Introdução a virtualização

Professor: Pedro Horchulhack

Agenda

1. Arquiteturas de SO
2. Virtualização e máquinas virtuais
3. Atividade

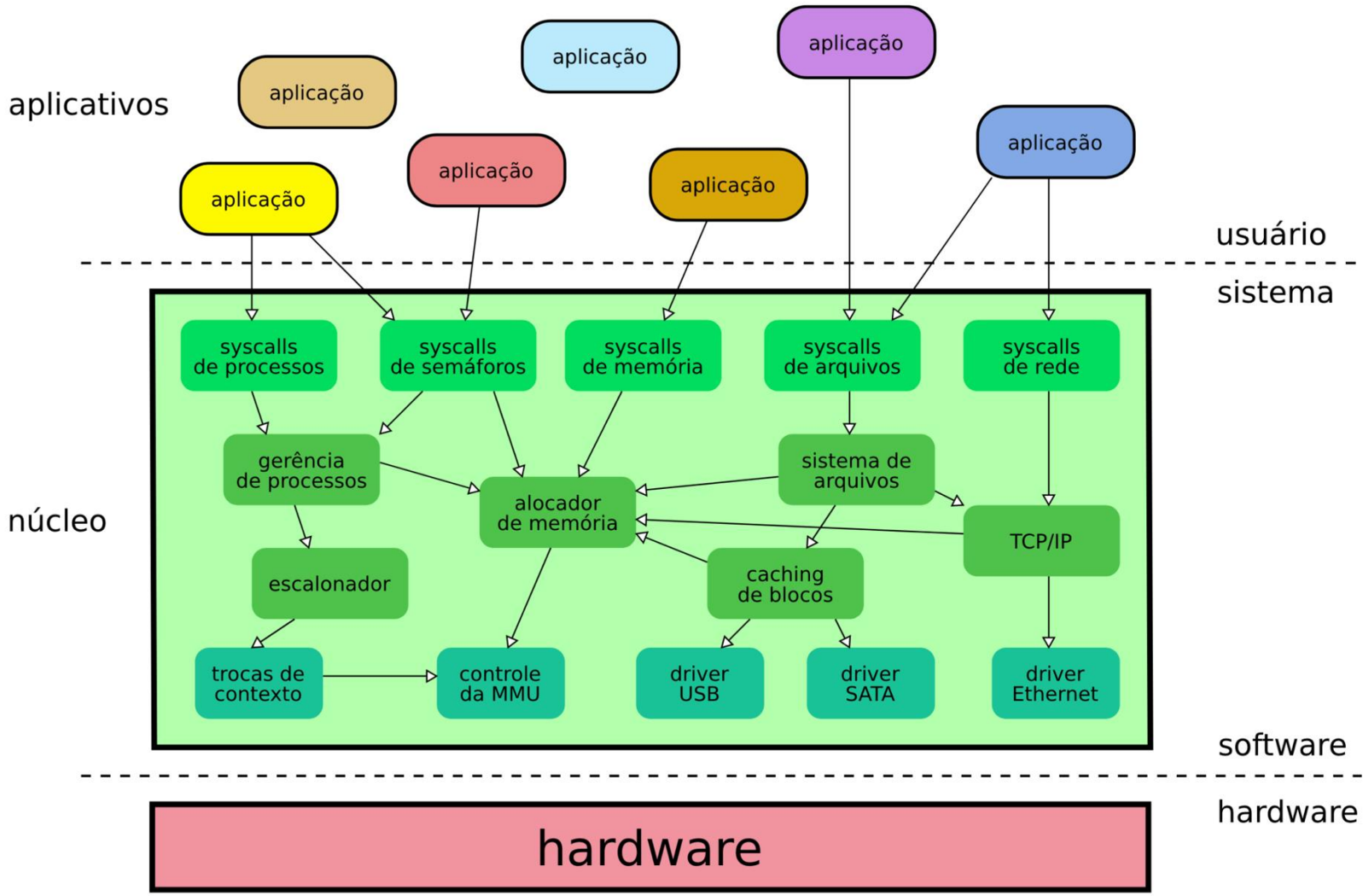
Arquitetura de SOs

Arquitetura de SOs

- Um SO pode ser organizado de diversas formas
  - Monolítico
  - Micronúcleo
  - Camadas
  - Híbridos

