TAL GARFINKEL AND
ANDREW WARFIELD

what virtualization can do for security

Tal Garfinkel is part of the advanced development group at VMware and is currently on leave from Stanford University, where he plans to submit his Ph.D. thesis any day now.

talg@vmware.com

Andrew Warfield has completed his Ph.D. at the University of Cambridge and now divides his time between XenSource and an Adjunct Professor position at the University of British Columbia. He lives in Vancouver.

andrew.warfield@xensource.com

VIRTUAL MACHINE (VM) TECHNOLOGY is rapidly gaining acceptance as a fundamental building block in enterprise data centers. It is most known for improving efficiency and ease of management. However, it also provides a compelling approach to enhancing system security, offering new ways to rearchitect today's systems and opening the door for a wide range of future security technologies.

Virtualization emerged as a technique for managing mainframes in the 1960s. In the late 1990s, it was rediscovered in the midst of a perfect storm. Widespread adoption of IT infrastructure based on inexpensive commodity PCs led to explosive growth in the number of machines in enterprise environments. Concurrently, intense growth in the commodity software industry left a legacy of large, feature-rich, and complex applications and operating systems—so complex, in fact, that the best practice for managing and securing them was (and still is) commonly held to be isolating each server application to its own, incredibly underutilized host.

Remedying this inefficiency through server consolidation is perhaps the most well known use of virtualization. More recently the benefits of virtualization for management, such as simplifying provisioning and allowing hardware upgrades without incurring downtime, are becoming equally well known. Another property of this technology that has received less attention is the benefits it can provide for security.

Our objective is to provide readers with a better sense of what virtualization can contribute in this area. We begin by looking at the basic security benefits VMs can provide today (e.g., its power as a mechanism for isolation). We then survey some of the emerging security technologies supported by virtualization that we may see in the years ahead.

Virtual Machines for Isolation

The oldest and simplest path to enhancing security with VMs is by separating multiple applications on a single OS into multiple VMs, with each application in its own VM. As with application sandboxes, jails, etc., this contains the damage an exploited application can inflict on other services. This works equally well in situations with stand-alone applications (e.g., protecting a DNS or email server from a compromised Web server) and for services

consisting of a more complicated aggregate of middleware, such as an e-commerce application with a Web server front end, back-end logic for dealing with user transactions, and a database—each of which could be run in its own VM.

This isolation presents two major benefits that we discuss in detail throughout this article. First, by virtue of running in separate virtual machines, applications are obviously much less vulnerable to compromises that start in other applications; that is, the result of a system compromise is better *confined* in a correctly configured virtualized environment. Second, many of the security mechanisms that we have today are best applied at a host granularity. The *encapsulation* afforded by more single-purpose VM-based environments allows much stricter security policies to be applied at a (virtual) host granularity.

MAKING MACHINE-LEVEL ISOLATION UBIQUITOUS

Isolating applications on their own machine to contain a compromise (e.g., in a company's DMZ) has long been considered best practice. By reducing the physical and management costs of whole system isolation, this practice can be applied far more ubiquitously. Of course, replacing physical isolation with virtualization does come at some cost in assurance, and there are some situations (e.g., separating machines in the DMZ from those inside the firewall) where the additional security this affords makes sense. However, in the common case, the benefits of virtual-machine-monitor-based (VMM-based) isolation outweigh the resulting loss in assurance. A hybrid approach seems most sensible—for example, relying on a VMM to compartmentalize applications on either side of the firewall, while putting a physical gap between these two sides.

Running several virtual machines on a single platform, each hosting a single application, incurs only nominal overhead over running these applications all on the same OS. Depending on the applications and platform configuration, this overhead can potentially even be less, as on multicore platforms, and virtualization can sometimes make it easier to reduce resource contention and expose latent concurrency.

Similarly, application setup times on a virtualized platform are often less than on a traditional platform. Uniform virtual hardware means only a single "base" operating system image is required, which can then be specialized on a per-application basis. Increasingly, administrators will simply have totally configured application VMs on hand in "template" form, where a new VM (e.g., a mail server) can simply be instantiated and deployed as needed. Also gaining popularity is the use of a prebuilt virtual appliance (i.e., an application that has been bundled along with an operating system in a VM by the software vendor).

