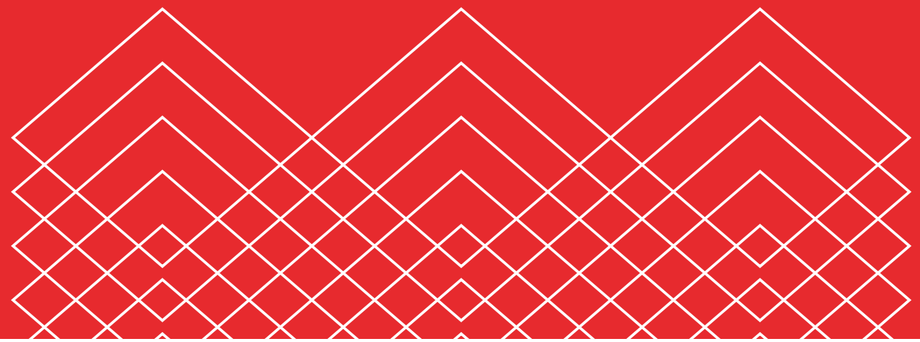
虚拟化Virtualization

CONTENTS

BACKGROUND

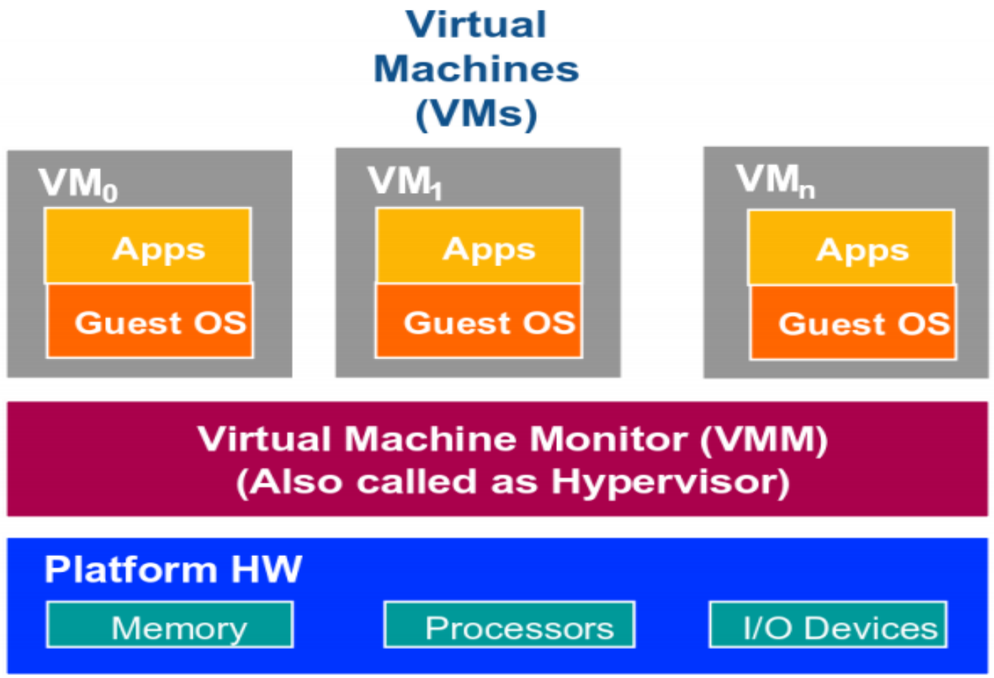


为什么需要虚拟化?

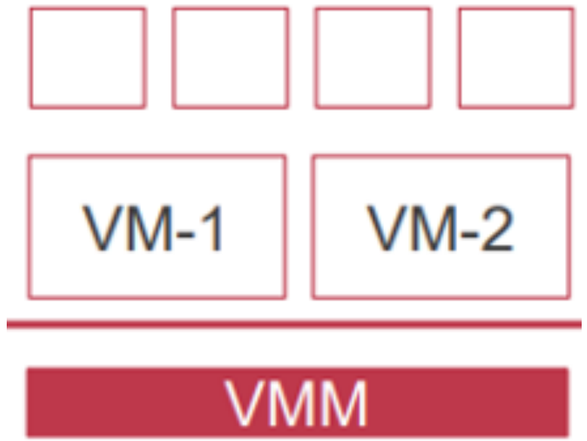
- 性能隔离
- 资源复用，提高利用率

虚拟化的挑战

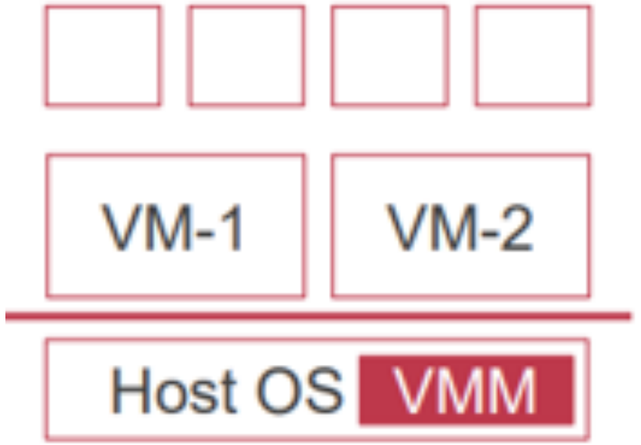
- 虚拟机之间的隔离。
- 支持不同操作系统跑不同的程序。
- 虚拟化的性能开销应该尽可能的小。



虚拟机的类型



Type-1(Xen)

