(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 January 2004 (08.01.2004)

PCT

(10) International Publication Number
WO 2004/003711 A2

(51) International Patent Classification[7]: G06F 1/00

(21) International Application Number: PCT/US2003/019597

(22) International Filing Date: 20 June 2003 (20.06.2003)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
10/185,887 26 June 2002 (26.06.2002) US

(71) Applicant: **INTEL CORPORATION** [US/US]; (a Delaware Corpporation), 2200 Mission College Boulevard, Santa Clara, CA 95052 (US).

(72) Inventors: **GRAWCOCK, David**; 8285 Southwest 184th Avenue, Aloha, OR 97007 (US). **POISNER, David**; 205 Penry Square, Folsom, CA 95630 (US).

(74) Agent: **MALLIE, Michael, J.**; Blakely, Sokoloff, Taylor & Zafman LLP, 12400 Wilshire Boulevard, 7th Floor, Los Angeles, CA 90025 (US).

(81) Designated States *(national)*: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, TJ, TM, TN, TR, TT, TZ, UA, UG, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) Designated States *(regional)*: ARIPO patent (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *without international search report and to be republished upon receipt of that report*

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

WO 2004/003711 A2

(54) Title: SLEEP PROTECTION

(57) Abstract: Methods, apparatus and machine-readable medium are described that attempt to protect secrets from sleep attacks. In some embodiments, the secrets are encrypted and a security enhanced environment dismantled prior to entering a sleep state. Some embodiments further re-establish a security enhanced environment and decrypt the secrets in response to a wake event.

SLEEP PROTECTION

BACKGROUND

[0001] Financial and personal transactions are being performed on computing devices at an increasing rate. However, the continual growth of such financial and personal transactions is dependent in part upon the establishment of security enhanced (SE) environments that attempt to prevent loss of privacy, corruption of data, abuse of data, etc. An SE environment may employ various techniques to prevent different kinds of attacks or unauthorized access to protected data or secrets (e.g. social security number, account numbers, bank balances, passwords, authorization keys, etc.). One type of attack that an SE environment may attempt to prevent is a sleep attack.

[0002] For example, many computing devices support a suspend-to-memory sleep state such as, for example, the S3 sleep state described in the Advanced Configuration and Power Interface (ACPI) Specification, revision 2.0, 27 July 2000. Upon entering the suspend-to-memory sleep state, the computing device removes power from various components and/or subcomponents of the computing device but continues to power the system memory to retain the contents of the system memory. As a result of removing power, the computing device may remove power from circuitry used to protect secrets stored in the system memory. Upon waking from the sleep state, the computing device may return power to the circuitry used to protect secrets stored in system memory. However, after returning power, the protection circuitry may be in a reset state and may not actually protect secrets in system memory. An attacker may successfully gain access to stored secrets prior to re-establishing the protections provided by the protection circuitry.

BRIEF DESCRIPTION OF THE DRAWINGS

[0003] The invention described herein is illustrated by way of example and not by way of limitation in the accompanying figures. For simplicity and clarity of illustration, elements illustrated in the figures are not necessarily drawn to scale. For example, the dimensions of some elements may be exaggerated relative to other elements for clarity. Further, where considered appropriate, reference numerals have been repeated among the figures to indicate corresponding or analogous elements.

[0004] FIG. 1 illustrates an embodiment of a computing device.

[0005] FIG. 2 illustrates an embodiment of a security enhanced (SE) environment that may be established by the computing device of FIG. 1.

[0006] FIG. 3 illustrates an embodiment of a sleep method of the computing device of FIG. 1.

[0007] FIG. 4 illustrates an embodiment of a wake method of the computing device of FIG. 1.

DETAILED DESCRIPTION

[0008] The following description describes techniques for protecting secrets from sleep attacks. In the following description, numerous specific details such as logic implementations, opcodes, means to specify operands, resource partitioning/sharing/duplication implementations, types and interrelationships of system components, and logic partitioning/integration choices are set forth in order to provide a more thorough understanding of the present invention. It will be appreciated, however, by one skilled in the art that the invention may be practiced without such specific details. In other instances, control structures, gate level circuits and full software instruction sequences have not been shown in detail in order not to obscure the invention. Those of ordinary skill in the art, with the

included descriptions, will be able to implement appropriate functionality without undue experimentation.

[0009] References in the specification to "one embodiment", "an embodiment", "an example embodiment", etc., indicate that the embodiment described may include a particular feature, structure, or characteristic, but every embodiment may not necessarily include the particular feature, structure, or characteristic. Moreover, such phrases are not necessarily referring to the same embodiment. Further, when a particular feature, structure, or characteristic is described in connection with an embodiment, it is submitted that it is within the knowledge of one skilled in the art to effect such feature, structure, or characteristic in connection with other embodiments whether or not explicitly described.

[0010] References herein to "symmetric" cryptography, keys, encryption or decryption, refer to cryptographic techniques in which the same key is used for encryption and decryption. The well known Data Encryption Standard (DES) published in 1993 as Federal Information Publishing Standard FIPS PUB 46-2, and Advanced Encryption Standard (AES), published in 2001 as FIPS PUB 197, are examples of symmetric cryptography. Reference herein to "asymmetric" cryptography, keys, encryption or decryption, refer to cryptographic techniques in which different but related keys are used for encryption and decryption, respectively. So called "public key" cryptographic techniques, including the well-known Rivest-Shamir-Adleman (RSA) technique, are examples of asymmetric cryptography. One of the two related keys of an asymmetric cryptographic system is referred to herein as a private key (because it is generally kept secret), and the other key as a public key (because it is generally made freely available). In some embodiments either the private or public key may be used for encryption and the other key used for the associated decryption.

[0011] As used herein, the term "object" is intended to be a broad term encompassing any grouping of one or more bits regardless of structure, format, or representation. Further, the verb "hash" and related forms are used herein to refer to performing an operation upon an operand or message to produce a digest

value or a "hash". Ideally, the hash operation generates a digest value from which it is computationally infeasible to find a message with that hash and from which one cannot determine any usable information about a message with that hash. Further, the hash operation ideally generates the hash such that determining two messages which produce the same hash is computationally impossible. While the hash operation ideally has the above properties, in practice one way functions such as, for example, the Message Digest 5 function (MD5) and the Secure Hashing Algorithm 1 (SHA-1) generate hash values from which deducing the message are difficult, computationally intensive, and/or practically infeasible.

