



SFD 2010

Carla Souza  
Otacílio Lacerda

# Virtualização

*"Em computação, é uma forma de esconder as características físicas de uma plataforma computacional dos utilizadores, mostrando outro hardware virtual, emulando um ou mais ambientes isolados."*

[Wikipedia]

# Virtualização

## Usos:

- Isolamento de Serviços
- WebHosting
- Teste de Software
- ...

# Virtualização

## Vantagens:

- Gerenciamento centralizado
- Instalações simplificadas
- Facilidade para a execução de backups
- Suporte e manutenção simplificados
- Independência de Hardware
- Migração para novo hardware de forma transparente
- Economia de espaço físico

# Virtualização

## Inconvenientes:

- Grande consumo da capacidade em disco - é necessário espaço para que cada máquina virtual tenha o seu próprio sistema operativo e as aplicações instaladas
- Dificuldade no acesso direto a hardware, como por exemplo placas gráficas ou dispositivos USB
- Grande consumo de memória RAM dado que cada máquina virtual vai ocupar uma área separada da mesma
- Overhead: Simulação de toda uma máquina além de uma camada de software extra entre o host e a máquina virtual

# Vamos sonhar

## E se...

...tivessemos como ter as melhores vantagens da virtualização sem as desvantagens citadas?

# Idéias

Alguém tem alguma?

# Solução: Contexto

- Apenas processos com o mesmo contexto se enxergam.
- Abstração que permite criar “servidores virtuais” sem utilizar virtualização ou para-virtualização.
- Qualquer e somente distribuições GNU/Linux podem ser usadas como servidor virtual.

Mas como isso é possível?  
Mágica?

# Desvendando a Mágica

- No linux qualquer coisa é um processo ou arquivo.
- Arquivos e processos estão organizados em árvores.
- O sistema é multi-tarefa e multi-usuário, assim permitindo que processos e arquivos sejam controlados por esquemas de permissões e capacidades.

# Desvendando a Mágica

## Árvore de Processos:

```
init-+-acpid
| -6*[agetty]
| -atd
| -crond
| -events/0
| -events/1
| -httpd-+-error-log.sh
|           \-17*[httpd]
| -khelper
| -5*[kjournald]
| -klogd
| -ksoftirqd/0
| -kswapd0
| -kthread-+-aio/0
|           | -exec-osm/1
|           | -kcryptd/0
|           | -kmirrord
|           | -kseriod
|           \-etc
| -migration/0
| -migration/1
| -munin-node
| -sshd---sshd---sshd---bash---pstree
| -syslogd
| -watchdog/0
\ -watchdog/1
```