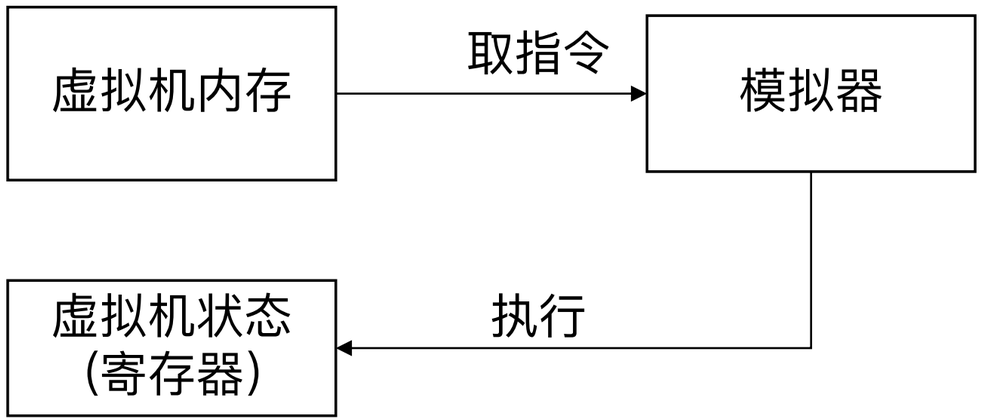


Type-2(KVM)