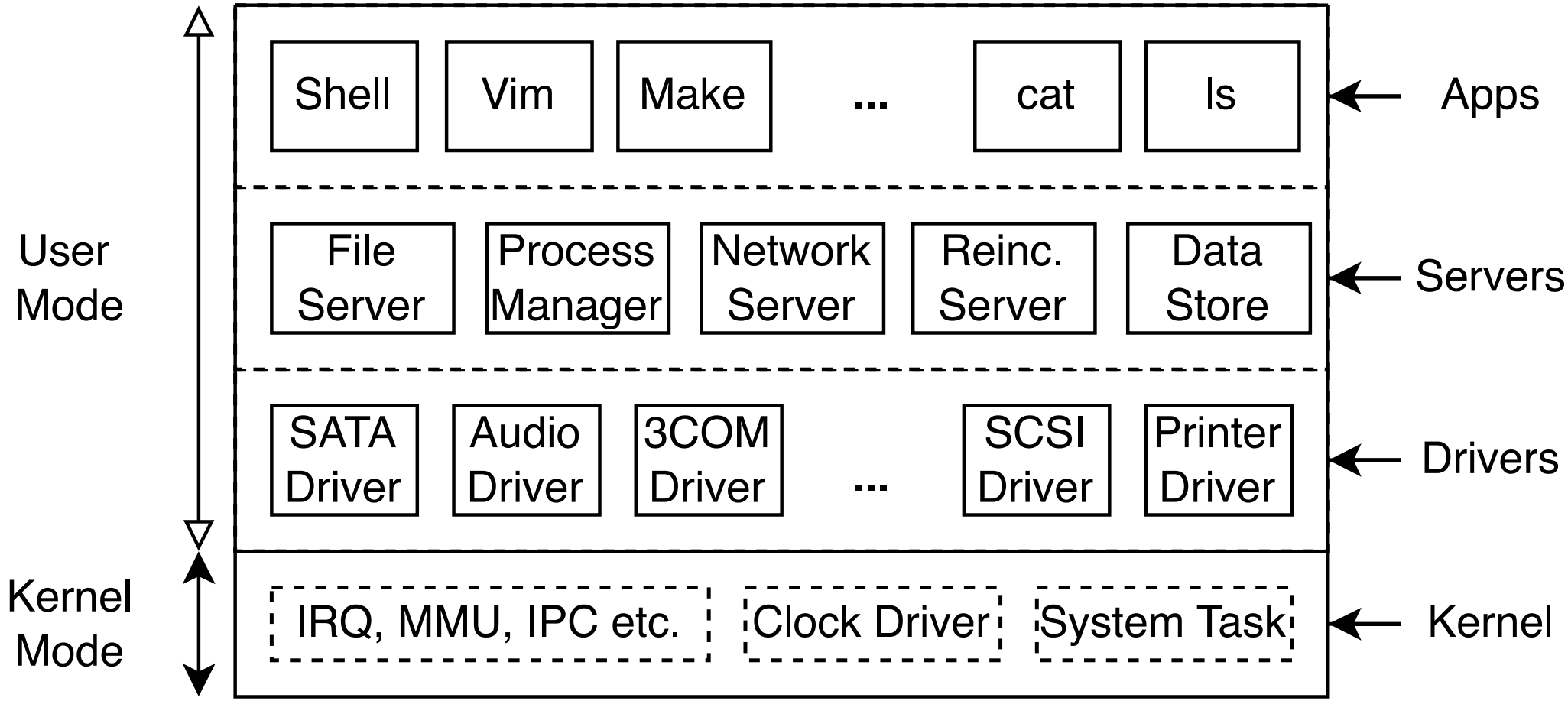
- Organizações ainda mais complexas:
  - Máquinas virtuais
  - Contêineres
  - Exonúcleo
  - Uninúcleo

Arquitetura de SOs



Núcleo de SO monolítico (MAZIERO, 2019)

Arquitetura de SOs



SO MINIX3 com micronúcleo (adaptado de HERDER *et al.*, 2006)

O que é virtualização?