## Sistema de Arquivos:

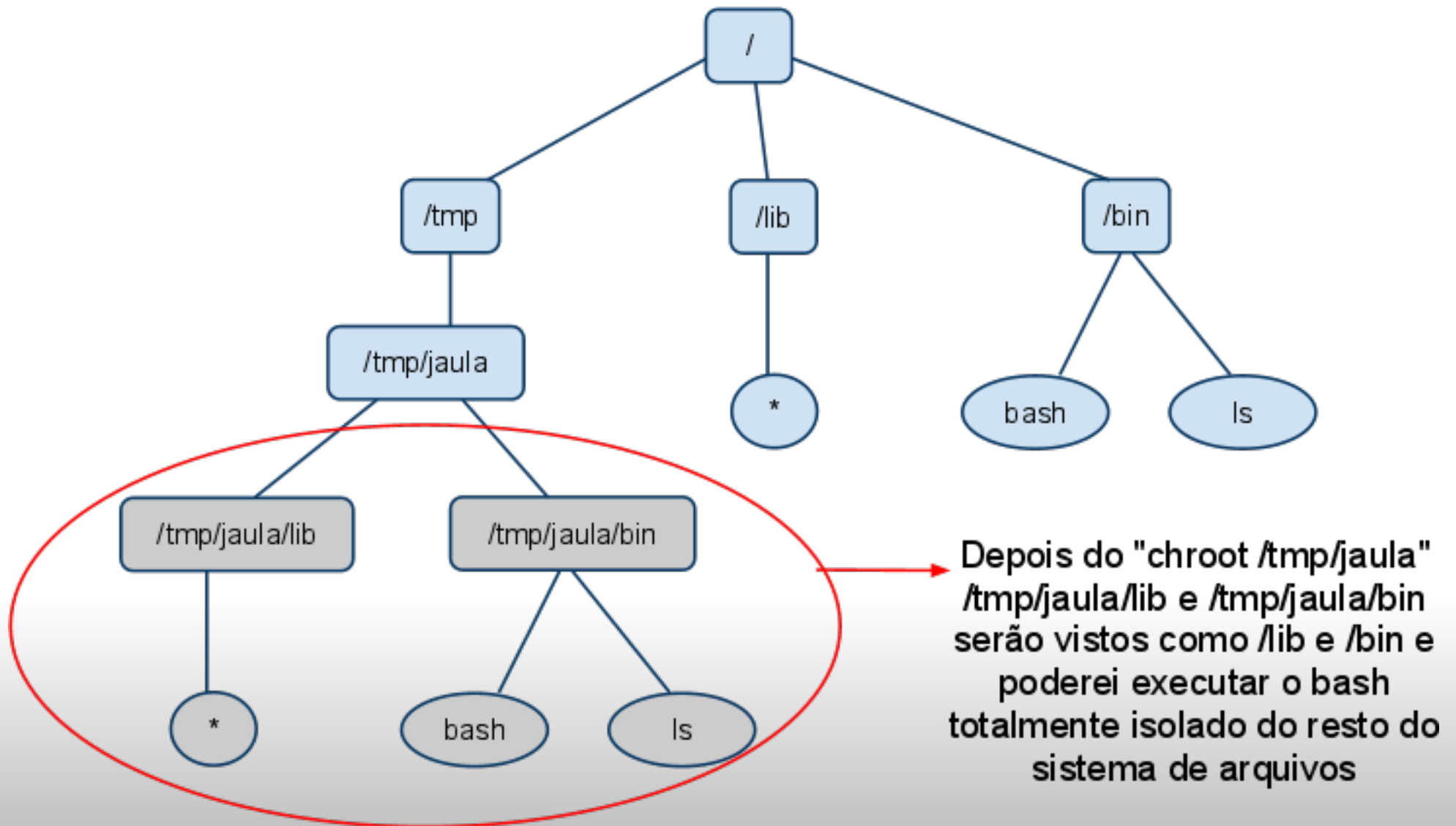
```
/
|-- bin
|   |-- arch
|   |-- ash
|   |-- awk -> gawk
|   |-- bash
|   \-- ...
|-- boot
|   |-- System.map
|   |-- config
|   |-- grub
|   \-- ...
|   \-- vmlinuz
|-- dev
|   |-- cdrom
|   |-- fd
|   \-- ...
...
|-- usr
|   |-- bin
|   |-- doc
|   \-- share
\..
```

# Desvendando a Mágica

- Vamos isolar uma porção do sistema onde processos serão executados sem que consigam escapar para outros locais
- Serviços inseguros poderão ser trancados em jaulas:

```
# mkdir -p /tmp/jaula/ {bin,lib}  
# cp -a /lib/* /tmp/jaula/lib/  
# cp -a /bin/{bash,ls} /tmp/jaula/bin/  
# chroot /tmp/jaula /bin/bash
```

# Desvendando a Mágica



# Resultado

Vantagens da virtualização  
+  
Sem overhead considerável

# Mas, isso é seguro?

É possível que um programa dentro de uma jaula escape para outros locais do sistema via `chroot()`.

Para os interessados:

<http://www.bpfh.net/simes/computing/chroot-break.html>



# O que é?

- Patch no kernel Linux.
- Aplicações no nível de usuário.
- Implementa isolamento de aplicações utilizando contextos.

# Linux VServer

- Introdução de “contextos” de processos via `chcontext()`.
- Isolamento de rede via `chbind()`.
- Atributos extendidos do sistema de arquivos para criar uma barreira à chamada `chroot()`.
- Implementação de capacidades POSIX dentro dos contextos.

# Contextos

- Campo adicional na estrutura de processos (pid, uid, gid, etc).
- Processos com o mesmo context id se enxergam.
- Contextos 0 e 1 são especiais.
- Cada vserver tem seu próprio esquema de usuários, arquivos e pacotes.

# Isolamento do Sistema de Arquivos

- A chamada `chroot()` não é reimplementada!
- O Linux-VServer usa atributos estendidos no sistema de arquivos para criar uma barreira sinalizadora que indica à `chroot()` que saídas de uma jaula estão bloqueadas.
- Os vservers “compartilham” o mesmo kernel

# Isolamento de Rede

- `chbind()`: isola um processo num IP.
- O processo apenas escutará conexões endereçadas a este IP.
- Mesmo serviços que tendem a escutar em todos IPs e interfaces da máquina podem ser isolados por essa chamada.