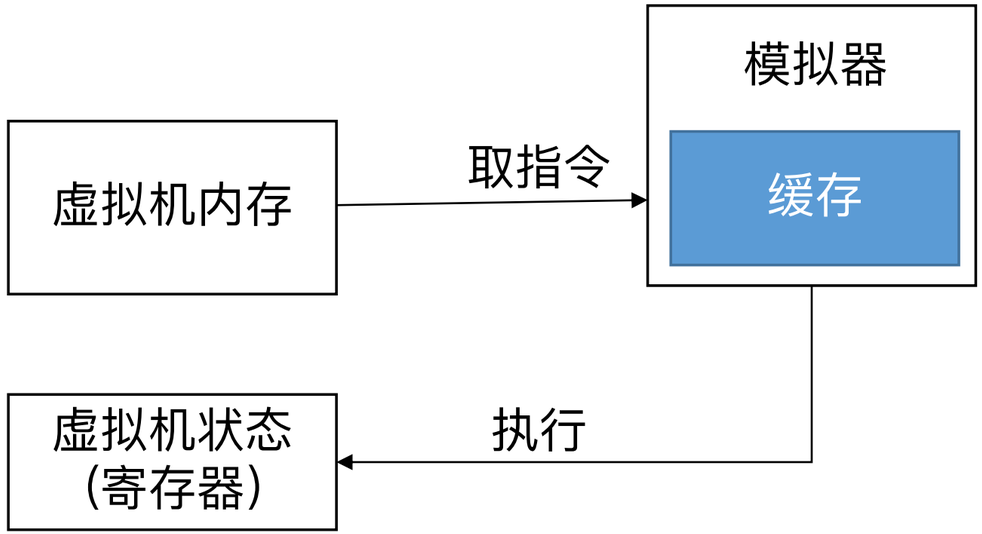
虚拟化的三大任务

CPU虚拟化：为虚拟机提供虚拟处理器的抽象并执行指令的过程。

软件模拟的方式：



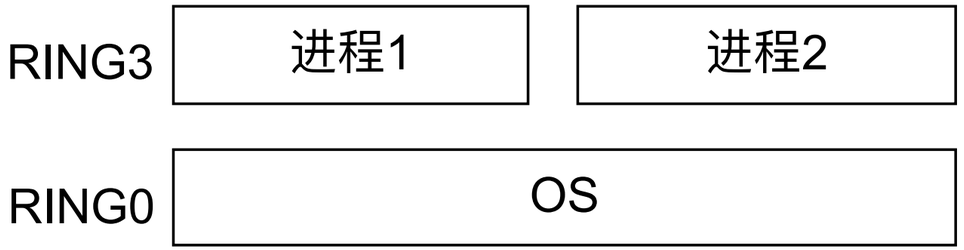
动态二进制翻译：

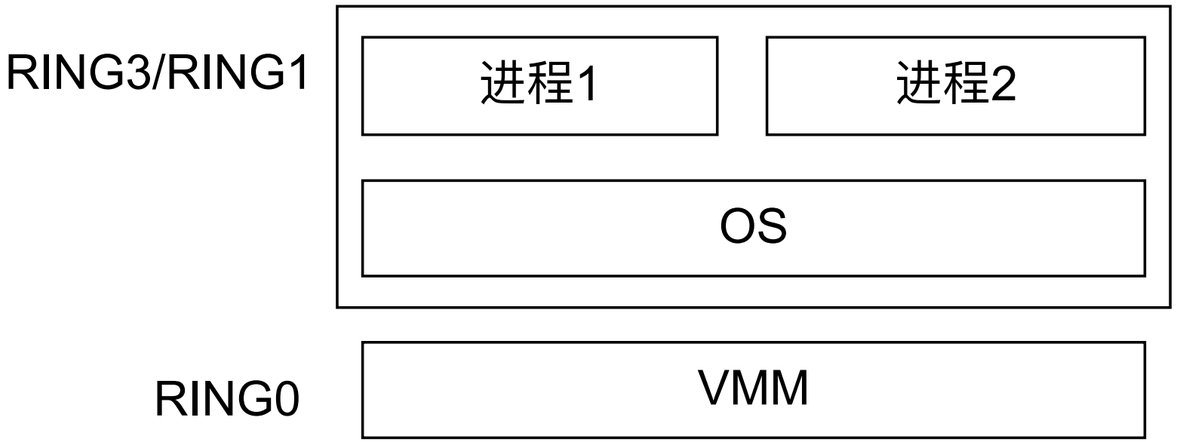


虚拟化的三大任务

CPU虚拟化：为虚拟机提供虚拟处理器的抽象并执行指令的过程。

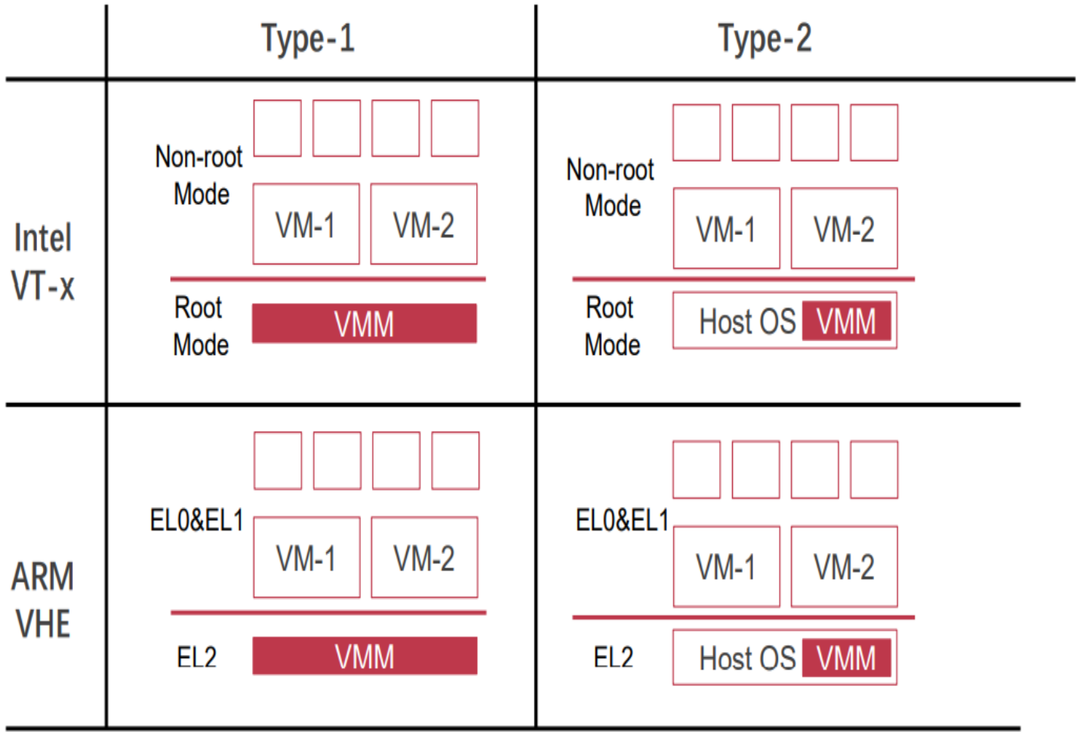
