Single-Address-Space FaaS with Jord

Yuanlong Li, Atri Bhattacharyya, Madhur Kumar,
Abhishek Bhattacharjee[†], Yoav Etsion[‡], Babak Falsafi, Sanidhya Kashyap, Mathias Payer
EcoCloud, EPFL *[†]Yale University* *[‡]Technion*

midgard.epfl.ch

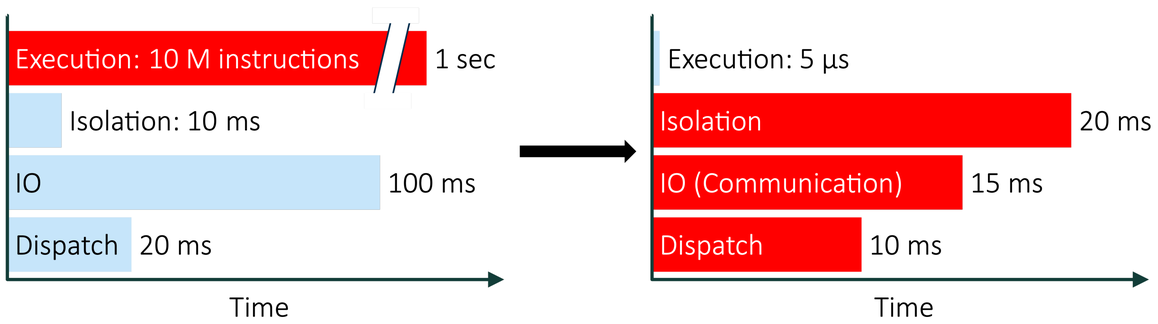
The OS "Address Space" Bottleneck

Address space was introduced for 80s' apps

- Monolithic, user-centric, long-running, little communication
- Sharing small physical memory while being isolated
- App's execution time dominated the runtime

Not suitable for today's datacenter apps

- Service-centric, multi-tenancy, short-running, frequent communication
- Function-as-a-service (FaaS) is an archetype



Millisecond-scale overheads incurred for microsecond-scale apps

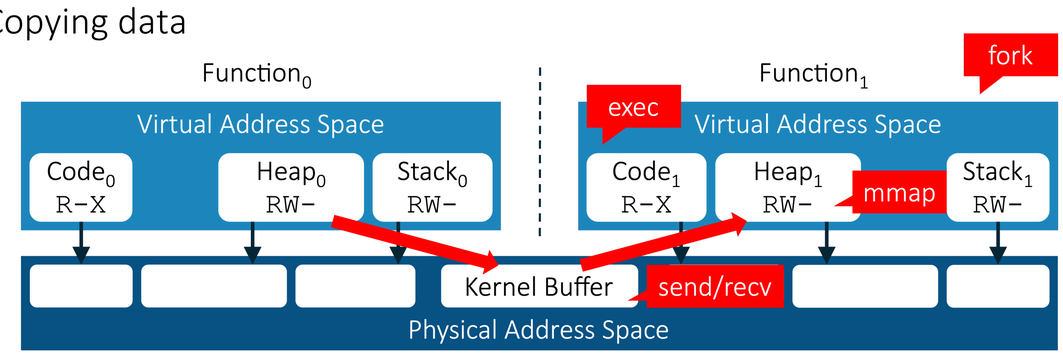
Address Space 101

OS manages address spaces for each function

- Organized using virtual memory areas (VMAs) like code, stack, and heap
- Each VMA represents a logical and contiguous memory region

OS overheads due to per-function address spaces

- Creating/destroying address spaces
- Mapping/unmapping VMAs
- Copying data



OS's address space management is in the millisecond timescale

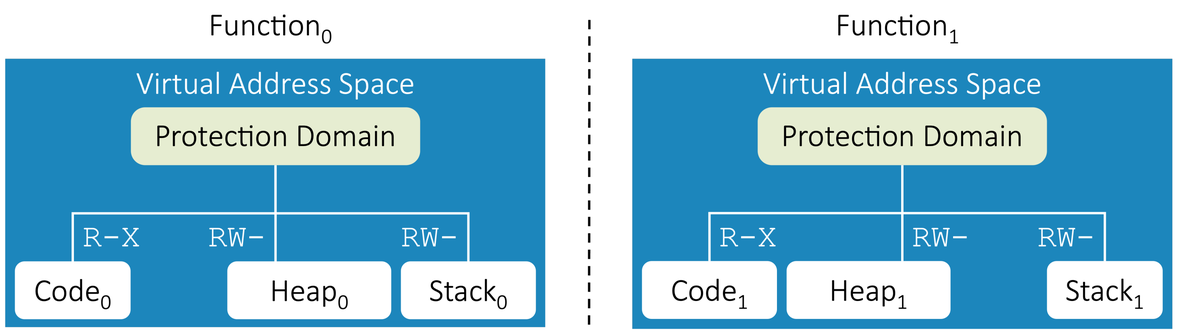
Isolation with Protection Domains

Protection domain in traditional systems

- Defining permissions for all VMAs
- Each address space constitutes a protection domain

