Linux-VServers e Segurança Por Contexto

http://slack.sarava.org/node/14

Índice

Introdução

O que é?

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

Que tipo de isolamento?

- Apenas processos com o mesmo “contexto” se enxergam.
- Abstração que permite criar “servidores virtuais” sem utilizar virtualização ou para-virtualização.
- Qualquer distribuição de GNU/Linux pode ser usada como servidor virtual.
- Cada vserver tem seu próprio esquema de usuários, arquivos e pacotes.

Exemplo

```
# ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0    680    72 ?        S    Apr12   0:00 init [3]
root         2  0.0  0.0      0     0 ?        S    Apr12   0:00 [migration/0]
root         3  0.0  0.0      0     0 ?        SN   Apr12   0:00 [ksoftirqd/0]
[snip]
root        85  0.0  0.0   1520   372 ?        Ss   Apr12   0:00 /usr/sbin/syslogd
root        88  0.0  0.0   1480   248 ?        Ss   Apr12   0:00 /usr/sbin/klogd -c 3 -x
root       158  0.0  0.0   3400   568 ?        Ss   Apr12   0:01 /usr/sbin/sshd
root       167  0.0  0.0   1680   376 ?        S    Apr12   0:00 /usr/sbin/crond -l10
[snip]
# ls /vservers/jaula
bin  boot  dev  etc  home  lib  mnt  opt  proc  root  sbin  sys  tmp  usr  var
# vserver jaula enter
# ps aux
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root      1271  0.0  0.0   1664   628 ?        Ss   Apr12   0:00 /usr/sbin/syslogd
root      1312  0.0  0.0   1716   588 ?        S    Apr12   0:00 /usr/sbin/crond -l10
root     25851  5.4  0.1   2388  1396 pts/3    S    13:35   0:00 /bin/bash -login
root     25904  1.0  0.0   2512   924 pts/3    R+   13:35   0:00 ps aux
[snip]
```

Vantagens

- Aumento da segurança.
- Mais organização.
- Praticidade: fácil reinstalação de um vserver.
- Sem overhead considerável: os vservers “compartilham” o mesmo kernel.

Sistemas POSIX:
Visão do Usuário

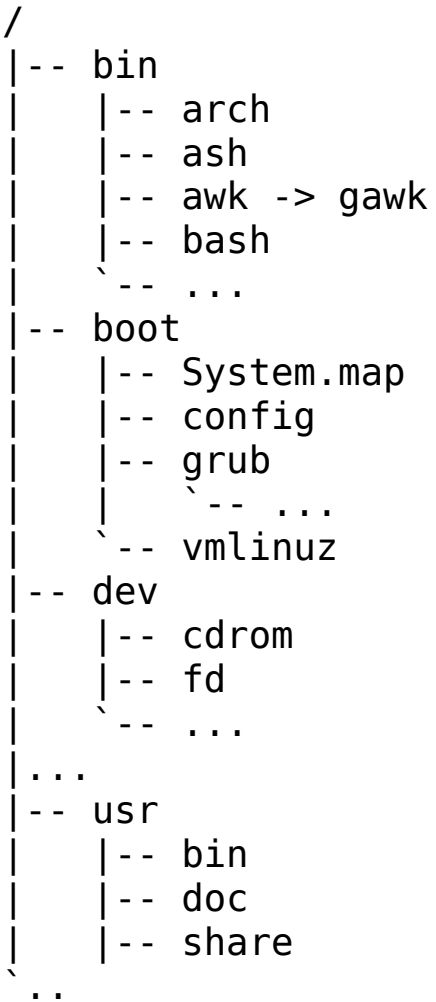
Sistemas POSIX: visão do usuário

- Qualquer coisa é um processo ou arquivo.
- Arquivos e processos estão organizados em árvores.
- O sistema é multi-tarefa e multi-usuário.

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



Segurança do sistema

- Multi-tarefa e multi-usuário: processos e arquivos controlados por esquemas de permissões e capacidades.
- UID 0: root, por padrão dententor de todas as capacidades.
- Pedidos de recurso ao sistema são feitos via chamadas especiais e são confrontadas com as capacidades do processo.

