

The Linux operating system as a managed object

The case for infrastructure management

Skill Level: Introductory

[Stephen B. Morris \(stephenbjm@yahoo.com\)](mailto:stephenbjm@yahoo.com)
CTO
Omey Communications

29 Jul 2008

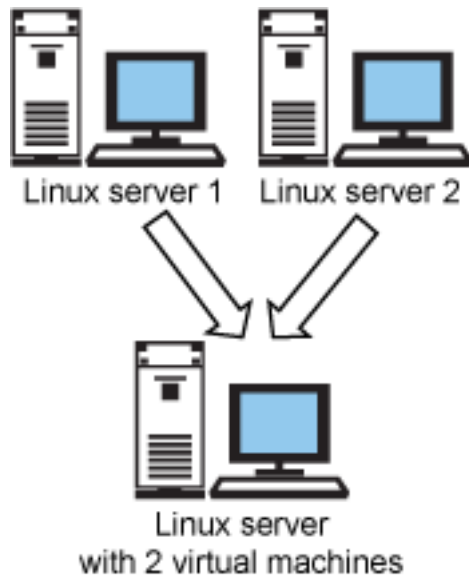
Organizations today face two major challenges: deployment of an increasingly rich service mix and managing the associated massive base of computing platforms. In this article, discover a new(ish) means of viewing a key component of the organizational architecture—treating operating systems themselves as individual managed objects.

Few can argue with the success of modern computing technology. For better or worse, the global proliferation of computing access is historically without precedent. So too are the consequences; witness the advent of spam and online fraud as more and more people go online. However, there are a great many positive aspects to global computerization, such as access to information and a well-informed, flexible global workforce.

Underpinning the success of modern computing is the not-so-well-known data center in which vast numbers of servers and other hardware reside. The demand for computing services has grown to such an extent that many data centers are rapidly approaching (or have already exceeded) physical limits in relation to size and power consumption.

Virtualization is a relatively new technology that is helping to reduce the strain on the data center. This saving is achieved as illustrated in Figure 1, where you see two physical Linux® servers being consolidated into virtual machines (VMs) hosted on one server.

Figure 1. Consolidating Linux servers using virtualization



The setup shown in Figure 1 might seem like a lot of trouble to go to just to reduce the number of data center machines, which leads to a key question: Why virtualize?

Why virtualize?

So, why would it be a good idea to subsume two Linux servers into two VMs? Well, for one thing, the power requirement is immediately halved, as is the need for networking between the two computers. In fact, the saving may be even greater if the new host server is used for other duties. But the conservative estimate is that the saving is 50 percent. Is there a downside?

With technology, you generally don't get anything for free, and this case is no exception. By hosting the two Linux servers on one computer, you get a single point of failure: If the new machine goes down, it will take the two virtual Linux machines down with it.

The technology of virtualization was once very specialized, but this is no longer the case. In fact, as you'll soon see, you're no more than a couple of downloads away from deploying some of the most advanced virtualized solutions available. With relative ease, you can now host virtual Linux servers and Linux applications in Microsoft® Windows®. (I have to say, I was amazed by this the first time I saw it in action.) Let's dive into a virtual Linux installation.

Download and build a virtualized Linux installation

Begin by downloading a virtual Linux image from the [thoughtpolice site](#). The Linux image I've selected for this article is Ubuntu-server-7.10-i386, but you can use any of the other images on the site. For example, your information technology (IT)

department might use Fedora as a standard. If this is the case, then some of the following instructions might need to change a bit. But the overall installation is relatively unchanged.

Resources for IT architects

IBM breaks IT architecture down into six main disciplines: enterprise, application, information, integration, infrastructure, and operations. Find definitions of these disciplines in [New to Architecture](#) and find resources to help you architect enterprise and software systems in [free IT architecture kits](#) from IBM. Then find technical [articles and tutorials](#), [tools downloads](#), [skills road maps](#), [forums](#), and other learning and community resources to help you develop skills to architect solutions in the [Architecture area on developerWorks](#).

