wilayah mempunyai alamat yang berurutan

Memori Virtual

- Alamat awal wilayah dimulai dari awal page memori virtual (page-aligned)
- Deskripsi properti untuk setiap wilayah ->vm_area_struct
  - berisikan : status read/write/execute /permission suatu proses, *associated file*

- **Physical view**
  - Menggunakan *hardware page table* per proses
  - Diatur oleh sejumlah routine dari *software interrupt handler* milik kernel

Memori Virtual

- Page dari setiap wilayah memory dapat disimpan dalam bentuk,
  - File
  - Nothing -> kosong (demand zero memory)
    - Page dengan isi nol
- Private atau Shared
  - private
    - Jika proses menulis pada wilayah private, perubahan berlaku hanya pada proses tersebut
  - shared
    - Jika proses menulis pada wilayah shared, perubahan berlaku pada proses lain

Memori Virtual

- Kernel membuat virtual address yang baru saat:

1. Sebuah proses menjalankan program baru dengan menggunakan system call **exec()**

    - Saat program baru dieksekusi, proses mendapatkan alamat virtual address yang baru.
    - Selanjutnya routine menempatkan alamat tersebut pada wilayah virtual memori

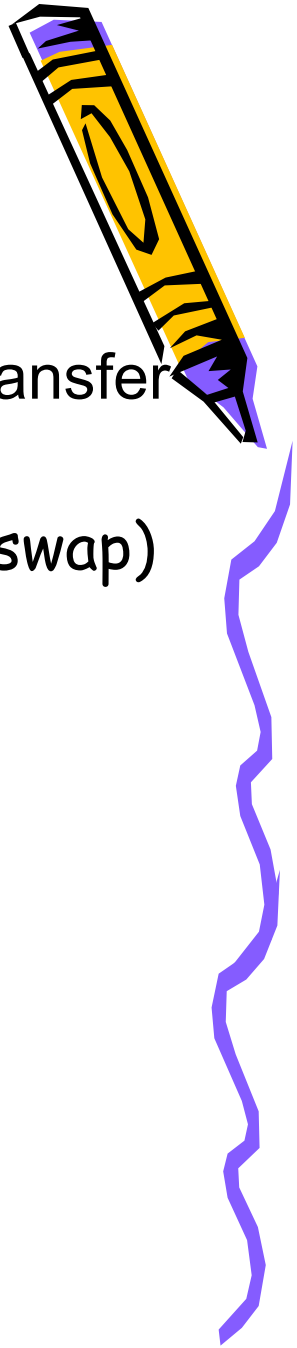
Memori Virtual

- 2. Saat membuat proses dengan system call **fork()**
  - Membuat duplikasi baru dari ruang virtual address yang sudah ada
  - Kernel menduplikasi vm_area_struct descriptor parent, kemudian membuat page table baru untuk child
  - Isi page table parent diduplikasi langsung ke page table milik child
  - Parent dan child *share* page yang sama pada memori fisik

Memori Virtual

- Paging melakukan relokasi dari memori fisik ke disk saat slot memori dibutuhkan oleh page lain
- Linux menggunakan mekanisme paging, tidak melakukan swapping seluruh proses
- Sistem Paging di linux dibagi menjadi dua seksi:
  - *Pageout-policy algorithm,* menentukan page mana yang akan ditulis ke disk dan kapan penulisan dilakukan
    - menggunakan ubahan algoritma clock (multiple-pass clock) & LFU

Memori Virtual



- Mekanisme paging yang menangani aktifitas transfer dan pengembalian page dari disk ke memori
  - support page out ke swap device (partisi swap) atau file

Memori Virtual

- Page table untuk kernel linux mempunyai label proteksi
- Tujuan : agar page tidak dapat dirubah atau dilihat saat prosesor berjalan dalam mode user.
- Memori virtual kernel terdiri dari dua *wilayah*:
  - statis, berisikan *page table* yang merujuk ke page pada memori fisik, penerjemahan dari fisik ke virtual terjadi ketika kode kernel dijalankan
    - isi : kernel utama, kumpulan page yang dialokasi oleh *page allocator*
  - umum (*general purpose*), wilayah ini dapat digunakan secara umum. entri page table dapat dimodifikasi untuk menunjuk lokasi yang berbeda dalam memori.

