



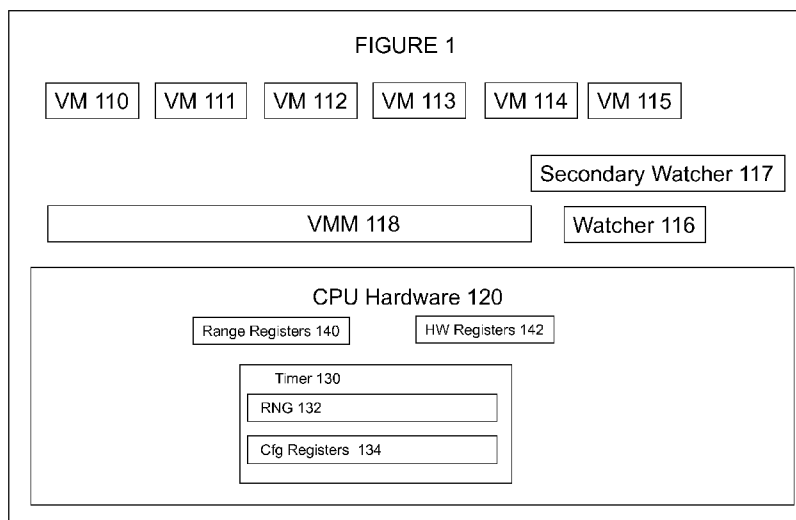
- (51) **International Patent Classification:**
G06F 21/53 (2013.01) *G06F 21/70* (2013.01)
- (21) **International Application Number:**
PCT/US2013/046573
- (22) **International Filing Date:**
19 June 2013 (19.06.2013)
- (25) **Filing Language:** English
- (26) **Publication Language:** English
- (30) **Priority Data:**
13/539,299 29 June 2012 (29.06.2012) US
- (71) **Applicant (for all designated States except US):** **INTEL CORPORATION** [US/US]; 2200 Mission College Boulevard, Santa Clara, California 95054 (US).
- (72) **Inventors; and**
- (71) **Applicants (for US only):** **THOMAS, Brent** [US/US]; 2184 W. Olive Way, Chandler, Arizona 85248 (US). **DATTA, Shamanna** [US/US]; 532 NE Lenox Street, Hillsboro, Oregon 97124 (US). **DURRANT, Scott** [US/US]; 5997 SE Lois St., Hillsboro, Oregon 97123 (US). **MUNOZ, Alberto** [US/US]; 1739 Hawkins Drive, Los Altos, California 94024 (US).
- (74) **Agents:** **LANE, Thomas R.** et al.; Intel Corporation, c/o CPA GLOBAL, P.O. Box 52050, Minneapolis, Minnesota 55402 (US).

- (81) **Designated States** (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.
- (84) **Designated States** (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:

— with international search report (Art. 21(3))

(54) **Title:** TIMER FOR HARDWARE PROTECTION OF VIRTUAL MACHINE MONITOR RUNTIME INTEGRITY WATCHER



(57) **Abstract:** An apparatus and method for hardware protection of a virtual machine monitor (VMM) runtime integrity watcher is described. A set of one or more hardware range registers that protect a contiguous memory space that is to store the VMM runtime integrity watcher. The set of hardware range registers are to protect the VMM runtime integrity watcher from being modified when loaded into the contiguous memory space. The VMM runtime integrity watcher, when executed, performs an integrity check on a VMM during runtime of the VMM. Execution of the VMM runtime integrity watcher is triggered by a timer event generated based on multiple frequency bands.



TIMER FOR HARDWARE PROTECTION OF VIRTUAL MACHINE MONITOR RUNTIME INTEGRITY WATCHER

FIELD

This field of invention relates generally to computer processor architecture, and
5 more specifically, to hardware protection of virtual machine monitor runtime integrity
watcher software.

BACKGROUND

Consumers (and potential consumers) of cloud computing services have expressed
reluctance to deploy sensitive data and workloads into cloud infrastructures due to concern
10 about the infrastructure's ability to protect their sensitive information. In a cloud
environment, this means providing adequate security for the virtualized environment.

For example, in a virtual environment (as most infrastructure as a service (IaaS)
deployments are), it is the job of the virtual machine monitor (VMM) to provide security
services to the virtual machines (VMs) it is managing. Accordingly, the VMM has a very
15 high asset value since if it is compromised, then security of all the VMs (workloads) that it
is monitoring is also compromised.