Jaulas

- Porção do sistema onde processos são executados sem que consigam escapar para outros locais.
- Serviços inseguros podem ser isolados em suas próprias jaulas.
- Syscall `chroot()`.

Exemplo de jaula

- Criando uma jaula simples:

```
# 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
```

- Traçando dependências:

```
$ ldd /usr/bin/bzip2
        libbz2.so.1 => /lib/libbz2.so.1 (0x4002e000)
        libc.so.6 => /lib/libc.so.6 (0x4003d000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x40000000)
```

Isso é seguro?

- É possível que um programa dentro de uma jaula escape para outros locais do sistema via `chroot()`.
- `http://www.bpfh.net/simes/computing/chroot-break.html`

Vservers

Vservers

- 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.

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.

Isolamento no sist. de arquivos

- A chamada `chroot()` não é re-implementada.
- O Linux-VServer usa atributos extendidos no sistema de arquivos para criar uma barreira sinalizadora que indica à `chroot()` que saídas de uma jaula estão bloqueadas.

“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.

Características adicionais

- Limitação de consumo de memória, disco, CPU e rede nos vservers.
- Controle de capacidades POSIX nos vservers.
- Proteção de entradas no /proc para que não apareçam dentro dos vservers.
- Namespaces.
- Unificação de instâncias.

Instalação

Instalação

- Aplicar o patch no kernel, compilá-lo e instalá-lo.
- Reiniciar a máquina.
- Instalar o pacote util-vserver.
- Criar a barreira dos vservers.
- Criar e executar os vservers.
- Scripts de inicialização.

Criando vservers

- Copiando o “sistema principal”.
- Imagens prontas.
- Via *util-vserver*.
- debootstrap ou sistema de pacotes da distribuição utilizada.

Exemplo

- Vserver em debian:

```
# vserver jaula build -m debootstrap --hostname jaula \
          --interface eth0:192.168.0.1/24 -- -d sarge
```

- Jaula criada em /vservers/jaula
- Configurações em /etc/vservers/jaula
- Após a configuração, a jaula pode ser iniciada com o comando “vserver jaula start” e um processo init (ou uma emulação dele) passa a rodar.

Detalhes

Contextos

- Campo adicional para os processos.
- Contexto 0: principal, onde rodam as aplicações do sistema (“servidor principal” ou “sistema hospedeiro”).
- Contexto 1: contexto especial que pode enxergar outros processos.

```
chcontext ps aux
chcontext --ctx context-id ps aux
chcontext --ctx 1 ps aux
vps aux
```

Resistência da jaula

- É permitido a um processo do contexto principal, rodando como usuário root, criar um processo num novo contexto.
- Em geral os vservers estão configurados para permitir o “recebimento” de processos originados no contexto principal.
- Processos originados dentro da jaula não conseguem escapar para fora, para outro contexto ou ouvir um outro IP; é possível ainda proibir que chroot seja chamada de dentro dela.

Barreira do chroot

- Em princípio, chroot pode se invocada mesmo dentro de um vserver.
- Para evitar que a raíz de um processo seja mudada para um nível acima da raíz do vserver, são utilizados sinalizadores no sistema de arquivos.
- Esses sinalizadores são atributos extendidos de sistemas de arquivos como ext3 e reiserfs.

Atributos extendidos

- Atributos são registros adicionais nos sistemas de arquivos usados para “etiquetar” os arquivos.
- Os bits de permissão de arquivo são atributos do sistema de arquivos.
- Sistemas como o ext3 e o reiserfs ainda possuem registros adicionais que podem ser usados de formas diversas.