[0012] Embodiments of the invention may be implemented in hardware, firmware, software, or any combination thereof. Embodiments of the invention may also be implemented as instructions stored on a machine-readable medium, which may be read and executed by one or more processors. A machine-readable medium may include any mechanism for storing or transmitting information in a form readable by a machine (e.g., a computing device). For example, a machine-readable medium may include read only memory (ROM); random access memory (RAM); magnetic disk storage media; optical storage media; flash memory devices; electrical, optical, acoustical or other form of propagated signals (e.g., carrier waves, infrared signals, digital signals, etc.), and others.

[0013] An example embodiment of a computing device 100 is shown in FIG. 1. The computing device 100 may comprise one or more processors 102 coupled to a chipset 104 via a processor bus 106. The chipset 104 may comprise one or more integrated circuit packages or chips that couple the processors 102 to system memory 108, a token 110, firmware 112, non-volatile storage 114 (e.g. hard disk, floppy disk, optical disk, flash, programmable read only memory, etc.) and/or other devices 116 (e.g. a mouse, keyboard, video controller, etc.).

[0014] The processors 102 may support execution of a secure enter (SENTER) instruction to initiate creation of a SE environment such as, for example, the example SE environment of FIG. 2. The processors 102 may further support a secure exit (SEXIT) instruction to initiate dismantling of a SE

environment. In one embodiment, the processor 102 may issue bus messages on processor bus 106 in association with execution of the SENTER, SEXIT, and other instructions.

[0015] The processors 102 may further comprise a key 118 such as, for example, a symmetric cryptographic key, an asymmetric cryptographic key, or some other type of key. The processor 102 may use the processor key 118 to authentic an authentic code (AC) module prior to executing the AC module. In one embodiment, the processor key 118 comprises an asymmetric private key to which only the processor 102 has access.

[0016] The processors 102 may support one or more operating modes such as, for example, a real mode, a protected mode, a virtual real mode, and a virtual machine mode (VMX mode). Further, the processors 102 may support one or more privilege levels or rings in each of the supported operating modes. In general, the operating modes and privilege levels of a processor 102 define the instructions available for execution and the effect of executing such instructions. More specifically, a processor 102 may be permitted to execute certain privileged instructions only if the processor 102 is in an appropriate mode and/or privilege level.

[0017] The chipset 104 may comprise one or more chips or integrated circuits packages that interface the processors 102 to components of the computing device 100 such as, for example, system memory 108, the token 110, non-volatile storage 114, and the other devices 116. In one embodiment, the chipset 104 comprises a memory controller 120. However, in other embodiments, the processors 102 may comprise all or a portion of the memory controller 120. In general, the memory controller 120 provides an interface for components of the computing device 100 to access the system memory 108. Further, the memory controller 120 of the chipset 104 and/or processors 102 may define certain regions of the memory 108 as security enhanced (SE) memory 122. In one embodiment, the processors 102 may only access SE memory 122 when in an appropriate operating mode (e.g. protected mode) and privilege level (e.g. 0P).

[0018] Further, the chipset 104 may comprise a key 124 that may be used to authentic an AC module prior to execution. Similar to the processor key 118, the chipset key 124 may comprise a symmetric cryptographic key, an asymmetric cryptographic key, or some other type of key. In one embodiment, the chipset key 124 comprises an asymmetric private key to which only the chipset 104 has access. In another embodiment, the chipset 104 comprises a hash of an asymmetric chipset key 124 stored in another component of the computing device 100. The chipset 104 may retrieve the chipset key 124 and authenticate the key 124 using the hash.

[0019] The chipset 104 may further comprise a secrets store 126 to indicate whether the system memory 108 might contain unencrypted secrets. In one embodiment, the secrets store 126 may comprise a flag that may be set to indicate that the system memory 108 might contain unencrypted secrets, and that may be cleared to indicate that the system memory 108 does not contain unencrypted secrets. In other embodiments, the secrets store 126 may be located elsewhere such as, for example, the token 110, the processors 102, or other components of the computing device 100.

[0020] In one embodiment, the secrets store 126 is implemented as a single volatile memory bit having backup power supplied by a battery. The backup power supplied by the battery maintains the contents of the secrets store 126 across a system reset, a sleep event, a system shutdown, a system power down, or other power removal/loss event. The chipset 104 may further comprise battery detection circuitry (not shown) to detect an interruption in power supplied by the battery. The circuitry may further update the secrets store 126 to indicate that the system memory 108 may contain secrets in response to detecting a power interruption. In another embodiment, the secrets store 126 is implemented as a non-volatile memory bit such as a flash memory bit that does not require battery backup to retain its contents across a power removal/loss event. In one embodiment, the secrets store 126 is implemented with a single memory bit that may be set or cleared. However, other embodiments may comprise a secrets store 126 having a different storage capacity and/or utilizing a different status encoding.

[0021] The chipset 104 may further protect the secrets store 126 from unauthorized updates. In one embodiment, chipset 104 comprises a processor interface 128 to decode transactions of the processor bus 106 and/or receive messages from the processors 102. The processors 102 may generate bus transactions and/or messages in response to executing one or more privileged instructions that request the chipset 104 to update the secrets store 126. The processor interface 128 may receive the bus transaction and/or messages and may update the secrets store 126 based upon the decoded bus transaction and/or messages. In one embodiment, valid execution of the privileged instructions is restricted to software executing at a particular processor privilege level. For example, in one embodiment valid execution of the privileged instructions is restricted to a monitor executing at the most privileged processor level. (See, FIG. 2).

[0022] The chipset 104 may further allow unprivileged updates of the secrets store 126. In one embodiment, the processors 102 in response to executing one or more privileged instructions may generate bus transactions and/or messages that the request the chipset 104 to allow unprivileged updates of the secrets store 126. Further, the processors 102 in response to executing one/or more unprivileged or privileged instructions may generate bus transactions and/or messages that request the chipset 104 to deny unprivileged updates of the secrets store 126. The processors 102 in response to executing one or more unprivileged instructions may generate bus transactions and/or messages that request the chipset 104 to update the secrets store 126. The processor interface 128 may receive the bus transactions and/or messages and may allow unprivileged updates, deny unprivileged updates, and/or update the secrets store 126 based upon the decoded bus transactions and/or messages. In one embodiment, valid execution of the privileged instructions to request unprivileged updates is restricted to software executing at a particular processor privilege level. For example, in one embodiment valid execution of these privileged instructions is restricted to a monitor executing at the most privileged processor level, thus allowing the monitor to grant selected non-privileged code (e.g. an AC module) write access to the secrets store 126.