Notice that the VMware Linux images have a large disk footprint. The Ubuntu-server-7.10-i386 image I'm using is about 193MB, so it might take a little time to download. An additional software item you need to go along with the VMware image is what's called a *player*—that is, an execution platform for the virtual operating system (so, it would reside on the machine in the bottom half of [Figure 1](#)). The player you'll be using is called *VMware Player*. Think of this VMware Player rendering or executing the virtual operating system machine in a manner not unlike the manner in which you play a CD track.

You don't have to use VMware Player, of course. Instead, you can use VMware Server, which provides similar facilities to VMware Player with additional features such as the facility for creating new VMs.

An important part of getting up to speed with virtualization technology is recognizing that the Ubuntu image is a full-fledged Linux operating system. As such, you can run this Linux instance and interact with it either locally or across a network. One interesting application for such an operating system facility is using it as a preinstalled platform on which to distribute software. Used in this way, the image and the associated software are available as an integrated environment. Now, it's time to get the player for the Linux image.

VMware Player is available as a free software download from the [VMware site](#). Note that VMware Player also has a big disk footprint: around 172 MB. After you install VMware Player, you can run it from the Windows Start button, as shown in [Figure 2](#).

Figure 2. Starting VMware Player



Figure 2 represents a starting point in virtualized Linux use. Remember that VMware Player effectively renders the Linux image in a manner conceptually similar to the way an audio file is rendered—you just point and click. All the user-driven commands available in VMware Player appear in the **Commands** area on the VMware Player startup page.

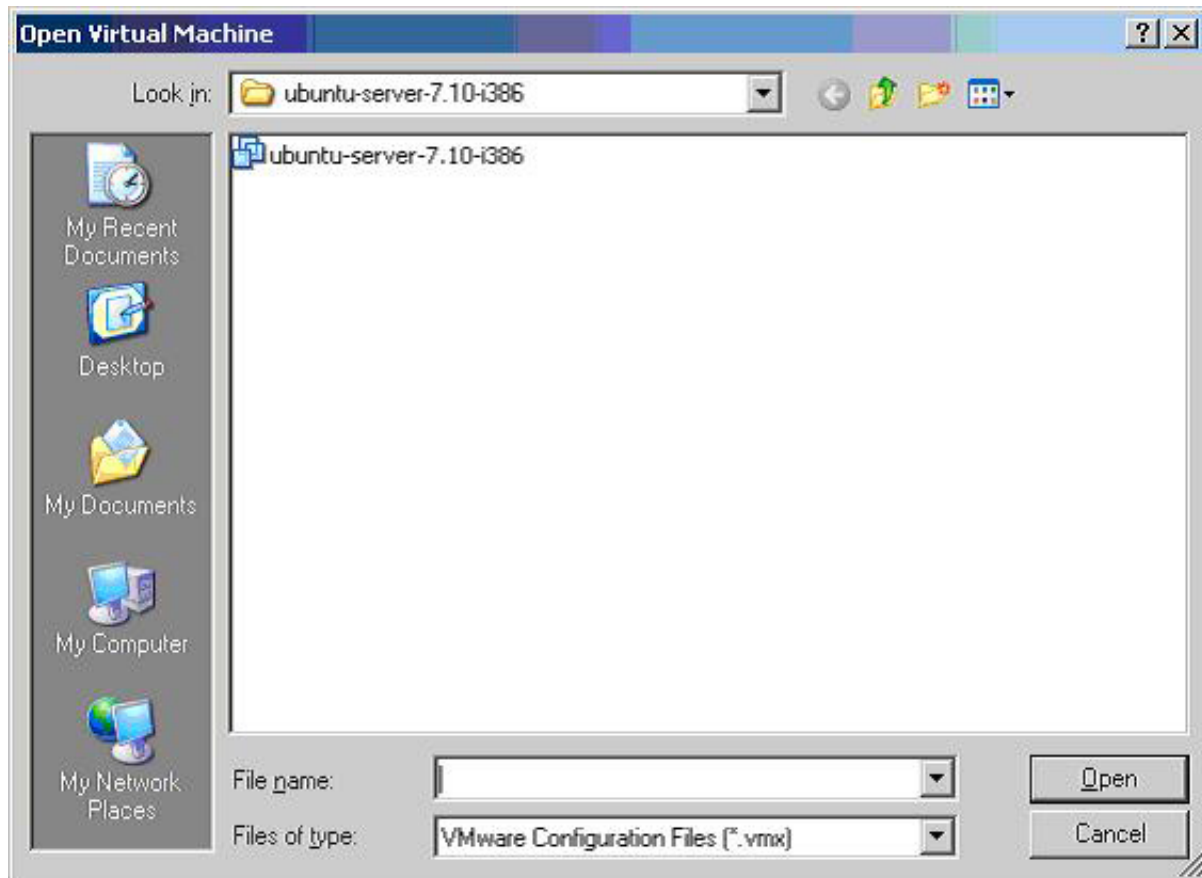
If you're interested in virtual appliances, have a look at the **Featured Virtual Appliance** area on the same VMware Player page. Before you actually run some virtual software, note that just below the **Commands** shown in Figure 2 is a list of recently run VM images. Chances are, this area is blank at the moment, because you've only just installed the software and haven't yet run a VM. You can do that now.