Atributos da barreira

- Série 2.x:

```
setattr --barrier /vservers
```

- Série 1.x:

```
chmod 000 /vservers
chattr +t /vservers
```

Isolamento de rede

- Aplicativos rodando com privilégios de superusuário podem ouvir conexões de rede em qualquer porta.
- Exemplo de alias de rede para um vserver:

```
eth0      Link encap:Ethernet  HWaddr 00:00:00:00:00:00
          inet addr:IP-DO-SERVIDOR
          Bcast:BCAST-DO-SERVIDOR  Mask:MASCARA-DO-SERVIDOR

eth0:jaul Link encap:Ethernet  HWaddr 00:00:00:00:00:00
          inet addr:10.0.1.1
          Bcast:BCAST-DO-SERVIDOR  Mask:MASCARA-DO-SERVIDOR

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
```

- O chbind trava a escuta do vserver jaula às conexões destinadas a 10.0.0.1.

Capacidades POSIX

- Conjunto de potencialidades disponíveis a um processo, como alterar a data do sistema, mudar permissão de arquivos, executar chroot e outras chamadas do sistema.
- Três conjuntos de capacidades: as *herdadas*, as *permitidas* e as *efetivas*.
- Capacidades estão associadas apenas a operações privilegiadas realizadas pelo sistema operacional.

Capacidades POSIX

- Permitidas: conjunto das capacidades que um processo dispõe.
- Efetivas: capacidades que um processo realmente pode utilizar.
- Herdadas: é o conjunto das capacidades que um processo passará para seus processos filhos.
- Efetivas <= Permitidas e Herdadas <= Permitidas

Capacidades POSIX

- Quando um processo tenta executar uma chamada do sistema, o kernel checa se o processo tem a capacidade correspondente no conjunto das *efetivas*.
- Um processo pode adicionar ou remover no conjunto das efetivas e no das herdadas qualquer capacidade que estiver listada no conjunto das *permitidas*.

Capacidades POSIX

- Lista de capacidades disponível no arquivo `/usr/include/linux/capability.h`
- CAP_CHOWN: mudança de usuário e grupo nos arquivos.
- CAP_KILL: ignora que o processo que está sendo morto precisa ter o mesmo uid do processo que está tentando matá-lo.
- CAP_SETGID: permite o uso de setgid.
- CAP_SETUID: permite o uso de setuid.
- CAP_NET_BIND_SERVICE: utilização de portas menores que 1024.
- CAP_SYS_CHROOT: permite chroot().

Capacidades em Vservers

- Processos em vservers tem um conjunto de capacidades menor do que processos do servidor principal.
- Em vservers, nem processos rodando como superusuário poderão executar certas operações.
- O Linux-VServer ainda implementa algumas capacidades específicas para vservers.
- É possível configurar as capacidades disponíveis para cada vserver.

Capacidades em Vservers

- SET_UTSNAME: mudanças na estrutura de nome do sistema.
- SET_RLIMIT: mudar limites de uso dos recursos do sistema.
- RAW_ICMP: uso de sockets ICMP (ping, por exemplo).
- SYSLOG: registrar mensagens do sistema via syslog.
- QUOTA_CTL: permite estabelecimento de quotas dentro do vserver.

Cap. Desabilitadas em Vservers

- SETPCAP: transf. de cap. do grupo das permitidas para os outros grupos.
- LINUX_IMMUTABLE: mod. de imutabilidade de um arquivo.
- NET_BROADCAST: envio de pacotes de rede em broadcast.
- NET_ADMIN: permite a configuração da interface de rede.
- NET_RAW: uso de pacotes no estado bruto.
- SYS_ADMIN: permite a execução de várias tarefas administrativas.
- SYS_NICE: modificar prioridades de execução de processos.
- SYS_RESOURCE: modificar limites e quotas, dentre outras coisas.
- SYS_TIME: permite modificar a data do sistema
- MKNOD: permite a criação de arquivos de dispositivo.

Flags

- Controlam se processos de outros contextos podem criar processos no contexto de cada vserver;
- Controlam se o escalonador do sistema operacional dará a mesma prioridade a todos os processos de um vserver.
- Aplicam limites ao vserver (memória, processamento, disco, etc).