Technology exists, such as the Intel Trusted eXecution Technology (TXT), that
ensures launch time integrity of a VMM. However, in a typical virtual IaaS environment,
a launched VMM may keep running for over a month at a time, thereby increasing the
20 vulnerability of this critical asset.

BRIEF DESCRIPTION OF THE DRAWINGS

The present invention is illustrated by way of example and not limitation in the
figures of the accompanying drawings, in which like references indicate similar elements
and in which:

25 Figure 1 illustrates an exemplary system for hardware protection of a virtual
machine monitor runtime integrity watcher according to one embodiment; and

Figure 2 is a flow diagram illustrating exemplary operations performed by a system
providing hardware protection of a VMM runtime integrity checking application
according to one embodiment.

DETAILED DESCRIPTION

30 In the following description, numerous specific details are set forth. However, it is
understood that embodiments of the invention may be practiced without these specific

details. In other instances, well-known circuits, structures and techniques have not been shown in detail in order not to obscure the understanding of this description.

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

As detailed earlier, if a virtual machine monitor (VMM) is compromised, then security of all VMs (workloads) that the VMM is managing is also compromised. Embodiments described herein provide runtime integrity protection for the root VMM.

Figure 1 illustrates an exemplary system for hardware protection of a virtual machine monitor runtime integrity watcher according to one embodiment. The CPU hardware 120 includes the hardware necessary to execute the virtual machines (VMs) 110-114, the main VMM 118 and the main VMM runtime integrity watcher program 116 (“watcher”). The VMM 118 is the stand-alone root VMM that runs on the bare platform hardware and provides security services to the VMs 110-114 and is able to retain selective control of processor resources, physical memory, interrupt management, and data I/O provided through the CPU hardware 120 for the VMs 110-114.

The watcher 116 is a program that monitors the content and behavior of the main VMM 118 (e.g., to detect malware in the VMM 118 or other malicious software or behavior caused by such malware). In one embodiment, the watcher 116 is loaded into a hardware range register protected contiguous memory space from which it has visibility into the entire VMM’s memory and resources. Thus, the watcher 116 may monitor the VMM’s memory and resources. In one embodiment, hardware range register(s) 140 are defined that define the bounds of the memory address space where the code for the watcher 116 resides.

The hardware range register(s) 140 may be part of the processor core page management unit and work in conjunction with the memory management architecture of the processor. In one embodiment, the VMM 118 sets the values of the hardware range register(s) 140 when it launches and sets a lock bit, which prevents the values to be subsequently changed (unless a reboot occurs).

The hardware range register protected contiguous memory space prevents other entities on the platform, including the VMM 118, from modifying the watcher 116 or affecting its behavior. For example, the CPU hardware 120 sets an internal identifier inside the page management hardware as soon as the code transitions into the hardware
5 range register protected contiguous memory space that is defined by the range registers (e.g., the execution of the watcher 116). This identifier indicates being in watcher mode. As will be described in greater detail later herein, execution of the watcher 116 will be triggered as a result of certain events occurring. Thus, the identifier is set only when the processor execution transfers control to the watcher 116. The identifier is reset when the
10 processor execution transfers control back to the VMM 118.

For example, memory accesses made from the watcher 116 to any other memory space (excluding System Management Code space) is allowed by the page management hardware. For example, the page management unit consults the internal identifier to determine whether memory access is allowed. Access from code that is not executing in
15 the space defined by the hardware range register(s) 140 into the space defined by the range register(s) 140 is directed into an abort page by the page management hardware unit. In such a case, for example, the code executing outside the space defined by the hardware range register(s) 140 will read all ones (0xff). By way of example, the VMM 118 is prevented from accessing the VMM integrity watcher 116 (e.g., prevented by the page
20 management hardware unit). For example, if the identifier is not set, then the page management unit directs the request to an abort page. The VMM 118 should not be able to make a memory access when the identifier is set, since execution of the VMM 118 is preempted when the event is detected and control moves to the watcher 116.

In one embodiment, access to the memory space defined by the range register(s) 140
25 from code executing in System Management Mode is allowed. Thus, System Management Mode is treated as a supervisory mode and is allowed to access the watcher 116. The code executing in System Management Mode may be inside a Trusted Computing Base. System Management Mode may allow supervisory functions such as Reliability, Availability, and Serviceability (RAS) to be performed.

