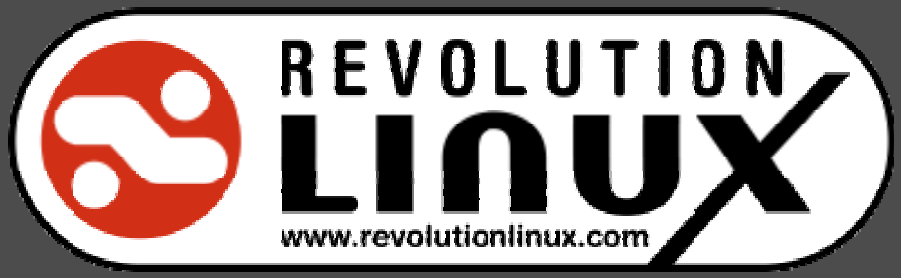
Virtualization of Linux based computers: the Linux-VServer project



Benoît des Ligneris, Ph. D.

Benoit.des.Ligneris@RevolutionLinux.com

Objectives:

Objectives:

1) ***Present the available programs that can provide a virtualization of Linux computers with different technologies.***

Objectives:

1) Present the available programs that can provide a virtualization of Linux computers with different technologies.

2) Focus on Linux-VServers: a very lightweight and effective technology for the regular Linux user not intersted in Kernel hacking.

REVOLUTION LINUX
www.revolutionlinux.com

Plan

Plan

- Introduction

Plan

- Introduction
- Overview of the available technology

Plan

- Introduction
- Overview of the available technology
- Classification of the problems: usage criteria

Plan

- Introduction
- Overview of the available technology
- Classification of the problems: usage criteria
- Comparative study of the existing technology

Plan

- Introduction
- Overview of the available technology
- Classification of the problems: usage criteria
- Comparative study of the existing technology
- Technology overview of Linux-VServers

Plan

- Introduction
- Overview of the available technology
- Classification of the problems: usage criteria
- Comparative study of the existing technology
- Technology overview of Linux-VServers
- Conclusion

Introduction

Introduction

- ***Why vservers?***

Introduction

- ***Why vservers?***
- ***Virtualization is now more and more acessible for regular users given the extreme processing power of the current computers***

Introduction

- ***Why vservers?***
- ***Virtualization is now more and more acessible for regular users given the extreme processing power of the current computers***
- ***The availability of COTS multi-processor 64 bit architecture accelerates the needs for a mature virtualization technique, as it's more and more difficult for a common application to use 100% of the available resources***

Introduction

- ***Why vservers?***
- ***Virtualization is now more and more acessible for regular users given the extreme processing power of the current computers***
- ***The availability of COTS multi-processor 64 bit architecture accelerates the needs for a mature virtualization technique, as it's more and more difficult for a common application to use 100% of the available resources***
- ***Virtualization also affect scientific computing and could become, in the near future, the corner stone of the so called «grid computing» as it solves elegantly most of the problems (security, resources consumption) of the current Grid technology***

Overview of the available technology

Overview of the available technology

- VMware

Overview of the available technology

- VMware
- plex86

Overview of the available technology

- VMware
- plex86
- Bochs

Overview of the available technology

- VMware
- plex86
- Bochs
- Linux-VServers

Overview of the available technology

- VMware
- plex86
- Bochs
- Linux-VServers
- User Mode Linux (UML)

Overview of the available technology

- VMware
- plex86
- Bochs
- Linux-VServers
- User Mode Linux (UML)
- Xen

Overview of the available technology

- VMware
- plex86
- Bochs
- Linux-VServers
- User Mode Linux (UML)
- Xen
- QEMU

VMware

VMware

« Vmware workstation is a powerfull virtual machine software for the desktop. VMware workstation runs multiple operating systems, including Microsoft Windows, Linux and Novell NetWare, simultaneously on a single PC in fully networked, portable virtual machines »

http://www.vmware.com/products/

VMware

- Provide complete multi-OS emulation on x86 CPU only

VMware

- Provide complete multi-OS emulation on x86 CPU only

- The whole installation process of a Linux distribution can be done with VMware

VMware

- Provide complete multi-OS emulation on x86 CPU only
- The whole installation process of a Linux distribution can be done with VMware
- Resource consumption is static (RAM, Disck, etc) and very important (up to 50% of the available computing power!)