WHY WHOLE MACHINE ISOLATION?

Using virtual machines for isolation frequently elicits the question, "Why use something so coarse-grained?" BSD jails, application sandboxes such as Systrace or AppArmor, and fine-grained OS-level access controls such as SELinux have long been available. The simple answer is that VMs offer potential benefits over these solutions by way of simplicity and assurance, in terms both of implementation and of configuration.

Correctly utilizing OS-level isolation often requires a deep understanding of both OS semantics and the behavior of the particular application(s) being

secured (e.g., file system usage), to securely configure a system and to debug reliability problems associated with configuration errors. In contrast, the basic abstraction that VMs provide is familiar and easy to understand—at most, an administrator must configure a firewall for containment or set up a network file system for sharing. Beyond these tasks, no OS-specific knowledge is required. Further, securing an OS against exploits is a demonstrably difficult problem, owing to their many functions and broad attack surfaces. The narrow interface that a virtual machine presents offers a comparatively harder target for attackers, thus potentially providing higher assurance isolation.

Of course, virtualization is not a substitute for OS-level mechanisms. If an application is compromised, using a defense-in-depth strategy to contain that compromise only improves protection. Thus, if an attacker must break out of a jail, sandbox, or other OS-based protection mechanisms before attempting to overcome the isolation imposed by the VM, all the better. Further, a VMM-based approach is not always suitable: VMMs excel at isolation, but in situations where there is a great deal of controlled sharing, an OS-level solution may be more suitable.

SYSTEM SPECIALIZATION

Beyond reducing the possibility that a compromise in one application can spread to another, putting each application in its own virtual machine conveys a variety of benefits. First, it eliminates complexity inside the guest, making access controls and other hardening measures easier to set up and allowing a thinner OS to be used, reducing the size of the application's trusted computing base. Next, each VM's attack surface can be reduced, as each VM only requires the interfaces (network ports, RPC daemons, etc.) a single application needs to be enabled.

This approach converges on a virtual appliance model. Just as on a hardware appliance, in a virtual appliance the operating system is specifically tailored from the ground up for the application that it's running. An example of the extreme end of this spectrum is BEA Systems Liquid VM [1], a custom operating system tailored specifically to run its JRocket JVM and middleware platform. Other vendors offer a middle ground, providing custom versions of existing operating systems specifically tailored to the needs of appliances. Both approaches offer the possibility of a smaller trusted computing base and reduced attack surface for applications. Additionally, this allows securing applications to shift from system administrators to ISVs, who can often use their greater knowledge of an application to employ more complex hardening measures.

DATA CENTER PARTITIONING

Technologies for segmenting data centers, such as VLANS, have become increasingly compelling, as they provide an intuitive model of separation. Using virtual machines, this model of separation can be pushed onto the end host, making network-based partitioning an end-to-end proposition. For example, prior to its adoption of virtualization, the NSA used to use physically separate machines to access networks with data at different levels of classification. This provides excellent isolation, but of course an architecture like this is expensive and unwieldy. This prompted the NSA's move to their Nettop architecture [5], which instead used virtual machines for isolation on the same physical host.

With the cost of such an end-to-end solution reduced, such an architecture becomes feasible for normal businesses. One common pattern is to partition a user's desktop into two parts: One partition has access to all company internal resources, while the other can communicate with the outside world using Web browsers, IM, etc., with all the ensuing perils this entails. This pattern can also be applied inside a company, as with the NSA example. Organizational units such as marketing, engineering, and sales may be configured to have strongly isolated virtual resources, compartmentalizing these organizational roles in the face of compromise.

Virtualization facilitates another interesting sort of partitioning. As backup, logging and monitoring, remote display, and other functionality migrate out of the application VM and into separate protection domains on the virtualization layer, a natural separation occurs between the management plane (with all the infrastructure services that are important for security and management) and the application plane (with services actually used by end users). This again naturally supports an end-to-end separation of duties, with management functionality living on a separate set of virtual machines, separate network, etc., from running services—again, making it easier to gain confidence in the integrity of this part of a data center, even in the face of application compromise.

Virtual Machines for Fine-Grained Protection

Another unfortunate consequence of the rapid growth in size and complexity in commodity operating systems has been a predictable reduction in our ability to have faith that these systems cannot be compromised.

