Techniques for secure non-volatile random access memory are provided. In one aspect, a request to write to secure non-volatile random access memory may be received. A processor may enter a system management mode. A lock password may be retrieved from a system management mode memory. The lock password may be written to a locking device, wherein the locking device may unlock the secure non-volatile random access memory. The lock password may have been generated at the time the system was booted. The secure non-volatile random access memory may then be written. The secure non-volatile random access memory may then be re-locked.
SECURE NON-VOLATILE RANDOM ACCESS MEMORY

BACKGROUND

[0001] The Unified Extensible Firmware Interface (UEFI) is a software interface between the operating system and the platform firmware of a computer system. UEFI is intended to be the replacement for the Basic Input Output System (BIOS) that has been present in computing systems for the past several decades, although UEFI still provides support for legacy BIOS. UEFI provides many capabilities that are not present in legacy BIOS.

[0002] One such capability is secure boot. Each UEFI component is digitally signed by the provider of that component. Any modifications to the component, such as that which may be performed by a malevolent user, such as a hacker, would cause the digital signature to no longer be valid for the component. When UEFI secure boot is enabled, each component's digital signature is verified. If the digital signature cannot be verified, it is assumed that the component has been tampered with, and may contain malevolent instructions. To ensure no harm comes to the system or the user of the system, the component is not allowed to run if the digital signature cannot be verified.

BRIEF DESCRIPTION OF THE DRAWINGS

[0003] FIG. 1 is an example of a system which may utilize the secure non-volatile random memory techniques described herein.

[0004] FIG. 2 is another example of a system which may utilize the secure non-volatile random memory techniques described herein.
FIG. 3 is an example of a high level flow diagram for writing to a secure non-volatile random access memory according to techniques described herein.

FIG. 4 is another example of a high level flow diagram for locking and writing to a secure non-volatile random access memory according to techniques described herein.

FIG. 5 is an example of a high level flow diagram for using the secure non-volatile memory techniques described herein to implement a secure boot system.

FIG. 6 is another example of a high level flow diagram for using the secure non-volatile memory techniques described herein to implement a secure boot system.

DETAILED DESCRIPTION

The techniques described herein provide for a secure location to store the keys that may be used to verify UEFI components. As mentioned above, to ensure that a UEFI component has not been altered, a digital signature of the component may be verified. In one implementation, a component's digital signature may be generated by creating a hash of the component's firmware image then encrypting the hash with a private key known by the component provider. The digital signature may then be added to the firmware image. To verify the component, the digital signature may be extracted from the image. The remainder of the image may then be hashed using the same hashing mechanism used when generating the digital signature. The extracted digital signature may then be decrypted using a public key associated with the component provider. If the hash generated during the verification process matches the decrypted digital signature, it can be ensured that the component has not been altered (e.g. because the hashes would not have matched had the component been altered.). Because the public keys, which will be referred to generally as keys, are used to determine authenticity, the keys should be stored in such a way that they cannot be tampered with.
The present disclosure provides for a secure non-volatile random access memory (NVRAM) to store the keys in a secure manner. Write access to the NVRAM is limited to only trusted portions of computer code. By limiting access to only trusted code, it can be ensured that bad actors are not able to alter the values stored in the NVRAM. Furthermore, although the techniques described herein are being applied to UEFI component verification, it should be understood that the techniques could also be applied to secure any other type of data.

FIG. 1 is an example of a system which may utilize the secure non-volatile random memory techniques described herein. The system 100 may include a processor 110, a locking device 130, and a secure non-volatile memory. The processor 110 may be any type of processor suitable for use in a computing device, such as a desktop or laptop computer, a server computer, a smartphone, a tablet, or any other device that may make use of a secure NVRAM. The techniques described herein are not limited to any particular type of computing device.

The processor may be coupled to a system management mode (SMM) memory 112. The SMM memory may be in any suitable form, such as dynamic random access memory (DRAM), static random access memory (SRAM), Flash memory, or any other form of volatile or non-volatile memory. The SMM memory may be a standalone component, or it may be a portion of a computers main memory. The defining characteristic of SMM memory is that it is a memory that is only accessible when the processor 110 is running in system management mode. Because only trusted code is allowed to run in SMM, it can be ensured that only trusted code has access to the SMM memory.