- plex86

plex86

« (...) a very lightweight Virtual Machine (VM) for running Linux/x86»

http://plex86.sourceforge.net/ (Feb/2005)

plex86

« (...) a very lightweight Virtual Machine (VM) for running Linux/x86»

http://plex86.sourceforge.net/ (Feb/2005)

- Use the same Vmware logic but is restricted only to Linux OS (native OS as well as guest OS)

plex86

« (...) a very lightweight Virtual Machine (VM) for running Linux/x86»

http://plex86.sourceforge.net/ (Feb/2005)

- Use the same Vmware logic but is restricted only to Linux OS (native OS as well as guest OS)
- It's needed to recompile the kernel on the guest OS

plex86

« (...) a very lightweight Virtual Machine (VM) for running Linux/x86»

http://plex86.sourceforge.net/ (Feb/2005)

- Use the same Vmware logic but is restricted only to Linux OS (native OS as well as guest OS)
- It's needed to recompile the kernel on the guest OS
- Very slow at the time of this writing

Bochs

Bochs

« Bochs is a highly portable open source IA-32(x86) PC emulator written in C++, that runs on most popular platforms. It includes emulation of the Intel x86 CPU, common I/O devices and a custom BIOS. Currently, Bochs can be compiled to emulate a 386, 486, Pentium, Pentium Pro or AMD64 CPU including optional MMX, SSE, SSE2 and 3DNow instructions »

http://bochs.sourceforge.net/ (Feb/2005)

Bochs

- The performance of bochs does not compare to Vmware or plex86 mainly because it emulates the CPU instead of using the native instruction set of the IA-32 CPUs

Bochs

- The performance of bochs does not compare to Vmware or plex86 mainly because it emulates the CPU instead of using the native instruction set of the IA-32 CPUs

- There is no locking mechanism for the disks.

The Linux-VServers

The Linux-VServers

« Linux-VServer allows you to create virtual private servers and security contexts which operate like a normal Linux server, but allow many independent servers to be run simultaneously in one box at full speed»

http://www.linux-vserver.org (Feb/2005)

The Linux-VServers

- The Linux-VServer project consists of a kernel patch and installation of userland tools

The Linux-VServers

- The Linux-VServer project consists of a kernel patch and installation of userland tools
- It manage resources dinamically: a single kernel is in charge of allocating resources.

The Linux-VServers

- The Linux-VServer project consists of a kernel patch and installation of userland tools
- It manage resources dinamically: a single kernel is in charge of allocating resources.
- Priority, Memory, Disk space, CPU ticks can be managed dynamically for a given vserver.

The Linux-VServers

- The Linux-VServer project consists of a kernel patch and installation of userland tools
- It manage resources dinamically: a single kernel is in charge of allocating resources.
- Priority, Memory, Disk space, CPU ticks can be managed dynamically for a given vserver.
- Because only one kernel access the hardware and interrupts, it uses the advanced management mechanism already present in the Linux Kernel

The Linux-VServers

- As a consequence, this is a very fast and lightweight system as only the necessary services are run (ssh, http, postfix, etc) and not a complete boot process.

The Linux-VServers

- As a consequence, this is a very fast and lightweight system as only the necessary services are run (ssh, http, postfix, etc) and not a complete boot process.
- Additional security occurs inside a vserver; the Linux-VServer use the POSIX capabilities to increase its security.

The Linux-VServers

- As a consequence, this is a very fast and lightweight system as only the necessary services are run (ssh, http, postfix, etc) and not a complete boot process.
- Additional security occurs inside a vserver; the Linux-VServer use the POSIX capabilities to increase its security.
- Network access, device access and many more capabilities can be given or taken in order to have a more secure virtual server.

User-Mode Linux (UML)

User-Mode Linux (UML)

« User-Mode Linux is a safe, secure way of running Linux versions and Linux processes. Run buggy software, experiment with new Linux Kernel or distributions, and poke around in the internals of Linux, all without risking your main Linux setup»

http://user-mode-linux.sourceforge.net/ (Feb/2005)

User-Mode Linux (UML)

« User-Mode Linux is a safe, secure way of running Linux versions and Linux processes. Run buggy software, experiment with new Linux Kernel or distributions, and poke around in the internals of Linux, all without risking your main Linux setup»

http://user-mode-linux.sourceforge.net/ (Feb/2005)

- very slow performance because only one program can run in privileged mode: the host Kernel that support the hosted ones

User-Mode Linux (UML)

« User-Mode Linux is a safe, secure way of running Linux versions and Linux processes. Run buggy software, experiment with new Linux Kernel or distributions, and poke around in the internals of Linux, all without risking your main Linux setup»

http://user-mode-linux.sourceforge.net/ (Feb/2005)