为什么不直接在硬件上运行？**不可虚拟化架构**

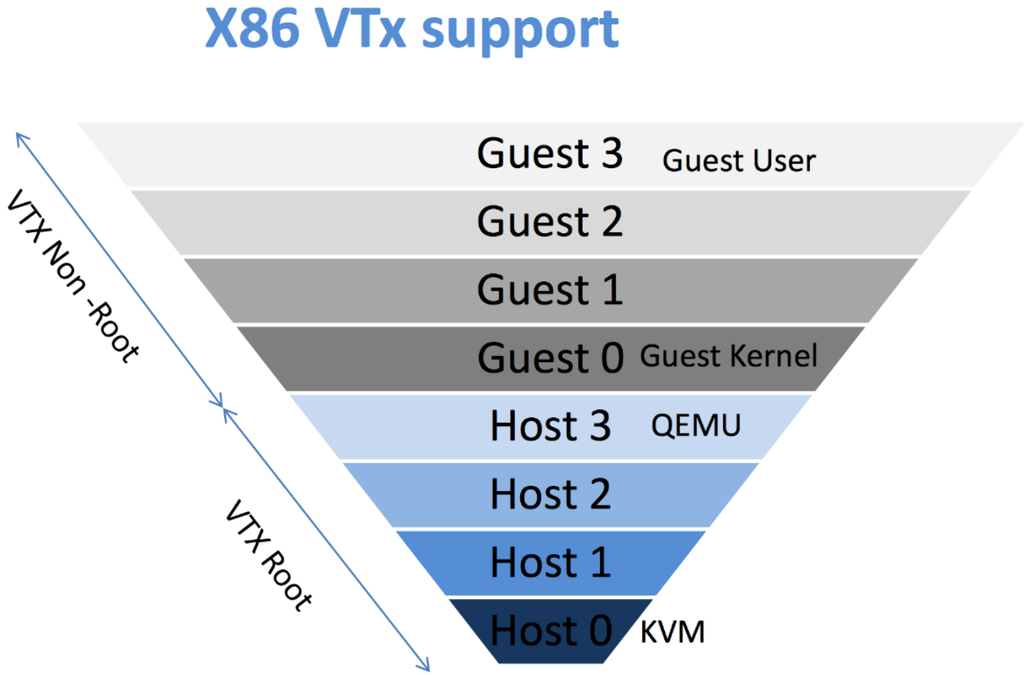




虚拟化的三大任务

CPU虚拟化：硬件上的支持

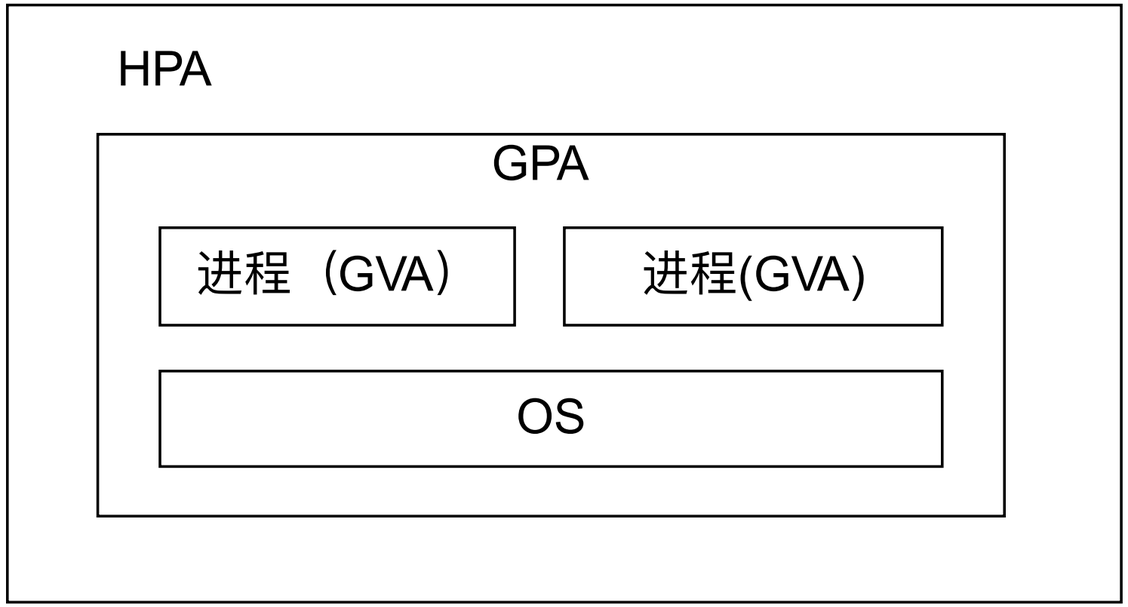




虚拟化的三大任务

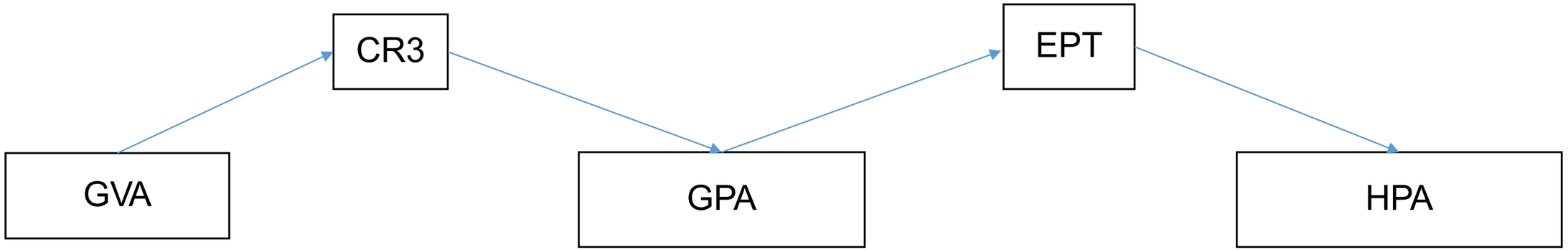
内存虚拟化

- GVA : Guest Virtual Address
- GPA : Guest Physical Address
- HPA : Host Physical Address



