NIST Special Publication 800-147B

BIOS Protection Guidelines for Servers

Andrew Regenscheid

http://dx.doi.org/10.6028/NIST.SP.800-147B

C O M P U T E R S E C U R I T Y

National Institute of Standards and Technology
U.S. Department of Commerce

NIST Special Publication 800-147B

BIOS Protection Guidelines for Servers

Andrew Regenscheid
Computer Security Division
Information Technology Laboratory

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.800-147B

August 2014

U.S. Department of Commerce
Penny Pritzker, Secretary

National Institute of Standards and Technology
Willie May, Acting Under Secretary of Commerce for Standards and Technology and Acting Director

Authority

This publication has been developed by NIST to further its statutory responsibilities under the Federal Information Security Management Act (FISMA), Public Law (P.L.) 107-347. NIST is responsible for developing information security standards and guidelines, including minimum requirements for Federal information systems, but such standards and guidelines shall not apply to national security systems without the express approval of appropriate Federal officials exercising policy authority over such systems. This guideline is consistent with the requirements of the Office of Management and Budget (OMB) Circular A-130, Section 8b(3), *Securing Agency Information Systems*, as analyzed in Circular A-130, Appendix IV: *Analysis of Key Sections*. Supplemental information is provided in Circular A-130, Appendix III, *Security of Federal Automated Information Resources*.

Nothing in this publication should be taken to contradict the standards and guidelines made mandatory and binding on Federal agencies by the Secretary of Commerce under statutory authority. Nor should these guidelines be interpreted as altering or superseding the existing authorities of the Secretary of Commerce, Director of the OMB, or any other Federal official. This publication may be used by nongovernmental organizations on a voluntary basis and is not subject to copyright in the United States. Attribution would, however, be appreciated by NIST.

National Institute of Standards and Technology Special Publication 800-147B
Natl. Inst. Stand. Technol. Spec. Publ. 800-147B, 32 pages (August 2014)
http://dx.doi.org/10.6028/NIST.SP.800-147B
CODEN: NSPUE2

This publication is available free of charge from:
http://dx.doi.org/10.6028/NIST.SP.800-147B

Comments on this publication may be submitted to:

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930
Email: 800-147comments@nist.gov

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analyses to advance the development and productive use of information technology. ITL's responsibilities include the development of management, administrative, technical, and physical standards and guidelines for the cost-effective security and privacy of other than national security-related information in Federal information systems. The Special Publication 800-series reports on ITL's research, guidelines, and outreach efforts in information system security, and its collaborative activities with industry, government, and academic organizations.

Abstract

Modern computers rely on fundamental system firmware, commonly known as the Basic Input/Output System (BIOS), to facilitate the hardware initialization process and transition control to the hypervisor or operating system. Unauthorized modification of BIOS firmware by malicious software constitutes a significant threat because of the BIOS's unique and privileged position within the PC architecture. The guidelines in this document include requirements on servers to mitigate the execution of malicious or corrupt BIOS code. They apply to BIOS firmware stored in the BIOS flash, including the BIOS code, the cryptographic keys that are part of the Root of Trust for Update, and static BIOS data. This guide is intended to provide server platform vendors with recommendations and guidelines for a secure BIOS update process.

Keywords

Basic Input/Output System (BIOS); information security; patch management; server security; firmware; Root of Trust; Root of Trust for Update

Acknowledgements

The author, Andrew Regenscheid of the National Institute of Standards and Technology (NIST), wishes to thank his colleagues who reviewed drafts of this document and contributed to its technical content. In particular, NIST appreciates the contributions from experts from industry and government who helped guide this work. These experts included Gary Simpson from AMD; Anthony Grieco, Bill Jacobs and Chirag Schroff from Cisco; Mukund Khatri and Frank Molsberry from Dell; Scott Dunham and Charles Palmer from IBM; and Lelia Barlow, Bob Hale and David Riss from Intel Corporation.

NIST would also like to acknowledge and thank Mike Boyle, Tom Broström, Bob Clemons, and Patrick Hagerty from the National Security Agency, who also provided substantial contributions to this document.

Table of Contents

Executive Summary **1**

1. Introduction **1-1**

1.1 Purpose and Scope 1-1
1.2 Audience 1-1
1.3 Document Structure 1-2

2. Background **2-1**

2.1 System BIOS 2-1
2.2 Server Architectures 2-1
2.3 System BIOS Update Mechanisms 2-2
2.4 Threats to the System BIOS 2-3
2.5 Root of Trust for Update 2-3

3. BIOS Security Principles **3-1**

3.1 BIOS Update Authentication 3-1
3.2 Secure Local Update 3-1
3.3 Firmware Integrity Protection 3-2
3.4 Non-Bypassability 3-2

4. Security Guidelines by Update Mechanism **4-1**

4.1 Update Mechanism 1: Secure BIOS Update at Anytime 4-1
4.2 Update Mechanism 2: Secure BIOS Update at Reboot 4-2
4.3 Update Mechanism 3: Secure BIOS Update Requiring Verification at Boot 4-2

5. Guidelines for Service Processors **5-1**

5.1 Service Processor as a Root of Trust 5-1
5.2 Non-Bypassability of BIOS Protections by Service Processor 5-1

List of Appendices

Appendix A— Summary of Requirements **A-1**

Appendix B— Example of Update Mechanism 1 ..... **B-1**

Appendix C— Example of Update Mechanism 2 ..... **C-1**

Appendix D— Example of Update Mechanism 3 ..... **D-1**

Appendix E— Glossary **E-1**

Appendix F— Acronyms and Abbreviations **F-1**

Appendix G— References **G-1**

Executive Summary

Modern computers rely on fundamental system firmware, commonly known as the Basic Input/Output System (BIOS), to facilitate the hardware initialization process and transition control to the hypervisor or operating system. The BIOS is typically developed by both original equipment manufacturers (OEMs) and independent BIOS vendors, and is distributed to end-users by motherboard or computer manufacturers. Manufacturers frequently update system firmware to fix bugs, patch vulnerabilities, and support new hardware. This document is the second in a series of publications on BIOS protections. The first document, SP800-147, *BIOS Protection Guidelines*, was released in April 2011 and provides guidelines for desktop and laptop systems deployed in enterprise environments.

Unauthorized modification of BIOS firmware by malicious software constitutes a significant threat because of the BIOS's unique and privileged position within modern computing architectures. Malicious BIOS modification could be part of a sophisticated, targeted attack on an organization—either a permanent denial of service or a persistent malware presence.

This document covers BIOS protections for basic, managed and blade servers. These types of servers often contain Service Processors—specialized microcontrollers that provide administrators with an interface to manage the host server. Servers, particularly those with Service Processors, may implement multiple BIOS update mechanisms. Servers implementing a single BIOS update mechanism may implement the guidelines in SP800-147, if the update mechanism is sufficiently similar to those in PC client systems, or they may implement the guidelines in this document.

The security guidelines in this publication do not attempt to prevent installation of an inauthentic BIOS through the supply chain, by physical replacement of the BIOS chip, or through secure local update procedures.

Security guidelines are specified for four system BIOS security features:

- **Authenticated BIOS update mechanisms**, where digital signatures prevent the execution of BIOS update images that are not authentic.
- An optional **secure local update mechanism**, which requires that an administrator be physically present at the machine in order to install BIOS images without signature verification.
- **Firmware integrity protections**, to prevent unintended or malicious modification of the BIOS outside the authenticated BIOS update process.
- **Non-bypassability** features, to ensure that there are no mechanisms that allow the main processor or any other system component to bypass the BIOS protections.

This document also provides additional information and recommendations for implementing BIOS protections using three BIOS update mechanisms that are commonly implemented in servers. This material is intended to help implementers design systems that meet the security requirements in this publication.

Service Processors are critical management components in many modern server designs. They are responsible for various management features, depending on the implementation of the system. Some, but not all, Service Processors are able to update the system BIOS. This document describes the possible roles of Service Processors in the system BIOS update process, and describes how the security guidelines apply to systems containing these components.

