NPS-CS-06-014
ISI-TR-621
September 2006

Trustworthy Commodity Computation and Communication

SecureCore Technical Report

Preliminary Security Requirements for SecureCore Hardware

Thuy D. Nguyen, Timothy E. Levin, Cynthia E. Irvine,
Terry V. Benzel, and Ganesha Bhaskara

This material is based upon work supported by the National Science Foundation under Grant No. CNS-0430566 and CNS-0430598 with support from DARPA ATO. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation or of DARPA ATO.

Author Affiliations

Naval Postgraduate School:

Thuy D. Nguyen, Timothy E. Levin, and Cynthia E. Irvine

Center for Information Systems Security Studies and Research
Computer Science Department
Naval Postgraduate School
Monterey, California 93943

USC Information Sciences Institute:

Terry V. Benzel and Ganesha Bhaskara

Information Sciences Institute
University of Southern California
4676 Admiralty Way, Suite 1001
Marina del Rey, Ca 90292

Preliminary Security Requirements for SecureCore Hardware

Abstract: *This document describes a set of preliminary high level security requirements for the SecureCore hardware base (SCHW). A SecureCore (SC) component is anticipated to be a mobile networked device capable of operating in different modes with different levels of trust. To promote rapid user acceptability, it is essential that security features implemented in the SC architecture must minimize changes to existing application-level software. The SCHW security requirements are specified in terms of the following capabilities: hardware virtualization, protected processing environment, protected memory management, secure I/O channels, secure boot, secure system maintenance, concealed execution mode, trusted platform attestation and hardware isolation of security critical functions.*

I. Introduction

The purpose of this document is to describe a set of preliminary high level requirements that the SecureCore hardware base (SCHW) must satisfy. To provide a coherent view of the desired hardware features, the features described herein include those that are currently available ("*C*"), are part of the next generation ("*N*" e.g., Intel's VTx), and those proposed for the future ("*F*").[1] In regard to the latter, we are currently engaged in NDA-level discussions with Intel. This document assumes the reader is familiar with the SecureCore architecture provided in the SecureCore project description [1].

II. SCHW Architecture Overview

A SecureCore component is intended to be a mobile networked device capable of operating in different modes with different levels of trust. Security features implemented in the SecureCore architecture must minimize changes to existing application-level software to promote rapid user acceptability. The current trend in hardware-assisted virtualization technology focuses mainly on processor virtualization for consolidation of workstations with different OSs and for fail-over purposes [2]. Since many other components within the workstation platform (i.e., on or attached to the motherboard) affect its security operation, the SCHW architecture will address virtualization issues of the entire platform, not just the processor. Furthermore, the proposed hardware features will afford mobile platforms with virtualization technology that is currently only available in servers and high-end desktops.

The SecureCore platform must be able to provide the following capabilities.

[1] Due to the research nature of these requirements, their imposition, interactions and tradeoffs will require continuing examination, especially for those designated as future (*F*).

A. *Hardware virtualization*

This capability must support Type I Virtual Machine Monitor (VMM) software architecture that can host multiple unmodified virtual machines (VMs) [3]. For the SecureCore project, we are exploring an architecture in which the LPSK and SCSS layers form a Type I VMM; the VMs are the guest commercial operating systems and the native (SecureCore) operating system.

The virtualization mechanism must provide a way for the VMM to virtualize all system resources such as processor, memory (including cache and TLB), interrupts, and I/O devices. In particular, hardware support for VM scheduling, as well as task scheduling within a VM, is required. (*F*)

B. *Protected processing environment*

The processor architecture must provide the following:

1. Support for a partially ordered privilege domain architecture, with at least 4 VM privilege domains and 2 VMM privilege domains. The VM privilege domains are available to both the VMs and VMM whereas the VMM privilege domains are only available to the VMM. (*N*)
2. Support for distinct execution entities (viz., multitasking). (*C*)
3. A method to partition the modules of a task into different privilege domains. (*C*)
4. A method for the processor to transfer control directly to the LPSK when a non-privileged portion of a task attempts to execute a privileged instruction that might violate the security policy. (*C*)
5. A method for virtualizing the interrupts handling mechanism, including support for hardware-assisted virtualization of the interrupt controller chip. (*F*)
6. A method for restricting access to I/O devices. (*C*)

C. *Protected memory management*

The platform MMU must provide the following:

1. Support virtual memory. (*C*)
2. A method to implement distinct, protected per-VM and per-task address spaces. (*C*)
3. A method to enforce fine-grained memory protection policy. Access to memory will be based on hardware attributes such as privilege levels and access modes.(*C*)
4. A method to restrict per-device DMA access. (*N*)
5. A method to prevent information leakage through memory areas (system memory, cache, and Flash) between VMs, including timing issues. (*N*)

D. *Secure I/O channels*

The platform must provide mechanisms to protect the confidentiality and integrity of communications between the I/O devices and other components on the platform. Minimally,

secure channels must be provided for the following I/O devices: keyboard, console display device, USB devices, and network interface cards. (*N*)

E. Secure boot

The platform must provide the following:

1. A mechanism to ensure the integrity of the BIOS code and data that can be used by a ratcheting bootstrap scheme [4], e.g., a "secure root of trust" such as might be provided by the SP extension discussed in Section 2.7. (*F*)
2. A privileged mechanism to disable selected privileged operations as the operating system initialization progresses, which could not be reset until the processor was re-initialized.(*F*) This feature would provide hardware protection against modification of structures that should not change after initialization – such as the x86 LDT for a static separation kernel.

F. Secure system maintenance

Most modern platforms support a special execution mode, i.e., system maintenance mode, which can bypass all protections on platform resources (e.g., memory, I/O devices). A typical use of this mode is to implement various power saving mechanisms (e.g., powering down unused devices, putting the system to sleep or hibernation). The SC platform must provide mechanisms to control the invocation of this special mode as well as to protect platform resources and prevent information leakage between VMs when it is invoked. (*F*)

G. Concealed execution mode

The platform must support the integration of the Secret Protected (SP) architecture to provide a safe environment for per-user cryptographic processing [5]. The original SP design must be enhanced to support virtualization to prevent covert channels. The SecureCore "SP virtualization" technical report describes a set of proposed enhancements to virtualize the SP component [6]. (*F*) Additionally, we are working with the SecureCore East team on an SP extension to support high integrity enterprise keys, which will support the "transient trust" functions of the SP architecture.

H. Trusted platform attestation

The platform must support the integration of a trustworthy hardware attestation component to provide the ability to attest, at a minimum, the presence of the VMM. (*N*)

I. Hardware isolation of security critical functions

We plan to work with hardware vendors to extend the current multicore architecture to isolate

processing units such that one or more units can be used securely for privileged mode processing (i.e., kernel functions), and for dedicated engines such as the SP and TPM. This isolation feature will also facilitate concurrent execution of VMs, i.e., multiple VMs can be isolated by hardware and also execute at the same time. (*F*)

III. Summary

This document presents a preliminary set of high level requirements for the SCHW architecture. Table 1 characterizes the requirements with respect to their technological maturity.

Table 1. SecureCore Hardware Requirements

Categories	Requirements
Currently available	II.B.2, II.B.3, II.B.4, II.B.6, II.C.1, II.C.2, II.C.3
Next generation	II.B.1, II.C.4, II.C.5, II.D, II.H
Future	II.A, II.B.5, II.E.1, II.E.2, II.F, II.G, II.I

References and Bibliography

[1] Cynthia Irvine, Terry Benzel, et. al., National Science Foundation Poster - CyberTrust meeting, Sept 26, 2005

[2] Intel Corp, "Intel® Virtualization Technology Specification for the IA-32 Intel® Architecture", 2005.

[3] R. Goldberg, "Architectural Principles for Virtual Computer Systems," Ph.D. thesis, Harvard University, Cambridge, MA, 1972.

[4] W. A. Arbaugh, D. J. Farber, and J. M. Smith, "A Secure and Reliable Bootstrap Architecture," *Proceedings of the 1997 IEEE Symposium on Security and Privacy*, pp. 65-71, May 1997.

[5] Ruby B. Lee, Peter C. S. Kwan, John Patrick McGregor, Jeffrey Dwoskin, and Zhenghong Wang, "Architecture for Protecting Critical Secrets in Microprocessors", *Proceedings of the 32nd International Symposium on Computer Architecture (ISCA 2005)*, pp. 2-13, June 2005.

[6] Ganesha Bhaskara, Timothy E. Levin, Thuy D. Nguyen, Terry V. Benzel, Cynthia E. Irvine, Paul C. Clark, "Integration of User Specific Hardware for SecureCore Cryptographic Services," NPS-CS-06-012, Naval Postgraduate School, Monterey, California, July 2006. (Also available as USC/ISI technical report TR-2006-619.)