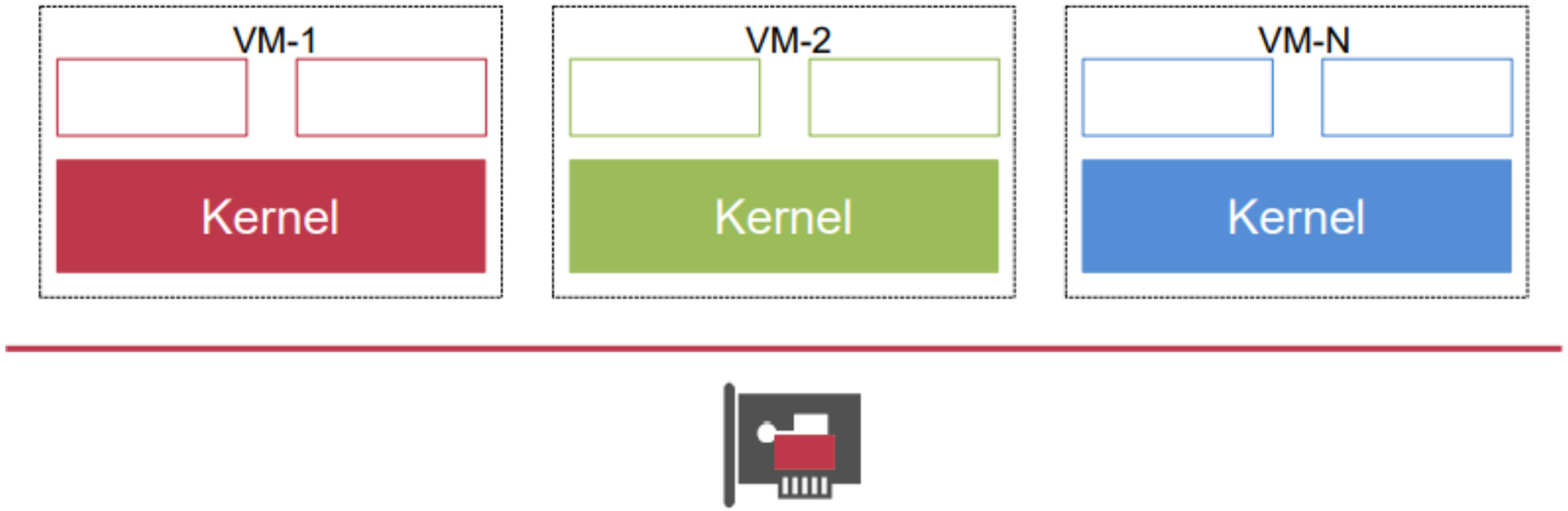
虚拟化的三大任务

内存虚拟化:硬件支持下的方式



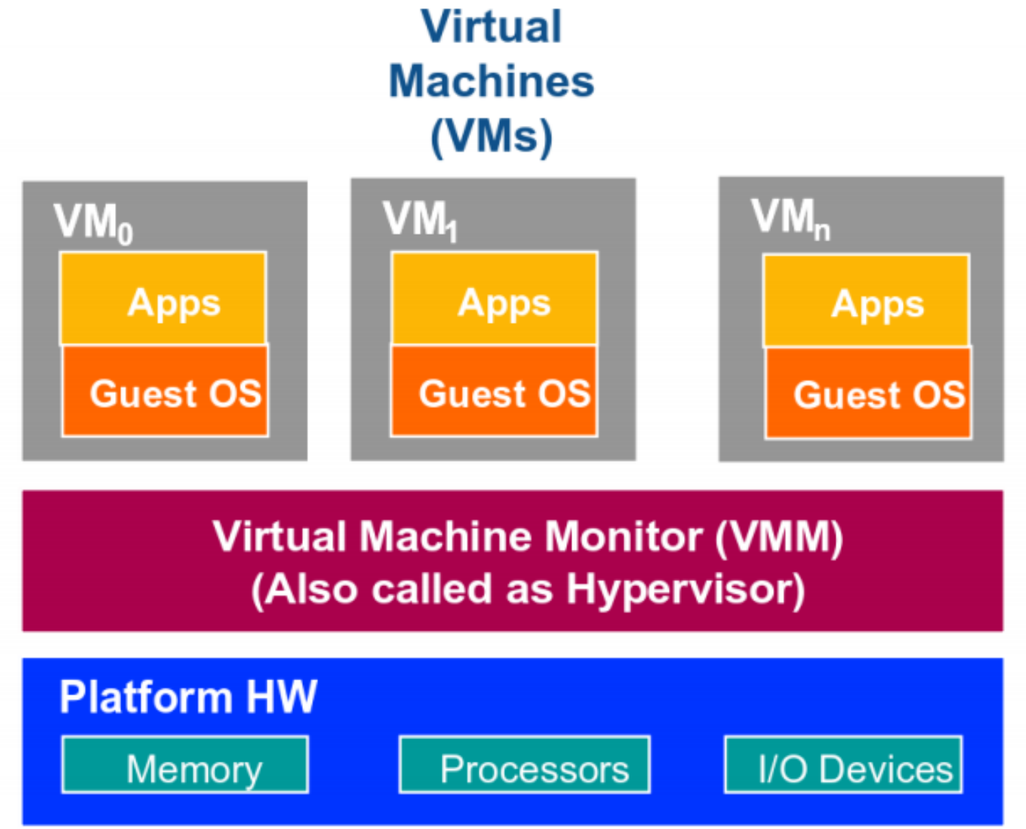
虚拟机的三大任务

I/O虚拟化



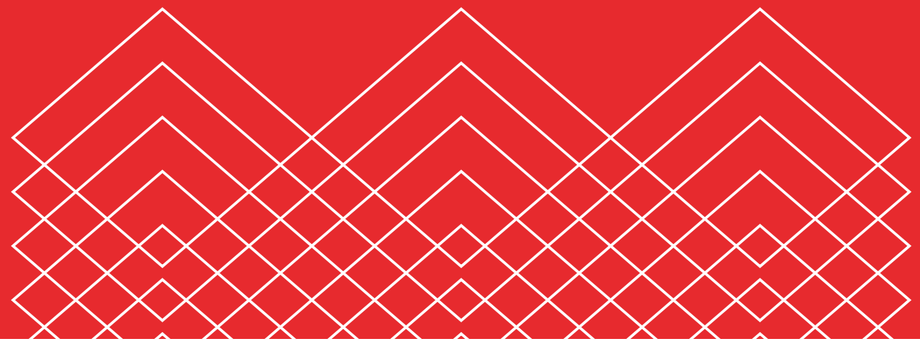
虚拟化的方法

- 软件模拟
- 半虚拟化
- 硬件虚拟化
- 容器



Design principles of Xen

(Xen and the Art of Virtualization)