1. Introduction

1.1 Purpose and Scope

This document provides guidelines for BIOS protections in server-class systems. It is the second in a series of publications on BIOS protections. The first document, SP800-147, *BIOS Protection Guidelines*, was released in April 2011 and provides guidelines for desktop and laptop systems deployed in enterprise environments.

Unauthorized modification of BIOS firmware by malicious software constitutes a significant threat because of the BIOS's unique and privileged position within modern computer system architectures. Malicious BIOS modification could be part of a sophisticated, targeted attack on an organization—either a permanent denial of service (if the BIOS is corrupted) or a persistent malware presence (if the BIOS is implanted with malware).

The three core principles of BIOS protection outlined in NIST SP 800-147 for client systems—authenticated firmware updates, integrity protection, and non-bypassability of protection mechanisms—apply to server-class machines. However, the architectural and operational complexity in servers due to the need to remotely manage them makes it more difficult to implement BIOS security protections in the same manner as clients; the core reason for the increased difficulty is that servers typically possess multiple BIOS update mechanisms. In addition, some servers have one or more Service Processors (SPs). SPs perform various management functions for the hosts, which may include BIOS updates. This introduces the SP as a security-critical component, and thus this document includes guidelines for SPs that are part of the BIOS update process.

Boot firmware refers to software or firmware used to facilitate the hardware initialization process and transition control to the hypervisor or operating system. This publication uses the term BIOS as a generic, recognized term for system boot firmware in servers, including, but not limited to, boot firmware based on conventional BIOS, Extensible Firmware Interface (EFI) BIOS, and Unified Extensible Firmware Interface (UEFI) BIOS. The guidelines in this document include requirements on servers to mitigate the execution of malicious or corrupt BIOS code. They apply to BIOS firmware stored in the BIOS flash memory, including the BIOS code, the cryptographic keys that are part of the Root of Trust for Update (RTU), and static BIOS data. Device firmware that is stored with the system BIOS firmware and is updated by the same mechanism is considered part of the BIOS for the purposes of this document (e.g., Option ROMs embedded in the BIOS flash). These guidelines do not apply to other device firmware or Option ROMs stored elsewhere in a server, such as on an option card itself.

This guide is intended to provide server platform vendors with recommendations and guidelines for a secure BIOS update process. System administrators should consult Section 3.2 of NIST SP 800-147 for recommended best practices for managing BIOS in client and server systems.

1.2 Audience

The intended audience for this document includes BIOS and platform vendors of server-class systems, and information system security professionals who are responsible for managing the servers' security. The material may also be of use when developing enterprise-wide procurement strategies and deployment.

The material in this document is technically oriented, and it is assumed that readers have at least a basic understanding of system and network security. The document provides background information to help

such readers understand the topics that are discussed. Readers are encouraged to take advantage of other resources (including those listed in this document) for more detailed information.

1.3 Document Structure

The remainder of this document is organized into the following major sections:

- Section 2 presents an overview of the BIOS, describes server architectures and update mechanisms, identifies potential threats to BIOS in servers, and explains the Root of Trust for Update.
- Section 3 identifies security controls for BIOS implementations that are required or recommended to mitigate threats to BIOS in servers.
- Section 4 provides additional security guidelines for each of the three update mechanisms.
- Section 5 provides additional security guidelines for Service Processors.

The document also contains appendices with supporting material:

- Appendix A contains a summary of the security guidelines for system BIOS implementations.
- Appendices B, C, and D describe examples of possible system designs for implementing the BIOS protections in servers.
- Appendix E defines terms used in this document.
- Appendix F contains a list of acronyms and abbreviations used in this document.
- Appendix G contains a list of references used in the development of this document.

2. Background

This section provides background information on the system BIOS and types of server architectures. It identifies the primary methods used for updating the system BIOS, and it also discusses security issues and threats to the system BIOS. Finally, it discusses Root of Trust for Update components.

2.1 System BIOS

Exclusive of certain configuration and security code that may be embedded into a platform's silicon components, e.g., processor, the system BIOS is the first software executed on the main central processing unit (CPU) when a computer is powered on. While the system BIOS was originally responsible for providing operating systems access to hardware, its primary role on modern machines is to initialize and test hardware components and load the operating system. In addition, the BIOS loads and initializes important system management functions, such as power and thermal management. The system BIOS may also load CPU microcode patches during the boot process.

There are several different types of BIOS firmware. Some computers use a 16-bit conventional BIOS, while many newer systems use boot firmware based on the UEFI specifications [UEFI], or use other customized boot firmware. In this document we refer to all types of boot firmware as BIOS firmware, the system BIOS, or simply BIOS. When necessary, we differentiate conventional BIOS firmware from UEFI firmware by calling them the conventional BIOS and UEFI BIOS, respectively.

System BIOS is typically developed by the original equipment manufacturer (OEM) of the server. Manufacturers frequently update system firmware to fix bugs, patch vulnerabilities, and support new hardware. The system BIOS is typically stored on electrically erasable programmable read-only memory (EEPROM) or other forms of flash memory. Typically, system BIOS firmware is updated using a utility or tool that has special knowledge of the non-volatile storage components in which the BIOS is stored. This tool may run on one of the CPUs in the server itself, or it could run on a management processor within the server chassis.

A given computer system can have BIOS in several different locations. In addition to the motherboard, BIOS may be found on hard drive controllers, video cards, network cards and other add-in cards. This additional firmware generally takes the form of Option ROMs (containing conventional BIOS and/or UEFI drivers). These are loaded by the system firmware during the boot process and executed on the host CPU. Option ROMs should not be confused with device firmware; there are many system devices in a server with microcontrollers that run their own firmware.

The guidelines in this document apply to BIOS firmware stored in the BIOS flash memory, including the BIOS code, the cryptographic keys that are part of the Root of Trust for Update, and static BIOS data. Option ROMs that are stored with the system BIOS firmware and are updated by the same mechanism are considered part of the BIOS for the purposes of this document. However, the guidelines do not apply to Option ROMs and other device firmware stored elsewhere in a server, such as on an add-on card.

For more information on system BIOS fundamentals, see Section 2 of NIST SP 800-147.

2.2 Server Architectures

In this subsection, we differentiate servers into three classes: Basic Servers, Managed Servers, and Blade Servers. The distinctions made here have to do with the mechanisms employed to update the system BIOS.

A **Basic Server** is architecturally similar to a client PC system with a single BIOS update mechanism. The requirements for protecting BIOS updates for a Basic Server can be satisfied by meeting the requirements listed in this document or those in Section 3.1 of NIST SP 800-147. Typically the Root of Trust for Update (RTU) on a Basic Server is part of the system BIOS (see Section 2.5 for more on RTUs).

A **Managed Server** is a computer system with a dedicated management channel used for device maintenance and additional server management features. In addition to host processors, a Managed Server may have a Service Processor (see below) capable of performing system management, possibly including BIOS updates. It is critical to prevent the execution of malicious or corrupt code on the Service Processor since it has direct control over other operational aspects of the server. As a potential BIOS update mechanism, the code executed on the Service Processor may contain an RTU and must be protected from unauthorized modification.

A **Blade Server** is a specialized server hardware platform design that is configured as a mounted device in a chassis/enclosure that can be installed in a rack system. These platforms often have shared power and cooling and a shared management interface. Blades function either autonomously in a chassis/enclosure/rack as a bank of servers or they may be controlled by a central bus manager. This system configuration also may have multiple paths for updating the BIOS used to run the individual blades. From the BIOS update perspective, a Blade Server is typically similar to a managed server. Industry also uses the term multi-node (or modular) servers which have the same requirements as the Blade Servers for BIOS protections.

Server architectures vary greatly. While most servers fit into one of the classes list above, in some cases it may be difficult to classify a server as strictly belonging to one of those classes. The classes are provided as examples to help clarify requirements, however the security guidelines in this document are intended to have broad applicability to any type of server.

A server may have a Service Processor, sometimes called a Baseboard Management Controller, which is a specialized microcontroller. The Service Processor is often a special add-in card or embedded on the server motherboard. The Service Processor microcontroller runs a specialized operating system typically stored in flash memory, and provides administrators with an interface to manage the host server. The Service Processor is typically a highly privileged component in modern servers, often capable of updating firmware and software on the system, changing configuration settings, and reading from system memory. This document assumes administrators will implement proper computer, network, and physical security controls to mitigate the unique risks to Service Processors. For instance, typically administrators interact with the Service Processor over a network interface. Security best practices dictate that the Service Processor should be on a private LAN accessible only to system administrators.