[0023] The chipset 104 may further comprise a sleep controller 130, a sleep type store 132, and a sleep enable store 134. The sleep controller 130 in one embodiment selectively powers components and/or subcomponents based upon the sleep type store 132 and the sleep enable store 134. In one embodiment, a value may be stored in the sleep type store 132 to indicate into which sleep state (e.g. ACPI sleep states S1, S2, S3, S4) the sleep controller 130 is to place the computing device 100. The sleep enable store 134 may be updated to invoke entry into the sleep state indicated by the sleep state store 132. For example, the sleep enable store 134 may comprise a flag that in response to being set causes the sleep controller 130 to place the computing device 100 in the requested sleep state.

[0024] The chipset 104 may further comprise sleep attack detection logic 136 that detects probable sleep attacks. In one embodiment, a sleep method updates the secrets store 126 to indicate that the the system memory 108 contains no unencrypted secrets prior to updating the sleep enable store 134 to initiate the sleep entry process. Therefore, the sleep attack detection logic 136 in one embodiment determines that a sleep attack is probable in response to (i) the secrets store 126 indicating that the system memory 108 might contain unencrypted secrets and (ii) the sleep enable store 134 requesting that the sleep entry process be invoked. In response to detecting a probable sleep attack, the sleep attack detection logic 136 initiates a sleep attack response such as, for example, generating a system reset event, a system halt event, a system shutdown event, a system power off event, or some other response to protect the secrets stored in system memory 108.

[0025] In another embodiment, the sleep attack detection logic 136 further determines based upon the sleep state to be entered whether to invoke a sleep attack response. For example, circuitry used to protect secrets stored in the SE memory 122 may remain effective during a given sleep state. Accordingly, the sleep attack detection logic 136 may either decide that no sleep attack is occurring or may decide not to invoke a sleep attack response if the sleep type

store 132 indicates a sleep state in which SE memory protections remain effective.

[0026] The chipset 104 may also support standard I/O operations on I/O buses such as peripheral component interconnect (PCI), accelerated graphics port (AGP), universal serial bus (USB), low pin count (LPC) bus, or any other kind of I/O bus (not shown). In particular, the chipset 104 may comprise a token interface 138 to connect chipset 104 with a token 110 that comprises one or more platform configuration registers (PCR) 140. In one embodiment, token interface 138 may comprise an LPC bus interface (LPC Interface Specification, Intel Corporation, rev. 1.0, 29 December 1997).

[0027] In general, the token 110 may record metrics in a security enhanced manner, may quote metrics in a security enhanced manner, may seal secrets to a particular environment (current or future), and may unseal secrets to the environment to which they were sealed. The token 110 may comprise one or more keys 142 that may be used to support the above operations. The token keys 142 may include symmetric keys, asymmetric keys, and/or some other type of key. The token 110 may further comprise one or more platform configuration registers (PCR registers) 140 to record and report metrics in a security enhanced manner. In one embodiment, the token 110 supports a PCR extend operation that records a received metric in an identified PCR register 140 in a security enhanced manner.

[0028] The token 110 may also support a PCR quote operation that returns a quote or contents of an identified PCR register 140. The token 110 may further support a seal operation and an unseal operation. In response to a seal operation, the token 110 generates a sealed object comprising an object sealed to the token 110 and a specified device environment. Conversely, the token 110 may return an object of a sealed object in response to an unseal operation only if the object was sealed with a key of the token 110 and the current device environment satisfies environment criteria specified for the sealed object. In one embodiment, the token 110 may comprise a Trusted Platform Module (TPM) as described in the Trusted

Computing Platform Alliance (TCPA) Main Specification, Version 1.1a, 1 December 2001 or a variant thereof.

[0029] In an embodiment, the firmware 112 comprises Basic Input/Output System routines (BIOS) 144. The BIOS 144 may comprise AC modules, sleep code, wake code, system start-up code and/or structures. For example, the BIOS 144 may comprise ACPI structures and ACPI Source Language (ASL) code which may be accessed and/or executed during sleep event processing, wake event processing, and/or computing device initialization.

[0030] One embodiment of an SE environment 200 is shown in FIG. 2. The SE environment 200 may be initiated in response to various events such as, for example, system startup, an application request, an operating system request, etc. As shown, the SE environment 200 may comprise a trusted virtual machine kernel or monitor 202, one or more standard virtual machines (standard VMs) 204, and one or more trusted virtual machines (trusted VMs) 206. In one embodiment, the monitor 202 of the SE environment 200 executes in the protected mode at the most privileged processor ring (e.g. 0P) to manage security and provide barriers between the virtual machines 204, 206.

[0031] The standard VM 204 may comprise an operating system 208 that executes at the most privileged processor ring of the VMX mode (e.g. 0D), and one or more applications 210 that execute at a lower privileged processor ring of the VMX mode (e.g. 3D). Since the processor ring in which the monitor 202 executes is more privileged than the processor ring in which the operating system 208 executes, the operating system 208 does not have unfettered control of the computing device 100 but instead is subject to the control and restraints of the monitor 202. In particular, the monitor 202 may prevent the operating system 208 and its applications 210 from directly accessing the SE memory 122 and the token 110.

[0032] The monitor 202 may further comprise sleep logic 212 and one or more monitor keys 214 to encrypt and/or otherwise protect information. The sleep logic

212 comprises code to perform one or more sleep operations such as, for example, encrypting and attesting to memory contents. The monitor keys 214 may comprise symmetric cryptographic keys, asymmetric cryptographic keys, or other keys to which the monitor 202 has exclusive control. For example, the monitor keys 214 may comprise a symmetric root key and one or more asymmetric keys that are encrypted with the symmetric root key.

[0033] The monitor 202 may perform one or more measurements of the trusted kernel 216 such as a hash of the kernel code to obtain one or more metrics, may cause the token 110 to extend a PCR register 140 with the metrics of the kernel 216, and may record the metrics in an associated PCR log stored in SE memory 122. The monitor 202 may further establish the trusted VM 206 in SE memory 122 and launch the trusted kernel 216 in the established trusted VM 206.