Run the Linux operating system image

If you're following along, you should have downloaded both VMware Player and one of the selected Linux VMware images. You can now run your selected virtual Linux image by clicking **Open** in the **Commands** area, then browsing to the location of

your VM. When located, the files appears in a new VMware Player window, as illustrated in Figure 3.

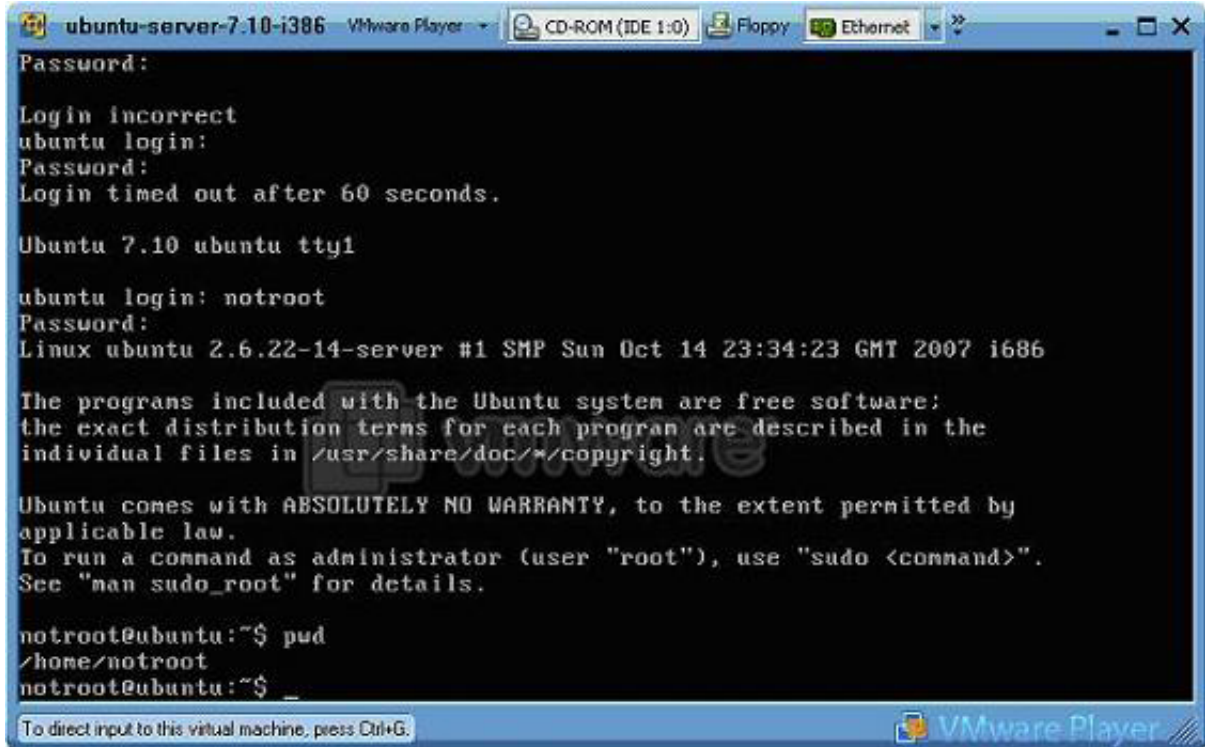
Figure 3. Browsing to your selected VM image



