Title: ACCESS TO MEMORY REGION INCLUDING CONFIDENTIAL INFORMATION

Abstract: Embodiments herein relate to accessing a memory region including confidential information. A memory request from a process may be received. The memory request may include a process ID (PID) of the process, a requested memory address, and a requested access type. The memory request may be compared to a permission set associated with a memory region including the confidential information. Access to the memory region by the process may be controlled based on the comparison.
Declarations under Rule 4.17:

— as to the identity of the inventor (Rule 4.17(i))
— as to applicant’s entitlement to apply for and be granted a patent (Rule 4.17(H))

— as to the applicant’s entitlement to claim the priority of the earlier application (Rule 4.17(in))

Published:

— with international search report (Art. 21(3))
ACCESS TO MEMORY REGION INCLUDING CONFIDENTIAL INFORMATION

PRIORITY INFORMATION

[0001] This application claims the benefit of priority on US Provisional Application No 61/509,078, filed July 18, 2011, the entire contents of which are incorporated herein in their entirety by reference.

BACKGROUND

[0002] A device may store confidential information, such as a cryptographic security parameter (CSP). However, an integrity or security of the confidential information may be compromised if accessed by an unauthorized party. Programming a process of the device to only access confidential information for which the process has authority to access, may be unreliable.

[0003] For example, the process may malfunction and/or be manipulated by an unauthorized party, such as a hacker, and access confidential information for which the process does not have authorization. Manufacturers, vendors, and/or users are challenged to provide more secure methods for protecting confidential information.

BRIEF DESCRIPTION OF THE DRAWINGS

[0004] The following detailed description references the drawings, wherein:

[0005] FIG. 1 is an example block diagram of a device for controlling access to a memory region;
FIG. 2 is an example block diagram of a permission module of the device of FIG. 1;

FIG. 3 is an example block diagram of modules included in the permission module of the device of FIG. 2;

FIG. 4 is another example block diagram of the permission module of the device of FIG. 1;

FIG. 5 is an example block diagram of a permission set of the permission module of FIG. 4;

FIG. 6 is another example block diagram of a device for controlling access to a memory region;

FIG. 7 is an example block diagram of a system including the device of FIG. 6;

FIG. 8 is an example block diagram of a computing device including instructions for controlling access to a memory region; and

FIG. 9 is an example flowchart of a method for controlling access to a memory region.

DETAILED DESCRIPTION

Specific details are given in the following description to provide a thorough understanding of embodiments. However, it will be understood by one of ordinary skill in the art that embodiments may be practiced without these specific details. For example, systems may be shown in block diagrams in order not to
obscure embodiments in unnecessary detail. In other instances, well-known processes, structures and techniques may be shown without unnecessary detail in order to avoid obscuring embodiments.

[0015] A device may store confidential information, such as a cryptographic security parameter (CSP). In order to maintain an integrity or security of this confidential information, the device may use a secondary processor to control access to the confidential information. However, this may increase a cost and complexity of the device and decrease performance. For example, a main processor of the device may have to communicate with the secondary processor before receiving access to the confidential information.

[0016] Alternatively, the confidential information may be separated and stored with the processes that may use the confidential information. However, in this case, the confidential information may not be accessible unless the corresponding process is loaded and running. Further, if the process is running, a memory management unit (MMU) may limit access to the process's memory. In addition, separating the confidential information among multiples locations may compromise the integrity and/or security of the confidential information. Moreover, the process may malfunction and/or be manipulated by an unauthorized party, such as a hacker, thus reducing an integrity or security of the confidential information.

[0017] Embodiments may increase a security and/or integrity of the confidential information without greatly increasing a complexity of the device. For example, in embodiments, a memory request from a process may be received. The memory
request may include a process ID (PID) of the process, a requested memory
address, and a requested access type. The memory request may be compared to
a permission set associated with a memory region including the confidential
information. Access to the memory region by the process may be controlled based
on the comparison. Thus, access permissions to each part of the confidential
information may be separately and selectively based on a variety of factors, such
as the PID, requested address, the requested access type, a purpose and a user
ID (UID).

[0018] Further, embodiments may also carry out additional operations if access
to the confidential information by the process is denied. For example, an audit log
may be generated when the process requests an improper type of access, in order
to maintain a record of failed attempts to access the confidential information.
Further, embodiments may carry out an action, such as clearing the confidential
information or generating an interrupt, if the PID of the process or the requested
address is incompatible.