[0034] Similarly, the trusted kernel 216 may take one or more measurements of an applet or application 218 such as a hash of the applet code to obtain one or more metrics. The trusted kernel 216 via the monitor 202 may then cause the physical token 110 to extend a PCR register 140 with the metrics of the applet 218. The trusted kernel 216 may further record the metrics in an associated PCR log stored in SE memory 122. Further, the trusted kernel 216 may launch the trusted applet 218 in the established trusted VM 206 of the SE memory 122.

[0035] In response to initiating the SE environment 200 of FIG. 2, the computing device 100 further records metrics of the monitor 202 and hardware components of the computing device 100 in one or more PCR registers 140 of the token 110. For example, the processor 102 may obtain hardware identifiers such as, for example, processor family, processor version, processor microcode version, chipset version, and physical token version of the processors 102, chipset 104, and physical token 110. The processor 102 may then record the obtained hardware identifiers in one or more PCR registers 140.

[0036] Referring now to FIG. 3, an embodiment of a method to enter a sleep state is illustrated. The computing device 100 may perform the method in

response to a sleep event. For example, a sleep event may be generated in response to a device and/or an operating system detecting that a device has remained idle for a predetermined length of time. In response to the sleep event, the operating system 208 may determine in block 300 whether an SE environment 200 is currently established. In response to determining that no SE environment 200 is established, the computing device 100 in block 302 may invoke a sleep entry process (described in more detail below) to place the computing device 100 into a requested sleep state.

[0037] In response to determining that an SE environment 200 is established, the request, the monitor 202 in block 304 may encrypt and attest to the contents of the SE memory 122. In one embodiment, the monitor 202 encrypts the pages of the SE memory 122 using one of the monitor keys 214 and replaces the pages with encrypted pages. The monitor 202 may leave portions of the SE memory 122 that contain the monitor 202 or the portions of the SE memory 122 that contain the sleep logic 212 of the monitor 202 unencrypted so that processors 102 may continue to execute the sleep logic 212.

[0038] The monitor 202 in block 304 may further attest to the contents of the SE memory 122. In one embodiment, the monitor 202 may generate a contents attestation by hashing the encrypted contents of the SE memory 122 to obtain a memory hash. In another embodiment, the monitor 202 may generate the contents attestation by hashing only the pages that will remain in the SE memory 122 after the wake process. For example, the wake process may reload the monitor 202 and/or other code from non-volatile storage 114. Since these portions of the SE memory 122 are reloaded, the computing device 100 may erase these portions from system memory 108 and/or may not save them to non-volatile storage 114 prior to entering the sleep state. In another embodiment, the monitor 202 may attest to the contents of the SE memory 122 by embedding a content attestation such as, for example, a watermark, signature, and/or other information in the attested contents of the SE memory 122.

[0039] In block 306, the monitor 202 may generate and attest to a data structure (e.g. a page table, page list, segment list, region list, etc.) that identifies pages/segments/regions of system memory 122 encrypted in block 304. In one embodiment, the monitor 202 may generate a data structure attestation by hashing the data structure to obtain a data structure hash. In another embodiment, the monitor 202 may attest to the data structure by embedding a data structure attestation such as, for example, a watermark, signature, and/or other information in the attested data structure.

[0040] The monitor 202 in block 308 may seal the content attestation, the data structure attestation, and/or the monitor keys 214 to protect them from unauthorized access and/or alteration. In one embodiment, the monitor 202 seals the content attestation, the data structure attestation, and the monitor keys 214 via one or more seal operations of the token 110 to obtain one or more sealed resume objects. In one embodiment, the seal operations use a PCR register 140 containing a metric of the monitor 202 to effectively prevent another monitor such as, for example, a rogue monitor from accessing and/or altering the unencrypted contents of the sealed resume objects.

[0041] In block 310, the monitor 202 dismantles the SE environment 200. The monitor 202 may perform various operations as part of the dismantling process. In one embodiment, the monitor 202 updates the secrets store 126 to indicate that the system memory 108 does not contain unencrypted secrets. For example, the monitor 202 may clear a flag of the secrets store 126 to indicate the system memory 108 does not contain unencrypted secrets. Further, the monitor 202 may shutdown the trusted virtual machines 206 and may exit the VMX processor mode. The monitor 202 may further erase regions of the system memory 108 that will be reloaded from non-volatile storage 114 during the wake process.

[0042] In block 312, the computing device 100 may cease execution of the monitor 202 and return to execution of the operating system 208. In one embodiment, as a result of returning to the operating system 208, the monitor 202 provides the operating system 208 with SE environment resume information that

identifies the location and size of the monitor 202 to be executed in response to waking and the location and size of the sealed resume objects. However, the computing device 100 may utilize other mechanisms to enable the operating system 208 to retrieve the monitor 202 and sealed resume objects during the wake process. For example, the monitor 202 and/or sealed resume objects may be stored at predetermined locations or at locations set by the BIOS 144.

[0043] The operating system 208 in block 314 may save the resume information so that it may be retrieved as part of the wake process. The operating system 208 may store the SE environment resume information at predetermined locations of the system memory 108, at locations set by the BIOS 144, non-volatile registers of the chipset 104, and/or other locations. In one embodiment, the monitor 202 in block 312 stores the information at the appropriate locations, thus relieving the operating system 208 of saving the information in block 314.

[0044] The operating system 208 and/or the BIOS 144 in block 302 may complete the sleep entry process. For example, the operating system 208 and/or the BIOS 144 may write a sleep type identifier to the sleep type store 132 to indicate which sleep state the computing device 100 is entering and may update the sleep enable store 134 to invoke entry into the sleep state. In one embodiment, the operating system 208 and/or BIOS 144 may cause the computing device 100 to enter a sleep state that is different than the sleep state requested. The operating system 208 and/or BIOS 144 may elect to change the sleep state for various reasons such as, for example, one or more components of the computing device 100 not supporting the requested sleep state. In response to updating the sleep type store 132 and sleep enable stores 134, the sleep controller 130 may cause the computing device 100 to enter the sleep state and may complete the sleep process. For example, the sleep controller 130 may remove power from components and/or subcomponents of the computing devices 100, may request components and/or subcomponents to enter a low power mode of operation, and/or may cause the contents of system memory 108 to be written to non-volatile storage 114.