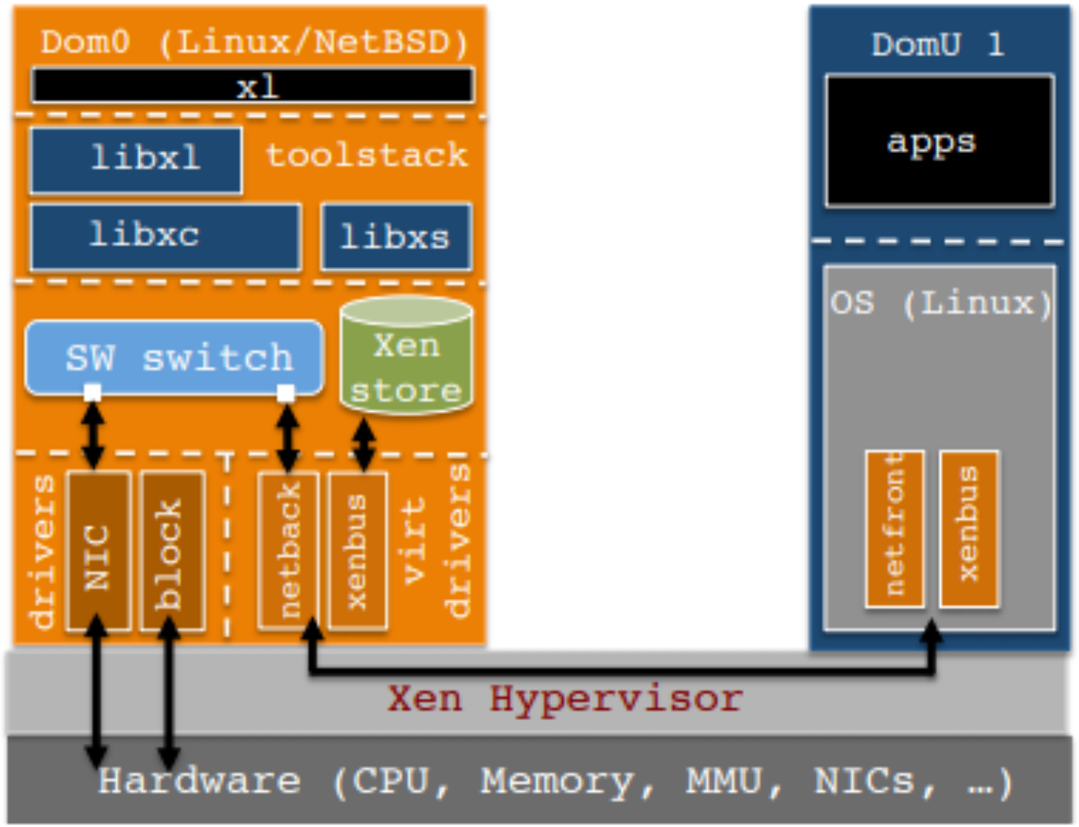
半虚拟化(Paravirtualization)思想

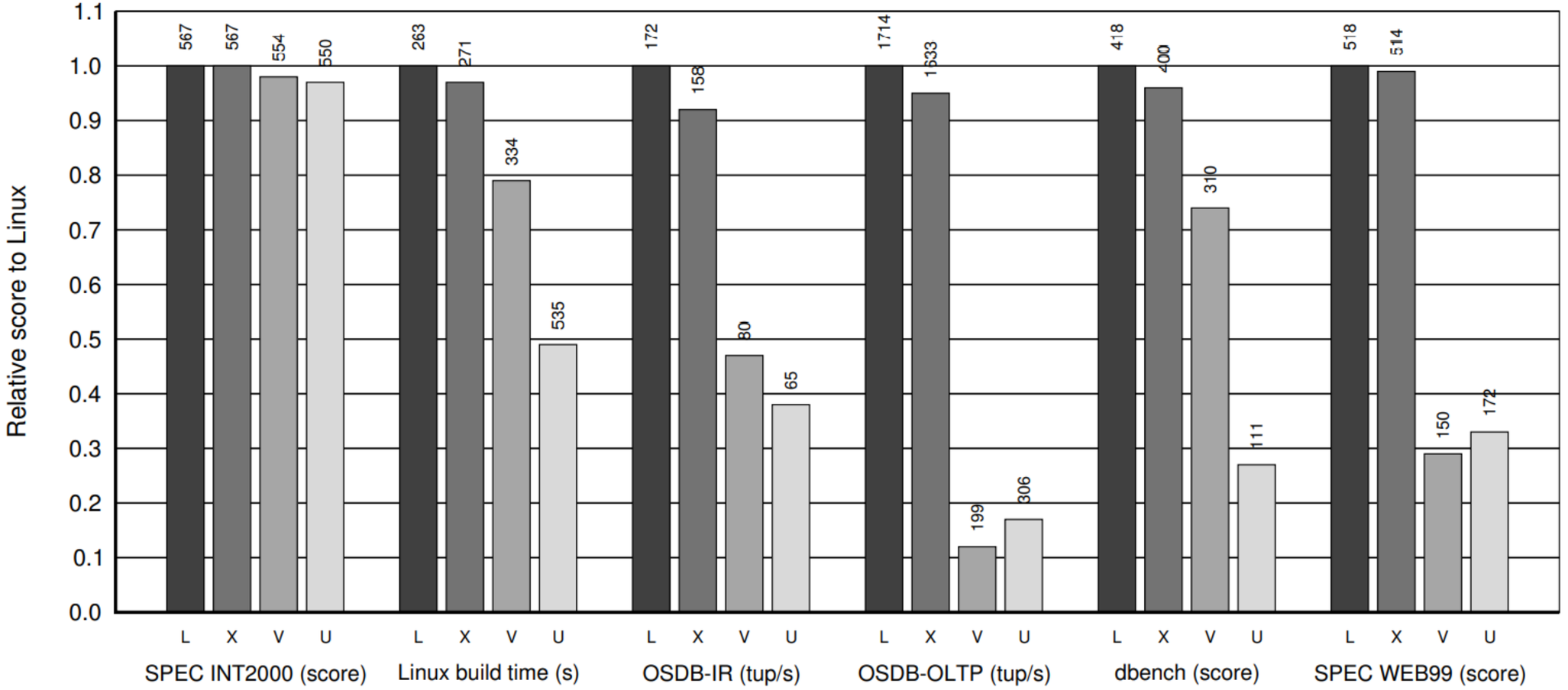
- 半虚拟化下**需要对客户操作系统源代码进行修改**。
- 操作系统可以意识到自己处于一个虚拟机内。