Service Processors are responsible for various features, depending on the implementation of the system. The Service Processor will typically monitor various sensors in the server, such as temperature and power sensors for thermal and power management. It might provide administrators with a mechanism to update software or firmware on the server and host operating system. Maintaining the security of the Service Processor is critical to maintaining the integrity and availability of the server. See Section 5 for security requirements for Service Processors.

2.3 System BIOS Update Mechanisms

The guidelines in this publication are intended to secure BIOS update mechanisms so that only authentic, authorized BIOS images are written to BIOS flash memory, a process sometimes referred to as “flashing” the image. While client systems typically only have one path for updating the BIOS, server systems may implement several update mechanisms to allow administrators to update the BIOS from different

environments. This publication identifies three general types of authenticated BIOS update mechanisms. A server will typically implement one or more of these update mechanisms. Each mechanism is briefly described below:

- **Update Mechanism 1 – Authenticated BIOS update that can occur anytime**. This update mechanism allows the BIOS flash memory to be securely updated irrespective of the operating state of the server. This includes being able to update the BIOS flash while the server operates, without requiring a reboot. This mechanism must protect the BIOS flash memory from unauthorized modifications.
- **Update Mechanism 2 – Authenticated BIOS update on reboot**. This BIOS update mechanism allows the flash process to be initiated during server runtime, but the actual update of the BIOS flash memory does not occur until the system reboots. Like the previous mechanism, this mechanism must protect the BIOS flash memory from unauthorized modifications.
- **Update Mechanism 3 – BIOS update at runtime and verification of BIOS on reboot**. With this BIOS update mechanism, the BIOS is verified to be authentic before it is executed on every boot. This mechanism is necessary if the BIOS flash locking mechanism is inadequate to protect the integrity of a running system's BIOS flash memory. If the BIOS flash memory is determined to be inauthentic during boot, a recovery process is initiated and the inauthentic BIOS is not executed.

In addition, some servers may include a secure local update mechanism that updates the system BIOS without using any of the three authenticated update mechanisms discussed above. A secure local update mechanism ensures the authenticity and integrity of the BIOS update image by requiring that an administrator be physically present at the server to conduct the update. The requirement for physical presence mitigates the risk of remote attacks against the system BIOS.

2.4 Threats to the System BIOS

A server is susceptible to the same forms of attack that threaten a client system. Execution of operating system-level malware on servers could precede a BIOS attack. Server BIOS updates that are executed without being authenticated as coming from a trusted source are vulnerable to attacks. Since servers may have multiple BIOS update mechanisms, each mechanism has a risk of vulnerability. The interaction between update mechanisms potentially can also introduce vulnerabilities.

The Service Processor in servers has elevated privileges to perform system management, which may include modifying BIOS. While the Service Processor might be controlled through an isolated communication channel, unauthorized access to this channel exposes great risk to the server. While many security practices focus on the data network, the management network might be less vetted and less protected without special efforts.

If insufficiently protected, BIOS image backups on servers (often maintained for recovery features) are vulnerable to rewrite attacks even while the primary BIOS is protected from modification. Subsequent to successful subversion of a BIOS backup, an adversary could employ other means to cause the server to reboot with the infected backup image.

2.5 Root of Trust for Update

A Root of Trust for Update (RTU) is an inherently trusted combination of hardware and firmware that performs a secure update of the BIOS and maintains the integrity of the BIOS. The RTU may comprise functionality for verifying digitally signed BIOS images, engaging and disengaging write-protection mechanisms, writing BIOS updates to flash, performing BIOS recovery, and updating the RTU itself.

Section 3 specifies requirements on the RTU for performing secure BIOS updates and maintaining BIOS integrity. The inherent trust of the RTU is derived from an isolated execution environment—minimizing the risk to subvert the functionality of the RTU and hence maintaining the inherent trust of the RTU.

Each functional component of the RTU may be considered a Root of Trust for the specific function:

- The **Verification Component** is responsible for verifying a digitally signed BIOS image to determine if control should be passed to the image. This component has a trusted execution path since it is entered from a known good state of the machine. The verification component can be used to extend trusted execution to code in unprotected memory locations. The verification component verifies the BIOS image and if the verification is successful then it passes control to the image. If the verification fails then the verification component returns to the trusted execution path and does not pass control to the image.
- The **Recovery Component** is responsible for initiating a return of the system to a known good state.
- The **Integrity Component** is responsible for maintaining the integrity of a BIOS image. This may include engaging hardware and firmware based locking mechanisms to prevent unauthorized modification of the image. It also prevents race/logic conditions from unauthorized modification of a BIOS image.
- The **Update Component** is responsible for performing a secure update of the RTU and maintaining the integrity of the RTU.

3. BIOS Security Principles

The security principles presented in NIST SP 800-147 for client systems—update authentication, flash region integrity, and non-bypassability—apply directly to server class machines. These security principles are intended to mitigate threats targeting the system BIOS. The complexity of server architectures and the multiple update paths for BIOS on servers require the extension of the guidelines in NIST SP 800-147. This section enumerates the requirements for servers to assert the security principles for BIOS update. The principles use the terms authorize and authenticate in the following context. Authentication of an image assures the integrity and origin of the image. It is typically rooted at the firmware or server manufacturer. Authentication is performed cryptographically using digital signatures. Authorization is permission for an update to be performed by the system. Authorization of updates is typically rooted in the server administrator.

3.1 BIOS Update Authentication

Authenticated BIOS update mechanisms shall be implemented in an RTU (see Section 2.5). These mechanisms employ digital signatures to ensure the authenticity and integrity of BIOS update images. The Verification Component of the RTU shall contain the digital signature verification algorithm and a key store. The key store shall include the public key needed to verify the signature of the BIOS update image or an approved cryptographic hash [FIPS180-4] of the key. In the latter case, the update mechanism shall hash the public key provided with the BIOS update image and ensure that it matches a hash which appears in the key store before using the provided public key to verify the signature on the BIOS update image.

BIOS images shall be signed in conformance with NIST SP 800-89, *Recommendation for Obtaining Assurances for Digital Signature Applications* [SP800-89], using an approved digital signature algorithm as specified in NIST FIPS 186-4, *Digital Signature Standard (DSS)* [FIPS186-4], that provides at least 112 bits of security strength, in accordance with NIST SP 800-131A, *Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths* [SP800-131A].

Authenticated update mechanisms shall ensure that BIOS update images have been digitally signed and that the digital signature can be verified using a key stored or verified by the RTU before updating the BIOS. These mechanisms may also verify digital signatures over BIOS code in flash memory prior to execution, particularly if the BIOS flash is not protected (see Section 3.3). Recovery mechanisms should also use an authenticated update mechanism, unless the recovery process meets the guidelines for a secure local update.

The system should provide mechanisms capable of preventing unauthorized update of the BIOS to an earlier authentic version. This limitation of the rollback mechanism may be accomplished, for example, by verifying that the version number of the BIOS image is higher than the currently installed BIOS image's version number. Appropriate rollback protections will depend upon the environment in which the server is used and the security and availability needs of the organization operating the server. The system may allow administrators to authorize rollback to an earlier version, disable these mechanisms, or otherwise configure these mechanisms based on their needs.

3.2 Secure Local Update

Servers may optionally include a secure local update mechanism that updates the system BIOS without using the authenticated update mechanism. Secure local update mechanisms shall authorize and authenticate the BIOS update image by requiring that an administrator be physically present at the server

to conduct the update. This requirement for physical presence mitigates the risk of a remote attacker conducting a malicious update to an inauthentic system BIOS image.

Interacting with a server via a remote console does not satisfy this requirement for physical presence. A secure local update mechanism can be used, for example, to recover from a corrupted BIOS that cannot be updated using an authenticated update or an automated recovery mechanism. The secure local update mechanism could also be used by a physically-present administrator to update to an earlier BIOS version on a system that does not allow rollback.