- Uma estratégia de executar aplicações em ambientes diferentes dos quais foram originalmente projetados

- Os softwares acessam o hardware por meio do SO
  - Interfaces que abstraem e padronizam o acesso

- Interfaces típicas:
  - *Instruction Set Architecture* (ISA)
  - *Application Binary Interface* (ABI)
  - *Syscalls*
  - *Libcalls*

Todas as interfaces mudarão de acordo com o hardware sobre qual executarão!

Máquinas virtuais

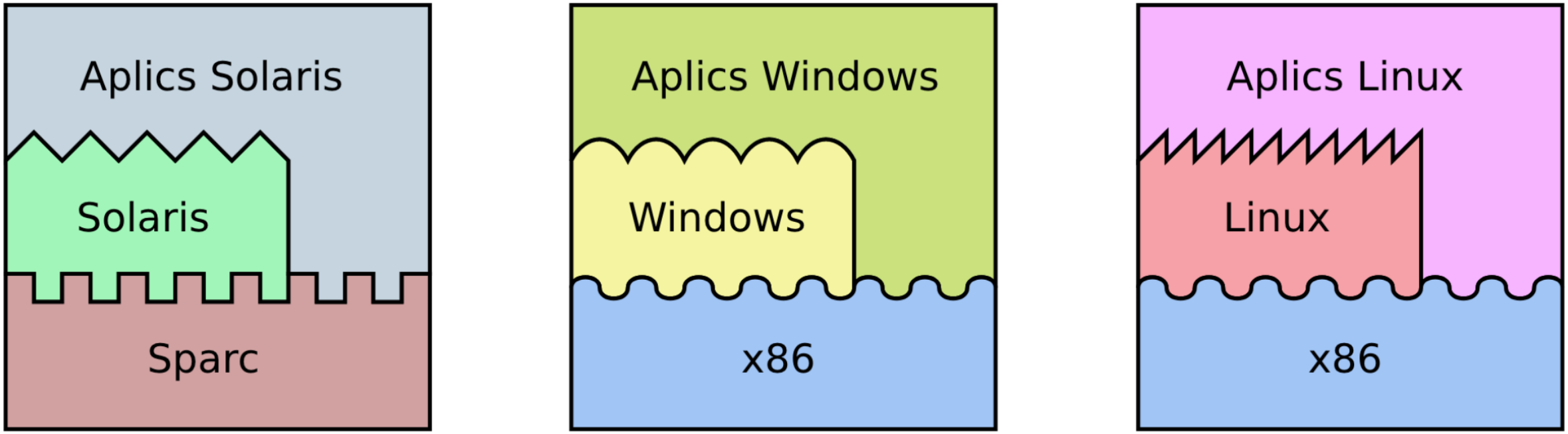
- Por que Máquinas Virtuais (VMs) são interessantes?
  - Softwares mais adaptativos
  - Flexibilidade
  - Elasticidade
  - Segurança

- Virtualização é a mesma coisa que emulação?
  - Depende…

  - Emulação de hardware
  - Emulação de SO
  - Otimização dinâmica
  - Replicação de hardware