[0019] Referring now to the drawings, FIG. 1 is an example block diagram of a
device 100 for controlling access to a memory region 190. The device 100 may be
included in any type of device that accesses a memory, such as a secure
microprocessor, a notebook computer, a desktop computer, an all-in-one system, a
slate computing device, a portable reading device, a wireless email device, a
mobile phone, and the like. In the embodiment of FIG. 1, the device 100 includes a
permission module 110. The permission module 110 further includes a permission set 120.

[0020] The permission module 110 may include, for example, a hardware device including electronic circuitry for implementing the functionality described below, such as a register or Boolean logic. In addition or as an alternative, the permission module 110 may be implemented as a series of instructions encoded on a machine-readable storage medium and executable by a processor.

[0021] The memory region 190 may be part of a machine-readable storage medium, such as any type of electronic, magnetic, optical, or other physical storage device capable of storing information, like data or instructions. Example of the machine-readable storage medium include Random Access Memory (RAM), an Electrically Erasable Programmable Read-Only Memory (EEPROM), a storage drive, a Compact Disc Read Only Memory (CD-ROM), and the like.

[0022] The permission module 110 is to receive a memory request from a process 180. The term process may refer to any part of a computer program or instance thereof. The memory request may include a process ID (PID) of the process 160, a requested memory address, and a requested access type. The permission module 110 may compare the memory request to the permission set 120, which is associated with the memory region 190 storing confidential information 192. The permission module 110 may then control access to the memory region 190 by the process 180 based on the comparison. For example, the permission module 110 may output an access control signal to the memory
region 190 that enables and/or disables an ability of the process 180 to access the memory region 190.

[0023] The confidential information 192 may any type of information to which access is restricted to authorized parties. Examples of the confidential information 192 may include a key, a certificate, information associated with platform security of the device 100, cryptographic information, private user data, and the like. Example operations of the permission module 110 will be explained in greater detail below with respect to FIGs. 2 to 4.

[0024] FIG. 2 is an example block diagram of a permission module 210 of the device 100 of FIG. 1. In the embodiment of FIG. 2, the permission module 210 includes a permission set 220, a PID module 230, an address module 240, an access type module 250, a first NAND gate 260, a first AND gate 270 and a NOT gate 280.

[0025] The permission set 220 includes one or more allowed PIDs 222, a memory address range 224, and one or more allowed access types 226 associated with the memory region 190. The one or more allowed PIDs 222 may indicate one or more processes that are allowed to access the memory region 190. The memory address range 224 may indicate the addresses that may be accessed when requesting access to memory region 190. The access types 226 include at least one of read, write and reference types. The read type indicates whether the process 180 has permission to read a value at the requested memory address. The write type indicates whether the process 180 has permission to modify, clear, write, and/or overwrite a value at the requested memory address. The reference type
indicates whether the process 180 has permission to provide a pointer to the requested memory address, instead of the value at the requested memory address. The permission set 220 will be explained in greater detail with respect to FIG. 5.

[0026] The PID module 230, address module 240 and access type module 250 may include, for example, a hardware device including electronic circuitry for implementing the functionality described below, such as a register or Boolean logic. In addition or as an alternative, the modules 230, 240 and 250 may be implemented as a series of instructions encoded on a machine-readable storage medium and executable by a processor.

[0027] The PID module 230 is to compare the PID of the process 180 to the one or more allowed PIDs 222 of the permission set 220. For example, if the PID of the process 180 is included in the one or more allowed PIDs 222, the PID module 230 may output a logic one or high. Otherwise, the PID module 230 may output a logic zero or low.

[0028] The address module 240 is to compare the requested memory address to the memory address range 224 of the permission set 220. For example, if the requested memory address is included in the memory address range 224, the address module 240 may output a logic one or high. Otherwise, the address module 240 may output a logic zero or low.

[0029] The access type module 250 is to compare the requested access type to the one or more allowed access types 226 of the permission set 220. For example,
if the requested access type is included in the one or more allowed access types 226, the access type module 250 may output a logic one or high. Otherwise, the access type module 250 may output a logic zero or low. The output of the access type module 250 is inverted by the NOT gate 280 and output from the permission module 210 as a fault signal.

[0030] The first NAND gate 260 logically NANDs the output of the PID module 230 and the address module 240 and outputs a miss signal out of the permission module 210. The first AND gate 270 logically ANDs the output of the PID module 230, the address module 240 and the access type module 250 and outputs an access control signal out of the permission module 210.

[0031] Thus, the permission module 210 allows access to the memory region 190 by the process 180 based on the comparisons of the PID, address, and access type modules 230, 240 and 250. Further, the permission module 210 asserts the miss signal if the PID module 230 indicates that the one or more allowed PIDs 222 does not include the PID of the process 180 and/or the address module 240 indicates that the memory address range 224 does not include the requested memory address. A more detailed example of the modules 230, 240 and 250 is provided below with respect to FIG. 3.