Virtualization can help us address this problem by adding additional protection for code running inside virtual machines. Virtualization may be used both to provide a higher-assurance protection layer than is afforded by the guest OS and to offer an additional degree of defense in depth, for example, preventing data from being leaked from a host, even if the guest OS is compromised.

QUIS CUSTODIET IPSOS CUSTODES?

The question, "Quis custodiet ipsos custodes?" ("Who will guard the guards?")is an important one for modern commodity operating systems. Today's antivirus and host-based intrusion detection and prevention systems are stuck with a difficult chicken-and-egg problem. Much of what they are seeking to detect is OS compromise, especially in light of the growing popularity of kernel rootkits. However, once the kernel has been compromised, the detection tools themselves are left unprotected. Thus, the arms race between attackers and defenders rapidly devolves into a complex game of core wars. This state of affairs has been institutionalized with the introduction of PatchGuard in Microsoft Windows Vista, and, unsurprisingly, the arms race to disable this mechanism is already in full swing [6].

Virtualization provides a way out of this situation by allowing the "guards" to be run in an entirely different protection domain, *outside* the influence of the OS that they are protecting. This can be accomplished in a number of ways. For example, a kernel rootkit detector could be protected by the VMM in situ (i.e., in the same address space as the operating system it is protecting) by preventing modifications to detector code and ensuring that it is executed with a specified frequency. Alternatively, the detector could be moved outside of the guest OS entirely and run in a totally separate VM [2]. Both approaches offer a simple and clear advantage over today's systems, where

intrusion detection and prevention systems must paradoxically rely for protection on the OS they are trying to protect.

GETTING BETTER HIPS

VMs can enhance monitoring and enforcement capabilities in AV and HIPS by allowing efficient interposition on hardware events in the guest OS. x86 virtualization has long played technically exciting tricks involving the virtualization of the hardware memory management unit (MMU); MMU virtualization is required to prevent VMs from accessing each other's memory, and providing high-performance implementations is one of the major challenges of effective x86 virtualization. Leveraging this capability for security, a system can enforce exactly what code should be permitted to execute on a host [4] or, alternately, perform up-to-the-moment detection of malware, offering superior detection capabilities when compared with the file-scanning approaches of today's AV systems. Interposing on devices offers many other possibilities: for example, preventing malware from being written to disk, scanning a USB disk when it is plugged into the machine, or filtering network traffic.

LOCKING DOWN DATA

Because it acts as a small privileged kernel that runs *under* an existing guest OS, the virtual machine monitor can impose nearly arbitrary protection policies inside the guest. Potential applications range from protecting key data in an SSL stack, to preventing sensitive documents from being leaked from a VM, to enforcing fine-grained information flow policies. Although few technologies offering this capability have been deployed to date, the possibilities are rich and promising.

Logging and Snapshots: More of What You Want and Less of What You Don't

With the shift from physical to virtual machines, what was once hardware state, locked away on devices, becomes entirely software-based. This change allows us to rethink how we manage and take advantage of system state.

DOWN WITH HARD STATE

One of the biggest challenges in managing security within an installed base of software is keeping it secure as it ages. Users and administrators generally accept that a freshly installed application running on a freshly installed OS usually works just fine. However, as other software is installed and uninstalled, applications are run, and time passes, confidence in the safe and correct configuration of a system diminishes. Moreover, once a system has been compromised—or in many cases even *may have been* compromised—the only way to restore confidence is to return to a "clean" state, which is often achieved by completely reinstalling the OS, or at least reimaging the system.

VMs provide a very compelling way to deal with this sort of complexity. They can trivially checkpoint and return to points in history, allowing the unknown permutations to a system to be dropped. Further, we may specify a positive filter on what changes we *do* want to keep, perhaps preserving the contents of a bookmarks file while reverting a Web browser VM to a freshly installed state with each restart. In this sense, VMs allow us to treat millions

of lines of complex software as a very coarse-grained transformation on a small and confined set of data. Any changes made by the VM to internal state other than the interested set can simply be dropped and reverted to a fresh install, providing much stronger levels of confidence in the system.