Limites

- Quotas de disco por contexto.
- Número máximo de processos.
- Limite de memória.
- Limites do escalonador.
- Limite de tráfego de rede (via QoS).

Segurança do /proc

- /proc: sistema de arquivos que age como interface entre o espaço do usuário e as estruturas de dados do kernel.
- Contém informações de processos, da rede e estado geral do sistema.
- O /proc dentro de um vserver é restrito e pode ser restringido ainda mais através de parâmetros de configuração.
- Restrições ao /proc também podem ser aplicadas enquanto um vserver está rodando.

Segurança do /proc

```
$ showattr /proc/cpuinfo
Awh-ui- /proc/cpuinfo
||||||
123456
||||| \_ sinalizador de imutabilidade
|||| \__ sinalizador de unificação
||| \___ sinalizador de barreira (apenas para pastas)
|| \____ ativação de visibilidade
| \_____ visibilidade no contexto 1 (supervisor)
 \______ visibilidade no contexto 0 (admin)
```

admin	supervisor	ativação	estado do arquivo
qualquer	qualquer	h	visível em todos os contextos
A	w	H	visível apenas no contexto 0
a	W	H	visível apenas no contexto 1
A	W	H	visível apenas nos ctxs 0 e 1
a	w	H	invisível em todos os ctxs

Segurança do /proc

```
# setattr --hide /proc/cpuinfo
# showattr /proc/cpuinfo
AwH-ui- /proc/cpuinfo
# setattr --~hide /proc/cpuinfo
# setattr --hide --~admin --watch /proc/interrupts
```

Namespaces e unificação

- Namespaces: diferentes vservers com visões diferentes do sistema de arquivos (i.e, partições montadas dentro dos vservers).
- Unificação: uso do atributo extendido de imutabilidade juntamente com *hardlinks* para economia de espaço em disco.
- Ambos esquemas ainda são pouco usados.

Comandos diversos

- vserver: serve para criar, iniciar, parar e reiniciar vservers.
- vps: mostra informações de processos de todos os contextos.
- vserver-stat: estatísticas dos vservers.
- vtop: top em todos os contextos.
- vserver-info: informações gerais de vservers e do ambiente de configuração do sistema.

Vservers em sistemas de produção

Roteando conexões

- NAT

```
iptables -A POSTROUTING -t nat -s 192.168.0.0/24 -d 0/0 \
         -j SNAT --to IP-DO-SERVIDOR
```

- Conexões entrantes

```
iptables --protocol tcp -t nat -A PREROUTING -i eth0 \
         -s 0/0 -d IP-DO-SERVIDOR --dport PORTA      \
         -j DNAT --to-destination DESTINO:PORTA2
```