- very slow performance because only one program can run in privileged mode: the host Kernel that support the hosted ones
- the performance penalty is very important and a complete boot process is necessary

Xen

Xen

« Xen is a virtual machine monitor for x86 that supports execution of multiple guest operating systems with unprecedented levels of performance and resource isolation»

http://www.cl.cam.ac.uk/Research/SRG/netos/xen/
(Feb/2005)

Xen

- this is achieved by installing a king of «mega-bios» layer (Xen) that hides the physical hardware and provides supported OS specific «Xen drivers» in order to interact with the Xen abstraction layer.

Xen

- this is achieved by installing a king of «mega-bios» layer (Xen) that hides the physical hardware and provides supported OS specific «Xen drivers» in order to interact with the Xen abstraction layer.
- the virtual servers interact with Xen hardware (including CPU) needs a specific kernel but applications can run unchanged.

Xen

- this is achieved by installing a king of «mega-bios» layer (Xen) that hides the physical hardware and provides supported OS specific «Xen drivers» in order to interact with the Xen abstraction layer.
- the virtual servers interact with Xen hardware (including CPU) needs a specific kernel but applications can run unchanged.
- a lightweight technology, but demands complete systems to be «booted» inside the Xen domains (virtual servers) so resource consumption (RAM, CPU, processes, etc) is much more important than the Linux-VServer project.

QEMU

QEMU

« QEMU is a generic and open source processor emulator which achieves a good emulation speed by using dynamic translation»

http://fabrice.bellard.free.fr/qemu/ **(Feb/2005)**

QEMU

- emulates only the x86 family of processors

QEMU

- emulates only the x86 family of processors
- supports emulation of user code on other architecture (ARM, SPARC, PowerPC)

QEMU

- emulates only the x86 family of processors
- supports emulation of user code on other architecture (ARM, SPARC, PowerPC)
- emulation, by default, very slow; a non-free layer (QEMU accelerator) gives a much better performance on the same architecture (x86 emulated on x86)

QEMU

- emulates only the x86 family of processors
- supports emulation of user code on other architecture (ARM, SPARC, PowerPC)
- emulation, by default, very slow; a non-free layer (QEMU accelerator) gives a much better performance on the same architecture (x86 emulated on x86)
- a young and still very experimental project

Classification of problems: usage criteria

Classification of problems: usage criteria

We present in the following several needs for computer virtualization and will use those criteria to compare the selected technology

Classification of problems: usage criteria

- Multi OS

Classification of problems: usage criteria

- Multi OS
- Kernel development / debugging

Classification of problems: usage criteria

- Multi OS
- Kernel development / debugging
- OS installation process

Classification of problems: usage criteria

- Multi OS
- Kernel development / debugging
- OS installation process
- Resources consumption

Classification of problems: usage criteria

- Multi OS
- Kernel development / debugging
- OS installation process
- Resources consumption
- Dynamical allocation of resources

Classification of problems: usage criteria

- Multi OS
- Kernel development / debugging
- OS installation process
- Resources consumption
- Dynamical allocation of resources
- Multi architecture

Classification of problems: usage criteria

- Multi OS
- Kernel development / debugging
- OS installation process
- Resources consumption
- Dynamical allocation of resources
- Multi architecture
- Maturity

Classification of problems: usage criteria

- Multi OS
- Kernel development / debugging
- OS installation process
- Resources consumption
- Dynamical allocation of resources
- Multi architecture
- Maturity
- Security

Multi OS

Multi OS

- Some virtualization technology only support a type of OS (Linux, Windows, FreeBSD, etc) while others are more generic and can run Linux on Windows, Windows on Linux, etc.

Multi OS

- Some virtualization technology only support a type of OS (Linux, Windows, FreeBSD, etc) while others are more generic and can run Linux on Windows, Windows on Linux, etc.

- Multi OS virtualization systems include VMware and Xen.

Kernel development / debugging

Kernel development / debugging

- Some users need to develop the kernel. This criteria will define if, yes or no, those tasks can be achieved with the chosen virtualization technique

Kernel development / debugging

- Some users need to develop the kernel. This criteria will define if, yes or no, those tasks can be achieved with the chosen virtualization technique

- UML has been designed for Kernel Hacking and development

OS installation process

OS installation process

- Some users need to reproduce the complete installation of a system (install CD, network boot, hard disk partitioning, etc).