The system management mode memory may include a lock password 114. Generation of the lock password is described in further detail below. For purposes of the description of FIG. 1, it may be assumed that the lock password has been stored in SMM memory, and is thus only accessible to the processor when the processor is running trusted code in system management mode.
[0014] The system 100 may also include a secure non-volatile random access memory (NVRAM) 150. The particular form of the memory is unimportant. For example, the memory may be a Flash memory or a Memristor based memory. The techniques described herein are applicable to any type of NVRAM. Regardless of the type of memory used for secure NVRAM, one characteristic is that the secure NVRAM, or at least portions of the secure NVRAM, may be write locked. Write locking a memory means that the portion of the memory that is locked cannot be written. However, the write locked portion of the memory can still be read. In some implementations, the secure NVRAM may be a component that contains an input, such as an input pin, which determines when write locking for the secure NVRAM is enabled or disabled. In other implementations, the secure NVRAM may be locked / unlocked by writing data to a particular memory location on the secure NVRAM. Regardless of how the locking is achieved, it should be understood that the secure NVRAM can be locked by an external entity.

[0015] The secure NVRAM may be accessible by the processor, in some implementations, the secure NVRAM may be accessible by the processor directly, while in other implementations, the secure NVRAM may be accessible by the processor through another component, as will be described in further detail with respect to FIG. 2. What should be understood is that the processor is able to access the secure NVRAM, and depending on the lock state of the secure NVRAM may be able to write to the secure NVRAM.

[0016] Secure keys 152 may be stored within the secure NVRAM. In the present description, the secure keys may be the keys used to verify a UEFI component. However, it should be understood that the secure keys may be any type of data that is to be stored in a secure manner, in other words, the secure keys may be any type of data that is not to be altered by unauthorized persons or computer code.

[0017] System 100 may also include a locking device 130 coupled to the processor. The locking device may be any type of device that may be used to lock the secure NVRAM. For example, in one implementation, the locking device may be a Complex programmable logic device (CPLD). In other
implementations, the locking device may be a field programmable gate array (FPGA). The particular form of the locking device is unimportant to the techniques described herein. What should be understood is that the locking device is able to write lock and unlock the secure NVRAM.

[0018] The locking device may include a lock password 132. The lock password may be the same as the lock password 114. The locking device may use the lock password to determine if the secure NVRAM is to be locked or unlocked. For example, in order to unlock the secure NVRAM, a user may need to provide a password to the locking device. If the provided password matches the lock password 132, the locking device may unlock the secure NVRAM. In some implementations, the lock password may be stored in a write once register on the locking device, such that once the lock password is written, and cannot be altered until the next time the system is booted.

[0019] In operation, the processor 110 may enter system management mode. For example, the processor may enter SMM after receipt of a system management mode interrupt. As mentioned above, when the processor enters system management mode, the processor may be granted access to the SMM memory 112. Because only trusted code is allowed to run in system management mode, it can be ensured that a bad actor is not able to gain access to the SMM memory. The processor may then retrieve the lock password 114 from the SMM memory. The processor may then provide the lock password 114 to the locking device 130.

[0020] The locking device 130 may compare the lock password 114 to the stored lock password 132. If the passwords match, the locking device may unlock secure NVRAM 150. Once the secure NVRAM is unlocked, the processor may be able to write to the secure NVRAM, including writing the secure keys 152. Once the processor has completed writing the secure keys, the processor may proceed to relock the secure NVRAM. In a UEFI secure boot implementation, the processor may write the keys used to verify a UEFI component into the secure NVRAM. Thus, the keys are ensured to be unaltered, because they were written while the processor was running in secure
management mode. The overall process is described in more detail with respect to FIG. 2.

FIG. 2 is another example of a system which may utilize the secure non-volatile random memory techniques described herein. Just as above, the system 200 may include a processor 210, a SMM memory 212, a locking device 230, and a secure NVRAM 250. These components may be similar to the processor 110, SMM memory 112, locking device 130, and secure NVRAM 150 described above with respect to FIG. 1, and as such the descriptions of those components are not repeated here. System 200 may also include a non-transitory processor readable medium 216, a management processor 270, a storage non-volatile random access memory 280, and a staging non-volatile random access memory 290.