[0032] FIG. 3 is an example block diagram of the modules 230, 240 and 250 included in the permission module 210 of the device of FIG. 2. In the embodiment of FIG. 3, the PID module 230 may include, for example, a bitwise XOR gate 232 and a first NOR gate 234. The PID of the process signal output from the process 180 and
the allowed PIDs signal output from the permission set 220 received by the bitwise XOR gate 232 may be multi-bit fields. For example, each of the bits of the PID of the process signal and the allowed PIDs signal may correspond to one of a plurality of different types of PIDs, where each of the PIDs corresponds to a different process or type of process.

[0033] Depending on a number of PIDs associated with the memory region 190, more than one bit may be set in the allowed PIDs signal. However, only one of the bits of the PID of the process signal may be set to indicate the identity of the process 180. Thus, the bitwise XOR gate 232 bitwise logically XORs the corresponding bits of the PID of the process signal and the allowed PIDs signal. The first NOR gate 234 logically NORs all the bits of the output of XOR gate 232, thus also combining all of the bits output from the bitwise XOR gate 232. Hence, the output of the first NOR gate 234, as well as the output of the PID module 230, is a logic one or high if the PID of the process 180 is included in the one or more allowed PIDs 222 of the permission set 220.

[0034] The address module 240 may include, for example, an adder 242, a first comparator 244, a second comparator 246 and a second NOR gate 248. The first adder 242 receives an initial address and a length from the memory address range 224. The initial address represents a first address of the memory region 190 and a length represents a number of addresses included in the memory region 190. The adder 242 adds the initial address and the length to output a last memory address of the memory region 190 to the second comparator 246. The first comparator 244
bitwise magnitude compares the requested address of the process 180 to the initial address of the memory address range 224. The first comparator 244 outputs logic one or high signal if the requested address is less than the initial address. Otherwise, the first comparator 244 outputs a logic low or zero signal.

[0035] The second comparator 246 bitwise magnitude compares the last memory address to the requested address of the process 180. The second comparator 246 outputs logic one or high signal if the last memory address is less than the requested address. Otherwise, the second comparator 246 outputs a logic low or zero signal. The second NOR gate 248 logically NORs the output of the first and second comparators 244 and 246. The output of the second NOR gate 248, as well as the output of the address module 240, is a logic one or high if the requested address of the process 180 is included in the memory address range 224 of the permission set 220.

[0036] The access type module 250 may include, for example, a bitwise AND gate 252 and a first OR gate 254. The allowed access types signal of the permission set 220 and the requested access type signal of the process 180 input to the bitwise AND gate 252 may be multi-bit fields. For example, each of the bits of the allowed access types signal and the requested access type signal may correspond to one of a plurality of different access types, such as the read, write, and reference types, as explained above.

[0037] Depending on a number of access types associated with the memory region 190, more than one bit may be set in the allowed access type signal.
However, only one of the bits of the requested access type may be set as the process 180 may only carry out one of the access types per access to the memory region 190. Thus, the bitwise AND gate 252 bitwise logically ANDs the corresponding bits of the allowed access types signal and the requested access type signal. The first OR gate 254 logically ORs all the bits of the output of the bitwise AND gate 252, thus also combining all of the bits output from the bitwise AND gate 252. Hence, the output of the first OR gate 254, as well as the output of the access type module 250, is a logic one or high if the requested access type is included in the one or more allowed access types 222 of the permission set 220. The first NAND gate 260, the first AND gate 270 and the NOT gate 280, may operate as explained above with respect to FIG. 2.

[0038] FIG. 4 is another example block diagram of the permission module 410 of the device 100 of FIG. 1. In the embodiment of FIG. 4, the permission module 410 includes a permission set 420, a PID module 430, an address module 440, an access type module 450, a first NAND gate 460, a first AND gate 470, a NOT gate 480 and a purpose module 490. The PID module 430, address module 440, access type module 450, first NAND gate 460, first AND gate 470 and NOT gate 480 of FIG. 4 may be respectively similar to the PID module 230, address module 240, access type module 250, first NAND gate 260, first AND gate 270 and NOT gate 280 of FIG. 2. However, the first AND gate 470 may also receive an output of the purpose module 490. In addition, the request of the process 180 may further include a user ID (UID) field and a requested purpose.
The permission set 420 of the FIG. 4 may be similar to the permission set 220 of FIG. 2. For example, permission set 420 of FIG. 4 includes one or more allowed PIDs 422, a memory address range 424 and one or more allowed access types 426, which is similar to the one or more allowed PIDs 222, the memory address range 224 and the one or more allowed access types 226 included in the permission set 220 of FIG. 2. However, the permission set 420 of FIG. 4 also receives the UID and includes one or more allowed purposes 428, as explained below.