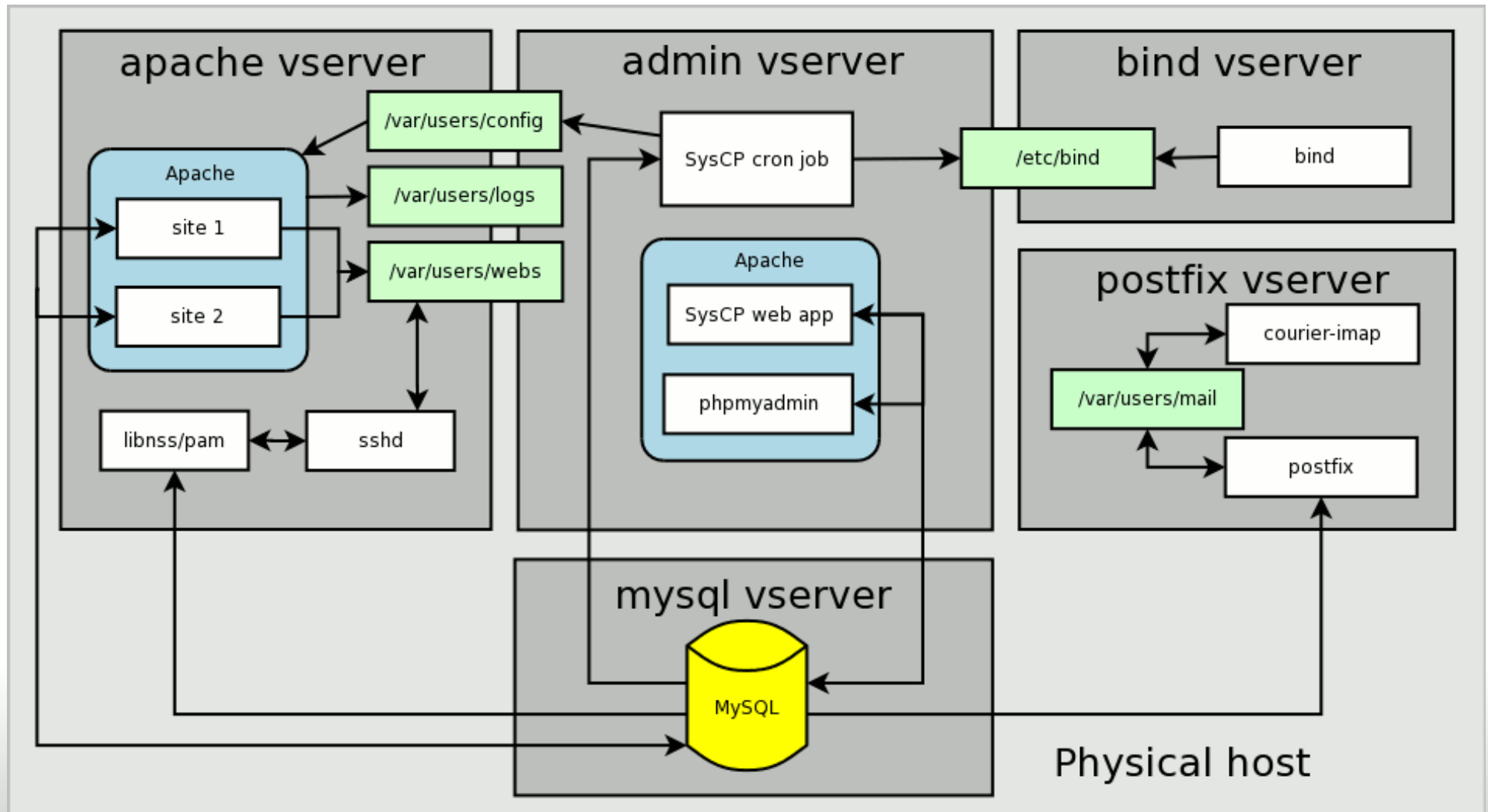
# "Virtualizando"

- Combinando as três chamadas (que na verdade são apenas uma) uma porção da máquina pode ser isolada como se fosse um sistema próprio.
- Isso pode ser feito com um conjunto de aplicativos: o pacote util-vserver juntamente com alguma distribuição GNU/Linux instalada numa pasta do sistema.

