論文の内容の要旨

論文題目

VMM-BASED SYSTEMS FOR ENHANCING APPLICATION SECURITY

(VMM を利用したアプリケーションプログラムの安全性を向上させるシステム)

氏名　尾　上　　浩　一

This thesis presents three security systems for controlling the behavior of application programs and protecting data relevant to those programs by using a virtual machine monitor (VMM), more precisely, from outside virtual machines (VMs). The goal of this work is to design and implement security systems that simultaneously meet the following two requirements: (i) providing fine-grained control and protection mechanisms, and (ii) making subversion and evasion of these mechanisms difficult.

Security systems running on the OS or application level (e.g., sandboxing systems and anti-virus tools) can control the behavior of untrusted programs and prevent attackers from leaking and tampering with security-sensitive data. However, these security systems are potentially vulnerable because they run in the same execution space as untrusted programs. If an attacker can hijack OS kernels and privileged programs, it is not hard for the attacker to compromise security systems residing in the same execution space.

A VMM virtualizes execution of separate computing environments for each VM and allows users to run any programs, including OS kernels, inside their VMs. In terms of security, a VMM has two advantages. First, it can isolate the control and protection mechanisms of security systems from untrusted VMs. Second, it can control accesses to physical resources (e.g., physical memory and disks) at a higher privilege level than that of the OS kernels and application programs running inside untrusted VMs. However, there are two key challenges in improving application program security from outside VMs. The first challenge is to identify and manipulate in-VM states in OS-level semantics from the hardware-level states that a VMM can observe. The second challenge is to enable security systems to intercept events that a VMM cannot capture.

This thesis proposes three types of VMM-based security systems with different security concerns. The proposed systems have control and protection functionalities at the application program (target program) granularity while maintaining the above two VMM security properties.

The first proposed system, ShadowVox, controls the system call execution of target programs. The security concern is to control the behavior of target programs in user mode. ShadowVox can control untrusted VMs at process and system-call granularity from outside those VMs. To overcome the first challenge noted above, all three of the proposed security systems provide a technique for obtaining OS-level states by using information on the process management and system calls of OS kernels. To overcome the second challenge, the three systems incorporate a technique for allowing a VMM to intercept an arbitrary processor instruction. Experimental

evaluation demonstrated that ShadowVox could control several types of application programs on different processor architectures. In addition, the system's effectiveness against an attack from a compromised program was demonstrated, and the overhead introduced by the system was measured.

The second proposed system, Shadowall, protects memory and virtual disk data involved with target programs. The security concern is to protect target program data in the user memory space and on virtual disks. Shadowall prevents compromised OS kernels and privileged programs from divulging or corrupting such data related to target programs running in the same execution space. Experimental results demonstrated Shadowall's effectiveness against malicious operations in existing application programs, and the performance penalties incurred by the system were measured.

Finally, the third proposed system, ShadowXeck, controls the behavior of OS kernels running on VMs. The security concern is to control the behavior of target programs in kernel mode. ShadowXeck addresses three problems with existing anti-malware systems: restriction on the use of kernel extensions, bypassing of system functionalities, and significant performance degradation. ShadowXeck does not conservatively impose any restriction on kernel extensions, and it reduces the performance impact on untargeted programs. Experimental evaluation demonstrated ShadowXeck's effectiveness against attacks by actual kernel-level malware, and the runtime overhead imposed by the system was measured.