Virtualization and Emulation in Linux

VIRTUES OF THE VIRTUAL

You'll find a virtualization solution for every Linux environment – from the desktop to the enterprise server.

In this month's cover story, we investigate some promising virtualization tools for Linux users.

BY TIM SCHÜRMANN

In the old days, an operating system was somehow monogamously tied to the hardware. Aside from a few experts and visionaries, no one even considered the possibility of many systems sharing the same iron simultaneously. And no one lost any sleep worrying about the option of one operating system running on a different operating system. In today's virtual world, however, your applications may not ever know where the hardware stops and where the software begins.

Virtualization provides several benefits for the Linux user: stability, manageability, security, and even nostalgia. This month, we take close look at some Linux virtualization options.

We start by comparing virtualization alternatives for the Linux desktop. Then we'll examine virtualization on IBM's System p servers. You'll also learn about the famous Bochs emulator, and we'll finish with a look at a Wine-based tool called IEs4Linux that lets you run Internet Explorer in Linux.

Competition and Emulation

More than 40 years ago, long before Linux was invented, IBM had a problem.

COVER STORY

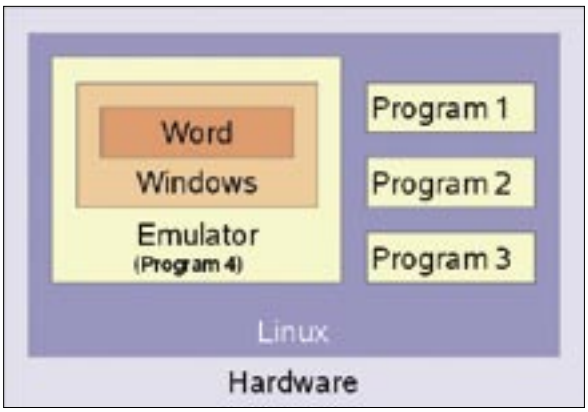

Figure 1: An emulator creates a PC within a PC. In this example, Windows and Word are running within the emulation. Neither application can access the underlying Linux system.

The new System/360 [1] worked in a way that was completely different from the antiquated 7070. To help potential customers migrate more easily, IBM wanted to let legacy applications run on the new system.

After various tests, IBM finally went for an idea submitted by engineer Larry Moss, who suggested a combination of software and a special hardware extension. Once started, the combined solution would monitor every single step of the legacy application and convert its commands into commands the new System/360 computer could understand. This way, the new computer would behave exactly like its predecessor.

This simple sleight of hand gave purchasers of the new model the ability to continue using their legacy 7070 applications. Because the software emulated the behavior of another computer, Larry Moss dubbed his invention an **emulator**.

Modern emulators copy the behavior of a complete computer at software level, and they do it so perfectly that you can launch a guest operating system along with applications designed for it in the emulator.

This gives users a virtual PC, or more generically, a **virtual machine** (see Figure 1). QEMU [2] and Bochs [3] are just two examples of popular, free emulators that simply reroute the screen output into an application window of their own. The operating system running on the virtual computer thinks it is using a standard display.

Emulators offer a number of benefits. First, it is quite simple to clone a system running on the emulator. Because the emulator copies a complete PC at software level, it is easy to freeze the current state and create a snapshot archive or let the virtual PC run on a completely different computer.

Because the simulated hardware does not change, you can even migrate across hardware boundaries. This gives users the ability to try out new programs without any risk. You simply save the current emulator state, install the application you want to test, and then restore the original state if you are unhappy with the test results.

In addition to these software-only solutions, some hardware emulators also exist. And just as in the case of the historic IBM System/360, there are also some hybrid solutions. A recent example of a software/hardware hybrid solution is Sony's PlayStation 3. The first version of PlayStation 3 includes a hardware component that allows PlayStation 2 games to run on the console. The current European version of the PlayStation 3 system uses a software-only emulator for this.