The management controller 270 may be an independent processor within the computer system 200 that is used for management functions. The management controller may commonly be referred to as a baseboard management controller (BMC). The management controller may be connected to the external world through a network (not shown). The management controller may be coupled to the processor 210 as well as secure NVRAM 250, storage NVRAM 280, and staging NVRAM 290.

Secure NVRAM 250 is substantially the same as secure NVRAM 150, including the secure keys 252, which are substantially similar to the secure keys 152. Storage NVRAM 280 may be another random access memory that is accessible to the management controller. Storage NVRAM may be used to store data, such as the UEFI components themselves, that does not need to be stored in as secure a manner as the data stored in the secure NVRAM. The collection of UEFI components may be referred to as an image, or system image. The storage NVRAM may be used to store a default system image 282. The staging NVRAM 290 may be used to store data, such as pending system images 292, while the system prepares to apply the system image to the storage NVRAM. Operation of the various NVRAMs is described in further detail below.
The non-transitory processor readable medium may contain thereon a set of processor executable instructions. These instructions, when executed by the processor, may cause the processor to implement the techniques described herein. For example, the medium may include secure NVRAM initialization instructions, secure NVRAM write instructions, and UEFI secure boot instructions.

The secure NVRAM initialization instructions may cause the processor to initialize the secure NVRAM. The instructions may cause the processor to generate a lock password, which is then stored in the SMM memory. A new password may be generated upon each reboot of the system, so that even if the password is compromised, it is only valid until the next time the system boots. The processor may then store the lock password in the locking device. The locking device may then write lock the secure NVRAM. The locking device may unlock the secure NVRAM upon presentation of a matching lock password.

The secure NVRAM write instructions may cause the processor to unlock the secure NVRAM and write data to the secure NVRAM. The processor may first enter system management mode through a SMM interrupt. The lock password may be retrieved from SMM memory. The processor may then provide the lock password to the locking device. If the provided lock password matches the lock password stored in the locking device, the locking device may then unlock the secure NVRAM. The processor may then be able to write data to the secure NVRAM. Upon completion of the write, the processor may cause the locking device to re-lock the secure NVRAM.

The UEFI secure boot instructions may cause the system to implement UEFI secure boot. Description of the UEFI boot process is described in further detail below, such as with respect to FIGS. 5 and 6. For purposes of the remainder of the description of FIG. 2, it can be assumed that a UEFI component is associated with a key. If the component is authentic, the digital signature of the component can be authenticated with the key. If the digital
signature cannot be verified, then the component has been altered and should be considered unsafe to use.

[0028] The operation of system 200 may be best described through the use of an example. A user may desire to load a new version of UEFI components onto the system. For purposes of this description, a system image will refer to a digitally signed collection of UEFI components. The system image contains the components themselves, as well as the keys (e.g. public keys) used to authenticate those components. The system image may be signed using public-private key pairs. For example, the system image may be encrypted with a private key whose corresponding public key is known by the management controller. Thus, if the system image is successfully decrypted, it is known that the image came from a source in possession of the management controller's private key, and thus the system image can be assumed to be authentic.

[0029] A user may cause the management controller to download a pending system image 292 over the network (not shown) and store the image in the staging non-volatile memory. At some point later, the user may decide to apply the pending system image 292. The management controller may unencrypt the pending system image, and store the image in the storage non-volatile memory 280 as the default system image 282. At this point, the keys used to authenticate the UEFI components may be contained within the default system image.

[0030] As explained previously, during initialization, the secure NVRAM may have been write locked by the locking device 230, using a password generated when the system was booted. The processor 210 may enter system management mode, for example upon receipt of a system management mode interrupt. Installing a new default system image on the system 200 may cause such an interrupt. The processor may then enter system management mode, which allows access to the SMM memory. The processor may then retrieve the lock password and provide the lock password to the locking device.

[0031] The locking device may receive the lock password, and if it matches the stored lock password 232, may unlock the secure NVRAM. The
processor may then extract the keys from the default system image 282 and store the keys as secure keys 252. The processor may then re-lock the secure NVRAM using the Socking device. What should be clear is that the secure keys were written to the secure NVRAM while the processor was running in system management mode. Because only trusted code is allowed to run in system management mode, it can be ensured that the keys that are stored in the secure NVRAM are the actual keys associated with the default image 282.