The UID field may identify a type of user and/or an identity of the user. The type of user may relate to user groups, such as staff or administrator. The identity of the user may relate to identifying a specific user. The permission set 420 may include a plurality of instances of at least part of the permission set 420. Each of the instances may be associated with one of a plurality of UIDs and include separate permissions for the memory region 190, as explained in greater detail below with respect to FIG. 5.

The requested purpose indicates how the process 180 seeks to use the confidential information 192 at the requested memory address. Examples of the requested purpose may include a storage destination of the confidential information 192, such as an external memory, a type of operation to be performed on the confidential information 192, such as a type of cryptographic operation, and the like.
The purpose module 490 may include, for example, a hardware device including electronic circuitry for implementing the functionality described below, such as a register or Boolean logic. In addition or as an alternative, the permission module 410 may be implemented as a series of instructions encoded on a machine-readable storage medium and executable by a processor.

The purpose module 490 is to compare the requested purpose to the one or more allowed purposes of the permission set 420. For example, the permission set 420 may send an allowed purpose types signal to the purpose module 480, where the allowed purpose types signal and the requested purpose signal are multi-bit signals. The purpose module 490 may carry out the comparison of the two multi-bit signals in a manner similar to that of the access type module 450. If the requested purpose is included in the one or more allowed purpose types 428, the purpose module 480 may output a logic a logic one or high. Otherwise, the purpose module 480 may output a logic a logic zero or low.

The first AND gate 470 logically ANDs the outputs of the purpose module 480, the PID module 430, the address module 440 and the access type module 450 and outputs an access control signal out of the permission module 410. Thus, the permission module 410 allows access to the memory region 190 by the process 180 based on the comparisons of the purpose, PID, address, and access type modules 490, 430, 440 and 450.

Similar to FIG. 2, the first NAND gate 460 logically NANDs the output of the PID module 430 and the address module 440 and outputs the miss signal out of
the permission module 410. Also, the NOT gate 480 inverts the output of the access type module 450 to output a fault signal. Thus, the permission module 410 asserts the fault signal if the one or more allowed access types 426 does not include the requested access type. However, embodiments may also, for example, assert the fault signal if the one or more purpose types 428 does not include the requested purpose.

[0046] FIG. 5 is an example block diagram of the permission set 520 of the permission module 410 of FIG. 4. In this embodiment, the permission set 520 includes seven fields 521 to 527, where at least some of the fields 525 to 527 include multiple instances. The first field 521 may store a valid bit to indicate whether the permission set 520 is active and assigned to the memory region 190. If the valid bit of the first field 521 is not set, the device 100 may ignore the permission set 520. The second field 522 may store the real address, such as the requested address of the process 180. The third field 523 may store the virtual address that corresponds to a location in the memory region 190 mapped to the real address. The fourth field 524 may store the memory address range of the memory region 190, such as the initial address and the length of the memory region 190. The fifth through seventh fields 525 to 527 may have n instances, where n is a natural number. Each of the instances of the fifth field 525-1 to 525-n may store the allowed access types for a corresponding one of the UIDs. Each of the instances of the sixth field 526-1 to 526-n may store the allowed PIDs for the corresponding one of the UIDs. Each of the
instances of the seventh field 527-1 to 527-n may store the allowed purpose types for
the corresponding one of the UIDs.

[0047] Each of the groups of instances for each UID may have different permissions. For example, the first instances 525-1 to 527-1 of the first UID may relate to staff users while the nth instances 525-n to 527-n of the nth UID may relate to supervisor users. In this example, the allowed access types of the first UID 525-1 may only enable the read access type while the allowed access type of the nth UID 525-n may enable all of the read, write and reference access types. The permission set 520 may choose one of the groups of instances to forward to the modules 480, 430 and 450, based on the user identified by the UID.

[0048] While the permission set 520 is shown to have seven fields 521 to 527 and multiple instances for three of those fields 525 to 527 for different UIDs, embodiments are not limited thereto. For example, embodiments may have more or less than seven fields and/or instances of more or less than three of the fields. Further, the instances may be created and/or grouped according to criteria other than the UID.