However, note that systems that implement the secure local update mechanism are potentially vulnerable to attacks by rogue administrators or other attacks with physical access to the server. Therefore, additional physical, environmental and technical security measures are essential to protecting these servers, but they are beyond the scope of this document.

3.3 Firmware Integrity Protection

To prevent the execution of inauthentic or malicious BIOS code, the integrity of the system BIOS shall be maintained between the verification of the system BIOS by the RTU and the execution of the system BIOS during the boot process.

To prevent unintended or malicious modification of the system BIOS outside the authenticated BIOS update process, the system flash memory containing BIOS (i.e., BIOS flash), excluding configuration data used by the system BIOS that is stored in non-volatile memory, should be protected from modifications outside the authenticated BIOS update mechanisms. If implemented, BIOS flash protections shall be engaged prior to execution of code outside of the RTU and shall be enforced by hardware mechanisms that are disengaged only by an authorized mechanism (e.g., system reset).

If BIOS flash protections are not implemented, then BIOS integrity shall be verified prior to each execution, using the Verification Component of the RTU to authenticate the BIOS image. If the verification fails, then the system shall initiate a recovery procedure, which may involve operator intervention or be automatic. Automatic recovery mechanisms which initiate recovery to a protected authentic BIOS should be supported. Automatic recovery mechanisms mitigate the risk of a denial of service attack, whereby an attacker would load an inauthentic BIOS and put the system in an unbootable state. The system may allow authorized administrators to configure security policies on the server that drive these recovery procedures.

Each RTU shall be protected from modifications outside the authenticated update mechanisms. Protection mechanisms that ensure the integrity of the RTU shall be engaged prior to execution of code outside of the RTU. RTU integrity protections shall be enforced by hardware mechanisms that are disengaged only by an authorized mechanism, for example system reset.

3.4 Non-Bypassability

The design of the system and accompanying system components and firmware shall ensure that there are no mechanisms to install and execute unauthenticated BIOS code, except through physical intervention and the secure local update mechanism.

In particular, the authenticated update mechanisms shall be the exclusive mechanisms for modifying the RTU absent physical intervention through the secure local update mechanism. The design of the system and accompanying system components and firmware shall ensure that there are no mechanisms that allow

the host processor or any other system component to bypass the authenticated update mechanism for updating the RTU, except for the secure local update mechanism.

A modern platform includes design features that give system components direct access to the RTU or system BIOS for performance and management improvements, such as shadowing the BIOS in RAM or for system management mode operations. A system component may have read access to system flash memory, but it shall not be able to directly modify the RTU unless that component serves as an extension to the RTU or as an RTU itself.

4. Security Guidelines by Update Mechanism

The following provides detailed recommendations for implementing the security guidelines in this document for three typical secure BIOS update mechanisms. A server may implement one or more of these mechanisms, and the method(s) implemented by a given server may be dictated by the hardware support on the platform. These methods differ on when an RTU can be established and the availability of a secure locking mechanism for the flash to prevent unintended or malicious modification of the code and data stored in the BIOS flash memory. All mechanisms depend on a digitally signed BIOS update image and the ability of the verification component of an RTU, using a public key, to verify the signature of that image.

4.1 Update Mechanism 1: Secure BIOS Update at Anytime

With this BIOS update mechanism, the secure update of the BIOS flash memory may occur during numerous operating states of the server. This includes being able to update the BIOS flash while the server operates, without requiring a reboot. While the new BIOS is not expected to execute until reboot, the flash can be updated at runtime using a System Management Interrupt (SMI) handler, a Service Processor, or other secure methods. The reboot that causes the new BIOS to execute could occur at any later time (potentially months after the actual flash update takes place).

Because this secure update mechanism prevents writing inauthentic code to the BIOS flash memory, it is not necessary to verify the BIOS during boot.

The relevant guidelines for implementing this BIOS update mechanism are listed below:

- The BIOS update image must be digitally signed in accordance with the BIOS Update Authentication requirements in Section 3.1. An implementation may divide the BIOS update image into multiple, individually digitally signed parts.
- An RTU, as defined in Section 2.5, must be available at runtime (and may be available during other operating states of the server) to update the BIOS flash memory.
- A locking mechanism must exist such that only an RTU can write to the BIOS flash. Only an RTU should be able to unlock the flash and, when unlocked, only the RTU may have the ability to write to the BIOS flash memory. An RTU may always have access to write to the BIOS flash memory.
- The digitally signed BIOS update image must be transferred to an RTU. The RTU must have the ability to store the BIOS update image in a location that does not allow unauthorized write access to the BIOS update image.

The general steps for implementing this mechanism are:

1. A digitally signed BIOS update image is transferred to the RTU.
2. The RTU stores the BIOS update image in a location that can only be written to by an RTU.
3. The RTU verifies the BIOS update image is authentic. If the BIOS update image is determined to be inauthentic, the BIOS update image will not be written to the BIOS flash memory. If the BIOS update image is authentic, the RTU configures the locking mechanism such that only an RTU has access to write the BIOS flash memory. A viable locking mechanism design could allow the RTU to always have access to write the BIOS flash memory as long as only RTUs have the same capability.
4. The RTU writes the BIOS update image into the BIOS flash memory.
5. The RTU ensures that the BIOS flash memory is locked prior to transferring control to code outside the RTU (e.g., Option ROMs), per the Firmware Integrity Protection requirements in Section 3.3.

See Appendix B for an example of this update mechanism.

4.2 Update Mechanism 2: Secure BIOS Update at Reboot

With this BIOS update mechanism, the BIOS flash memory process is initiated at server runtime. However, the actual update of the BIOS flash memory does not occur until a server reboot. This BIOS update mechanism prevents inauthentic code from ever being written to the BIOS flash memory.

The relevant guidelines for implementing this BIOS update mechanism are listed below:

- The BIOS update image must be digitally signed in accordance with the BIOS Update Authentication requirements in Section 3.1. An implementation may divide the BIOS update image into multiple, individually digitally signed parts.
- A locking mechanism must exist for the BIOS flash memory such that no entity except an RTU has write access to the BIOS flash memory at runtime. Although it is not necessary for an RTU to have write access to the BIOS flash memory for this update mechanism, additional BIOS update mechanisms may be supported that do have this requirement.
- An RTU, as defined in Section 2.5, must be available during system boot that can update the BIOS flash memory at runtime. The RTU must execute on reboot before the BIOS flash memory is updated. The RTU will verify the BIOS update image's digital signature before any changes to the flash occur.
- A memory location whose contents are preserved on reboot must exist in which the signed BIOS update image can be buffered at runtime and which the RTU can access during the reboot to verify the BIOS update image's digital signature before updating the contents of the BIOS flash memory. During reboot, the memory location holding the BIOS update image during verification must not be accessible by any entities other than an RTU.

The general steps for implementing this mechanism are:

1. A digitally signed BIOS update image is buffered into a storage location whose contents are preserved when the server is rebooted.
2. When the server is rebooted, execution is transferred to the RTU.
3. The RTU verifies the BIOS update image is authentic. If found to be authentic, the RTU will unlock the BIOS flash memory if necessary and write the update into the BIOS flash memory. If the BIOS update image is determined to be inauthentic, the BIOS flash memory will not be updated.
4. The locking mechanism for the BIOS flash memory is enabled before executing untrusted code, including Option ROMs.

See Appendix C for an example of this update mechanism.

4.3 Update Mechanism 3: Secure BIOS Update Requiring Verification at Boot

With this BIOS update mechanism, a locking mechanism to protect the BIOS flash memory at runtime does not exist or has limitations such that writes to the BIOS flash memory by an entity other than an RTU are not prevented. A rogue update of the BIOS flash memory might occur as the BIOS flash memory is not write-protected or locked. However, the contents of the BIOS flash memory shall be authenticated before being executed on every boot. If the BIOS flash memory is determined to be inauthentic during boot, a recovery process is initiated and the inauthentic BIOS is not executed.