This example shows another main use for emulators – the ability to instill new

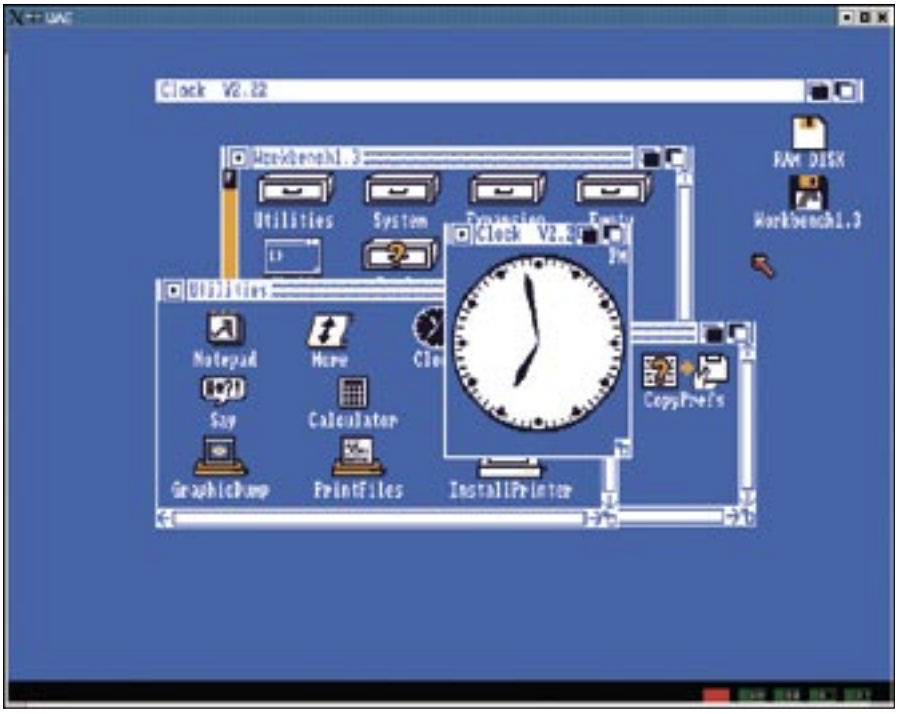

Figure 2: The UAE emulator revives the legacy Commodore Amiga. You can see the Amiga GUI running on a window in Linux here.

Unknown Virtual Machines

Virtualization solutions crop up at the most unexpected places. For example, the Open Firmware alternative BIOS is also a virtual machine [16]. The hardware installed in the computer can store its own extensions to the basic configuration at this location no matter what kind of hardware platform it runs on.

Some CPUs or computers support a compatibility mode. All of today's x86 CPUs can run programs for ancient relatives. CPU manufacturer Transmeta takes this a step further – to retain compatibility with Intel and AMD CPUs, Transmeta simply translates *all* instructions into Tansmeta's own format before executing.

The Commodore 128 from the 1980s is another example of virtualization. The home computer had a native operating mode, but it could also run applications designed for its smaller brother, the C=64, and for CP/M, the state-of-art PC operating system at the time.

GLOSSARY

Emulator: A word for a program that copies the behavior of another application, derived from the Latin word "aemulare" (to compete, emulate).

life into defunct hardware and software. Just imagine running the good old accounts program from the heady days of DOS or the legendary WordPerfect processor from the same era.

Emulators of this kind tend to focus on video game consoles and home computers from the last century. Fans have created emulators that copy these legacy treasures with bit precision. This gives classic machines such as the Commodore Amiga [4], Atari ST [5], or the C = 64 [6] a new lease of life on Linux PCs (see Figure 2).

Because they are what they are, emulators can cause issues. An emulator needs to copy a computer's hardware components as precisely as possible, so you'll need to make sure the system has sufficient computational power.

Older systems built for less-advanced hardware are often less taxing. As a rule of thumb, the older an operating system is, the faster the system will run in an emulator.

In contrast to this, running a recent version of Ubuntu on QEMU gives you performance similar to running Ubuntu on a machine that is well past its prime.

Figure 3: ScummVM brings old adventures back to life by giving them what they need – a script interpreter.

The closer you look, the less sense it makes to emulate a complete Intel CPU if you already have one in your computer. Instead, you could use the CPU directly and would only need to simulate the rest of the required target system.

You could apply the same principle to any other component by just emulating

the hardware you need at the software level. In this case, we no longer refer to the approach as emulation. Instead, we call it virtualization.

If you simply remove the CPU simulation from an emulator, this is referred to as hardware or system virtualization. You can install the KQEMU extension for QEMU to use this mode. The commercial VMware [7] product also does without CPU emulation. The downside of this approach is the restriction to a specific CPU type. For example, neither of these solutions will work on a PowerPC out of the box.

Para-Virtualization

An emulator runs as a normal program on Linux or Windows, but theoretically you could eliminate the host operating system. To allow this to happen, you need special software that runs directly on the hardware.

Programs of this kind are known as hypervisors or virtual machine monitors. The hypervisor manages the guest operating systems installed on the computer independently of the operating system, and thus ensures trouble-free parallel operations.

In this kind of environment, guest systems do not work directly with the hardware, but pass requests to the hypervisor. For example, if one of the guest Linux systems running in parallel on a system needs disk access, the guest Linux system issues an access request to the hypervisor. The hypervisor then handles physical access and returns the results to the requesting system.

To allow all the guests to talk to the hypervisor, it offers them a standardized interface to the physical hardware, which other programs and operating systems can use. This technique, which is known as para-virtualization, has the advantage of amazingly fast execution speeds compared with other solutions. The vendors of para-virtualization solutions refer to performance hits of just 0.5 to 3 percent compared with physical hardware. The free Xen [8] product and the commercial ESX Server by VMware [7] are the most popular examples of this technology.

The requirement for the guest system to support the hypervisor is an obstacle to para-virtualization because it implies modifying the operating system. And with a battened-down system like Microsoft Windows, this task of modifying the operating system is obviously difficult. Another complication is that the hypervisor itself has to handle a number of operating system tasks. For example, the hypervisor needs to know what kind of graphics adapter the system has and how to address it.