OS installation process

- Some users need to reproduce the complete installation of a system (install CD, network boot, hard disk partitioning, etc).

- VMware supports perfectly the simulation of the installation process for the supported Linux distributions

Resources consumption

Resources consumption

- This criteria will define how much resources a virtual computer need to use in order to be fully functional.

Resources consumption

- This criteria will define how much resources a virtual computer need to use in order to be fully functional.
- For each virtualization technique, the approximative resource consumption of a fully functional virtual server has been estimated.

Resources consumption

- This criteria will define how much resources a virtual computer need to use in order to be fully functional.
- For each virtualization technique, the approximative resource consumption of a fully functional virtual server has been estimated.
- VMware needs a lot of resources, as does UML, then Xen and finally Linux-VServers.

- Dynamical allocation of resources

Dynamical allocation of resources

- Some users need to dynamically change the resources used by a virtual computer. Some virtualization programs allow the user to live change the resources available for the virtual server while others can not do this.

Dynamical allocation of resources

- Some users need to dynamically change the resources used by a virtual computer. Some virtualization programs allow the user to live change the resources available for the virtual server while others can not do this.
- UML, Xen and Linux-VServers can dynamically alocate resources an ensure QoS criteria between the virtual servers and the host system.

Multi architecture

Multi architecture

- Some virtualization technology only support a type of architecture, x86 for the most part.

Multi architecture

- Some virtualization technology only support a type of architecture, x86 for the most part.

- UML and Linux-VServers support several architectures.

Maturity

Maturity

- This is a relative indicator of the maturity of the technology.

Maturity

- This is a relative indicator of the maturity of the technology.
- VMware is very mature (but not well supported with 2.6 kernel and more experimental kernels)

Maturity

- This is a relative indicator of the maturity of the technology.
- VMware is very mature (but not well supported with 2.6 kernel and more experimental kernels)
- UML and Linux-VServer are production ready

Maturity

- This is a relative indicator of the maturity of the technology.
- VMware is very mature (but not well supported with 2.6 kernel and more experimental kernels)
- UML and Linux-VServer are production ready
- Xen is more experimental

Security

Security

- While all virtualization techniques increases security by allowing system administrators to cleanly separate services on different virtual servers, some of them offers additional protections with rules/roles and additional security models that can make a virtual server more robust than a real one.

Security

- Linux-VServer share some code with the guest OS and this can be considered as a vulnerability.

Security

- Linux-VServer share some code with the guest OS and this can be considered as a vulnerability.
- We did not consider this as a vulnerability because we consider that if a security problem occurs in the kernel in a primitive method used by a Linux-VServer (chroot, chcontext, chbind, etc) then every Linux server (vserver or not) has this problem and has to be upgraded.

Security

- In this context, the Linux-VServer project is the more «security oriented» because it offers additional security features (POSIX capabilities).

Security

- In this context, the Linux-VServer project is the more «security oriented» because it offers additional security features (POSIX capabilities).
- The other technologies do not provide additional security.

Comparative study of the existing technology

Comparative study of the existing technology

- ***Only the major virtualization techniques will be analyzed***

Comparative study of the existing technology

- ***Only the major virtualization techniques will be analyzed***

- ***The Bochs and plex86 projects will not be compared with the others as they are not yet fully functional***

Comparative study of the existing technology

Name	Multi OS	Kernel Development	Intall Process	Resources	Dynamical Resources	Sercurity	Maturity	Architecture
Vmware	Yes	No	Yes	2 Gb	No	No	Good	x86
Linux-VServer	No	No	No	256 Mb	Yes	Yes	Excelent	x86, IA64, x86_64
UML	No	Yes	No	1 Gb	No	No	Good	x86, IA64, x86_64
Xen	Yes	Exp.	Yes	1 Gb	No	No	Young	x86
QEMU	Exp.	No	Exp.	1 Gb	No	No	Young	x86

Comparative study of the existing technology

- ***Based on the needs from the user, one should be able to easily choose the best suited virtualization technique***

- ***In order to facilitate this process, we have established some basic use-cases for the virtualization of computers:***

- ***In order to facilitate this process, we have established some basic use-cases for the virtualization of computers:***
  - Hosting

- ***In order to facilitate this process, we have established some basic use-cases for the virtualization of computers:***
  - Hosting
  - Testing one application

- ***In order to facilitate this process, we have established some basic use-cases for the virtualization of computers:***
  - Hosting
  - Testing one application
  - Build environment or development environment