The relevant guidelines for implementing this BIOS update mechanism are listed below:

- The BIOS image must be digitally signed in accordance with the BIOS Update Authentication requirements in Section 3.1.
- An RTU, as defined in Section 2.5, must verify the digital signature of the BIOS update image before writing the image to the BIOS flash memory.
- As the system BIOS is not protected from modification at runtime, the RTU must include a verification component capable of verifying the system BIOS prior to execution, per the Firmware Integrity Protection requirements in Section 3.3. The verification component must execute on boot and verify the system BIOS before any updatable BIOS code is executed.
- If it is determined that the system BIOS is inauthentic, the RTU must initiate a recovery process, and the inauthentic system BIOS must never be executed.

The general steps for implementing this mechanism are:

1. A digitally signed BIOS image is verified and written into the BIOS flash memory.
2. On every boot, execution is transferred to the RTU.
3. If the verification component of the RTU determines the BIOS flash memory is authentic, execution is transferred to the BIOS.
4. If it is determined that the BIOS flash memory is inauthentic, the RTU initiates a recovery process, and the inauthentic BIOS is never executed.

See Appendix D for an example of this update mechanism.

5. Guidelines for Service Processors

A key distinction between servers and clients is the inclusion of a Service Processor in the system. The Service Processor plays a critical role in managing and monitoring the server, and may have a role in updating the system BIOS. This section describes the Service Processor and provides more detailed security requirements for systems that contain a Service Processor.

5.1 Service Processor as a Root of Trust

Service Processors in managed and blade servers might have the ability to directly update the BIOS flash memory, the BIOS execution memory, or their own flash or other storage media. In these cases, some or all of the Service Processor environment may be employed as an RTU for the system BIOS. This applies to Service Processors that are capable of updating BIOS flash on systems that use BIOS flash protection mechanisms to prevent unauthorized modifications to BIOS code (e.g., those that employ update mechanism 1 or 2, as described in Section 4). It also applies to Service Processors that are capable of modifying the RTU used to verify BIOS code in systems that verify BIOS code prior to execution (e.g., those that employ update mechanism 3).

In these situations, the execution environment of the Service Processor (SP) must be protected from malicious code that could update the BIOS or the SP flash. To comply with the BIOS security principles in Section 3, Service Processors employed as a Root of Trust to protect BIOS shall meet the following guidelines:

1. Updates to the SP code, cryptographic keys, and static data stored on the SP flash shall be by way of an authenticating update mechanism.
2. The SP environment shall be controlled such that only authenticated code can be executed on the SP.
3. Authorization should be required for user interaction with the Service Processor.

5.2 Non-Bypassability of BIOS Protections by Service Processor

Some servers with Service Processors might not employ the Service Processor as an RTU for BIOS updates. To ensure the SP environment cannot bypass BIOS protections, Service Processors in these systems shall not have direct write access to the BIOS flash memory from which the BIOS code is verified or executed. Furthermore, the Service Processor shall be restricted such that is not capable of interfering with BIOS update processes, nor the memory from which the BIOS or RTU code is executed.

Service Processors that do not implement these restrictions may be capable of bypassing the BIOS protections outlined in this document. As such, they must be trusted to protect BIOS, even if in their normal operation they are not used for BIOS update. Such Service Processors shall be considered Roots of Trust and meet the requirements in Section 5.1.

Appendix A— Summary of Requirements

This appendix contains a summary of the secure BIOS update guidelines for system BIOS implementations found in Section 3. These guidelines are intended for platform vendors designing, implementing, or selecting a system BIOS implementation. Readers should consult the relevant sections in the main body of this document for additional informative text that further describes the intent and context of the guidelines.

1. BIOS Update Authentication

1-A Authenticated BIOS update mechanisms shall be implemented in an RTU (see Section 2.5).

1-B The Verification Component of the RTU shall contain the digital signature verification algorithm and a key store. The key store shall include the public key needed to verify the signature of the BIOS update image or an approved cryptographic hash [FIPS180-4] of the key. In the latter case, the update mechanism shall hash the public key provided with the BIOS update image and ensure that it matches a hash which appears in the key store before using the provided public key to verify the signature on the BIOS update image.

1-C BIOS images shall be signed in conformance with NIST SP 800-89, *Recommendation for Obtaining Assurances for Digital Signature Applications* [SP800-89], using an approved digital signature algorithm as specified in NIST FIPS 186-4, *Digital Signature Standard* [FIPS186-4], that provides at least 112 bits of security strength, in accordance with NIST SP 800-131A, *Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths* [SP800-131A].

1-D Authenticated update mechanisms shall ensure that BIOS update images have been digitally signed and that the digital signature can be verified using a key stored or verified the RTU before updating the BIOS.

1-E Recovery mechanisms should also use an authenticated update mechanism, unless the recovery process meets the guidelines for a secure local update.

1-F The system should provide mechanisms capable of preventing unauthorized update of the BIOS to an earlier authentic version.

2. Secure Local Update (Optional)

Servers may optionally include a secure local update mechanism, where physical presence authorizes installation of BIOS update images without necessarily using the authenticated update mechanisms.

2-A Secure local update mechanisms shall authorize and authenticate the BIOS update image by requiring that an administrator physically touch the server itself to conduct the update.

3. Integrity Protection

3-A The integrity of the system BIOS shall be maintained between the verification of the system BIOS by the RTU, and the execution of the system BIOS during the boot process.

3-B System flash memory containing BIOS, excluding configuration data based by the system BIOS that is stored in non-volatile memory, should be protected from modifications outside the authenticated BIOS update mechanisms.

3-B.i If implemented, BIOS flash protections shall be engaged prior to execution of code outside the RTU.

3-B.ii If implemented, BIOS flash protections shall be enforced by hardware mechanisms that are disengaged only by an authorized mechanism (e.g., system reset).

3-C If BIOS flash protections are not implemented, then BIOS integrity shall be verified prior to each execution, using the Verification Component of the RTU to authenticate the BIOS image.

3-C.i If verification fails, then the system shall initiate a recovery procedure.

3-C.ii Automatic recovery mechanisms which initiate recovery to a protected, authentic BIOS should be supported.

3-D Each RTU shall be protected from modifications outside the authenticated update mechanisms.

3-D.i Protection mechanisms that ensure the integrity of the RTU shall be engaged prior to execution of code outside of the RTU.

3-D.ii RTU integrity protections shall be enforced by hardware mechanisms that are disengaged only by an authorized mechanism, for example system reset.

4. Non-Bypassability

4-A The design of the system and accompanying system components and firmware shall ensure that there are no mechanisms to install and execute unauthenticated BIOS code, except through physical intervention and the secure local update mechanism.

4-B The authenticated BIOS update mechanism shall be the exclusive mechanism for modifying the RTU absent physical intervention through the secure local update mechanism.

4-C The design of the system and accompanying system components and firmware shall ensure that there are no mechanisms that allow the host processor or any other system component to bypass the authenticated update mechanism for updating the RTU, except for the secure local update mechanism.

4-D While a system component may have read access to system flash memory, it shall not be able to directly modify the RTU unless that component serves as an RTU itself.

5. Service Processors

5-A Service Processors employed as a Root of Trust to protect BIOS, shall meet the following requirements:

5-A.i Updates to the SP code, cryptographic keys, and static data stored on the SP flash memory shall be by way of an authenticating update mechanism.

5-A.ii The SP environment shall be controlled such that only authenticated code can be executed on the SP.

5-A.iii Authorization should be required for user interaction with the SP.

5-B Service Processors that are not employed as a Root of Trust to protect BIOS shall meet the following requirements:

5-B.i SPs shall not have direct write access to the BIOS flash memory which the BIOS code is verified or executed.

5-B.ii SPs shall be restricted such that it is not capable of interfering with BIOS update processes, nor the memory from which the BIOS or RTU code is executed.

Appendix B—Example of Update Mechanism 1

This appendix provides an example of how Update Mechanism 1 could be implemented in a server. Update Mechanism 1 includes authenticated BIOS updates at server runtime. In this example, runtime BIOS updates are performed by the Service Processor (SP). Figure 1, a system block diagram for this example, shows the BIOS flash SPI controller accessible by both the host and the SP. Extra logic along the host-to-SPI controller path may be needed to prevent write access by the host to the BIOS flash memory. This logic may either be internal chipset registers or external to the chipset. Because it has write access to the system flash, the SP environment in this example must be a secure, protected environment as described in Section 5.