Before moving on from this window, notice the `.vmx` file extension of the Linux image. This extension represents the file format used for storing a VMware VM. The Linux instance is also referred to as a *guest operating system*, because it executes inside another operating system. To actually execute the guest operating system, simply select the `.vmx` file, and then click **Open**. Doing so launches the virtual Linux instance as a guest operating system. At this point, if all is well with your setup, you'll see the complete boot-up sequence for your chosen version of Linux.

What's kind of magical about all this is that it happens in a console window as an independent application under Windows. At the moment of login, you may have to enter (by clicking inside) the VM, and then press the Return key. You must do this to be prompted to type your user name and password. When you're prompted to log in to the Ubuntu Linux VM, type the user name `notroot` and the password `thoughtpolice`. (This is normal for the www.thoughtpolice.co.uk Ubuntu Linux distribution). The login window is illustrated Figure 4.

Figure 4. The virtual Ubuntu Linux instance has started up and is ready for use



```
ubuntu-server-7.10-i386 VMware Player
Password:
Login incorrect
ubuntu login:
Password:
Login timed out after 60 seconds.

Ubuntu 7.10 ubuntu tty1

ubuntu login: notroot
Password:
Linux ubuntu 2.6.22-14-server #1 SMP Sun Oct 14 23:34:23 GMT 2007 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

notroot@ubuntu:~$ pwd
/home/notroot
notroot@ubuntu:~$ _

To direct input to this virtual machine, press Ctrl+G. VMware Player
```

You've now got a virtual Linux instance running on your Windows computer. How do you move between Windows and the virtual Linux instance? Moving between the guest operating system and the native Windows operating system is easy: Simply press Control-Alt. This key pair signals to VMware Player that you want to move the focus back to the native Windows host.

Close the operating system image

If you want to shut down a Linux image in VMware Player, type the command:

```
sudo shutdown -r now
```

Then, you're prompted for the root password followed by the notroot password. This is because for safety, the Ubuntu VMware image doesn't have a root password. You can, of course, create a root password if you wish. That's where the `sudo` command comes into play. Rather than logging in as the root user, the `sudo` command is a more secure, accepted, and safe method of executing superuser (that is, root-level) commands.

So, that's the basic infrastructure setup. As you can see, you can run any number of Linux virtual instances on a Windows computer, and it's relatively simple to get up and running with VMs. So, that solves part of the original problem: reducing the required number of machines in the data center. What about the issue of managing

Linux itself?

Viewing Linux as a managed object

In the remainder of this article, I look at these Linux instances as managed objects. This means that each Linux server is seen as a separate and distinct manageable entity. Here's a brief review of the way in which modern servers and software are typically managed.

Traditional view of an operating system

In the traditional model for managing instances of Linux and other operating systems, problems tend to be solved as they arise:

- Server memory runs low: Shut down some processes.
- Processor utilization is running too high: Shut down some processes.
- Application component fails: Restart the component.
- Insufficient server capacity: Add more machines.