[0049] FIG. 6 is another example block diagram of a device 600 for controlling access to a memory region 190. In this embodiment, the device 600 includes a plurality of the permission modules 110-1 to 110-n of FIG. 1, where n is a natural number. Embodiments of the device 600 may also include a plurality of other types of permission modules, such as the permission modules 210 or 410 of FIG. 2 or 4. The device 600 further includes a second AND gate 620, a third AND gate 640, a second OR gate 670, an action module 630 and an audit module 650.
The action and audit modules 630 and 650 may include, for example, a hardware device including electronic circuitry for implementing the functionality described below, such as a register or Boolean logic. In addition or as an alternative, the action and audit modules 630 and 650 may be implemented as a series of instructions encoded on a machine-readable storage medium and executable by a processor.

Each of the permission modules 110-1 to 110-n may include separate permission sets that are associated with different memory regions, as explained in greater detail below with respect to FIG. 7. However, all of the permission modules 110-1 to 110-n may receive the same PID of the process 180, the requested memory address, and the requested access type, from the process 180. The second AND gate 620 logically ANDs the miss signals of all the permission modules 110-1 to 110-n and outputs an action signal to the action module 630. Thus, the action signal is only asserted if all of the permission modules 110-1 to 110-n assert the miss signal.

The action module 630 is to carry out an action if the action signal is asserted, e.g. the PID module 230 and/or the address module 240 of the permission modules 110-1 to 110-n determine that the one or more allowed PIDs 222 do not include the PID of the process 180 and/or the memory address range 224 does not include the requested memory address. The action carried out by the action module 630 may include doing nothing, erasing a value at the requested memory address, erasing all values at the memory region and/or generating an...
interrupt. The interrupt may block access to and/or halt operation of a functionality of the device 600 and/or components related to the device 600, such as a processor or memory. Some of the above actions may improve security by preventing an unauthorized party from making further attempts to access the requested memory address.

[0053] The third AND gate 620 logically ANDs the fault signals of all the permission modules 110-1 to 110-n and outputs an audit signal. Thus, the audit signal is only asserted if all of the permission modules 110-1 to 110-n assert the fault signal. The audit module 650 is to generate an audit log 660 if the audit signal is asserted, e.g. the access type module 250 of the permission modules 110-1 to 110-n determines that the one or more allowed access types 226 does not include the requested access type. The audit log 660 may include the PID of the process 180, the requested address, the requested access type, the UID, a number of failed access attempts, a timestamp of the one or more failed access attempts, and/or the like. The audit log 660 may be configurable. For example, the audit module 650 may be configured to vary the contents and/or length of the audit log 660. Further, a method for saving the audit log 660 may also be configured, such as a flat file method that stops when the audit log 660 is full, a circular file method that overwrites old data with new data when the audit log 660 is full, and/or combination thereof.

[0054] The second NOR gate 620 logically NORs the access control signals of all the permission modules 110-1 to 110-n and outputs an access control signal. Thus,
the device 600 asserts the access control signal to grant the process 180 access to the memory region 180, if at least one of the permission modules 110-1 to 110-n grants access to the memory region 180. While the action and audit modules 630 and 650 are shown to operate in conjunction with a plurality of permission modules 110-1 to 110-n, the action and audit modules 630 and 650 may also be operate in conjunction with a single permission module 110.

[0055] FIG. 7 is an example block diagram of a system 700 including the device 600 of FIG. 6. In the embodiment of FIG. 7, the system 700 includes a device 705, a processor 710, a memory management unit 720 and memory 730. The device 705 of FIG. 7 may be similar to the device 600 of FIG. 6. However, embodiments of the system 700 may also include other similar devices, such as the device 100 of FIG. 1.

The memory 730 includes a plurality of memory regions 190-1 to 190-n (n is a natural number), where each of the memory regions 190-1 to 190-n may be similar to the memory region 190 of FIG. 1. Each of the permission modules 110-1 to 110-n corresponds to one of the memory regions 190-1 to 190-n. For example, each of the permission modules 110-1 to 110-n may include a permission set 120 that controls access to the confidential information 192 of one of the corresponding memory regions 190-1 to 190-n.

[0056] The processor 710 may be, at least one central processing unit (CPU), at least one semiconductor-based microprocessor, at least one graphics processing unit (GPU), other hardware devices suitable for retrieval and execution of instructions of
the process 180. The processor 710 may output the memory request of the process 180 and/or receive the action signal.

[0057] The MMU 720 may share responsibilities with the device 700 for controlling aspects of memory map management. For example, at least one of the device 700 and the MMU 210 may define the address ranges for different parts of the memory 730, such as the plurality of memory regions 190-1 to 190-n, and/or map real (e.g. internal or physical) addresses to virtual (e.g., logical or external) addresses. Further, the MMU 720 may manage access control for regions of the memory 730 that do not include the confidential information 192 while the device 700 may manage access control for regions of the memory 730 that do include the confidential information 192.