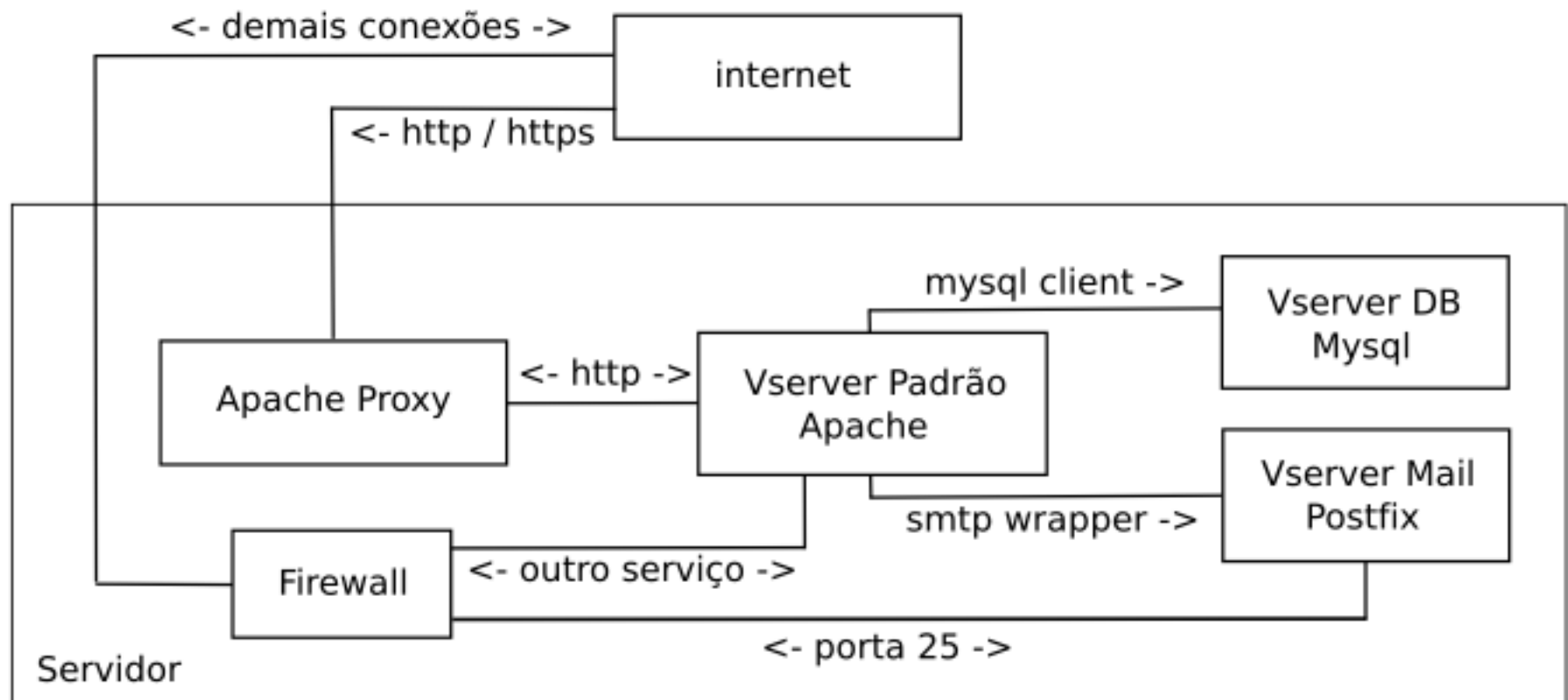
# Possíveis Aplicações

- Toolchains para construção de distros.
- Sandbox para administração de servidores.
- Construção de pacotes para não sujar o sistema principal.
- Servidores compartilhados por grupos e projetos diferentes.
- Replicabilidade de sistemas.
- Isolamento de serviços e aplicações possivelmente inseguras.

# Exemplo: Provedor de Sites



# Exemplo: Servidor LAMP



# Características adicionais

Possibilidade de controle de:

- Memória
- Disco
- CPU
- Rede

# Inconvenientes

- Requer que um patch para versão do kernel do host.
- Todas os servidores virtuais compartilham o mesmo kernel então todos precisam ser Linux.
- Rede é baseada em isolamento e não virtualização, ou seja, isso não permite configurar roteamento interno ou firewall.

# Referências

## Virtualização:

- <http://pt.wikipedia.org/wiki/Virtualiza%C3%A7%C3%A3o>

## Vserver:

- [http://linux-vserver.org/Welcome\\_to\\_Linux-VServer.org](http://linux-vserver.org/Welcome_to_Linux-VServer.org)
- [http://linux-vserver.org/Welcome\\_to\\_Linux-VServer.org](http://linux-vserver.org/Welcome_to_Linux-VServer.org)

Principal: <http://slack.sarava.org/vservers>

Dúvidas?  
Angústias?  
Desilusões?