- ***In order to facilitate this process, we have established some basic use-cases for the virtualization of computers:***
  - Hosting
  - Testing one application
  - Build environment or development environment
  - Testing distributed application and/or complex upgrade process

- ***In order to facilitate this process, we have established some basic use-cases for the virtualization of computers:***
  - Hosting
  - Testing one application
  - Build environment or development environment
  - Testing distributed application and/or complex upgrade process
  - Security usage

- ***In order to facilitate this process, we have established some basic use-cases for the virtualization of computers:***
  - Hosting
  - Testing one application
  - Build environment or development environment
  - Testing distributed application and/or complex upgrade process
  - Security usage
  - High availability

- ***In order to facilitate this process, we have established some basic use-cases for the virtualization of computers:***
  - Hosting
  - Testing one application
  - Build environment or development environment
  - Testing distributed application and/or complex upgrade process
  - Security usage
  - High availability
  - Disaster recovery

Hosting

Hosting

- An Internet provider or someone that simply have to provide access to one or several hosts on a real system.

Hosting

- An Internet provider or someone that simply have to provide access to one or several hosts on a real system.
- The resources consumption is very small because only the needed processes are started on the vservers.

Hosting

- An Internet provider or someone that simply have to provide access to one or several hosts on a real system.
- The resources consumption is very small because only the needed processes are started on the vservers.
- Additional security is provided by the POSIX capabilities

Hosting

- On demand servers can be created in seconds and delivered to the customer.

Hosting

- On demand servers can be created in seconds and delivered to the customer.
- Every Linux-VServer consist only of files that can be easily backuped and restored on another server if needed.

Hosting

- On demand servers can be created in seconds and delivered to the customer.
- Every Linux-VServer consist only of files that can be easily backuped and restored on another server if needed.
- Unification is a mechanism at the package level that allows Linux-VServers to share programs and library

Testing one application

Testing one application

- Perform stress tests or unitary testing on one application.

Testing one application

- Perform stress tests or unitary testing on one application.
- Is easy to move a Linux-VServer on different hardware to compare performance.

Testing one application

- Perform stress tests or unitary testing on one application.
- Is easy to move a Linux-VServer on different hardware to compare performance.
- Because the regular device drivers are used, the virtualization layer impact on performance measurement is expected to be nebligible.

- Build environment or development environment

- Build environment or development environment
  - Easily to create on deman different versions of distributions from a host system

- Build environment or development environment
  - Easily to create on deman different versions of distributions from a host system
  - Development starting from a clean virtual server:

- Build environment or development environment
  - Easily to create on deman different versions of distributions from a host system
  - Development starting from a clean virtual server:
    - Greatly increases bug reproducibility and process of development

Build environment or development environment

- Easily to create on deman different versions of distributions from a host system
- Development starting from a clean virtual server:
  - Greatly increases bug reproducibility and process of development
  - When a bug is found, the vserver where the bug can be triggered can be easily copied and «given » to the developer in charge.

- Testing distributed application and/or complex upgrade process

Testing distributed application and/or complex upgrade process

- One of the problems for complex applications is the fact that it is very difficult to reproduce, in the laboratory, an evironment similar to the production one.

Testing distributed application and/or complex upgrade process

- One of the problems for complex applications is the fact that it is very difficult to reproduce, in the laboratory, an evironment similar to the production one.
- As a consequence, and while this is certainly not the best practices, developers often need to develop on or «near» the production systems.

Testing distributed application and/or complex upgrade process

- One of the problems for complex applications is the fact that it is very difficult to reproduce, in the laboratory, an evironment similar to the production one.
- As a consequence, and while this is certainly not the best practices, developers often need to develop on or «near» the production systems.
- With one of the virtualization techniques it is very easy to duplicate the production environment in the laboratory: just copy your production virtual computer on a development system.

Security usage

Security usage

- The KISS[1] principle encourages the deployment of simple systems that only deliver one service per system.

[1] Keep It Simple and Stupid

Security usage

- The KISS[1] principle encourages the deployment of simple systems that only deliver one service per system.
- This principle is rarely used on the field because this will lead to a very big increase of the physical computers number.

Security usage

- The KISS[1] principle encourages the deployment of simple systems that only deliver one service per system.
- This principle is rarely used on the field because this will lead to a very big increase of the physical computers number.
- In turn, because modern computers have a huge computing power, those computers will be under-used

High availability

High availability

- While Xen is presently one of the first to manage load balancing between lives computers, one can easily set up a high availability system with any virtualization technique.