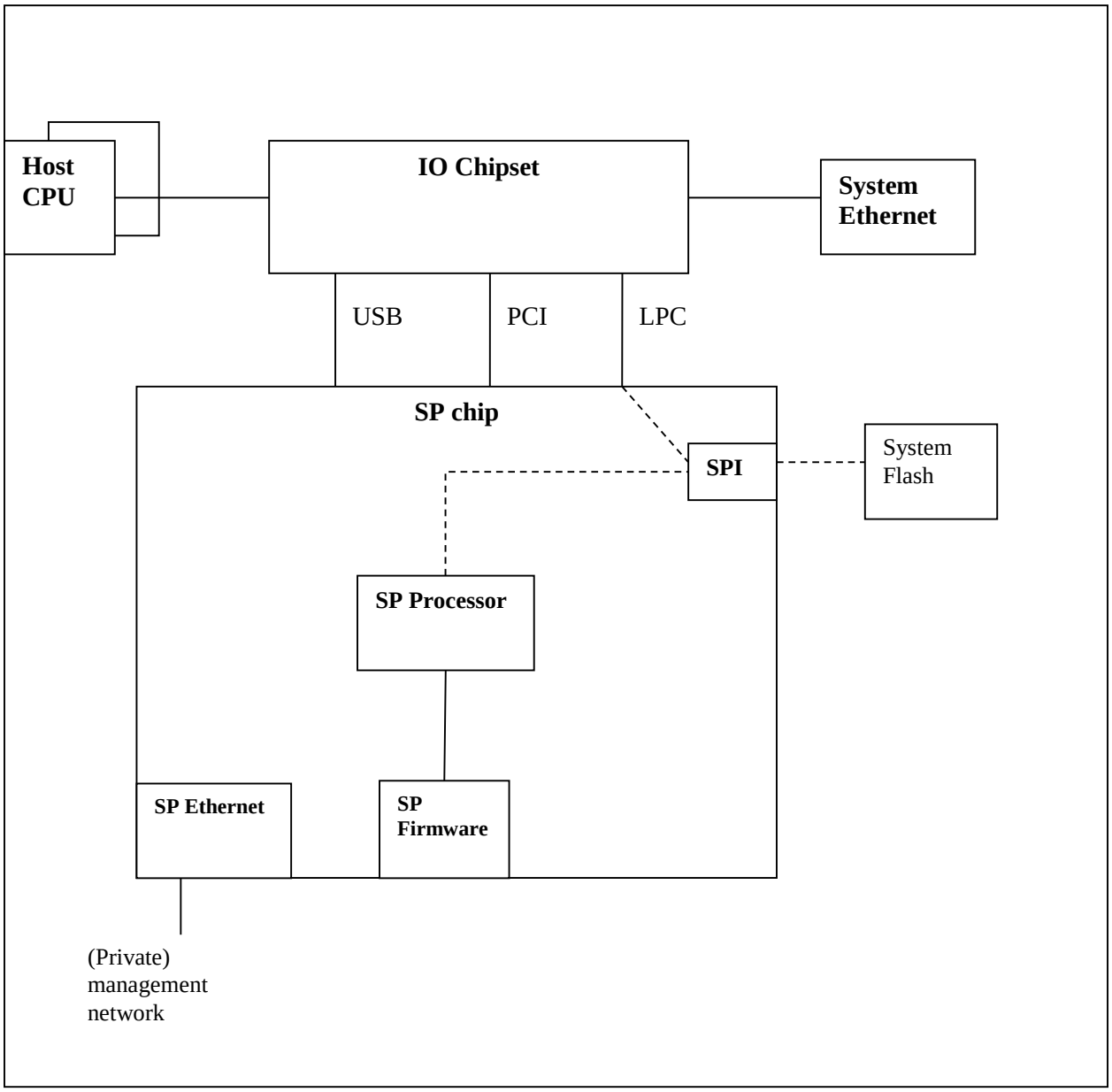

Figure 1: Example System Block Diagram for Update Mechanism 1

In this example, the system BIOS is given control of the host CPU at system reset. All SPI flash sectors are unlocked at system reset. The system BIOS, prior to allowing execution of untrusted code (e.g., Option ROMs, bootloaders) locks host-side write access to the system flash sectors containing the system BIOS. This locking is done via interaction with the logic to "lock until reset" the sectors containing the BIOS image. When the lock is set, the access to this lock register becomes read-only so that the "lock until reset" setting itself cannot be modified. Write access to the BIOS flash sectors remains available to the SP.

When conducting a BIOS update, system management software on the host can interact with the SP to send a candidate update image to the SP. Alternatively, the BIOS update image can arrive at the SP through out of band communication via the SP Ethernet. The SP, as an RTU for the system BIOS, verifies the BIOS update image using the digital signature verification algorithm and key store on the SP. If the image is authenticated, then the SP, which continues to have write access to the BIOS flash memory even after host OS boot, can perform the flash update via interaction with the SPI controller of the BIOS flash memory.

Appendix C—Example of Update Mechanism 2

This appendix provides an example of how Update Mechanism 2 could be implemented in a server. In this example, BIOS update images are verified and flashed during the boot process, while the RTU implemented as part of the BIOS has control of the system. Figure 2 shows an example system block diagram for a server using this update mechanism. The BIOS flash SPI controller is accessible by just the host. A potential BIOS update image is stored in the SP environment.