[0032] When the processor attempts to load and run a UEFI component, the digital signature of the component is verified using the secure keys that were previously stored in the secure NVRAM. If the signature is verified, the component is authentic and can be safely loaded and executed. If verification is not successful, the component may have been tampered with, and as such may not be loaded and executed.

[0033] In addition to verifying the authenticity of a UEFI component, the secure NVRAM may be used to store UEFI variables in a secure manner. For example, some UEFI components may have data that is to be stored securely. The techniques described herein also provide for secure storage of UEFI variables in the secure NVRAM. A UEFI component wishing to store a secure variable may first prove its authenticity through the same key verification method described above. If the UEFI component is authentic, it can be trusted to write data to the secure NVRAM. As such, the processor may enter system management mode to retrieve the lock password and unlock the secure NVRAM, as has been described above. The process of the image writing to the secure NVRAM is described in further detail below.

[0034] FIG. 3 is an example of a high level flow diagram for writing to a secure non-volatile random access memory according to techniques described herein. The techniques described in FIG. 3 may be implemented by a processor, such as processor 110, by executing instructions, such as instruction stored on a non-transitory processor readable medium. The instructions may cause the processor to implement the techniques described. In block 310, a request to write to a secure non-volatile random access memory (NVRAM) may be received. For example, the request may come from application code, such as a
UEFI firmware driver. However, it should be understood that the secure NVRAM techniques described herein are not dependent on any particular application.

[0035] In block 320, system management mode may be entered by the processor. As explained above, system management mode is a mode that may be entered by the processor when certain conditions are met. For example, the processor may enter system management mode upon receipt of a system management mode interrupt. Entering system management mode may allow the processor to access certain resources, such as system management mode memory, that are not accessible outside of system management mode. Because the system management mode memory is accessible only when the processor is in system management mode, it can be ensured that the contents of system management mode memory are secure.

[0036] In block 330, a lock password may be retrieved from system management mode memory. Storing the lock password will be described below, but for purpose of the description of FIG. 3, it may be assumed that the lock password previously existed in system management mode memory. In block 340, the lock password may be written to a locking device, wherein the locking device unlocks the secure NVRAM when the lock password matches a password stored in the locking device at the time the processor was booted.

[0037] In other words, when the processor is booted, the secure NVRAM is write locked by the locking device using a password. The password may then be stored in system management mode memory. If the password is provided to the locking device, such as by writing the password to the locking device, and the password matches that which was used to lock the secure NVRAM at the time the processor was booted, the locking device may then unlock the secure NVRAM for writing.

[0038] In block 350, the secure NVRAM may be written to. As explained above in block 340, once the password has been verified, the locking device may then unlock the secure NVRAM, and the secure NVRAM may be written to. In other words, once the proper password is provided, the secure NVRAM is no longer write locked. In block 350, the secure NVRAM may be re-locked. For example, once the desired data has been written to the secure NVRAM, the
locking device may once again lock the secure NVRAM, meaning that the secure NVRAM can no longer be written to. As such, the data that is contained within the secure NVRAM once again becomes read only data, because the secure NVRAM can no longer be written.

[0039] FIG. 4 is another example of a high level flow diagram for locking and writing to a secure non-volatile random access memory according to techniques described herein. The techniques describe in FIG. 4 may be implemented by a processor, such as processor 110, by executing instructions, such as instruction stored on a non-transitory processor readable medium. The instructions may cause the processor to implement the techniques described. Many of the blocks in FIG. 4 are similar to those described with respect to FIG. 3. For purposes of efficiency, the description of blocks that are substantially similar to those described above is omitted.

[0040] In block 405, a random password may be generated upon every processor reboot. The particular mechanism to generate the random password is relatively unimportant. In addition, it is not required that process used to generate the password be random in the absolute statistical sense. For purposes of the techniques described herein, the password generation may be random in the sense that it is practically not feasible to know in advance the password that is generated.

[0041] In block 410, the generated password may be stored in the locking device, wherein the locking device locks the secure NVRAM. The particular technique to lock the secure NVRAM is relatively unimportant to the techniques described herein. What should be understood is that the locking device locks the secure NVRAM for purposes of writing to the secure NVRAM. In order to unlock the secure NVRAM, the locking device may be provided with a password. If the provided password and the previously stored randomly generated password match, the locking device may unlock the secure NVRAM.