30 In one embodiment, the watcher 116 is executed due to a hardware guaranteed immutable triggering mechanism that invokes the execution of the watcher 116. For example, in one embodiment, a hardware timer 130 is used to generate a timer event that causes the execution of the VMM 118 to be preempted and control brought into the watcher 116 (which is loaded in the protected memory space). Timer events may be

communicated securely and privately to VMM 118 and/or other software such as Integrity Measurement Software (IMS) using encryption and/or endpoint attestation and link shielding.

Design of the timer is based on several considerations. If the frequency of timer events is too high, performance may be affected. If the frequency of timer events is too low, an attack may take place between timer events. If the frequency of timer events is predictable, attacks might be synchronized to take place between timer events. Therefore, embodiments of the present invention may use a multi-frequency band, randomized timer.

For example, timer 130 may be programmed to generate timer events using four separate frequency bands. The timer may generate an event from one of the four bands according to a probability distribution, such that the number of events is distributed amongst the bands as desired. For example, 99.4% of events may be generated based on a frequency band with a period of 200msec, 0.35% of events may be generated based on a frequency band with a period of 10msec, 0.16% of events may be generated based on a frequency band with a period of 1msec, and 0.09% of events may be generated based on a frequency band with a period of 0.3msec. In this way, the frequency of events is more variable and less predictable, and not too high or too low. The selection of band may be based on a random number from random number generator 132 or other random number source, where the random number is used to select a band according to the desired probability distribution, then to select a random delay from within the selected band. Timer 130 may include programmable registers 134 to provide for configuration of the timer according to the desired probability distribution. In various embodiments, the probability distribution may be automatically or otherwise adjusted based on any desired factors, such as time of day, power requirements, etc. Any number of bands and any distribution among bands may be used within the scope of the present invention.

In other embodiments, the watcher 116 may also be invoked due to certain events (referred herein as “voluntary events”) such as a bad I/O port and control register or machine specific register (MSR) execution, and control is brought to the watcher 116.

After the execution of the watcher 116 is invoked, it operates in the hardware protected region unaffected by malware and cryptographically verifies the integrity of the VMM 118. In one embodiment, the watcher 116 uses well known cryptographic algorithms (e.g., SHA-2) to compute hash(es) of portions of code and/or data during the first invocation (e.g., when the machine initially boots and performs the first invocation) and storing it in the hardware range register protected memory space. Then, upon

subsequent invocations of the watcher 116 during the runtime of the VMM 118, the watcher 116 repeats the hashing operation on those portions of code and/or data to determine whether the hash values are the same. If they are the same, then integrity of the VMM 118 is not compromised. If they are different, then the integrity of the VMM 118 is compromised. In another embodiment, the hash value(s) of the VMM 118 is generated differently, for example during compilation (in such a case it may be signed).

In one embodiment, a secondary watcher program 117 is included within the VMM 118. The secondary watcher program 117 may be scattered throughout memory. The secondary watcher program 117 can run in VMM context. In addition, the secondary watcher program 117 can run with interrupts on. In embodiments where a secondary watcher program 117 is included within the VMM 118, the watcher program 116 verifies the integrity of the secondary watcher program 117 and then transfers responsibility of verifying the integrity of the VMM 118 either partially or fully to the secondary watcher.

In one embodiment, a set of hardware register(s) 142 of the CPU hardware 120 are used for a secure reporting mechanism to report findings of the watcher 116. In one embodiment, the set of hardware register(s) 142 can be only written by the watcher 116, but can be read by any other software entity. For example, a set of one or more uncore registers may be used for reporting. The set of uncore registers may be writable only by the watcher 116 but able to be read by any other software entity. For example, the set of uncore registers may only be written when in the watcher mode, as identified by the identifier previously described. Thus, if not in watcher mode, a write will not be allowed. When in watcher mode, only the watcher should be executing (since the VMM is preempted) and is allowed to write to the set of uncore registers. In an embodiment, the frequency of reporting may also be based on timer events from timer 130.