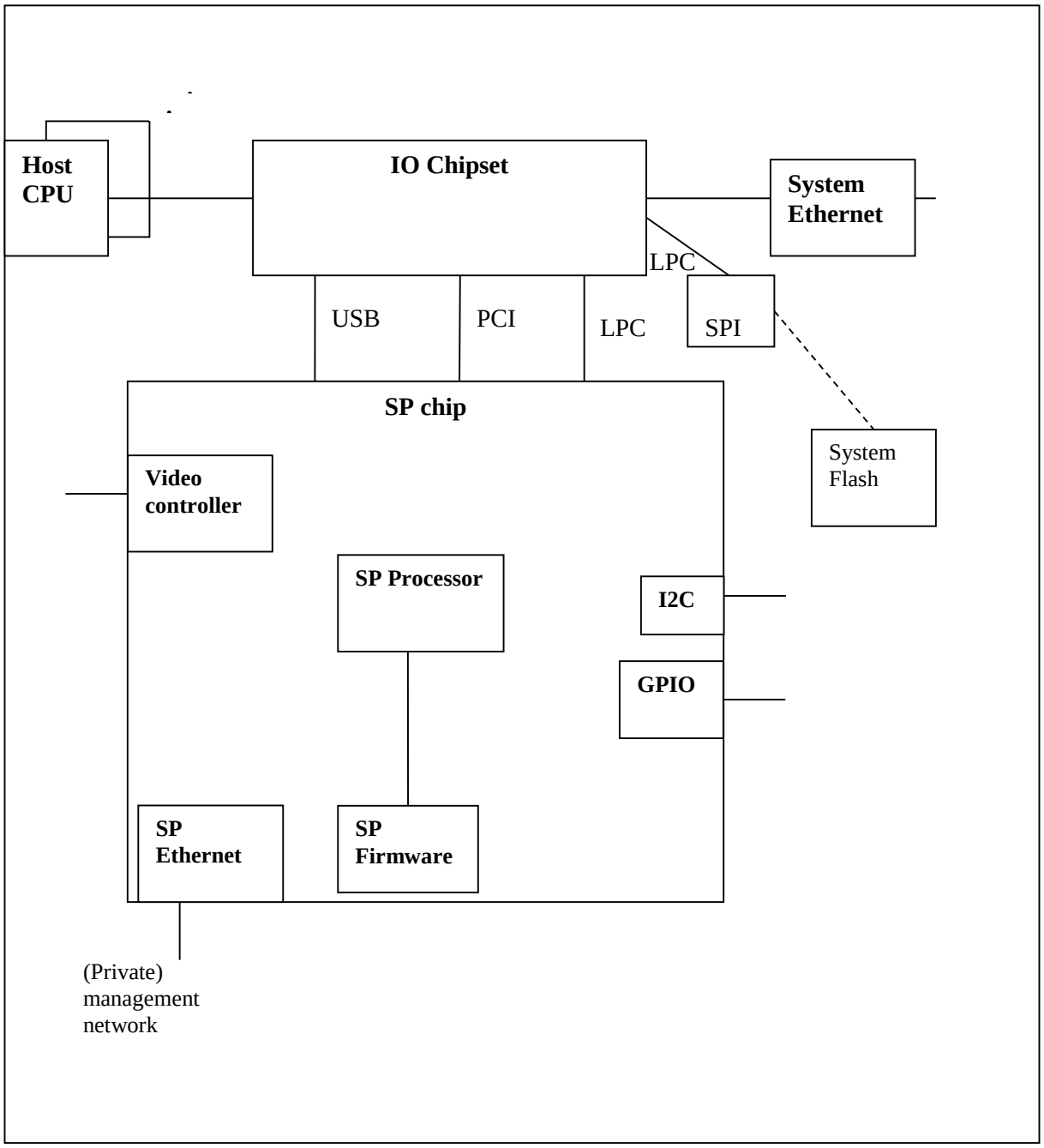

Figure 2: Example System Block Diagram for Update Mechanism 2

In this example, to initiate a BIOS update, system management software on the host can interact with the SP to send a BIOS update image to be stored in the SP environment for later access by BIOS. Alternatively, the candidate BIOS update image can arrive at the SP through out of band communication via the SP Ethernet.

The RTU, implemented as part of the system BIOS, gets control of the host side at system reset. All SPI flash sectors are unlocked at system reset. Within the system flash, the RTU is not separated from the remainder of the system BIOS. The system BIOS, prior to executing untrusted code (e.g., Option ROMs) is the RTU. It communicates with the SP to check for a candidate BIOS update image. If one exists, it is read from the SP into host memory (which is only writable by the system BIOS during RTU execution) and performs verification. If the BIOS update image is authentic, the system BIOS performs the update of the system flash via interaction with the SPI flash controller. At completion of the BIOS update, the system BIOS forces a system reset and restarts execution from the new image.

If the SP indicated that no candidate BIOS update image existed, or if the candidate BIOS update image failed verification, BIOS locks the BIOS flash memory via SPI controller interaction to "lock until reset" the sectors containing the BIOS image. When the lock is set, the access to this SPI sector lock register becomes read-only so that the "lock until reset" setting itself cannot be modified. This sector lock is done prior to exiting the RTU portion of the BIOS.

Appendix D—Example of Update Mechanism 3

This appendix provides an example of how Update Mechanism 3 could be implemented in a server. In this example, the BIOS flash is not strongly protected and therefore the integrity of the BIOS code is verified during the boot process. An example system block diagram for update mechanism 3 shows the BIOS flash SPI controller accessible by both the host and the SP. Because it has write access to the BIOS flash memory, the SP environment in this example must be a secure, protected environment as described in Section 5.

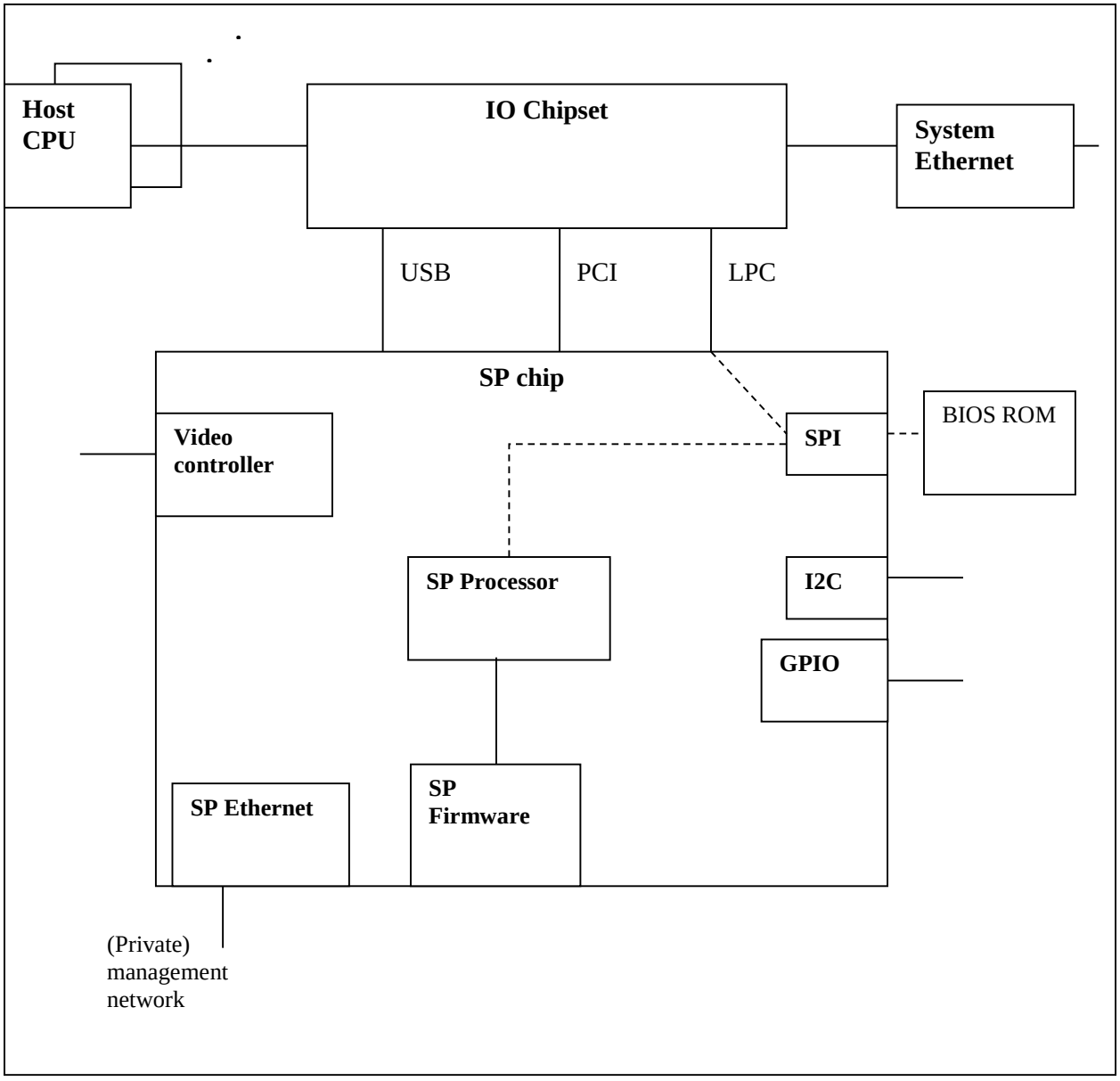

Figure 3: Example System Block Diagram for Update Mechanism 3

In this example, to initiate a BIOS update, system management software on the host can interact with the SP to send a BIOS update image to the SP. Alternatively, the BIOS update image can arrive at the SP through out of band communication via the SP Ethernet. The SP, acting as part of the RTU, then

authenticates the BIOS update image. If the image is authentic, then the SP, which continues to have write access to the system flash even after host OS boot, can then perform the flash update via interaction with the SPI controller of the system flash.

In this example, the system BIOS contains its own verification component of the RTU, which authenticates the remainder of the system BIOS each boot. This verification component resides in the system flash and will be labeled the RTU-V for the purposes of describing this update mechanism. The RTU-V is given control of the host side at system reset. System reset also unlocks all of the SPI flash sectors. Within the system flash, the RTU-V is lockable, independently of the remainder of system BIOS. The RTU-V is also updatable as follows: Similar to update mechanism example 2, the RTU-V update occurs upon reboot and is updated by the RTU-V itself. It communicates with the SP to check for a candidate RTU-V update image. If one exists, it is read from the SP into host memory and performs authentication. If the image authenticates, the RTU-V does the update of the RTU-V portion of the BIOS flash ROM via interaction with the SPI flash controller. At completion of the RTU-V update, the RTU-V forces a system reset and restarts execution from the new image.