FORENSICS AND REMEDIATION IN THE LOG-STRUCTURED DATA CENTER

Carrying this notion of logging and reverting the state of a system one step further allows us to consider simply recording to a log everything any VM in a data center ever does. Considerable research work on virtualization has explored the notion of logging and replaying a VM's execution at a fine granularity, using techniques as fine-grained as cycle-accurate instruction replay or as loose as logging network traffic and periodic VM checkpoints. Regardless of the technique used, it's quite reasonable to imagine production systems that include a facility to return to arbitrary points in the history of their execution.

This detailed logging has the potential to solve the very challenging task of separating good from bad in an exploited system. In normal situations, once a system is found to have been compromised the best that an administrator can possibly do is to revert to a backup and attempt to comb through the file system, searching for changes and determining which should be preserved. More often, this represents an unrealistic effort and the post-backup changes are simply lost.

Having a detailed log of a system's execution allows a compromise to be analyzed retrospectively. Detailed analysis tools may be built and run against the system's execution log and attempts can be made to isolate malicious changes, improving our ability to recover data. As a gedanken experiment for what this means in restoring compromised systems, imagine the ability to rewind the execution of a system to just before the point that it was attacked and there insert a firewall rule that refuses to admit the exploit traffic. The system would then play forward without maliciousness, ideally preserving a considerable amount of "good" activity.

Logging and analysis similarly provides the ability to more speedily understand malicious software and attacks on systems, because it allows forensic analysis to be run both forward and backward in time to diagnose the root of a system compromise and determine what malicious activities took place.

The Managed Desktop

The desktop is an inevitable final frontier for virtualization. Its support for a wide range of hardware and the latency-sensitive nature of its applications have posed a higher technical barrier for desktop entry than for the server space. However, from a security perspective the rewards are high: Desktop systems are exactly where many of the advantages that we have described here are most desirable.

In addition to richer policies, virtualization of the desktop allows security policies to be applied uniformly across the enterprise. Firewall policies, network quarantine, monitoring, and the like can be applied whether the user is at his or her desk or in the data center, regardless of the integrity of the guest OS.

Virtualization also provides the ability to strictly limit the actual hardware to which applications contained in VMs have access. For example, the discreet use of a USB memory stick to transfer applications or data off of a machine

may be disallowed, a VM's access to the network may be very tightly controlled, and VM data stored on disk can be automatically encrypted.

A Brave New World

We believe the benefits of virtualization for efficiency, platform flexibility, and ease of management alone will make it ubiquitous in enterprise data centers. As with provisioning and management before it, the benefits of virtualized platforms for improving security will take time to be realized. Part of the current unrealized potential is a lack of deep operational experience in modern IT environments, widespread understanding of how this technology can be leveraged, and tools that help facilitate best practices.

Our enthusiastic endorsement of virtualization's potential should be tempered with the observation that the flexibility and power that make it such a boon for system management also give rise to a variety of new security challenges [3]. Coping with these will require a varied combination of new technologies and best practices. As always, the responsibility for ensuring the secure adoption of virtualized platforms will lay in the hands of both platform vendors and those deploying them.

REFERENCES

[1] BEA Systems, Inc., "Stop Worrying About Computing Capacity—and Start Making the Most of It," 2007: http://www.bea.com/content/news_events/white_papers/BEA_Virtualization_wp.pdf.

[2] T. Garfinkel and M. Rosenblum, "A Virtual Machine Introspection Based Architecture for Intrusion Detection," *Proceedings of the Network and Distributed Systems Security Symposium,* February 2003.

[3] T. Garfinkel and M. Rosenblum, "When Virtual Is Harder Than Real: Security Challenges in Virtual Machine Based Computing Environments," *Proceedings of the 10th Workshop on Hot Topics in Operating Systems (HotOS X)*, May 2005.

[4] L. Litty and D. Lie, "Manitou: A Layer-Below Approach to Fighting Malware," *Proceedings of the Workshop on Architectural and System Support for Improving Software Dependability (ASID)*, October 2006.

[5] R. Meushaw and D. Simard, "NetTop: Commercial Technology in High Assurance Applications," 2000: http://www.vmware.com/pdf/TechTrendNotes.pdf.

[6] Uninformed, "PatchGuard Reloaded. A Brief Analysis of PatchGuard Version 3," September 2007: http://uninformed.org/index.cgi?v=8&a=5.