[0045] Referring now to FIG. 4, a method of waking from a sleep state is illustrated. The computing device 100 may perform the wake method in response to a wake event. A wake event may be generated in response various stimuli such as, for example, a modem detecting a ring event, a network controller detecting network activity, a keyboard controller detecting a key press, etc. In response to the wake event, the sleep controller 130 in block 400 may perform one or more wake operations such as, for example, waking the processors 102 and transferring saved state information from the non-volatile storage 114 to the system memory 108. The sleep controller 130 in one embodiment may perform one or more of the wake operations in response to executing ASL and/or other code of the BIOS 144. After performing the wake operations, the sleep controller 130 may transfer control to the operating system 208. In one embodiment, the sleep logic 212 invokes execution of the operating system 208 from a location identified by a wake vector.

[0046] The operating system 208 in block 402 may perform one or more wake operations, such as, waking network controllers, modems, and/or other devices of the computing device 100. In block 404, the operating system 208 determines whether to restore an SE environment 200 based upon stored resume information and/or the lack of stored resume information. In response to determining to restore the SE environment 200, the operating system 208 performs various operations. For example, the operating system 208 may load, authenticate, and initiate execution of AC modules that configure the computing device 100 and/or verify the configuration of the computing device 100. Further, the operating system 208 in block 406 may load and invoke execution of the monitor 202 identified by the resume information.

[0047] In block 408, the monitor 202 may unseal the sealed resume objects to obtain the contents attestation, the data structure attestation, and the monitor keys 214 via one or more unseal operations of the token 110. In response to detecting that the unseal operation failed (block 410), the monitor 202 in block 412 invokes a sleep attack response to address a probable sleep attack. In one embodiment, the monitor 202 invokes the sleep attack response by writing to a reset register of

the chipset 104 to invoke a system reset. However, the monitor 202 may respond in other ways such as, for example, halting the processors 102, erasing system memory 108, invoking a system shutdown, removing power from the computing device 100, and/or other actions that protect the secrets from unauthorized access and/or alteration.

[0048] In block 414, the monitor 202 verifies the authenticity of the data structure base upon the data structure attestation. In one embodiment, the monitor 202 hashes the data structure to obtain a computed data structure attestation. The monitor 202 further compares the computed data structure attestation to the data structure attestation obtained from the sealed resume objects and determines that the data structure is authentic in response to the computed attestation having a predetermined relationship (e.g. equal) to the unsealed attestation. In response to determining that the data structure may not be authentic and by be altered, the monitor 202 in block 412 invokes a sleep attack response to address the probable sleep attack.

[0049] The monitor 202 in block 416 may decrypt portions of system memory 108 and store the decrypted portions in SE memory 122. The monitor 202 may decrypt the portions of the system memory 108 identified by the data structure using one or more unsealed monitor keys 214. In block 418, the monitor 202 may verify the authenticity of the encrypted or decrypted SE memory contents. In one embodiment, the monitor 202 may hash the decrypted contents added to the SE memory 122 to obtain a computed contents attestation. In another embodiment, the monitor 202 may hash the encrypted contents to be added to the SE memory 122 to obtain a computed contents attestation. The monitor 202 may further compare the computed contents attestation to the unsealed contents attestation and may determine that the contents are authentic (e.g. unaltered) in response to the computed attestation having a predetermined relationship (e.g. equal) to the unsealed attestation. In response to determining that the contents are not authentic (e.g. altered), the monitor 202 in block 412 may invoke an attack response to the probable sleep attack. Conversely, in response to determining

that the contents are authentic, the monitor 202 completes the wake process by invoking execution of the operating system 208.

[0050] The above embodiments of the sleep and wake methods help protect secrets from attack. However, an attacker may attempt to circumvent the sleeping method of FIG. 3 to place the computing device 100 in a sleep state in which unencrypted secrets reside in system memory 108 and/or non-volatile storage 114 unprotected. To protect against such circumvention, the sleep attack detection logic 136 may invoke a system reset event or another attack response in response to detecting a probable sleep attack. In one embodiment of the sleep method of FIG. 3, the monitor 202 updates the secrets store 126 to indicate that the system memory 108 contains no unencrypted secrets prior to updating the sleep enable store 134 to initiate the sleep entry process. Accordingly, the sleep attack detection logic 136 may invoke a sleep attack response in response to the sleep enable store 134 being updated if the secrets store 420 indicates that the system memory 108 might contain unencrypted secrets.

[0051] In another embodiment of the sleep method of FIG. 3, the monitor 202 encrypts the SE memory 122 and updates the secrets store 126 to indicate that the system memory 108 contains no unencrypted secrets only if the requested sleep state would result in the SE memory 122 being unprotected. Accordingly, sleep attack detection logic 136 may invoke a sleep attack response in response to the sleep enable store 134 being updated if the secrets store 420 indicates that the system memory 108 might contain unencrypted secrets and the sleep type store 132 indicates a sleep state in which the SE memory 122 map be unprotected.

[0052] While certain features of the invention have been described with reference to example embodiments, the description is not intended to be construed in a limiting sense. Various modifications of the example embodiments, as well as other embodiments of the invention, which are apparent to persons skilled in the art to which the invention pertains are deemed to lie within the spirit and scope of the invention.

What is claimed is:

1. A method comprising

detecting a probable sleep attack, and

invoking a sleep attack response that protects secrets from the probable sleep attack.

2. The method of claim 1 further comprising

determining whether a memory might contain secrets; and

invoking the sleep attack response in response to determining that the memory might contain secrets.

3. The method of claim 1 further comprising

determining whether a memory might contain secrets in response to a sleep event; and

invoking the sleep attack response in response to determining that the memory might contain secrets.

4. The method of claim 1 further comprising

encrypting one or more portions of a memory in response to a sleep event.

5. The method of claim 4 further comprising

generating a contents attestation that attests to the one or more portions of the memory.

6. The method of claim 4 further comprising

generating a structure that identifies the one or more portions of the memory; and

generating one or more attestations that attest to the structure and the one or more portions of the memory.

7. The method of claim 6 further comprising

sealing the structure and the one or more attestations to a monitor of a computing device.

8. The method of claim 1 further comprising

generating a system reset in response to invoking the sleep attack response.

9. A chipset comprising

sleep attack detection logic to detect a sleep attack and to invoke an attack response in response to a detected sleep attack.

10. The chipset of claim 9 further comprising a secrets store to indicate whether a memory might contain secrets,

the sleep attack detection logic to detect a sleep attack based upon the secrets store.