If the SP indicated that no candidate RTU-V update image existed, or if the candidate RTU-V update image failed verification, the RTU-V locks until reset the RTU-V sectors of the flash via interactions with the SPI controller. When the lock is set, the access to this SPI sector lock register becomes read-only so that the "lock until reset" setting itself cannot be modified. This sector lock is done prior to exiting the RTU-V (e.g., before execution of the rest of BIOS).

Because the system flash regions containing BIOS code are not lockable and are subject to modification by malicious code, the RTU-V authenticates the remainder of system BIOS. The remainder of the system BIOS is kept in separate flash sectors in the system flash. If authentication passes, the RTU-V passes control to the remainder of the system BIOS. In this example, the remainder of system BIOS does not lock the flash sectors containing the remainder of the system BIOS. A possible reason for not locking the flash sectors may be to simplify access to those sectors by the BIOS RTU at runtime. Another possible reason would be that a portion of these sectors need to be writeable at runtime and the sector locking granularity is not sufficient to lock the BIOS without also locking the area that needs to be writeable.

If authentication of the BIOS fails, it is not executed. Instead, the RTU-V communicates with the BIOS RTU on the SP to inform it of the failure. The SP must then access an authentic BIOS image previously stored in the SP environment (possibly from the previous authenticated BIOS update), authenticate it and perform the flash update. The SP then forces a system reset to restart the RTU-V which will subsequently authenticate and execute the remainder of system BIOS.

Appendix E—Glossary

Selected terms used in the publication are defined below.

Basic Input/Output System (BIOS): Refers collectively to boot firmware based on the conventional BIOS, Extensible Firmware Interface (EFI), and the Unified Extensible Firmware Interface (UEFI).

Conventional BIOS: Legacy boot firmware used in many x86-compatible computer systems. Also known as the legacy BIOS.

Extensible Firmware Interface (EFI): A specification for the interface between the operating system and the platform firmware. Version 1.10 of the EFI specifications was the final version of the EFI specifications, and subsequent revisions made by the Unified EFI Forum are part of the UEFI specifications.

Firmware: Software that is included in read-only memory (ROM).

Option ROM: Firmware that is called by the system BIOS. Option ROMs include BIOS firmware on add-on cards (e.g., video card, hard drive controller, network card) as well as modules which extend the capabilities of the system BIOS.

System Management Mode (SMM): A high-privilege operating mode found in x86-compatible processors used for low-level system management functions. System Management Mode is only entered after the system generates a System Management Interrupt and only executes code from a segregated block of memory.

System Flash Memory: The non-volatile storage location of system BIOS, typically in electronically erasable programmable read-only memory (EEPROM) flash memory on the motherboard. While system flash memory is a technology-specific term, guidelines in this document referring to the system flash memory are intended to apply to any non-volatile storage medium containing the system BIOS.

Trusted Platform Module (TPM): A tamper-resistant integrated circuit built into some computer motherboards that can perform cryptographic operations (including key generation) and protect small amounts of sensitive information, such as passwords and cryptographic keys.

Unified Extensible Firmware Interface (UEFI): A possible replacement for the conventional BIOS that is becoming widely deployed in new x86-based computer systems. The UEFI specifications were preceded by the EFI specifications.

Appendix F—Acronyms and Abbreviations

This appendix contains a list of selected acronyms and abbreviations used in the guide.

ACPI	Advanced Configuration and Power Interface
BIOS	Basic Input/Output System
CPU	Central Processing Unit
EEPROM	Electrically Erasable Programmable Read-Only Memory
EFI	Extensible Firmware Interface
FIPS	Federal Information Processing Standard
FISMA	Federal Information Security Management Act
GPIO	General Purpose Input/Output
I2C	Inter-Integrated Circuit
IO	Input/Output
IT	Information Technology
ITL	Information Technology Laboratory
LAN	Local Area Network
LPC	Low Pin Count
NIST	National Institute of Standards and Technology
OEM	Original Equipment Manufacturer
OMB	Office of Management and Budget
OS	Operating System
PC	Personal Computer
PCI	Peripheral Component Interconnect
ROM	Read-Only Memory
RTU	Root of Trust for Update
RTU-V	Root of Trust for Update verification component
SMI	System Management Interrupt
SMM	System Management Mode
SP	Service Processor
SP	Special Publication
TPM	Trusted Platform Module
UEFI	Unified Extensible Firmware Interface
USB	Universal Serial Bus

Appendix G—References

The list below provides references for this publication.

[DuGr09] Loïc Duflot, Olivier Grumelard, Olivier Levillain and Benjamin Morin. "ACPI and SMI handlers: some limits to trusted computing." *Journal in Computer Virology*. Volume 6, Number 4, 353-374.

[EFI] *EFI 1.10 Specification*. Intel. 1 November 2003. http://www.intel.com/technology/efi/

[EmSp08] Shawn Embleton, Sherri Sparks, and Cliff C. Zou. "SMM Rootkits: A New Breed of OS Independent Malware," *Proceedings of 4th International Conference on Security and Privacy in Communication Networks (SecureComm)*, Istanbul, Turkey, September 22-25, 2008.

[FIPS180-4] FIPS 180-4, *Secure Hash Standard*. March 2012.

[FIPS186-4] FIPS 186-4, *Digital Signature Standard*. July 2013.

[Graw09] D. Grawrock. *Dynamics of a Trusted Platform: A Building Block Approach*. Hillsboro, OR: Intel Press, 2009.

[Heas07a] J. Heasman. "Firmware Rootkits: A Threat to the Enterprise." Black Hat DC. Washington, DC. 28 February 2007. http://www.nccgroup.com/Libraries/Document_Downloads/02_07_Firmware_Rootkits_The_Threat_to_the_Enterprise_Black_Hat_Washington_2007_sflb.sflb.ashx

[Heas07b] J. Heasman. "Hacking the Extensible Firmware Interface." *Black Hat USA*. Las Vegas, NV. 2 August 2007. https://www.blackhat.com/presentations/bh-usa-07/Heasman/Presentation/bh-usa-07-heasman.pdf

[Intel03] *Intel Platform Innovation Framework for EFI- Architecture Specification v0.9*. Intel. September 2003. http://www.intel.com/technology/framework/

[KGH09] A. Kumar, G. Purushottam, and Y. Saint-Hilaire. *Active Platform Management Demystified*. Hillsboro, OR: Intel Press, 2009.

[Sal07] Salihun, Darmawan. *BIOS Disassembly Ninjutsu Uncovered*. Wayne, PA: A-LIST, 2007.

[SaOr09] A. Sacco, A. Ortéga. "Persistant BIOS Infection." *Phrack*. Issue 66. 6 November 2009. http://www.phrack.com/issues.html?issue=66&id=7

[SP800-57] NIST SP 800-57 Part 1, *Recommendation for Key Management – Part 1: General (Revision 3)*. July 2012.

[SP800-89] NIST SP 800-89, *Recommendation for Obtaining Assurances for Digital Signature Applications*. November 2006.

[SP800-131A] NIST SP 800-131A, *Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths*. January 2011.

[Sym02] *W95.CIH Technical Details*. Symantec. 25 April 2002. http://www.symantec.com/security_response/writeup.jsp?docid=2000-122010-2655-99

[TCG05] *PC Client Work Group Specific Implementation Specification for Conventional Bios Specification, Version 1.2*. Trusted Computing Group. July 2005. http://www.trustedcomputinggroup.org/resources/pc_client_work_group_specific_implementation_specification_for_conventional_bios_specification_version_12

[UEFI] *UEFI Specification Version 2.3*. Unified EFI Forum. May 2009. http://www.uefi.org/specs/

[Wech09] F. Wecherowski. "A Real SMM Rootkit: Reversing and Hooking BIOS SMI Handlers." *Phrack*. Issue 66. 6 November 2009. http://www.phrack.com/issues.html?issue=66&id=11

[WoTe09] R. Wojtczuk and A. Tereshkin. "Attacking Intel BIOS." *Black Hat USA*. Las Vegas, NV. 30 July 2009. http://www.blackhat.com/presentations/bh-usa-09/WOJTCZUK/BHUSA09-Wojtczuk-AtkIntelBios-SLIDES.pdf