OS subsection	# lines	
	Linux	XP
Architecture-independent	78	1299
Virtual network driver	484	–
Virtual block-device driver	1070	–
Xen-specific (non-driver)	1363	3321
Total	**2995**	**4620**
(Portion of total x86 code base	**1.36%**	**0.04%)**

Xen的架构

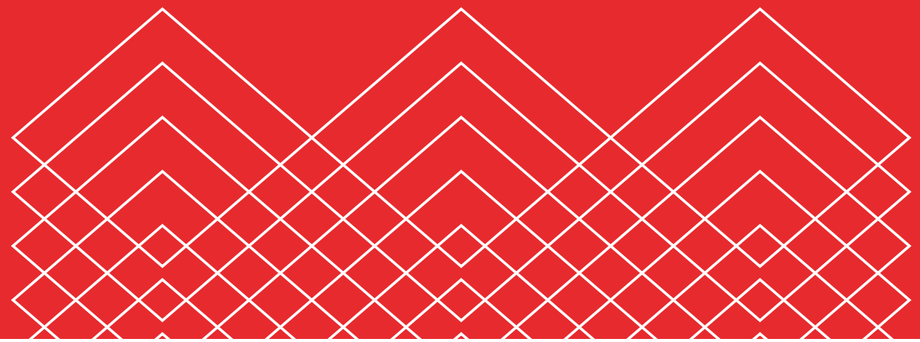
- 策略与机制分离
- CPU虚拟化: Hypercall
- IO虚拟化:**半虚拟化IO**，前后端的方式
- 内存虚拟化:直接页表




Relative score to Linux
1.1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
567
567
554
550
263
271
334
535
172
158
80
65
1714
1533
199
306
418
400
310
111
518
514
150
172
L X V U
SPEC INT2000 (score)
Linux build time (s)
OSDB-IR (tup/s)
OSDB-OLTP (tup/s)
dbench (score)
SPEC WEB99 (score)

Design Optimization of Xen

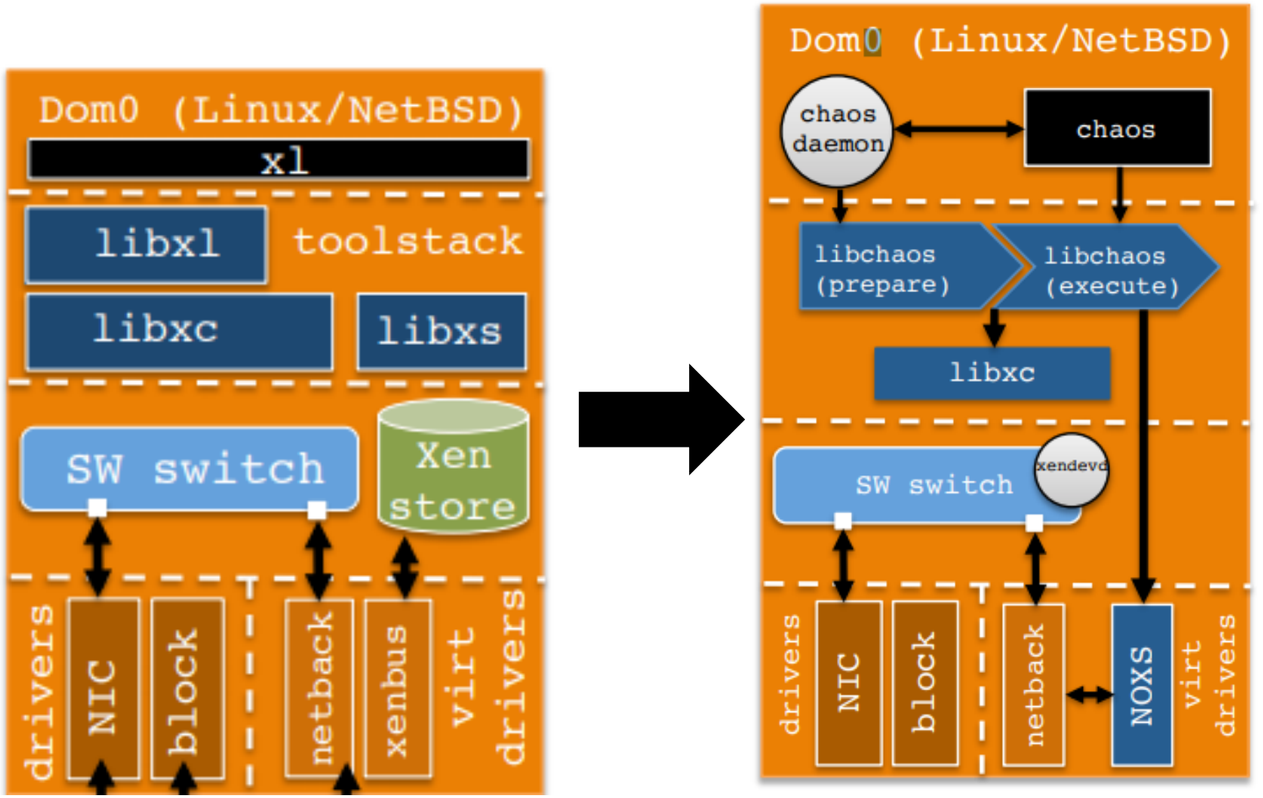
(My VM is Lighter (and Safer) than your Container