To avoid drowning in a sea of driver modifications, hypervisor developers typically opt for one of the following approaches:

- The Xen project's hypervisor simply chooses one of the parallel operating systems as its favorite guest. Once this relationship is established, the hypervisor uses the guest system's drivers. In other words, if another operating system running on the machine accesses a USB interface, the hypervisor passes this access request to the privileged guest operating system.
- The second approach converts an existing Linux kernel to produce a hypervisor – Linux has more or less everything the hypervisor needs. The KVM project [9], for example, uses this virtualization technique. The project provides a kernel module that converts the current Linux kernel into a hypervisor. The hypervisor then uses a modified QEMU emulator to launch other operating systems. This virtualization method is known as kernel-based virtualization.

Virtualizing Operating Systems

If you simply need to clone an instance of Linux that is running on your machine, operating system virtualization is probably your best option. In contrast to the other virtualization solutions, operating system virtualization means that the computer only has to run one operating system, which is cloned based on templates. The templates specify the configuration of the new clone; for example, administrators can restrict access to disk space.

During cloning, the virtualization solution typically just copies the system environment. Under the hood, there is

Licenses

To run a commercial operating system on a virtual machine, you do need a valid license. For example, to run a Microsoft system in a virtual environment, you need to purchase your own copy of Windows. Check out your software manufacturer's licensing details to be on the safe side.

Another issue – and one that particularly affects older computers or operating systems – is the patented firmware, which may be protected by copyright or patent law. For example, you need a matching BIOS for a full PC emulation. Older Apple computers actually store part of the operating system on chips in an approach that was very popular in the days of home computers. In the case of the PC, fortunately, there are various free BIOS alternatives [17] and most virtualization products include them. For all other computers, there is no alternative but to grab the firmware using a special program.

INFO

[1] IBM System/360: *http://en.wikipedia.org/wiki/System/360*

[2] Qemu and KQemu: *http://fabrice.bellard.free.fr/qemu/*

[3] Bochs PC Emulator: *http://bochs.sourceforge.net*

[4] UAE Amiga Emulator: *http://uae.coresystems.de*

[5] StonX Atari ST Emulator: *http://stonx.sourceforge.net*

[6] VICE Commodore 64 Emulator: *http://www.viceteam.org*

[7] VMware: *http://www.vmware.com*

[8] Xen: *http://www.cl.cam.ac.uk/research/srg/netos/xen/*

[9] KVM: *http://kvm.qumranet.com/kvmwiki*

[10] VServer: *http://linux-vserver.org/Welcome_to_Linux-VServer.org*

[11] OpenVZ: *http://openvz.org*

[12] Java programming language: *http://www.java.com*

[13] ScummVM: *http://www.scummvm.org*

[14] Intel Virtualization Technology: *http://www.intel.com/technology/platform-technology/virtualization/index.htm*

[15] AMD Virtualization: *http://www.amd.com/us-en/Processors/ProductInformation/0,,30_118_8796_14287,00.html*

[16] Open Firmware: *http://www.openbios.info*

[17] Free PC BIOS: *http://www.linuxbios.org/Welcome_to_LinuxBIOS*

still a single Linux kernel, which all the active programs use no matter which clone they happen to be running on. Operating system virtualization simply locks the application away in its own system environment, which gives you a fast and secure system. Programs for other operating systems are not supported, however.

Linux VServer [10] and OpenVZ [11], an open source derivative of the commercial Virtuozzo, are examples of free operating system virtualizers. Both variants use a modified Linux kernel.

Application Virtualization

In some cases, it is unnecessary to clone a whole system environment to keep programs separate. Application virtualization gives users a trouble-free package for any application containing the resources the application needs to run.

These resources could include configuration files, libraries, or auxiliary programs. The package bundle this creates is called a (virtual) runtime environment.

Programs packaged in this way don't need to be installed. Applications developed in the Java programming language are examples of this category. To execute a Java program, you need the Java Runtime Environment [12]. ScummVM [13], which is responsible for reincarnating a couple of older adventure games, also relies on this principle (Figure 3).

Partial Virtualization

Partial virtualization is a special case in the virtualization world. This solution involves pretending that a hardware component exists multiple times. If this is done for RAM memory, the process is known as address space virtualization. In this case, programs running on the system think they have exclusive access to memory. Other processes are hidden to the application.

Server Virtualization

Internet service providers are particularly interested in virtualization solutions. For one thing, these solutions support shared hosting, that is, the ability for multiple customers to share a physical Internet server. These customers assume – and they are not entirely wrong – that they are working on their own servers. Website operators can use this technology to enhance their response to Internet requests by distributing the load over their servers while reducing operating costs by sharing hardware.

The virtualization solutions used for server virtualization rely on the models I looked at earlier, but they also depend on server-level features such as load distribution or special administrative interfaces.

Examples of products for server virtualization include VServer or OpenVZ. Both systems use a modified Linux kernel, which launches and manages the individual virtual environments.

Conclusions

Apart from the huge volume of buzzwords bandied about by technical wizards and marketing consultants, virtualization offers interesting options for the Linux user.

You can run different operating systems at the same time, lock away critical applications, and try out new configurations without any risk to existing systems. We hope you enjoy this month's Virtualization cover story. ■