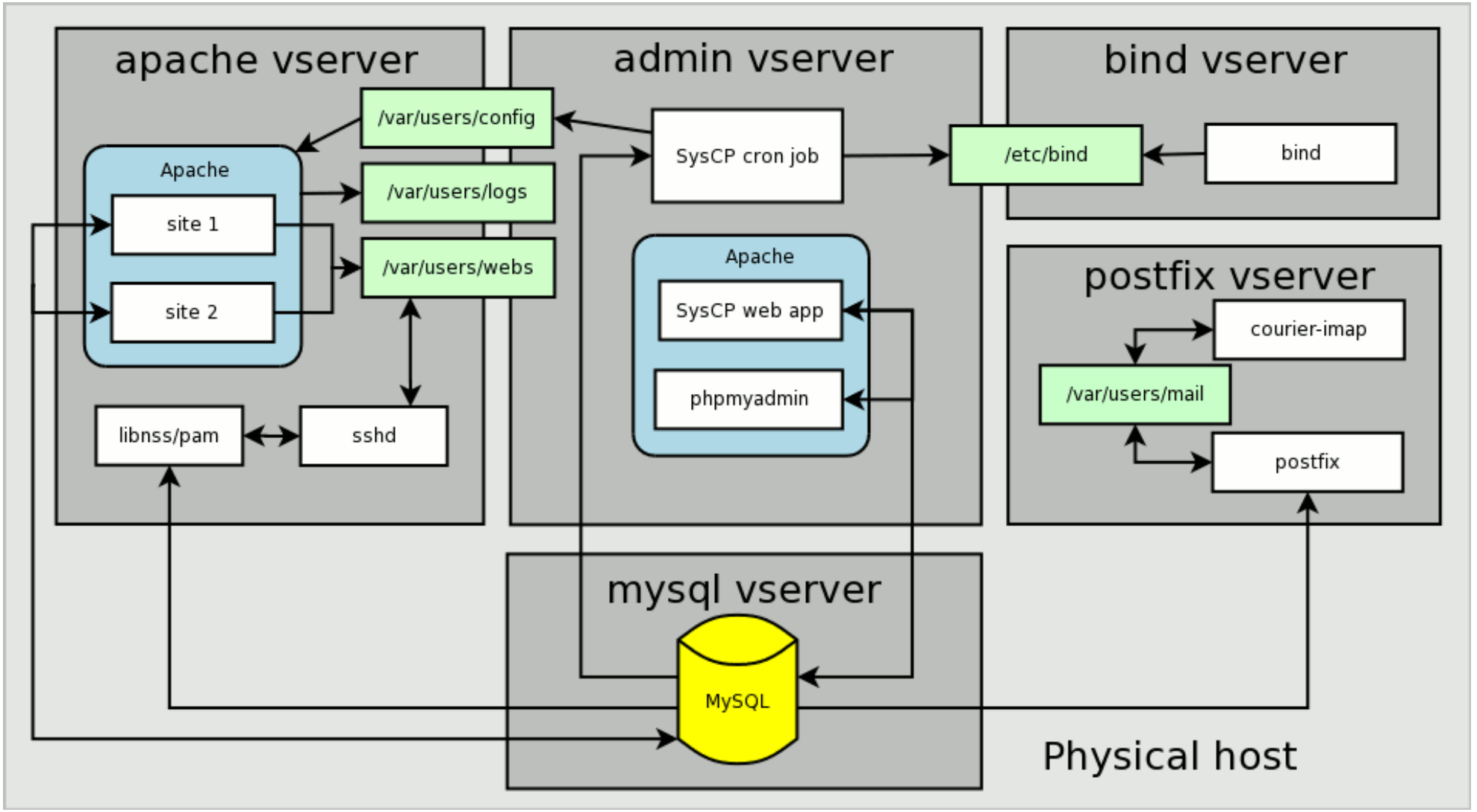
Compartilhando um IP

- Usando um proxy para o serviço.
- Utilizar portas fora do padrão.

