Can virtualization empower the creators of malware16 to carry out their mischievous activities with

impunity and with minimal danger of being detected? How difficult is it to implement such a system?

What are the means to prevent this type of malware to be put in place? The answers to these questions

are discussed in this section.

It is well understood that in a layered structure a defense mechanism at some layer can be disabled

by malware running a layer below it. Thus, the winner in the continuous struggle between the attackers

and the defenders of a computing system is the one in control of the lowest layer of the software stack

– the one that controls the hardware (see Figure 5.10).

Recall that a VMM allows a guest operating system to run on virtual hardware. The VMM offers to

the guest operating systems a hardware abstraction and mediates its access to the physical hardware.We

argued that a VMM is simpler and more compact than a traditional operating system; thus, it is more

secure. But what if the VMM itself is forced to run above another software layer so that it is prevented

from exercising direct control of the physical hardware?

A 2006 paper [194] argues that it is feasible to insert a “rogue VMM” between the physical hardware

and an operating system. Such a rogue VMM is called a virtual machine-based rootkit (VMBR). The

term rootkit refers to malware with privileged access to a system. The name comes from root, themost

privileged account on a Unix system, and kit, a set of software components.

It is also feasible to insert the VMBR between the physical hardware and a “legitimate VMM.” As

a virtual machine running under a legitimate VMM sees virtual hardware, the guest OS will not notice

any change of the environment; so the only trick is to present the legitimate VMM with a hardware

abstraction, rather than allow it to run on the physical hardware.

Before we address the question of how such an insertion is possible, we should point out that in this

approach the malware runs either inside a VMM or with the support of a VMM; but a VMM is a very potent engine for the malware. It prevents the software of the guest operating system or the application

from detecting malicious activities. A VMBR can record key strokes, system state, data buffers sent to

or received from the network, and data to be written to or read from the disk with impunity; moreover,

it can change any data at will.

The only way for a VMBR to take control of a system is to modify the boot sequence and to first load

the malware and only then load the legitimate VMM or the operating system. This is only possible if

the attacker has root privileges. Once the VMBR is loaded it must also store its image on the persistent

storage.

The VMBR can enable a separate malicious OS to run surreptitiously and make this malicious

OS invisible to the guest OS and to the application running under it. Under the protection of the

VMBR, the malicious OS could (i) observe the data, the events, or the state of the target system;

(ii) run services such as spam relays or distributed denial-of-service attacks; or (iii) interfere with the

application.

Proof-of-concept VMBRs to subvert Windows XP and Linux and several services based on this

platform are described in [194]. We should stress that modifying the boot sequence is by no means an

easy task, and once an attacker has root privileges he or she is in total control of a system.