[0058] The processor 710 may communicate with the MMU 710 when the processor 710 seeks to access the memory 730, such as after executing an instruction of the process 180. For instance, the processor 710 may transmit one or more requests to the MMU 720. In turn, the MMU 720 may transmit all of the memory requests to the device 700 or transmit only the memory requests to the device 700 that relate to the confidential information 192 managed by the permission modules 110-1 to 110-n. If all the memory requests are transmitted to the device 700, the device 700 may include additional processing logic to carry out some functions also generally carried out at the MMU 720, but overall performance may be increased. If only the memory requests that relate to the confidential information 192 managed by the permission modules 110-1 to 110-n are sent to
the device 700, the device 700 may have a more compact design and allow for
easier integration into the system 700. The memory request is transmitted to all of
the plurality of permission modules 110-1 to 110-n when received by the device
700.

[0059] FIG. 8 is an example block diagram of a computing device 800 including
instructions for controlling access to a memory region (not shown). In the
embodiment of FIG. 8, the computing device 600 includes a processor 810 and a
machine-readable storage medium 820. The machine-readable storage medium 820
further includes instructions 822, 824, 826 and 828 for controlling access to the
memory region (not shown).

[0060] The computing device 800 may be, for example, a chip set, a notebook
computer, a slate computing device, a portable reading device, a wireless email
device, a mobile phone, or any other device capable of executing the instructions
822, 824, 826 and 828. In certain examples, the computing device 800 may include
or be connected to additional components such as memories, sensors, displays, etc.

[0061] The processor 810 may be, at least one central processing unit (CPU), at
least one semiconductor-based microprocessor, at least one graphics processing unit
(GPU), other hardware devices suitable for retrieval and execution of instructions
stored in the machine-readable storage medium 820, or combinations thereof. The
processor 810 may fetch, decode, and execute instructions 822, 824, 826 and 828 to
implement controlling access to the memory region. As an alternative or in addition
to retrieving and executing instructions, the processor 810 may include at least one
integrated circuit (IC), other control logic, other electronic circuits, or combinations thereof that include a number of electronic components for performing the functionality of instructions 822, 824, 826 and 828.

[0062] The machine-readable storage medium 820 may be any electronic, magnetic, optical, or other physical storage device that contains or stores executable instructions. Thus, the machine-readable storage medium 820 may be, for example, Random Access Memory (RAM), an Electrically Erasable Programmable Read-Only Memory (EEPROM), a storage drive, a Compact Disc Read Only Memory (CD-ROM), and the like. As such, the machine-readable storage medium 820 can be non-transitory. As described in detail below, machine-readable storage medium 820 may be encoded with a series of executable instructions for controlling access to the memory region.

[0063] Moreover, the instructions 822, 824, 826 and 828 when executed by a processor (e.g., via one processing element or multiple processing elements of the processor) can cause the processor to perform processes, such as, the process of FIG. 9. For example, the compare PID instructions 822 may be executed by the processor 810 to compare a PID of a process (not shown) to one or more allowed PIDs. The compare requested address instructions 824 may be executed by the processor 810 to compare a requested address to an allowed address range.

[0064] The compare requested access type instructions 826 may be executed by the processor 810 to compare the requested access type to one or more allowed access types. The allow instructions 828 may be executed by the
processor 810 to allow the process to access the memory region including confidential information (not shown), if the one or more allowed PIDs includes the PID of the process, the allowed address range includes the requested address and the one or more allowed access types includes the requested access type.

[0065] The machine-readable storage medium 820 may also include instructions (not shown) to at least one of generate an audit log and carry out an action if the one or more allowed PIDs does not include the PID of the process, the allowed address range does not include the requested address and/or the one or more allowed access types does not include the requested access type. The audit log includes at least one of the PID of the process, the requested memory address, the requested access type, a user ID (UID), a requested purpose, a number of failed access attempts, and a timestamp of the one or more failed access attempts. The action includes at least one of doing nothing, erasing a value at the requested memory address, erasing all values at the memory region, and generating an interrupt. An operation of the device 800 may be described in more detail with respect to FIG. 9.

[0066] FIG. 9 is an example flowchart of a method 900 for controlling access to a memory region 190. Although execution of the method 900 is described below with reference to the device 600, other suitable components for execution of the method 900 can be utilized, such as the device 100. Additionally, the components for executing the method 900 may be spread among multiple devices (e.g., a processing device in communication with input and output devices). In certain scenarios, multiple
devices acting in coordination can be considered a single device to perform the method 900. The method 900 may be implemented in the form of executable instructions stored on a machine-readable storage medium, such as storage medium 820, and/or in the form of electronic circuitry.