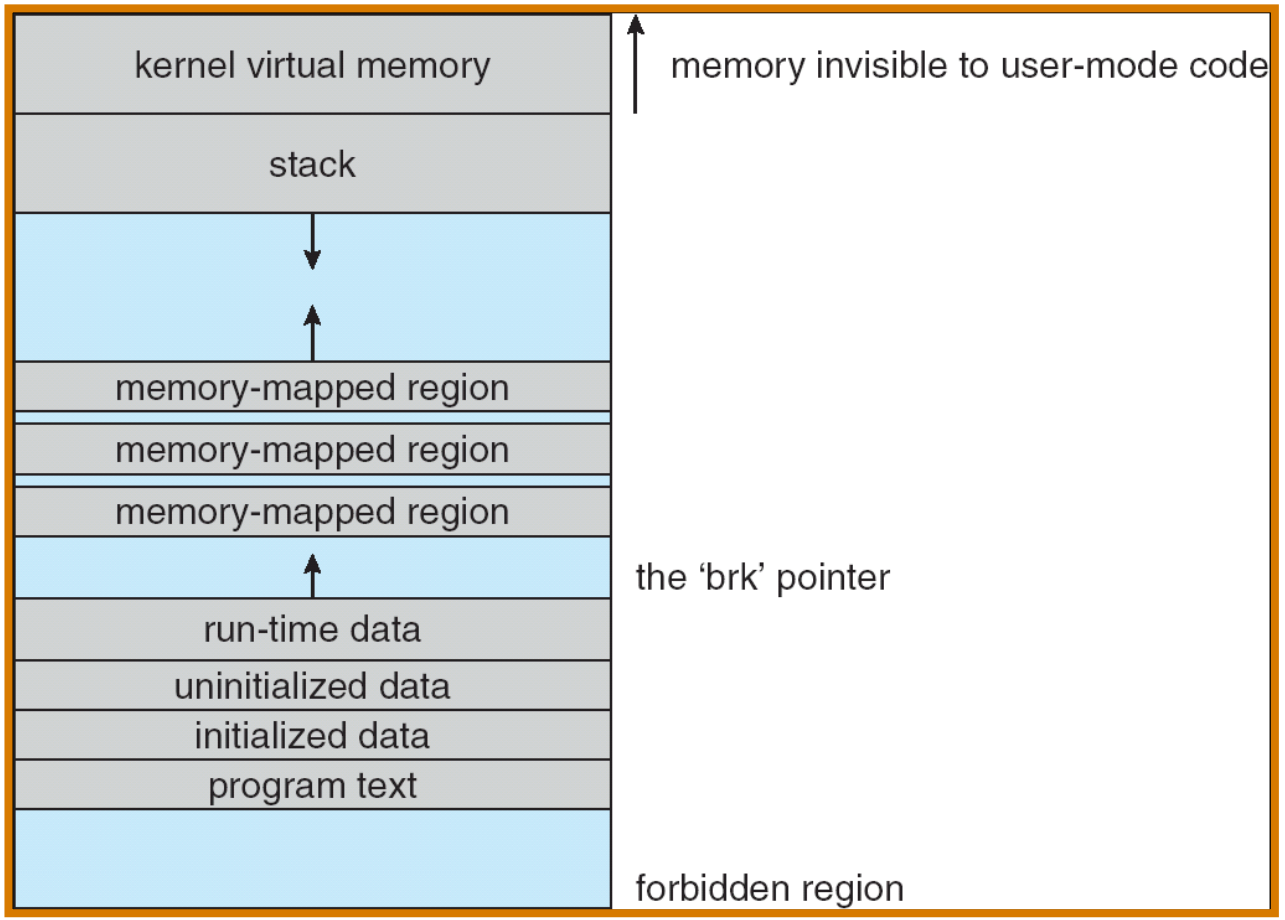
Eksekusi dan Loading program user

- Eksekusi program user dipicu oleh system call exec()
- Overwrite konteks eksekusi aktual dengan konteks prgram yang baru
- Cek permission proses terhadap file
- Kernel memanggil loader routine untuk menjalankan program
- Loader melakukan pemetaan program ke memori virtual
- Linux menggunakan tabel yang berisikan fungsi loader
  - alasan : perbedaan format binary pada kernel 1.0 dan 1.2

Pemetaan program ke Memori

- Linux memasukkan file binary per page ke memori
  - menggunakan demand paging
  - tugas binary loader milik kernel
  - ELF binary file
    - header dan beberapa seksi page
  - ELF loader
    - membaca header dan memetakan seksi page kewilayah memori virtual yang berbeda
- Tugas loader : inisialisasi pemetaan memori untuk eksekusi awal program

Pemetaan program ke Memori



Static dan Dynamic linking

- Static Linking
  - library terintegrasi dalam program binary
  - Tidak efisien
- Dynamic Linking
  - library diluar program binary
  - menggunakan bantuan linker
  - berjalan pada mode user
  - *link function* terintegrasi dalam program
  - *link function* merujuk ke library yang berada di virtual memori