[0042] In block 415, the generated password may be stored in system management mode memory as the lock password. As described above, system management mode memory may only be accessible when the processor has entered system management mode. As such, the lock password is inaccessible
to the processor, and hence any applications running on the processor, except when the processor is running in system management mode. Because only secure code is run when the processor is in system management mode, it can be ensured that malicious code that may attempt to steal the lock password, cannot be executed. Thus, the lock password may remain secure, and is only accessible while running in system management mode.

In block 420, just as in block 310, a request to write to secure NVRAM may be received. In block 425, just as in block 320, the processor may enter system management mode. In block 430, just as in block 330, the lock password may be retrieved from system management mode memory. In block 435, just as in block 340, the lock password may be written to the locking device, wherein the locking device unlocks the secure NVRAM. In block 440, just as in block 350, the secure NVRAM may be written to. In block 445, just as in block 360, the secure NVRAM may be re-locked.

FIG. 5 is an example of a high level flow diagram for using the secure non-volatile memory techniques described herein to implement a secure boot system. The techniques described in FIG. 5 may be implemented by a processor, such as processor 110, by executing instructions, such as instruction stored on a non-transitory processor readable medium. The instructions may cause the processor to implement the techniques described. In block 505, the system may boot. As will be described in further detail below, other steps may occur after the system boots.

In block 515 it may be determined if a request to reset secure NVRAM has been received. In other words, it may be determined if the user of the system, or a program running on the system, is attempting to reset the values that are store in the secure NVRAM back to their default values. In block 520, it may be determined if secure boot is enabled. The determination of whether or not secure boot is enabled may be used to determine which operations are allowed. For example, some operations may not be allowed if secure boot is enabled.

If it is determined that secure boot is not enabled, the process moves to block 525. In block 525, default values may be retrieved from a
storage NVRAM. The default values may be written to the secure NVRAM. As mentioned before, the storage NVRAM may store system images that have been verified by the management controller through public/private key encryption of the system images. As such, the system images stored in the storage NVRAM are ensured to be unaltered system images. Furthermore, because secure boot is not yet enabled, there is no need to ensure that the contents of the secure NVRAM are not overwritten.

[0047] If it is determined in block 520 that secure boot is enabled, the process moves to block 530. In block 530, the request to reset the secure NVRAM may be ignored. The request may be ignored because secure boot has been enabled, and as such, the contents of the secure NVRAM may contain data that should not be erased or overwritten. As such, any request to reset the secure NVRAM to the default values once secure boot has been enabled may be ignored. It should be noted that in some cases, only the request to reset the defaults stored in secure NVRAM is ignored. There may be additional data stored in other locations, such as a non-secure NVRAM. A reset request may reset this non-secure data, regardless of if secure boot is enabled or not. Thus, even in the case where secure boot is not enabled, block 525 still moves to block 530, wherein other data that is not stored in secure NVRAM may be reset to the default values.

[0048] FIG. 6 is another example of a high level flow diagram for using the secure non-volatile memory techniques described herein to implement a secure boot system. The techniques described in FIG. 6 may be implemented by a processor, such as processor 110, by executing instructions, such as instruction stored on a non-transitory processor readable medium. The instructions may cause the processor to implement the techniques described. Many of the blocks in FIG. 6 are similar to those described with respect to FIG. 5. For purposes of efficiency, the description of blocks that are substantially similar to those described above is omitted.

[0049] In block 650, just as above in block 505, the system may boot. In block 610, the secure NVRAM may be initialized. The process of initializing the secure NVRAM may be substantially similar to portions of the process
described in FIG. 4. In particular, a random password may be generated, the password stored in system management mode memory, and the password used by a locking device to lock the secure NVRAM. In block 615, just as above in block 515, it may be determined if a request to reset the secure NVRAM has been received.