[0067] At block 905, the device 600 receives from a process 180 a request to access a memory region 190 having confidential information 192. The request includes a process ID (PID) of the process, a requested address, and a requested access type. Then, at block 910, the device 600 compares the PID of the process 180 to one or more allowed PIDs 222 and flows to block 915. If the one or more allowed PIDs does not include the PID of the process 180, the device 600 generates an action at block 920. As explained above, the action includes at least one of doing nothing, erasing a value at the requested memory address, erasing all values at the memory region, and generating an interrupt.

[0068] If the one or more allowed PIDs include the PID of the process 180, the device 600 compares the requested address to an allowed address range at block 925 and then flows to block 930. If the allowed address range does not include the requested address, the device 600 generates the action at block 920.

[0069] If the allowed address range includes the requested address, the device 600, compares the requested access type to one or more allowed access types at block 935 and then flows to block 940. If the one or more allowed access types include the requested access type, the device 600 allows the process 180 to access the memory region 190 at block 945. If the one or more allowed access
types does not include the requested access type, the device 600 flows to block 950 to determine if an audit log is enabled. If the audit log is not enabled, the device 600 denies the process 180 access to the memory region 190 at block 960. If the audit log is enabled, the device 600 generates the audit log 660 at block 955 and then denies the process 180 access to the memory region 190 at block 960.

[0070] According to the foregoing, embodiments may increase a security and/or integrity of the confidential information without greatly increasing a complexity of the device. For example, in embodiments, a memory request from a process may be received. The memory request may include PID of the process, a requested memory address, and a requested access type. The memory request may be compared to a permission set associated with a memory region including the confidential information. Access to the memory region by the process may be controlled based on the comparison.
CLAIMS

We claim:

1. A device comprising:

   a permission module to receive a memory request from a process, the
   memory request including a process ID (PSD) of the process, a requested memory
   address, and a requested access type, wherein

       the permission module compares the memory request to a permission set
       associated with a memory region storing confidential information, and

       the permission module controls access to the memory region by the process
       based on the comparison.

2. The device of claim 1, wherein,

       the permission set includes one or more allowed PIDs, a memory address
       range, and one or more allowed access types associated with the memory region,
       the access types include at least one of read, write and reference types, and
       the reference type relates to providing a pointer to the requested memory
       address.

3. The device of claim 2, wherein the permission module includes,

       a PID module to compare the PID of the process to the one or more allowed
       PIDs;
an address module to compare the requested memory address to the memory address range; and
an access type module to compare the requested access type to the one or more allowed access types, wherein
the permission module allows access to the memory region by the process based on the comparisons of the PSD, address, and access type modules.

4. The device of claim 2, wherein,
the memory request further includes a user ID (UID) field to identify at least one of a type of user and an identity of the user, and
the permission set includes a plurality of instances of part of the permission set for the memory region, each of the instances to be associated with one of a plurality of UIDs.

5. The device of claim 2, wherein,
the memory request further includes a requested purpose to indicate how the confidential information at the requested memory address is to be used,
the permission set includes one or more allowed purposes,
the permission module includes a purpose module to compare the requested purpose to the one or more allowed purposes, and
the permission module allows access to the memory region by the process based on the comparisons of the PID, address, access type, and permission modules.

6. The device of claim 2, further comprising:
   an audit module to generate an audit log if the access type module determines that the one or more allowed access types does not include the requested access type, wherein
   the audit log includes at least one of the PID of the process, the requested memory address, the requested access type, a user ID (UID), a requested purpose, a number of failed access attempts, and a timestamp of the one or more failed access attempts.

7. The device of claim 2, further comprising:
   an action module to carry out an action if the access type module determines that the one or more allowed access types does not include the requested access type, wherein
   the action includes at least one of doing nothing, erasing a value at the requested memory address, erasing all values at the memory region, and generating an interrupt, and
   the interrupt at least one of blocks access to and halts operation of a functionality of the device.
8. The device of claim 2, wherein,

the permission module is to assert a miss signal if at least one of the PID module indicates that the one or more allowed PiDs does not include the PID of process and the address module indicates that the memory address range does not include the requested memory address, and

the permission module is to assert a fault signal if the access type module indicates that the one or more allowed access types does not include the requested access type.