11. The chipset of claim 10 further comprising a sleep enable store to invoke a sleep entry,

the sleep attack detection logic to detect a sleep attack based further upon the sleep enable store.

12. The chipset of claim 11 further comprising a sleep type store to indicate a requested sleep state,

the sleep attack detection logic to detect a sleep attack based further upon the sleep type store.

13. The chipset of claim 11 further comprising an interface that prevents untrusted modification of the secrets store.

14. The chipset of claim 11 further comprising an interface that requires receipt of one or more messages prior to allowing updates to the secrets store.

15. A system comprising an operating system and a more privileged monitor,

the operating system to receive a sleep event and to transfer processing of the sleep event to the monitor, and

the monitor, in response to the sleep request, to encrypt one or more pages of a memory and to indicate that the memory contains no unencrypted secrets.

16. The system of claim 15, wherein the monitor is to further update a secrets store to indicate that the memory contains no unencrypted secrets.

17. The system of claim 15, wherein

the monitor is to return processing of the sleep event to the operating system, and

the operating system is to write encrypted and non-encrypted pages of memory to non-volatile storage.

18. The system of claim 15, wherein

the monitor is to return processing of the sleep event to the operating system, and

the operating system is to cause the system to enter a sleep state.

19. The system of claim 18, wherein the operating system is to update a sleep type store to indicate the sleep state to be entered, and is to update a sleep enable store to invoke entry into the sleep state.

20. The system of claim 15, wherein the monitor is to further generate a contents attestation that attests to the encrypted pages of the memory.

21. The system of claim 20, wherein the monitor is to further generate a structure that identifies the encrypted pages, and is to generate a structure attestation that attests to the structure.

22. The system of claim 21, wherein the monitor is to further seal to the monitor the contents attestation, the structure attestation, and a monitor key to decrypt the encrypted pages.

23. A system comprising

volatile memory comprising security enhanced regions,

a secrets store to indicate whether the volatile memory might contain unencrypted secrets,

a sleep enable store to invoke entry into a sleep state,

a processor to encrypt the security enhanced regions in response to a sleep event and to update the secrets store to indicate that the volatile memory contains no unencrypted secrets in response to encrypting the security enhanced regions, and

sleep attack detection logic to invoke a sleep attack response in response to the sleep enable store being updated to invoke entry into sleep state and the secrets store indicating that the volatile memory might contain unencrypted secrets.

24. The system of claim 23, wherein the processor is to further generate a contents attestation that attests to the security enhanced regions and is to invoke

a sleep attack response in response to a wake event if the contents attestation indicates that the security enhanced regions are not authentic.

25. The system of claim 24, wherein the processor is to further seal the contents attestation and a key to decrypt the security enhanced regions to the system.

26. The system of claim 25, wherein the processor is to further invoke a sleep attack response in response to a wake event if unsealing the contents attestation and the key fails.

27. A machine-readable medium comprising a plurality of instructions that in response to being executed, result in a system

encrypting contents of a memory in response to a sleep event, and

generating a contents attestation that attests to the contents of the memory.

28. The machine-readable medium of claim 27 wherein the plurality of instructions in response to being executed further result in the system

using the contents attestation to verify the authenticity of the contents in response to a wake event, and

invoking a sleep attack response in response to determining that the contents of the memory are not authentic.

29. The machine-readable medium of claim 28 wherein the plurality of instructions in response to being executed further result in the system

sealing the contents attestation and a key to decrypt the contents of the memory to the system in response to a sleep event,

unsealing the contents attestation and the key in response to a wake event, and

invoking a sleep attack response in response to a failure in unsealing the contents attestation and the key.

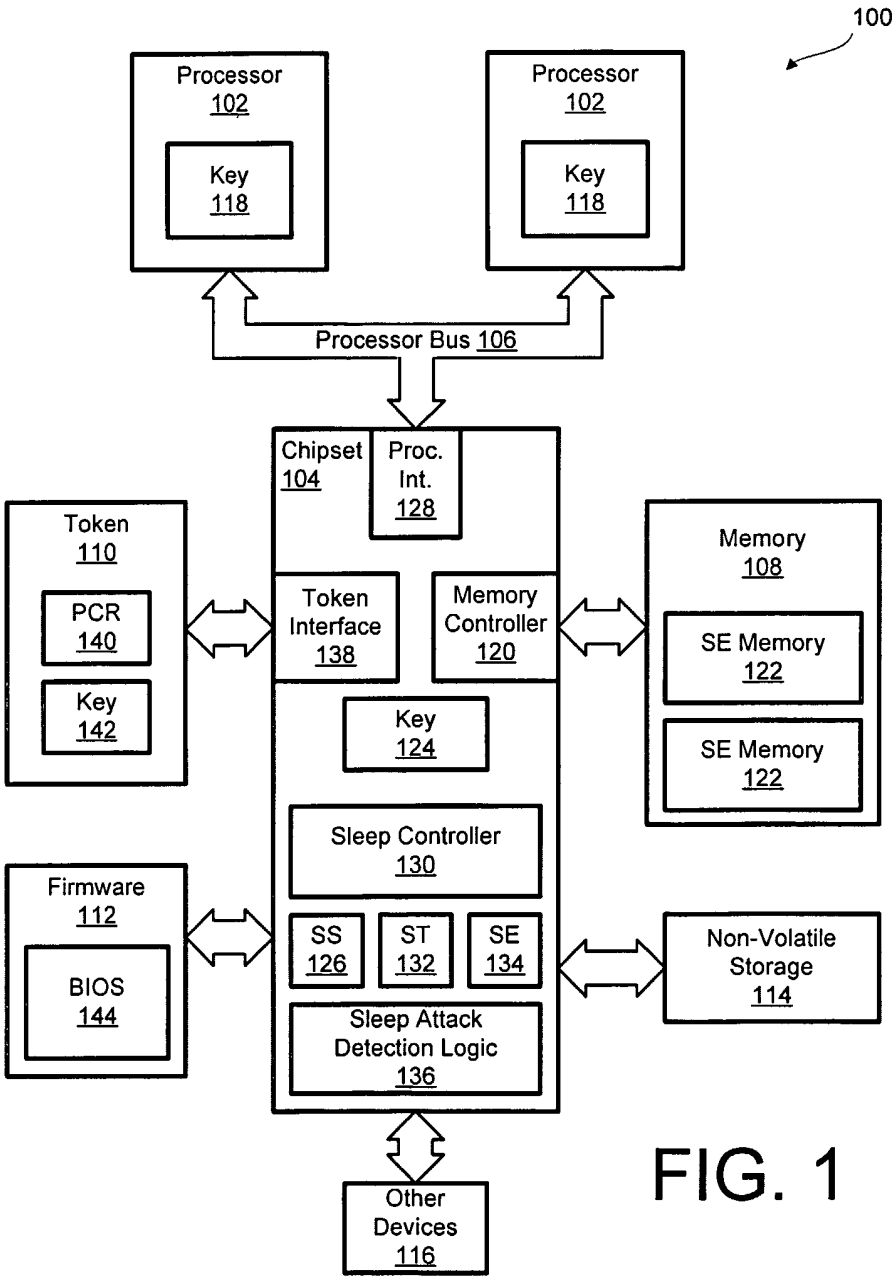
100
Processor
102
Key
118
Processor
102
Key
118
Processor Bus 106
Chipset
104
Proc.
Int.
128
Token
110
PCR
140
Key
142
Token
Interface
138
Memory
Controller
120
Memory
108
SE Memory
122
SE Memory
122
Key
124
Sleep Controller
130
Firmware
112
BIOS
144
SS
126
ST
132
SE
134
Non-Volatile
Storage
114
Sleep Attack
Detection Logic
136
Other
Devices
116