Protection domain in single-address-space OSes

- Multiple protection domains can co-exist in the single address space
- Isolation relies on the OS or programming language support



Existing isolation schemes are not efficient or widely adoptable

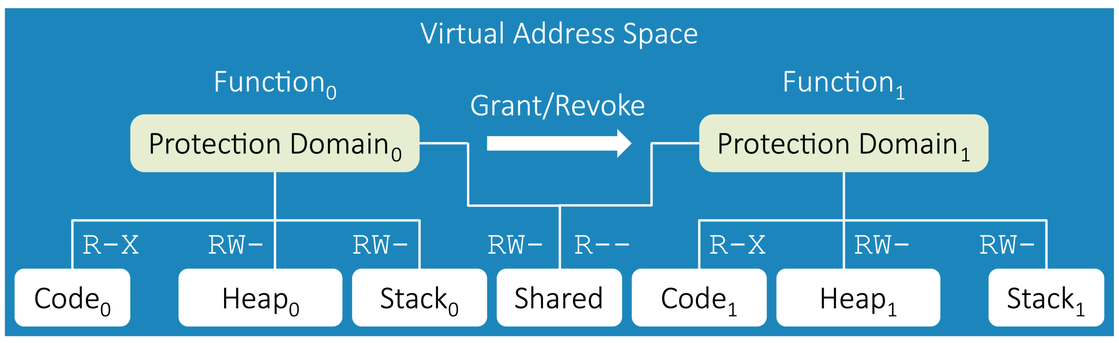
Jord: A Single-Address-Space FaaS

Reviving FaaS's original vision: running a function as a function

- Per-function protection domain
- Nanosecond-scale user-level VMA management with hardware support

Lightweight function serving

- Granting/revoking VMA access among protection domains
- Dispatching function with load balancing



Jord eliminates address space bottleneck with HW-SW co-design

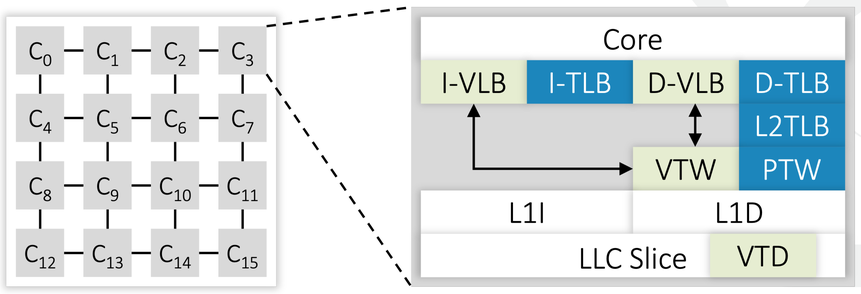
Hardware Support

Nanosecond-scale user-level VMA management

- Direct-addressable user-level VMAs with protection
- Hardware-coherent and protection-domain-aware VMA caches

Compatibility to traditional systems

- Separate address translation path
- Enabled only when translating an address managed by Jord

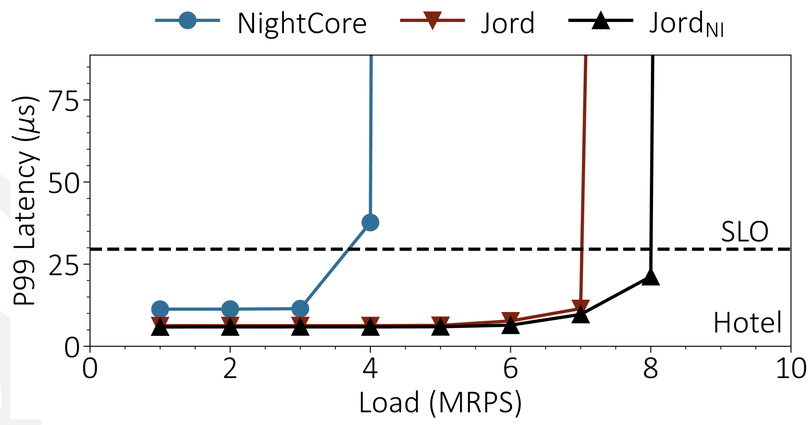


Jord accelerates all VMA operations in hardware

Evaluation

Methodology

- Timing simulation using QFlex + FPGA prototyping
- Linux 6.10 + DeathStarBench/Google's OnlineShop



- Jord achieves throughput under SLO within 13% of the ideal ($Jord_{NI}$)
- Jord only incurs ~11% overhead on average

Jord revives FaaS's original vision