Máquinas virtuais

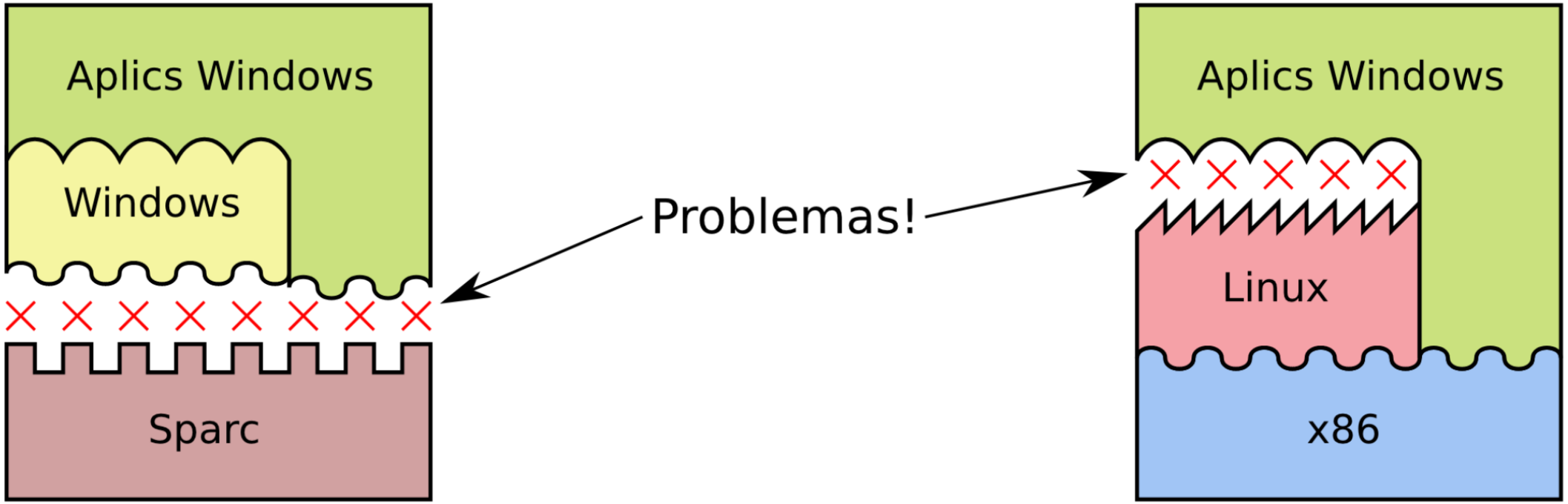
– Já entedemos que uma aplicação compilada no Windows não executará como esperado em um sistema baseado no GNU/Linux



Problema de compatibilidade entre interfaces (SMITH e NAIR, 2004, apud MAZIERO 2019)

Máquinas virtuais

– Já entedemos que uma aplicação compilada no Windows não executará como esperado em um sistema baseado no GNU/Linux



Problema de compatibilidade entre interfaces (SMITH e NAIR, 2004, apud MAZIERO 2019)

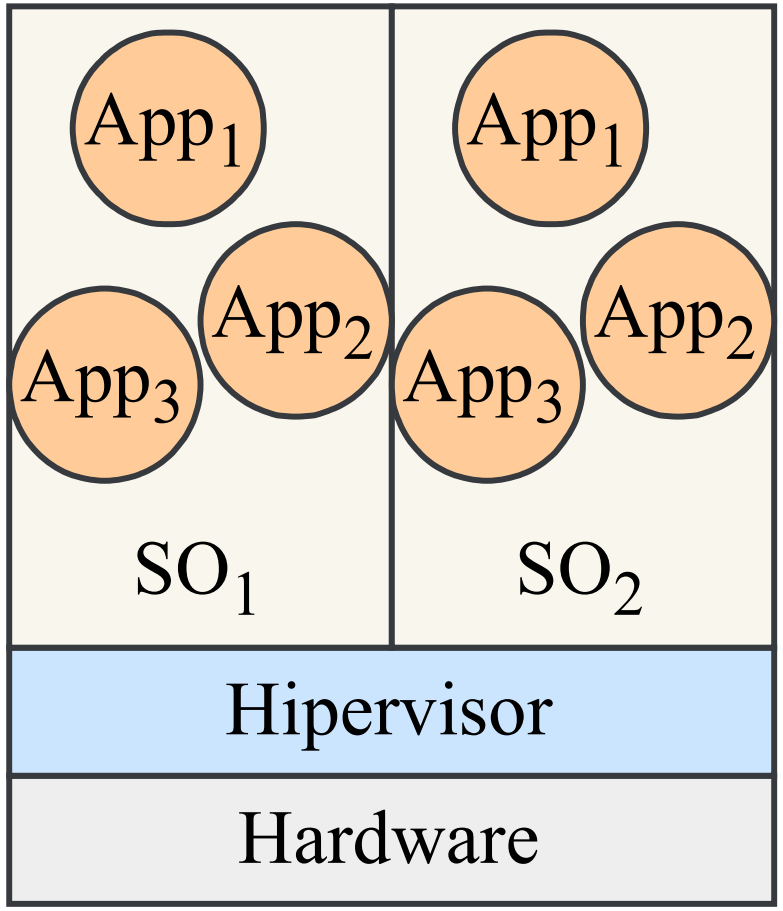
Máquinas virtuais

- Uma solução para a incompatibilidade pode ser uma *camada de virtualização* construida em software.
  - *Hypervisor* ou monitor de máquina virtual (VMM)