9. A system, comprising:

a plurality of the permission modules of claim 8;

a memory including a plurality of memory regions, each of the permission modules corresponding to one of the memory regions;

a processor to output the request and to receive the miss; and

a memory management unit (MMU) to transmit the request to the plurality of permission modules.

10. The system of claim 9, wherein,

each of the permission modules include separate permission sets associated with one of the memory regions,

the system asserts an action signal only if all of the permission modules
assert the miss signal,

    the system asserts an audit signal only if all the of the permission modules
assert the fault signal, and

    the system asserts an access control signal to grant the processor access to
the memory, if at least one of the permission modules allows access.

11. The system of claim 9, wherein,

    the MMU receives a plurality of memory requests, and
the MMU one of transmits all of the memory requests and only the memory
requests related to the memory regions associated with at least one of the
permission modules.

12. A method for memory access, comprising:

    receiving from a process a request to access to a memory region having
confidential information, the request including a process ID (PID) of the process, a
requested address, and a requested access type;
    comparing the PID of the process to one or more allowed PIDs;
    comparing the requested address to an allowed address range, if the one or
more allowed PIDs include the PID of the process;
    comparing the requested access type to one or more allowed access types,
if the allowed address range includes the requested address; and
    allowing the process to access the memory region, if the one or more
allowed access types includes the requested access type.

13. The device of claim 12, further comprising:

   generating an action if at least one of the one or more allowed PIDs does not include the PSD of the process and the allowed address range does not include the requested address; and

   generating an audit log if the one or more allowed access types does not include the requested access type.

14. A non-transitory computer-readable storage medium storing instructions that, if executed by a processor of a device, cause the processor to:

   compare a process ID (PID) of a process to one or more allowed PIDs;

   compare a requested address to an allowed address range;

   compare the requested access type to one or more allowed access types;

   and

   allow the process to access a memory region including confidential information, if the one or more allowed PIDs includes the PID of the process, the allowed address range includes the requested address and the one or more allowed access types includes the requested access type.
15. The non-transitory computer-readable storage medium of claim 14, further comprising instructions that, if executed by the processor, cause the processor to;

at least one of generate an audit log and carry out an action if at least one of the one or more allowed PIDs does not include the PID of the process, the allowed address range does not include the requested address and the one or more allowed access types does not include the requested access type, wherein

the audit log includes at least one of the PID of the process, the requested memory address, the requested access type, a user ID (UID), a requested purpose, a number of failed access attempts, and a timestamp of the one or more failed access attempts, and

the action includes at least one of doing nothing, erasing a value at the requested memory address, erasing all values at the memory region, and generating an interrupt.
FIG. 9

905 Receive a request to access to a memory region

910 Compare the PID of the process to allowed PIDs

915 Allowed PIDs include process PID?

Yes

920 Compare the requested address to an allowed address range

No

930 Allowed address range include the requested address?

Yes

940 Allowed access types include the requested access type?

No

950 Audit log enabled?

Yes

955 Generate an audit log

No

960 Deny the process access to the memory region

Yes

935 Compare the requested access type to allowed access types

No

945 Allow the process to access the memory region

Yes

900
INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2012/023385

A. CLASSIFICATION OF SUBJECT MATTER

G11C 7/24(2006.01)i, G06F 12/14(2006.01)1, G06F 21/22(2006.01)1

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)
G11C 7/24; G06F 11/34; G06F 1/00; G06F 13/28

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: memory, request, permission, compare

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>A</td>
<td>US 2010-0312940 A1 (SHINOHARA) 09 December 2010 See abstract, claim 1, and figures 2-3.</td>
<td>1-15</td>
</tr>
<tr>
<td>A</td>
<td>US 6553492 B1 (HOSSE) 22 April 2003 See abstract, claim 1, and figure 2.</td>
<td>1-15</td>
</tr>
<tr>
<td>A</td>
<td>US 05669564A A (JUNYA) 21 November 1995 See abstract, claim 1, and figure 1.</td>
<td>1-15</td>
</tr>
</tbody>
</table>

Further documents are listed in the continuation of Box C.

* 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 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
22 MAY 2012 (22.05.2012)

Date of mailing of the international search report
22 MAY 2012 (22.05.2012)

Name and mailing address of the ISA/KR
Korean Intellectual Property Office
189 Cheonsga-ro, Seo-gu, Daejeon Metropolitan City, Republic of Korea
Facsimile No. 82-42-472-7140

Authorized officer
Yoon, Nan Young

Telephone No. 82-42-481-8188

Form PCT/ISA/210 (second sheet) (July 2009)
<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 06047376A A</td>
<td>04.04.2000</td>
</tr>
</tbody>
</table>