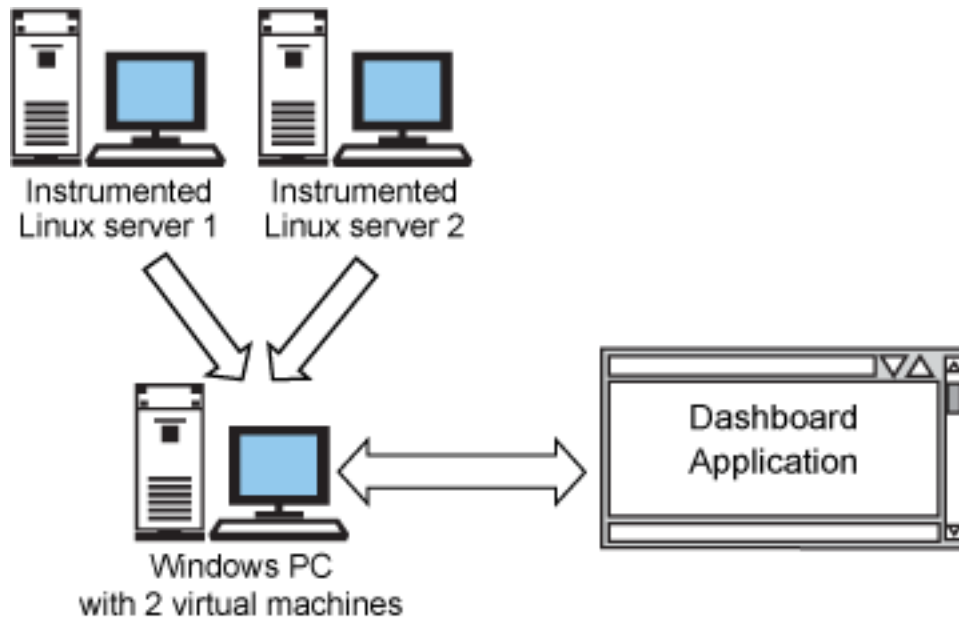
As you can see, these problems and their solutions are essentially reactive in nature—that is, they require human intervention. In many cases, the administrators may well be guided in their actions by dashboard software. The latter usually attempts to collect data and present it in an accessible format to aid decision making. On a more subtle level, these traditional solutions are not very agile.

This reactive model works fine as long as service level agreements (SLA) aren't too stringent and staff cost isn't too great an issue. However, as users become more demanding and costs escalate, the reactive model loses its appeal. So, what's the alternative? In a word, *instrumentation*.

Compress the operating system

By adding targeted instrumentation to Linux, the latter can become far more manageable. To some extent, this is part of the promise of [IBM autonomic computing](#), where instrumentation takes the form of touch points. Regardless of the approach used, the result becomes similar to that shown in Figure 5.

Figure 5. Instrumented Linux instances and a management dashboard



In Figure 5, each Linux instance hosts management software that enables it to become a managed object. You can then monitor and control the operating system managed object using the dashboard application.

Managing the compressed operating system today

The ideas presented in the preceding sections aren't especially new. Systems administrators have been struggling to manage servers for decades, and the form this has taken is generally some combination of:

- Software and scripts to monitor memory, processor, and disk loading.
- Reactively handling outages (for example, problems caused by software bugs).
- Restarting servers.
- Using intuition and guesswork.

What I propose here is a set of guidelines for Linux instrumentation to facilitate effective administration. This necessarily involves software structured to allow for both an overview and a specific view of the state of a Linux instance. The key elements of such software are:

- Standardization.
- Simplicity.
- Extensibility.

Standard solutions help to avoid vendor lock-in—that is, it should be possible to use a dashboard from any compliant vendor. Simplicity should facilitate ease of understanding in using a management solution. Extensibility should allow for accommodation of future requirements without having to resort to forklift upgrades.

Architectural guidelines for a better compressed operating system

So, what architectural guidelines can you use to facilitate improved (nonproprietary) compressed operating system management? The following is a list of desirable architectural quality attributes that apply to management instrumentation:

- Simplicity
- Usability
- High availability, low impact
- High performance
- Modifiability
- Security
- Testability—for unit, integration, and system tests

I place simplicity and usability at the top of the list, because previous generations of management technology have all suffered from being overly complex. Any management infrastructure must always be available and sufficiently lightweight so as not to affect a system under stress. High performance helps to reduce the impact on the managed system. Changes in management requirements are almost inevitable and should not result in massive disruption. Because management software often has system-level access, it is important for it to be secure and testable.

Conclusion

Data center crowding is a growing problem that in some ways has been exacerbated by the problem of overprovisioning. The latter is the procedure by which additional machines are added to cope with projected demand. Rather than "right size," the approach generally taken has been to "up-size." The result is that many data centers are now bursting at the seams, having run out of power and space.

Virtualization technology has helped to alleviate some of the overcrowding. Indeed, some virtualization tools are easy to understand and use. However, when virtualization has been exhausted as an interim solution, a more fundamental review will be needed at the Linux infrastructure management level.