[0050] If so, the process may move to block 620, where just as above in block 520, it may be determined if secure boot is not enabled. If secure boot is not enabled, the process may move to block 625, where just as above in block 525, the default values may be retrieved from a storage NVRAM and written to the secure NVRAM. The process of writing the data into the secure NVRAM may substantially follow that process that was described with respect to FIG. 4. In particular, the lock password may be retrieved from system management mode memory and provided to the locking device, which may then unlock the secure NVRAM. If secure boot is determined to be enabled in block 620, the process may move to block 630. In block 630, just as in block 530, the request to write to the secure NVRAM may be ignored. However, just as above, other portions of memory, such as those portions that are not within the secure NVRAM may still be overwritten with default values.

[0051] If it is determined in block 615 that a request to reset the secure NVRAM was not received, the process moves to block 635. In block 635, it may be determined if a request to switch to legacy boot mode has been received. As mentioned above, secure boot mode is an option, and as such may be turned on and off. However, if secure boot is enabled requests to switch to legacy boot mode may be an attempt to circumvent the security provided by secure boot. If it is determined that a request to switch to legacy boot mode has been received, the process may move to block 640.

[0052] In block 640, it may be determined if secure boot is enabled. If secure boot is not enabled, then the boot mode may be changed to legacy boot mode, thus bypassing secure boot. The process may then return to block 605, and the system is booted. If it is determined in block 640 that secure boot is enabled, then a request to switch to legacy boot mode is an attempted security violation. The process may then move to block 655, in which a security violation
is logged. The security violation may record information regarding the entity that was attempting to switch to legacy boot mode. Offline processes may then be used to determine why the switch to legacy boot mode was being attempted. In any case, the switch to legacy boot mode is not performed, and the process moves to block 650, to continue the boot process.

[0053] In block 650, a unified extensible firmware (UEFI) image may be retrieved. For example, the image may be retrieved from the storage NVM described above with respect to Fig.1. In block 660, it may be determined if secure boot is enabled. If secure boot is enabled, the process may move to block 665. In block 665, a signature of the UEFI image may be verified using a key stored in the secure NVRAM to determine that the image is authentic. In some implementations, the image may be digitally signed and as long as the image is unaltered, the signature may be verified using the key stored in the secure NVRAM. However, any alteration to the image will result in a change to the hash of the image. As such, the verification will fail. In block 675, it is determined that the image is not authentic (e.g. the verification has failed), the process moves to block 680, in which a security violation may be logged.

[0054] If it is determined that the image is authentic in block 675, the process moves to block 670, where the image is loaded. In block 670, the secure boot is either enabled and the image has proved itself to be authentic, or secure boot is not enabled, in which case authenticity of the image is unimportant. In either case the image is loaded and executed.

[0055] As mentioned above, in some cases, the UEFI image may desire to write to the secure NVRAM. For example, the UEFI image may need to keep track of a variable or some other piece of data in a secure manner. Regardless of the reason, the UEFI image may need to write to secure NVRAM. In block 685, a request from the image to write to secure NVRAM may be received. In block 690, it may be determined if the image is determined to be authentic, and if so, the secure NVRAM may be written to. The process of determining if the UEFI image is authentic may be substantially similar to that described above with respect to block 660, 665, and 675. The process of writing to the secure
NVRAM may be substantially similar to the process that was described with respect to FIG. 4.
We Claim:

1. A system comprising:
   a processor;
   a system management mode memory coupled to the processor, the system management mode memory accessible to the processor only when the processor is running in system management mode, the system management mode memory storing a lock password;
   a secure non-volatile memory accessible to the processor, at least a portion of the secure non-volatile memory being write lockable; and
   a locking device coupled to the processor and the secure non-volatile memory to lock and unlock the at least a portion of the secure non-volatile memory based on the lock password.

2. The system of claim 1 further comprising:
   a management controller coupled to the processor and the secure non-volatile memory, wherein processor access to the secure non-volatile memory is through the management controller; and
   a storage non-volatile memory coupled to the management controller, the storage non-volatile memory storing a default version of a system image, the default image having been digitally signed and verified by the management controller.

3. The system of claim 1 further comprising:
   a staging non-volatile memory coupled to the management controller, the staging non-volatile memory to store a pending system image uploaded by an authorized user, the pending system image to be transferred to the storage non-volatile memory upon occurrence of a determined event.

4. The system of claim 1 wherein the lock password is randomly generated upon each reboot of the system.
5. The system of claim 1 wherein the lock password is stored in a write once register of the locking device.