In another embodiment, the CPU hardware 120 may provide a secure pathway through other microcontroller(s) that are embedded inside the processor and/or chipset in order to export the report to a management entity (in addition to or in lieu of using the set of hardware register(s) 142 for reporting). Such a management entity may include, for example, a system management application or a cloud management application.

After reporting the results of the integrity check, execution of the watcher 116 exits and control is transferred back to the VMM 118 to resume its execution. By way of example, the watcher 116 issues an instruction architecture trigger (e.g., VMRESUME) instruction causing the VMM 118 to resume its execution.

Figure 2 is a flow diagram illustrating exemplary operations performed by a system providing hardware protection of a VMM runtime integrity checking application according to one embodiment. Figure 2 will be described with reference to the exemplary embodiment of Figure 1. However, it should be understood that the operations described with reference to Figure 2 can be performed by embodiments different than that described with reference to Figure 1, and the embodiment described with respect to Figure 1 can perform operations that are different than that are described with respect to Figure 2.

At operation 202, timer 130 is programmed with a desired probability distribution amongst a number of frequency bands. At operation 204, a random number is generated.

At operation 206, the random number is used to select one of a number of frequency bands according to the desired probability distribution. At operation 208, a delay value is randomly selected from within the selected frequency band. At operation 210, timer 130 waits for the selected delay. At operation 212, timer 130 generates an event to trigger the execution of the watcher 116.

At operation 215, the event causes the hardware 120 to preempt the execution of the VMM 118 and executes the watcher 116. For example, upon the event occurring, the chipset asserts a signal to cause the processor to enter watcher mode at the next instruction boundary. As previously described, in one embodiment the watcher 116 is loaded in a region of contiguous memory space that is protected via hardware range registers from being modified or its behavior affected (including by the VMM 118).

Flow then moves to operation 220, where the watcher 116 verifies the integrity of the VMM. For example, this may include the watcher 116 analyzing the cause of entry (e.g., the reason that it was executed) and generating cryptographic hash(es) of portions of code and/or data of the VMM 118 and comparing it to previously stored values.

In an embodiment where there is a single watcher (that is, in an embodiment where there is not a secondary watcher 117), the watcher 116 performs all analysis and may spend no more than 200 microseconds analyzing the cause of entry since interrupts are off.

In an embodiment where there is a secondary watcher 117 in addition to the watcher 116, the watcher 116 verifies the secondary watcher 117, which runs in VMM context.

The secondary watcher 117 can run with the interrupts on. The secondary watcher 117 may either full, or partially, perform the responsibility of VMM malware checking.

Flow moves from operation 220 to operation 225. At operation 225, the watcher 116 reports its findings. For example, the watcher 116 writes to the register(s) 142 to indicate whether the integrity of the VMM 118 has been compromised. As another

example, the watcher 116 may, in addition to or in lieu of writing to the register(s) 142, reports the findings (e.g., the status of the integrity of the VMM 118) to a management entity such as a system management application or a cloud management application. In such a case, the CPU hardware 120 may provide a secure pathway through

5 microcontroller(s) that are embedded inside the processor and/or the chipset.

In an embodiment where there is a secondary watcher 117, the secondary watcher 117 returns its verification results to the primary watcher 116 and the primary watcher 116 reports the result as described above in one embodiment. In another embodiment, a signed report of the results from the secondary watcher 117 is produced using a platform based
10 cryptographic hardware component (a Trusted Platform Module (TPM)).

In one embodiment, external server(s) (management console(s)) use the results of the integrity check. For example, the results written to the register(s) 142 may be exported by the chipset to management console(s), which may manage multiple servers. As another example, a signed report (signed using a TPM component) is exported to the management
15 console(s). These management consoles may take action upon a determination that integrity was compromised. For example, they may cause the machine to shut down. As another example, they may cause a notification (e.g., email, text message, etc.) to be automatically generated and sent to an administrator and/or a phone call to be automatically placed to an administrator. The management consoles may generate
20 forensic cookies or other log files that can later be used and analyzed by enforcement agencies.

Flow moves from operation 225 to operation 230, where the watcher 116 exits. For example, the watcher 116 issues a trigger to exit the watcher mode (e.g., a VMRESUME instruction), which will cause the VMM 118 to resume its execution. Flow then moves to
25 operation 235 where the VMM 118 resumes its execution.