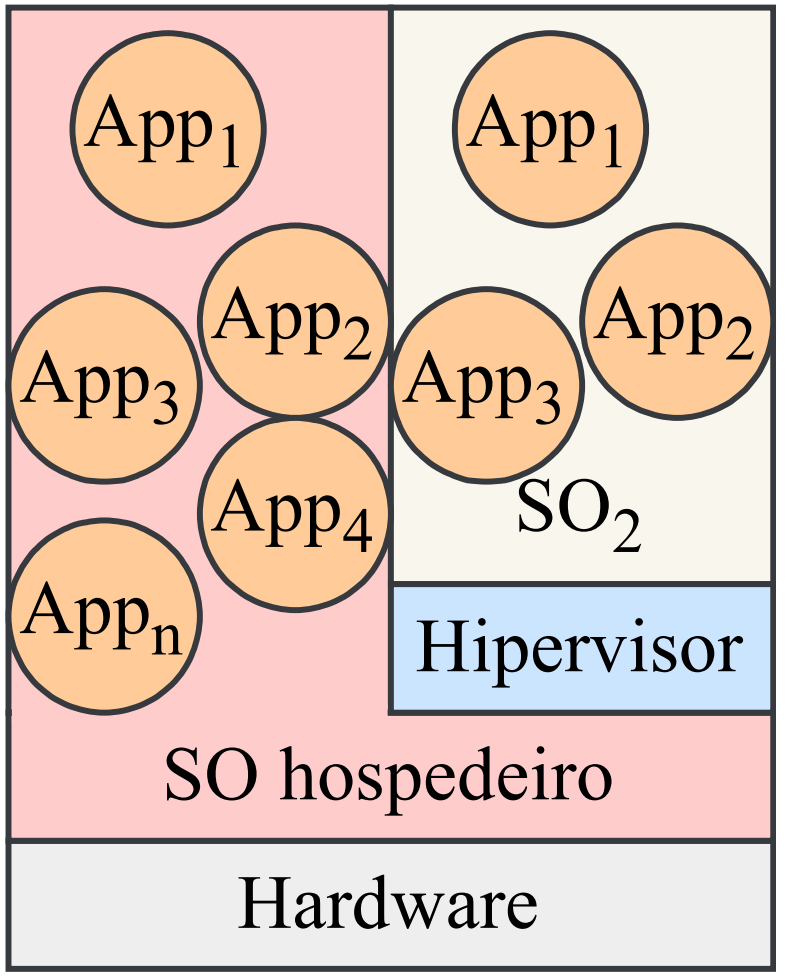
- Uma máquina virtual utiliza do monitor de máquina virtual para acessar o hardware

- Três componentes:
  - Host system
  - VMM
  - Guest system

Máquinas virtuais



Hipervisor nativo



Hipervisor convidado

Máquinas virtuais

- Alguns problemas que temos em máquinas virtuais:
  - Alto consumo de recursos
  - *Sobrealocação* de recursos

- Existe outra alternativa para lidar com esses problemas:
  - **Contêineres** ou **Máquinas Virtuais de Sistema Operacional**