6. A non-transitory processor readable medium containing thereon a set of instructions, which when executed by the processor cause the processor to:
   receive a request to write to a secure non-volatile random access memory (NVRAM);
   enter system management mode with the processor;
   retrieve a lock password from system management mode memory;
   write the lock password to a locking device, wherein the locking device unlocks the secure NVRAM when the lock password matches a password stored in the locking device at the time the processor was booted;
   write to the secure NVRAM; and
   re-lock the secure NVRAM.

7. The medium of claim 6 further comprising instructions to:
   initialize the secure NVRAM, wherein initializing the secure NVRAM comprises instructions to:
   generate a random password upon every processor reboot;
   store the generated password in the locking device, wherein the locking device locks the secure NVRAM; and
   store the generated password in system management mode memory as the lock password.

8. The medium of claim 7 wherein the locking device is a complex programmable logic device (CPLD) and the generated password is stored in a write once register of the CPLD.

9. A method comprising:
   determining, at a processor, receipt of a request to reset secure non-volatile random access memory (NVRAM) to default values;
   determining that secure boot is enabled;
retrieving default values from a storage NVRAM and writing the default values to the secure NVRAM when it is determined that the reset request was received and secure boot is disabled; and

ignoring the request to reset the secure NVRAM when it is determined that the reset request was receive and secure boot is enabled.

10. The method of claim 9 further comprising:
   determining receipt of a request to switch to a legacy boot mode;
   determining that secure boot is enabled;
   changing the boot mode to legacy when it is determined that the request to switch to legacy boot mode was received and that secure boot is not enabled, switch the boot mode to legacy further comprising rebooting the processor; and
   ignoring the request to switch to legacy boot mode when it is determined that the request to switch to legacy boot mode was received and that secure boot is enabled, wherein ignoring the request to switch to legacy boot mode further comprises logging a security violation.

11. The method of claim 10 further comprising:
   determining that secure boot is enabled;
   retrieving a unified extensible firmware interface (UEFI) image;
   verifying a signature of the UEFI image using a key stored in the secure NVRAM to determine that the image is authentic;
   loading the image when the image is determined to be authentic; and
   logging a security violation when the image is determined to not be authentic.

12. The method of claim 11 further comprising:
   receiving a request from the image to write to the secure NVRAM; and
   writing to the secure NVRAM when the image is determined to be authentic.

13. The method of claim 9 further comprising:
initializing the secure NVRAM, wherein initializing the secure NVRAM further comprises:
   generating a random password upon every processor reboot;
   storing the generated password in a locking device, wherein the locking device locks the secure NVRAM; and
   storing the generated password in system management mode memory as a lock password.

14. The method of claim 13 wherein writing to the secure NVRAM further comprises:
   entering system management mode with the processor;
   retrieving the lock password from system management mode memory;
   writing the lock password to the locking device, wherein the locking device unlocks the secure NVRAM when the lock password matches the generated password stored in the locking device at the time the processor was booted;
   writing to the secure NVRAM; and
   re-locking the secure NVRAM.

15. The method of claim 14 wherein the locking device stores the generated password in a write once register.
310. Receive a request to write to a secure non-volatile random access memory (NVRAM).

320. Enter system management mode with the processor.

330. Retrieve a lock password from system management mode memory.

340. Write the lock password to a locking device, wherein the locking device unlocks the secure NVRAM when the lock password matches a password stored in the locking device at the time the processor was booted.

350. Write to the secure NVRAM.

360. Re-lock the secure NVRAM.

FIG. 3
405. Generate a random password upon every processor reboot.

410. Store the generated password in the locking device, wherein the locking device locks the secure NVRAM.

415. Store the generated password in system management mode memory as the lock password.

420. Receive a request to write to a secure non-volatile random access memory (NVRAM).

425. Enter system management mode with the processor.

430. Retrieve a lock password from system management mode memory.

435. Write the lock password to a locking device, wherein the locking device unlocks the secure NVRAM when the lock password matches a password stored in the locking device at the time the processor was booted.

440. Write to the secure NVRAM.

445. Re-lock the secure NVRAM.

FIG. 4
FIG. 5

SYSTEM BOOT

REQUEST TO RESET SECURE NVRAM RECEIVED?

SECURE BOOT ENABLED?