Possíveis aplicações

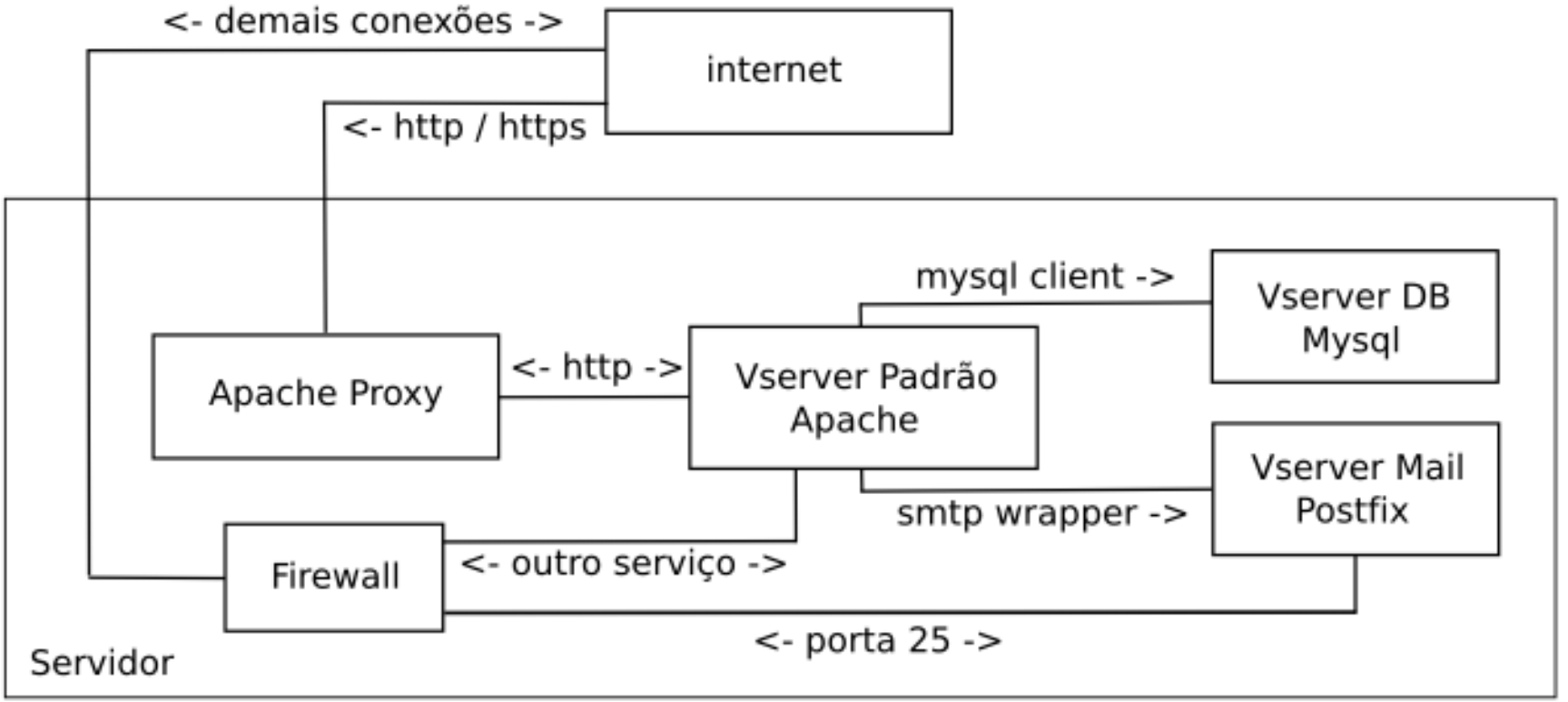
- Toolchains para construção de distros.
- Sandbox para administração de servidores.
- Construção de pacotes para não sujar o sistema principal.
- Servidores compartilhados por grupos e projetos diferentes.
- Replicabilidade de sistemas.
- Isolamento de serviços e aplicações possivelmente inseguras.

Exemplo: provedor de sites



- Fonte: `http://deb.riseup.net/web-server/syscp`

Exemplo: servidor LAMP



Entrada no kernel oficial

- Existe a discussão de oferecer esse tipo de infra-estrutura no kernel oficial.
- Discussões estão em torno de fundir o OpenVZ e o Linux-VServer.
- Vai demorar um tempão :)

Referências

- Página oficial: `http://linux-vserver.org`
- Página do projeto: `http://www.13thfloor.at/vserver`
- Lista: `http://list.linux-vserver.org`
- Estes slides: `http://slack.sarava.org/node/14`
- Grimório Debian: `http://deb.riseup.net/vserver`

Esquemas similares

- OpenVZ: `http://openvz.org`
- FreeVPS: `http://www.freevps.com`
- BSD Jail
- Solaris' Zones

(para)-Virtualizadores

- Qemu: `http://http://fabrice.bellard.free.fr/qemu`
- UML: `http://http://user-mode-linux.sourceforge.net`
- Xen: `http://www.cl.cam.ac.uk/Research/SRG/netos/xen`

?

- Dúvidas?
- Angústias?
- Desilusões?

Licença e Contato

- Email: rhatto em riseup.net
- Sítio: http://slack.sarava.org
- Chave pública: ID 0x6B566777 / keyserver.noreply.org
-