High availability

- While Xen is presently one of the first to manage load balancing between lives computers, one can easily set up a high availability system with any virtualization technique.
- A cold swap server that is synced either periodically (cron is your friend) of live, either at the application level (replication for MySQL, PostGreSQL, LDAP, etc) or with a low level tool like DR-DB/

High availability

- Then the hot or cold backup virtual-server can monitor failures from the other virtual-server provide a very inexpensive high availability layer.

High availability

- Then the hot or cold backup virtual-server can monitor failures from the other virtual-server provide a very inexpensive high availability layer.
- One can even use this procedure on a single hardware system: this will provide what we called «software high availability» and protect the user from software bugs.

Disaster recovery

Disaster recovery

- Virtualization deeply modify this area of modern computing providing an abstraction layer between the hardware and the virtual servers.

Disaster recovery

- Virtualization deeply modify this area of modern computing providing an abstraction layer between the hardware and the virtual servers.
- This meands that heterogeneous hardware can easily be used, without additional risk, to provide disaster recovery capacities.

Technology overview of Linux-VServers

Technology overview of Linux-VServers

- http://linux-vserver.org

Technology overview of Linux-VServers

- http://linux-vserver.org
- Created by Jacques Gelinas, a well know Linux hacker from Quebec (Linuxconf, insmod/modprobe, umsdos, etc).

Technology overview of Linux-VServers

- http://linux-vserver.org
- Created by Jacques Gelinas, a well know Linux hacker from Quebec (Linuxconf, insmod/modprobe, umsdos, etc).
- Project is leaded now by Herbert Poetzl and a lot of development occurs

Technology overview of Linux-VServers

- http://linux-vserver.org
- Created by Jacques Gelinas, a well know Linux hacker from Quebec (Linuxconf, insmod/modprobe, umsdos, etc).
- Project is leaded now by Herbert Poetzl and a lot of development occurs
- The community is very active and supportive

Technology overview of Linux-VServers

The Linux-VServer project can be see as the integration of 4 concepts, half of them having been specifically developed for the project:

Technology overview of Linux-VServers

The Linux-VServer project can be see as the integration of 4 concepts, half of them having been specifically developed for the project:

- chroot: disk isolation

Technology overview of Linux-VServers

The Linux-VServer project can be see as the integration of 4 concepts, half of them having been specifically developed for the project:

- chroot: disk isolation
- chcontext: process isolation

Technology overview of Linux-VServers

The Linux-VServer project can be see as the integration of 4 concepts, half of them having been specifically developed for the project:

- chroot: disk isolation
- chcontext: process isolation
- chbind: network isolation

Technology overview of Linux-VServers

The Linux-VServer project can be see as the integration of 4 concepts, half of them having been specifically developed for the project:

- chroot: disk isolation
- chcontext: process isolation
- chbind: network isolation
- capabilities: additional security

chroot: disk isolation

chroot: disk isolation

- Once called, the chroot system call allow the following commands to start from a different filesystem root.

chroot: disk isolation

- Once called, the chroot system call allow the following commands to start from a different filesystem root.
- This provides what we can call «disk isolation».

chroot: disk isolation

- Once called, the chroot system call allow the following commands to start from a different filesystem root.
- This provides what we can call «disk isolation».
- It is very common to use a chrooted environment for security sensible services (FTP, Bind, etc).

chroot: disk isolation

- Once called, the chroot system call allow the following commands to start from a different filesystem root.
- This provides what we can call «disk isolation».
- It is very common to use a chrooted environment for security sensible services (FTP, Bind, etc).
- If the chrooted service is hacked, only the files writable inside the chroot can be compromised.

chroot: disk isolation

Briefly:

the root of all the commands run in a Linux-VServer is not the same as the host system root. This provides file system isolation.

chcontext: process isolation

chcontext: process isolation

- This is a specific vserver system call that creates a new security context.

chcontext: process isolation

- This is a specific vserver system call that creates a new security context.
- This provide what we call «process isolation».

chcontext: process isolation

- This is a specific vserver system call that creates a new security context.
- This provide what we call «process isolation».
- The usual or «hosted» security context is the context "0", which has the same privileges of the root user (UID 0): can see and kill other tasks in the other contexts.

chcontext: process isolation

- If we except the context number 1 which is used to «view» other contexts but can not affect them, then the context isolation is complete: processes from one context can not see neither interact with processes from another context.