RETRIEVE DEFAULT VALUES FROM A STORAGE NVRAM AND WRITE THE DEFAULT VALUES TO THE SECURE NVRAM

IGNORE THE REQUEST TO RESET THE SECURE NVRAM

505

515

520

525

530
INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2014/050921

A. CLASSIFICATION OF SUBJECT MATTER

G06F 21/79(2013.01); G06F 12/14(2006.01)

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F 21/79; G06F 9/24; G06F 21/71; G06F 1/26; G06F 7/04; G06F 21/00; G06F 12/14

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Korean utility models and applications for utility models

Japanese utility models and applications for utility models

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

eKOMPASS(KIPO internal) & Keywords: UEFI system, image, CPLD, key, non-volatile, memory, password, register boot, unlock, processor

C. DOCUMENTS CONSIDERED TO BE RELEVANT

<table>
<thead>
<tr>
<th>Category</th>
<th>Citation of document, with indication, where appropriate, of the relevant passages</th>
<th>Relevant to claim No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>US 2008-0244252 AI (MICHAEL A. ROTHMAN et al.) 02 October 2008</td>
<td>1-15</td>
</tr>
<tr>
<td></td>
<td>See paragraphs [0022], [0028], [0036], [0054], [0057], [0071] ; and figure 3.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See paragraphs [0006], [0014], [0022] ; and figure 4.</td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td>WO 2014-063330 AI (INTEL CORPORATION) 01 May 2014</td>
<td>2, 11-12</td>
</tr>
<tr>
<td></td>
<td>See page 5, lines 19-24, and figures 1, 8.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See page 8, lines 1-5; page 134, lines 1-10; tables 1-3.</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>US 2013-0159729 AI (STEFAN THOM et al.) 20 June 2013</td>
<td>1-15</td>
</tr>
<tr>
<td></td>
<td>See paragraphs [0025], [0033] ; and figure 1.</td>
<td></td>
</tr>
</tbody>
</table>

Further documents are listed in the continuation of Box C. See patent family annex.

* Special categories of cited documents:
  "A" document defining the general state of the art which is not considered to be of particular relevance
  "E" earlier application or patent but published on or after the international filing date
  "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
  "O" document referring to an oral disclosure, use, exhibition or other means
  "P" document published prior to the international filing date but later than the priority date claimed
  "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
  "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
  "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
  "&" document member of the same patent family

Date of the actual completion of the international search: 31 March 2015 (3.01.2015)

Date of mailing of the international search report: 01 April 2015 (01.04.2015)

Name and mailing address of the ISA/KR
International Application Division
Korean Intellectual Property Office
189 Cheongsa-ro, Seo-gu, Daejeon Metropolitan City, 302-701, Republic of Korea
Facsimile No. +82 42 472 7140

Authorized officer
AHN, Jeong Hwan
Telephone No. +82-42-481-8440

Form PCT/ISA/210 (second sheet) (January 2015)
# INTERNATIONAL SEARCH REPORT

Information on patent family members

<table>
<thead>
<tr>
<th>Patent document cited in search report</th>
<th>Publication date</th>
<th>Patent family member(s)</th>
<th>Publication date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>US 2010-0332813 Al</td>
<td>30/12/2010</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 7395420 B2</td>
<td>01/07/2008</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 8281116 B2</td>
<td>02/10/2012</td>
</tr>
<tr>
<td>WO 2014-063330 Al</td>
<td>01/05/2014</td>
<td>None</td>
<td></td>
</tr>
<tr>
<td>US 2013-0159729 Al</td>
<td>20/06/2013</td>
<td>CN 103748594 A</td>
<td>23/04/2014</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EP 2737429 A</td>
<td>04/06/2014</td>
</tr>
<tr>
<td></td>
<td></td>
<td>EP 2737429 A4</td>
<td>05/11/2014</td>
</tr>
<tr>
<td></td>
<td></td>
<td>KR 10-2014-0054003 A</td>
<td>08/05/2014</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 2013-031374 Al</td>
<td>31/01/2013</td>
</tr>
<tr>
<td></td>
<td></td>
<td>US 8375221 BI</td>
<td>12/02/2013</td>
</tr>
<tr>
<td></td>
<td></td>
<td>WO 2013-019369 Al</td>
<td>07/02/2013</td>
</tr>
</tbody>
</table>