In another embodiment, the CPU hardware 120 may provide a secure pathway through other microcontroller(s) that are embedded inside the processor and/or chipset in order to export the report to a management entity (in addition to or in lieu of using the set of hardware register(s) 142 for reporting). Such a management entity may include, for
30 example, a system management application or a cloud management application.

Thus, embodiments described herein describe the ability to check VMM integrity after the initial VMM has been loaded through a processor hardware driven run-time VMM integrity monitor (the watcher). As previously described, the check may be performed periodically and/or upon certain event(s) occurring, thereby providing ongoing

assurance of VMM integrity. In addition, embodiments provide for a protected execution space for the run-time integrity checker described herein. Thus, the run-time integrity of VMM can be protected thereby resolving a security concern that exists today for deployment of virtualized servers in the cloud environment.

5

CLAIMS

What is claimed is:

1. An apparatus, comprising:
a set of one or more hardware range registers to protect a contiguous memory
5 space that is to store a virtual machine monitor (VMM) runtime integrity
watcher, wherein the set of hardware range registers are to protect the
VMM runtime integrity watcher from being modified when loaded into the
contiguous memory space;
a timer to generate a timer event based on a plurality of frequency bands; and
10 the VMM runtime integrity watcher to be invoked based on the timer event to
perform an integrity check on a VMM during runtime of the VMM.
2. The apparatus of claim 1, wherein the timer event is based on one of the plurality
of frequency bands selected from plurality of frequency bands according to a probability
distribution of desired number of events per frequency band.
- 15 3. The apparatus of claim 2, wherein the selection of the one of the plurality of
frequency bands is based on a random number.
4. The apparatus of claim 3, wherein the timer event is generated after a delay
randomly selected from the selected frequency band.
5. The apparatus of claim 2, wherein execution of the VMM is preempted upon the
20 event being generated.
6. The apparatus of claim 1, wherein the VMM runtime integrity watcher is further
to, when executed, report results of the integrity check.
7. The apparatus of claim 6, wherein the VMM runtime integrity watcher is to report
results of the integrity check to one of a system management application and a cloud
25 management application.6
8. The apparatus of claim , further comprising:
a set of one or more hardware reporting registers; and
wherein the VMM runtime integrity watcher is to write to the set of hardware
reporting registers to indicate whether the VMM has been compromised.
- 30 9. The apparatus of claim 8, wherein the set of hardware reporting registers are
writable only by the VMM runtime integrity watcher and readable by any entity.
10. A computer-implemented method in a computing system, comprising:
loading a virtual machine monitor (VMM) runtime integrity watcher into a
contiguous memory space of the computing system, wherein the contiguous

memory space is protected by a set of one or more hardware range registers that protect the loaded VMM runtime integrity watcher from being modified;

generating a timer event to invoke execution of the VMM runtime integrity

5 watcher, wherein the generation of the timer event is based on a plurality of frequency bands;

responsive to the timer event, executing the VMM runtime integrity watcher; and

performing, by the VMM runtime integrity watcher, an integrity check on a VMM during runtime of the VMM.

10 11. The computer-implemented method of claim 10, wherein the timer event is based on one of the plurality of frequency bands selected from plurality of frequency bands according to a probability distribution of desired number of events per frequency band.

12. The computer-implemented method of claim 11, wherein the selection of the one of the plurality of frequency bands is based on a random number.

15 13. The computer-implemented method of claim 12, wherein the timer event is generated after a delay randomly selected from the selected frequency band.

14. The computer-implemented method of claim 10, wherein responsive to the detected event, preempting execution of the VMM.

15. The computer-implemented method of claim 10, further comprising:

20 reporting, by the VMM runtime integrity watcher, a result of the integrity check.

16. The computer-implemented method of claim 14, wherein reporting includes writing the result of the integrity check to a set of one or more hardware reporting registers.

17. The computer-implemented method of claim 14, wherein reporting includes

25 reporting the result of the integrity check to one of a system management application and a cloud management application.

18. A system, comprising:

a virtual machine monitor (VMM); and

a processor including

30 a set of one or more hardware range registers to protect a contiguous memory

space that is to store a VMM runtime integrity watcher, wherein the set of hardware range registers are to protect the VMM runtime integrity watcher

from being modified when loaded into the contiguous memory space; and

a timer to generate a timer event based on a plurality of frequency bands;

the VMM runtime integrity watcher to be invoked based on the timer event to perform an integrity check on a VMM during runtime of the VMM.