FIG. 1

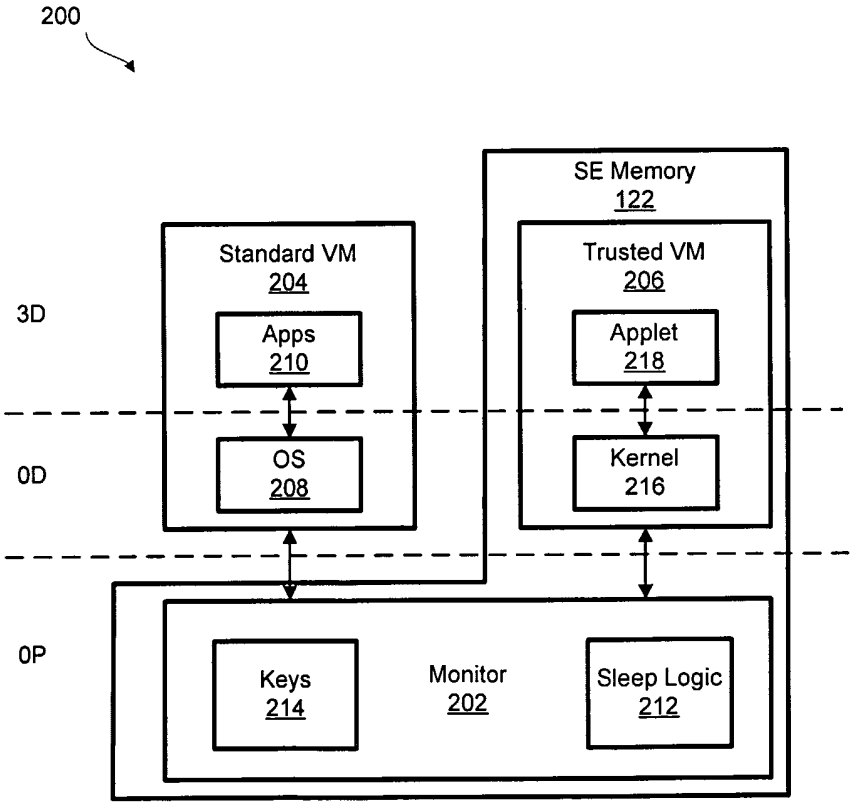
200
SE Memory
122
Standard VM
204
Trusted VM
206
3D
Apps
210
Applet
218
OS
208
Kernel
216
0D
0P
Keys
214
Monitor
202
Sleep Logic
212

FIG. 2

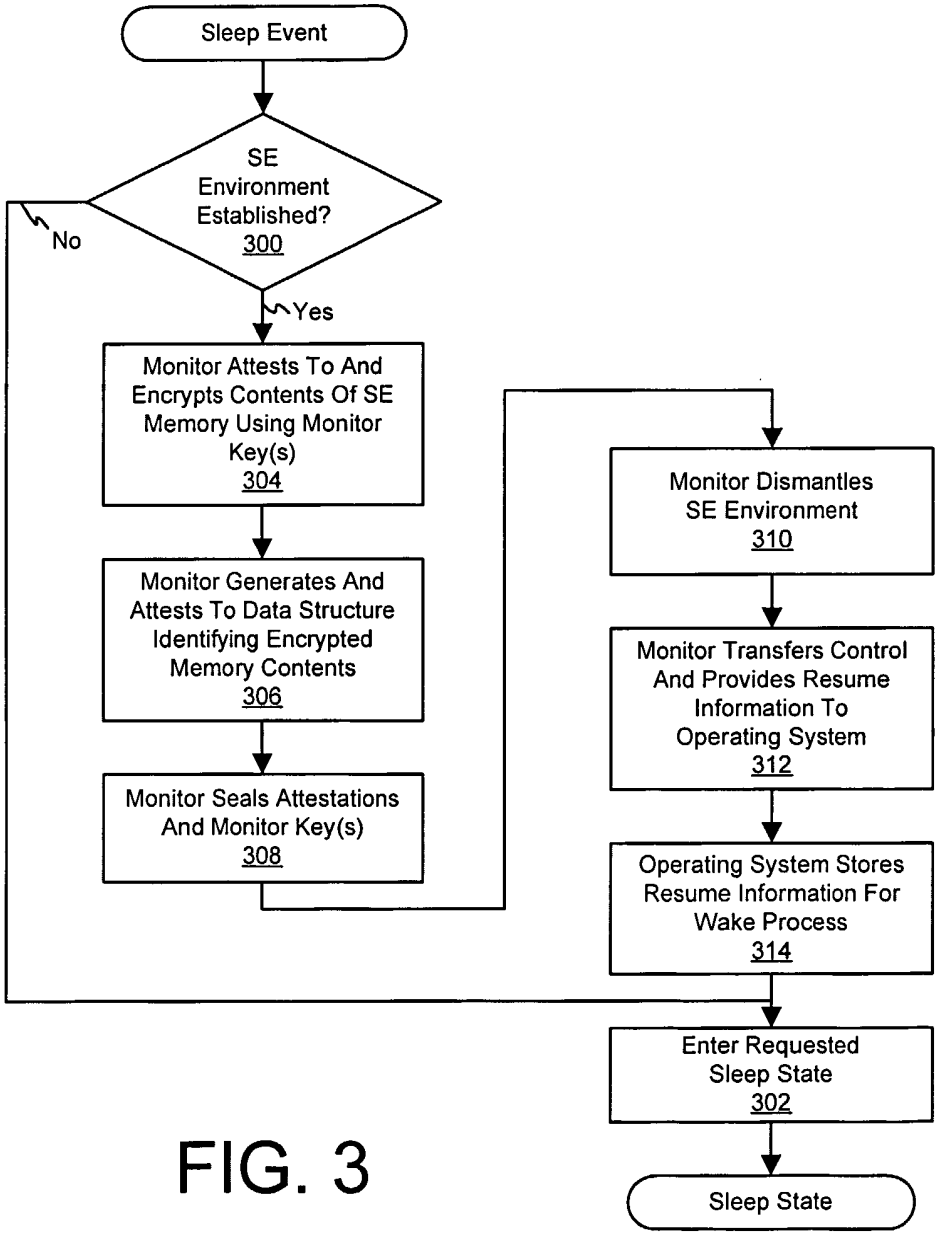
Sleep Event
SE Environment Established? 300
No
Yes
Monitor Attests To And Encrypts Contents Of SE Memory Using Monitor Key(s) 304
Monitor Generates And Attests To Data Structure Identifying Encrypted Memory Contents 306
Monitor Seals Attestations And Monitor Key(s) 308
Monitor Dismantles SE Environment 310
Monitor Transfers Control And Provides Resume Information To Operating System 312
Operating System Stores Resume Information For Wake Process 314
Enter Requested Sleep State 302
Sleep State