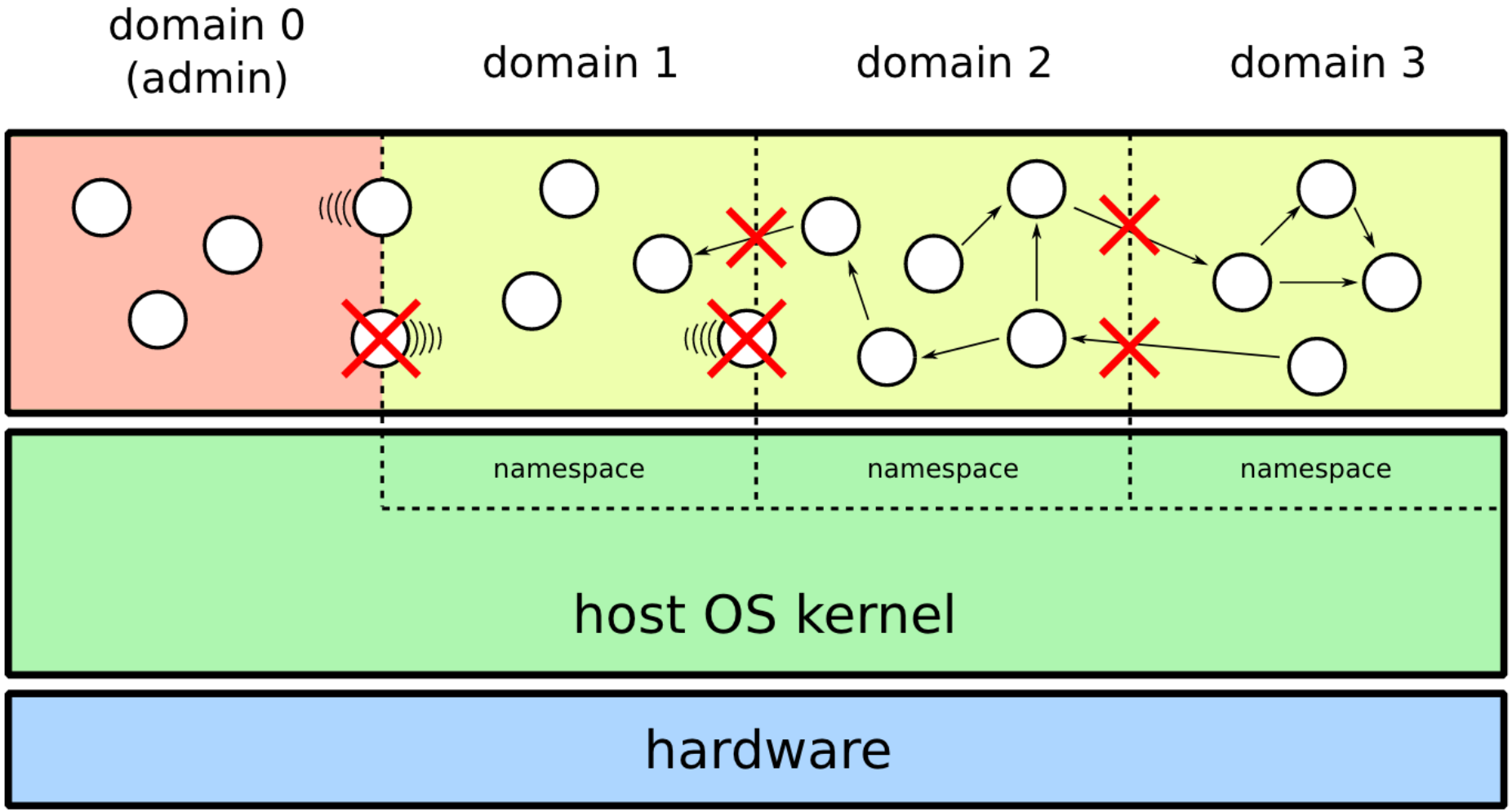
Contêineres

- É uma estratégia de virtualização mais leve do que máquinas virtuais

- Compartilham o núcleo do sistema operacional subjacente
  - Não é necessário a emulação de instruções
  - Virtualização do espaço do usuário

- É compartilhado uma fração dos recursos físicos para *domínios* ou *contêineres*