Linux is now a key component of the global computing infrastructure. Despite this notable success, management systems continue to suffer from age-old problems: excessive complexity, proprietary nature, noninteroperability, and so on. So, if you want to manage Linux platforms, you must create your own or use a proprietary solution. This is not a future-proof approach.

Previous generations of management technologies all suffer from the problems of excessive complexity, heavyweight implementations (excessive resource consumption), too fine-grained an approach, and so on. In effect, the area of Linux systems management infrastructure is wide open for a root and branch review. A good start for this dialog is a set of architectural guidelines similar to those I provided here.

Resources

Learn

- "[Enable C++ applications for Web service using XML-RPC](#)" (Karthik Subbian and Ramakrishnan Kannan, developerWorks, 20 Jun 2006) is a step-by-step guide to exposing C++ methods as services.
- With the "[Windows-to-Linux road map](#)" (developerWorks, November 2003), start moving your operational skills from a Windows to a Linux environment. This nine-part series shows you how to "think in Linux" and covers common commands, point-and-click configuration, working with filesystems and logs, using networking and recovery tools, and compiling packages from available source code.
- Our [Linux fundamentals and certification-prep tutorials](#) run the gamut of Linux administration tasks. Take over 25 tutorials in sequence to build fundamental Linux skills from the ground up, or zero in on the exam topics you need to study in order to attain Linux system administrator certification from the Linux Professional Institute.
- In "[Anatomy of the Linux file system](#)" (developerWorks, October 2007), learn why Linux is the Swiss Army knife of operating systems. Linux supports a large number of file systems, from journaling to clustering to cryptographic. Linux is a wonderful platform for using standard and more exotic file systems and also for developing file systems. This article explores the virtual file system (VFS) in the Linux kernel.
- In "[Anatomy of the Linux kernel](#)" (developerWorks, June 2007), delve into the Linux kernel, the core of a large and complex operating system. While huge, the kernel is well organized in terms of subsystems and layers. This article walks you through the general structure of the Linux kernel and points out major subsystems, core interfaces, and related resources to help you dig deeper.
- Take the "[Hacking the Linux 2.6 kernel, Part 1: Getting ready](#)" tutorial (developerWorks, July 2005) to learn about system and environment requirements, the best ways to acquire Linux source code, how to configure and boot your new kernel, and how to print messages during startup.
- Stay current with [developerWorks technical events and webcasts](#).
- In the [developerWorks Architecture zone](#), get the resources you need to advance your skills in the architecture arena.
- In the [developerWorks Linux zone](#), find more resources for Linux developers, and scan our [most popular articles and tutorials](#).
- Browse the [technology bookstore](#) for books on these and other technical topics.

Get products and technologies

- Download [IBM product evaluation versions](#) and get your hands on application development tools and middleware products from DB2®, Lotus®, Rational®, Tivoli®, and WebSphere®.
- [Order the software evaluation kit for Linux](#), a two-DVD set containing the latest IBM trial software for Linux from DB2, Lotus, Rational, Tivoli, and WebSphere.
- With [IBM trial software](#), available for download directly from developerWorks, build your next development project on Linux.

Discuss

- Get involved in the [developerWorks community](#) through blogs, forums, podcasts, and spaces.
- Check the [Linux tech support forum](#), for help with IBM trial products for Linux on x86 systems. Your host, Ian Shields, a Senior Programmer on the developerWorks staff, will also help you find other resources.

About the author

Stephen B. Morris

Stephen Morris is the CTO of Omev Communications in Ireland. For the past 20 years, Stephen has worked for some of the world's largest networking companies on a wide range of software projects, including J2EE/J2SE-based network management systems, billing applications, porting and developing SNMP entities, network device technologies, and GSM mobile networking applications. He is the author of *[Network Management, MIBs and MPLS: Principles, Design and Implementation](#)* (Prentice Hall PTR, 2003).

Trademarks

IBM, the IBM logo, ibm.com, DB2, developerWorks, Lotus, Rational, Tivoli, and WebSphere are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.