FIG. 3

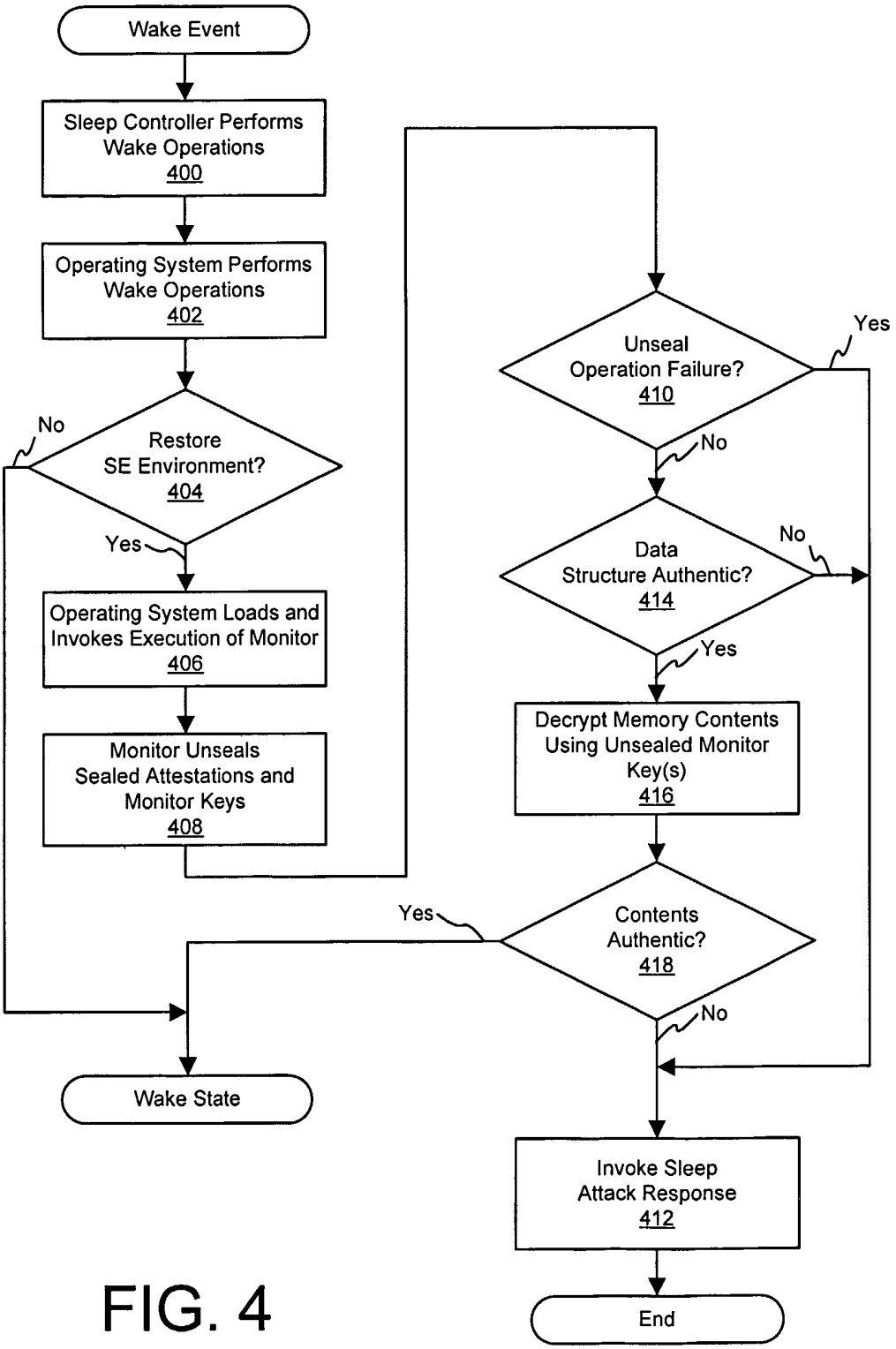
Wake Event
Sleep Controller Performs Wake Operations 400
Operating System Performs Wake Operations 402
Restore SE Environment? 404
No
Yes
Operating System Loads and Invokes Execution of Monitor 406
Monitor Unseals Sealed Attestations and Monitor Keys 408
Unseal Operation Failure? 410
Yes
No
Data Structure Authentic? 414
No
Yes
Decrypt Memory Contents Using Unsealed Monitor Key(s) 416
Contents Authentic? 418
Yes
No
Wake State
Invoke Sleep Attack Response 412
End

FIG. 4