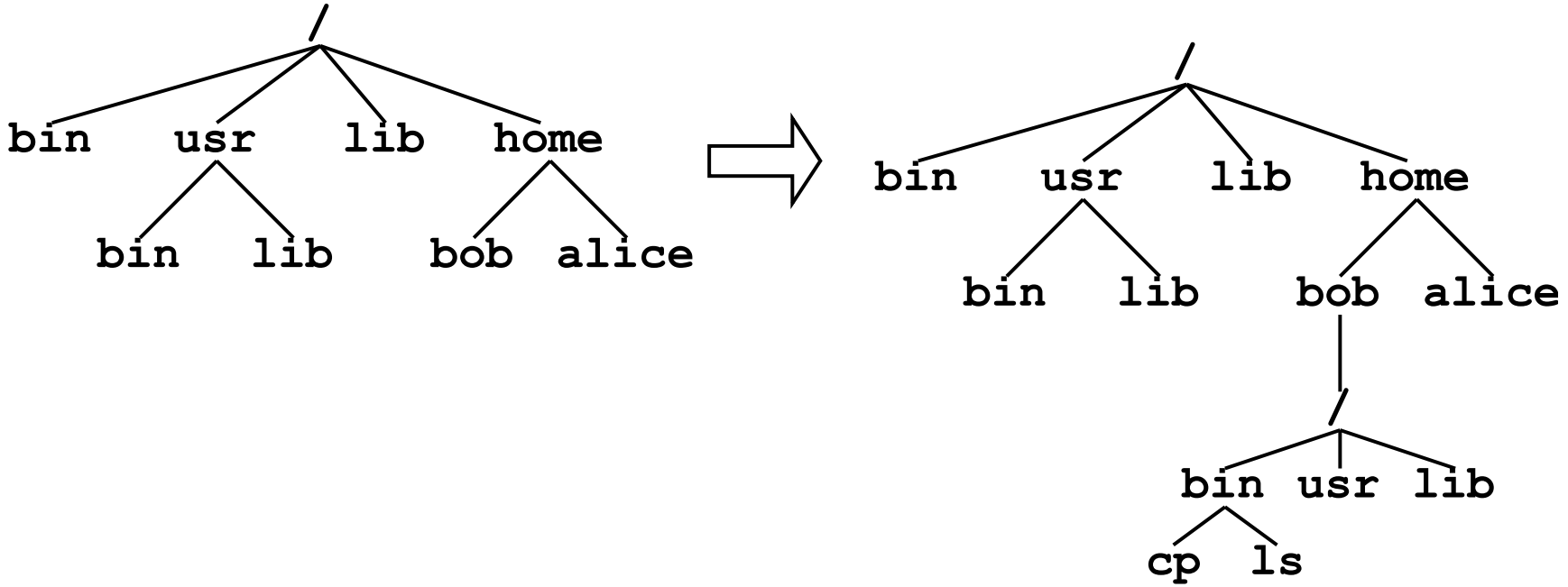
Contêineres



Contêineres (MAZIERO, 2019)

Contêineres

- É possível fazer esse tipo de virtualização através da chamada de sistema **`chroot()`**
  - Atua diretamente sobre o acesso do sistema de arquivos
  - Filhos do processo irão herdar a mesma visão do sistema de arquivos



Contêineres

- Existem algumas tecnologias que implementam **frameworks** que permitem trabalharmos com contêineres
  - Docker
  - Linux Containers (LXC)
  - runc

Atividade

Vamos simular um contêiner usando o **`chroot`**.

1. Crie um diretório na pasta `/home` chamado `container`
2. Entre no diretório criado
3. Crie outro diretório chamado `bin`
4. Instale a aplicação `tree` (`sudo apt update && sudo apt install tree -y`)
5. Copie recursivamente os diretórios `/usr/lib` para a pasta corrente
6. Copie o arquivo `/usr/bin/cat` para `./bin`
7. Copie o arquivo `/usr/bin/clear` para `./bin`
8. Copie o arquivo `/usr/bin/tree` para `./bin`
9. Copie o arquivo `/usr/bin/bash` para `./bin`
10. Entre no chroot do diretório corrente (`sudo chroot /home/ubuntu/container bin/bash`)

Referências

1. MAZIERO, Carlos A. **Sistemas operacionais: conceitos e mecanismos**. [s.l.]: Ufpr, 2019.
2. TANENBAUM, Andrew S.; BOS, Herbert. **Modern operating systems**. 4. ed. Boston: Prentice Hall, 2015.
3. LINUX**. Introduction – The Linux Kernel Documentation.** c2024**.** Disponível em: https://linux-kernel-labs.github.io/refs/heads/master/lectures/intro.html. Acesso em: 02 de abril de 2024.
4. HERDER, Jorrit N.; BOS, Herbert; GRAS, Ben; et al. **MINIX 3: a highly reliable, self-repairing operating system**. ACM SIGOPS Operating Systems Review, v. 40, n. 3, p. 80–89, 2006.

Contato: p.horchulhack@pucpr.br