chcontext: process isolation

- If we except the context number 1 which is used to «view» other contexts but can not affect them, then the context isolation is complete: processes from one context can not see neither interact with processes from another context.
- This provide the ability to run similar contexts on the same computer without any interaction possible at the application level.

chcontext: process isolation

Briefly:

the root of all the commands run in a Linux-VServer is not the same as the host system root. This provides file system isolation.

chbind: network isolation

chbind: network isolation

- The other vserver specific system call that provides «network isolation».

chbind: network isolation

- The other vserver specific system call that provides «network isolation».
- Once called, all traffic sent by any of the network interface is alterated so that it comes from the argument given to chbind (an ipv4 or ipv6 address).

chbind: network isolation

- The other vserver specific system call that provides «network isolation».
- Once called, all traffic sent by any of the network interface is alterated so that it comes from the argument given to chbind (an ipv4 or ipv6 address).
- Processes run from one chbind send packets with one IP address while processes run from another chbind send packets with another IP adress.

chbind: network isolation

- The other vserver specific system call that provides «network isolation».
- Once called, all traffic sent by any of the network interface is alterated so that it comes from the argument given to chbind (an ipv4 or ipv6 address).
- Processes run from one chbind send packets with one IP address while processes run from another chbind send packets with another IP adress.
- This uses the virtual device infrastructure that allow a computer with a single NIC to have numerous IP address.

chbind: network isolation

Briefly:

each packet send from a Linux-VServer has its origin sent to a well defined IP address. This provides network isolation.

capabilities: additional security

capabilities: additional security

- The POSIX capabilities were designed to «hardened» a POSIX system.

capabilities: additional security

- The POSIX capabilities were designed to «hardened» a POSIX system.
- A root account in a default Linux-VServer has much less privileges than a root account on a regular Linux server

capabilities: additional security

- The POSIX capabilities were designed to «hardened» a POSIX system.
- A root account in a default Linux-VServer has much less privileges than a root account on a regular Linux server
- For instance, IP addresses cannot be changed (no ifconfig!), nodes can not be created (no mknod), hardware time can not be set, etc.

capabilities: additional security

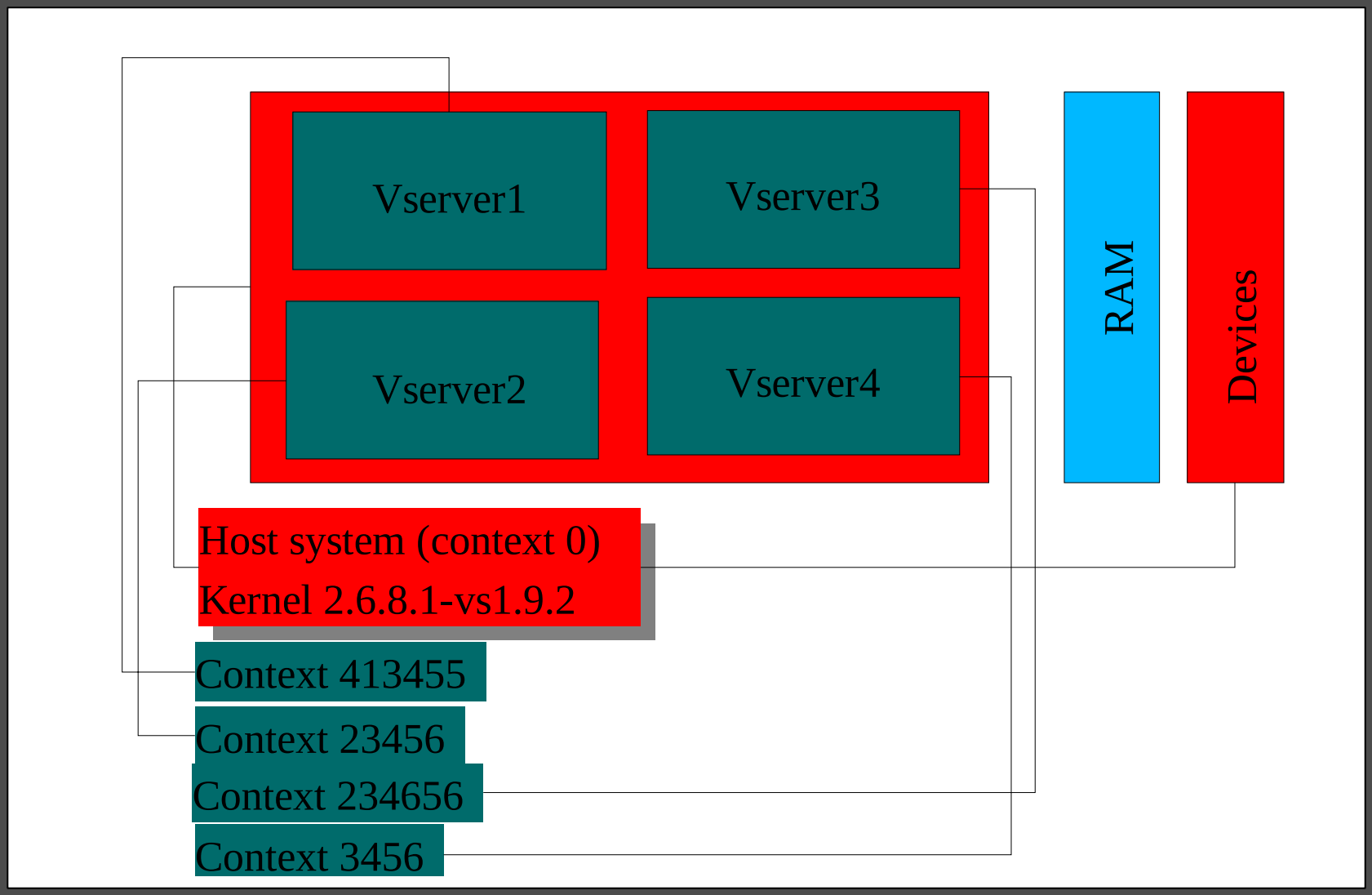
- This is specially intersting because fits very nicely with the Linux-VServer model where only the host server can set up certain properties of the vserver (IP address, time, network interface, etc) and the Linux-VServers can not alter those settings (for obvious security reasons).