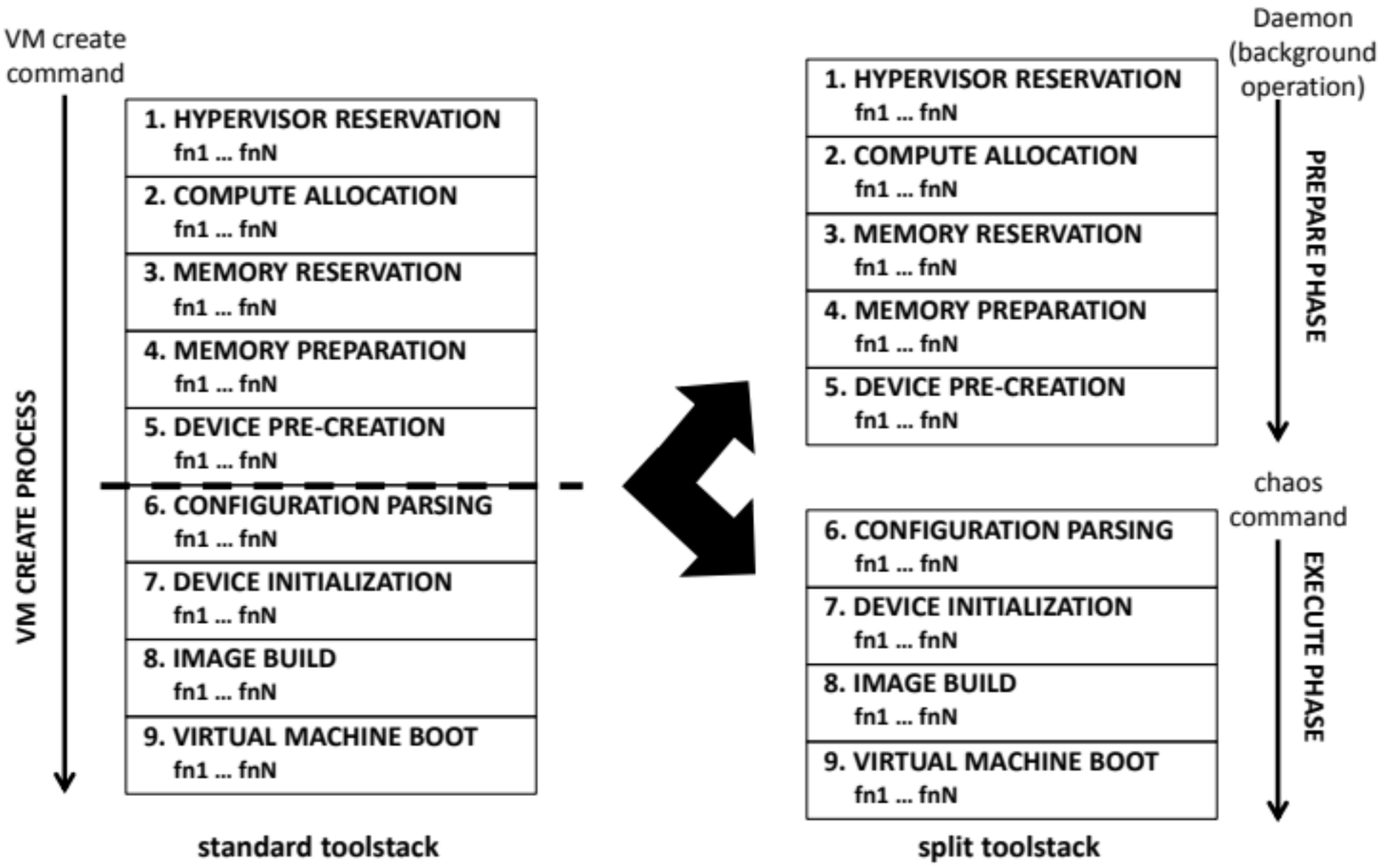


针对Xen性能问题的优化

证明Xen在优化到一定程度的情况下速度可以达到甚至超过Container

- Noxs解决VM creation & boottime 的 scalability
- Split Toolstack减少VM的启动时间
- TinyX裁剪内核减少VM的启动时间和整体体积




VM create command
VM CREATE PROCESS
1. HYPERVISOR RESERVATION
fn1 … fnN
2. COMPUTE ALLOCATION
fn1 … fnN
3. MEMORY RESERVATION
fn1 … fnN
4. MEMORY PREPARATION
fn1 … fnN
5. DEVICE PRE-CREATION
fn1 … fnN
6. CONFIGURATION PARSING
fn1 … fnN
7. DEVICE INITIALIZATION
fn1 … fnN
8. IMAGE BUILD
fn1 … fnN
9. VIRTUAL MACHINE BOOT
fn1 … fnN
standard toolstack
Daemon (background operation)
PREPARE PHASE
1. HYPERVISOR RESERVATION
fn1 … fnN
2. COMPUTE ALLOCATION
fn1 … fnN
3. MEMORY RESERVATION
fn1 … fnN
4. MEMORY PREPARATION
fn1 … fnN
5. DEVICE PRE-CREATION
fn1 … fnN
chaos command
EXECUTE PHASE
6. CONFIGURATION PARSING
fn1 … fnN
7. DEVICE INITIALIZATION
fn1 … fnN
8. IMAGE BUILD
fn1 … fnN
9. VIRTUAL MACHINE BOOT
fn1 … fnN
split toolstack