capabilities: additional security

Briefly:

each Linux-VServer has a set of capabilities (none by default) in order to be able to work. Strictly speaking, this means that a root on a Linux-VServer has much less «privileges» than a root account on a regular Linux server. This provides «root»-isolation.

How it works?



How it works?

How it works?

- Context 0 has power over all the others contexts

How it works?

- Context 0 has power over all the others contexts
- Context 1 : can only watch the other contexts (special)

How it works?

- Context 0 has power over all the others contexts
- Context 1 : can only watch the other contexts (special)
- Other contextes : can only see themselves.

How it works?

- Context 0 has power over all the others contexts
- Context 1 : can only watch the other contexts (special)
- Other contextes : can only see themselves.
- Devices : it's the host server (context 0) that decides who have acess to what

How it works?

- Context 0 has power over all the others contexts
- Context 1 : can only watch the other contexts (special)
- Other contextes : can only see themselves.
- Devices : it's the host server (context 0) that decides who have acess to what
- Exemple : network, mount points, /proc, etc.

Conclusion

Conclusion

- ***Because of its maturity (several production systems with more than 20 Linux-VServers in production for years) and because this is the more lightweight virtualization technique, we believe that Linux-VServer is the best tool for virtualizing Linux servers on a Linux operating system host.***

Conclusion

- ***There are some cases where other techniques are necessary, mainly running another OS and kernel development, but beside this two cases, the Linux-VServer is really the best virtualization technique available.***

Conclusion

- ***The use of a single kernel for all the Linux-VServers hosted on one system provides the project several key advantages when compared to other virtualization techniques:***

Conclusion

- ***The use of a single kernel for all the Linux-VServers hosted on one system provides the project several key advantages when compared to other virtualization techniques:***

- Lightweight: only services are started on the hosted Linux-VServer, not all the processes resulting from a complete boot process.

Conclusion

- ***Uses the latest Linux kernel development easily:*** *for instance, with the O(1) scheduler, all the processes are well prioritized.*

Conclusion

- ***Uses the latest Linux kernel development easily:*** ***for instance, with the O(1) scheduler, all the processes are well prioritized.***

- Native usage of device drivers: with the Linux-VServer project, one can use the latest kernel drivers without any performance penalty introduced by the virtualization layer.

Acknowledgements

This research has been funded by the National Research Council's Industrial Research Assistance Program (NRC-IRAP), project number 547017

Jacques Gelinas, for the original idea and valuable discussion.

Herbert Poetzl, the current project leader

The Linux-VServer community for their positive attitude.


REVOLUTION
LINUX
www.revolutionlinux.com