19. The apparatus of claim 18, wherein the timer event is based on one of the plurality of frequency bands selected from plurality of frequency bands according to a probability distribution of desired number of events per frequency band.
- 5 20. The apparatus of claim 19, wherein the timer event is generated after a delay randomly selected from the selected frequency band.

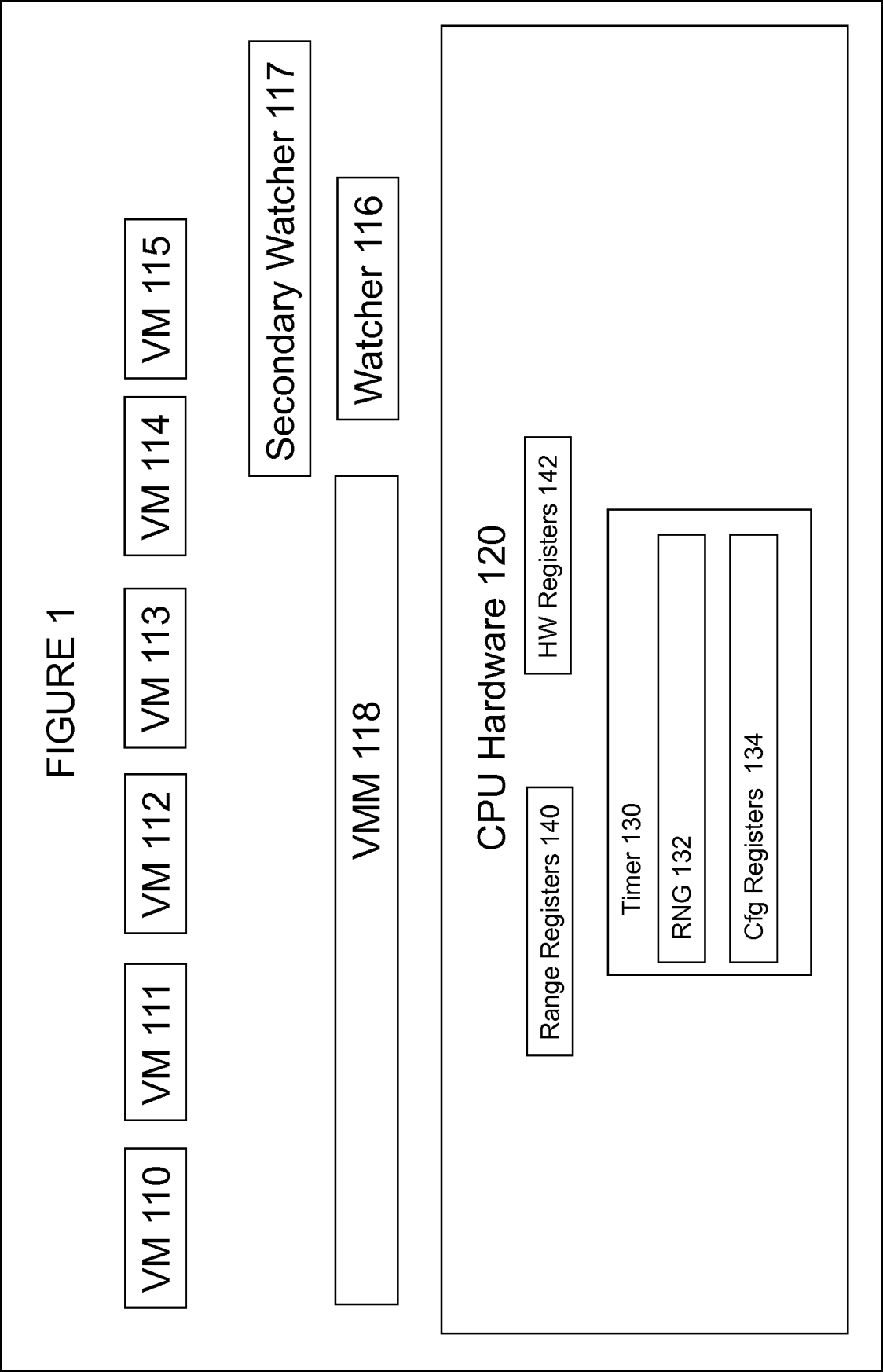
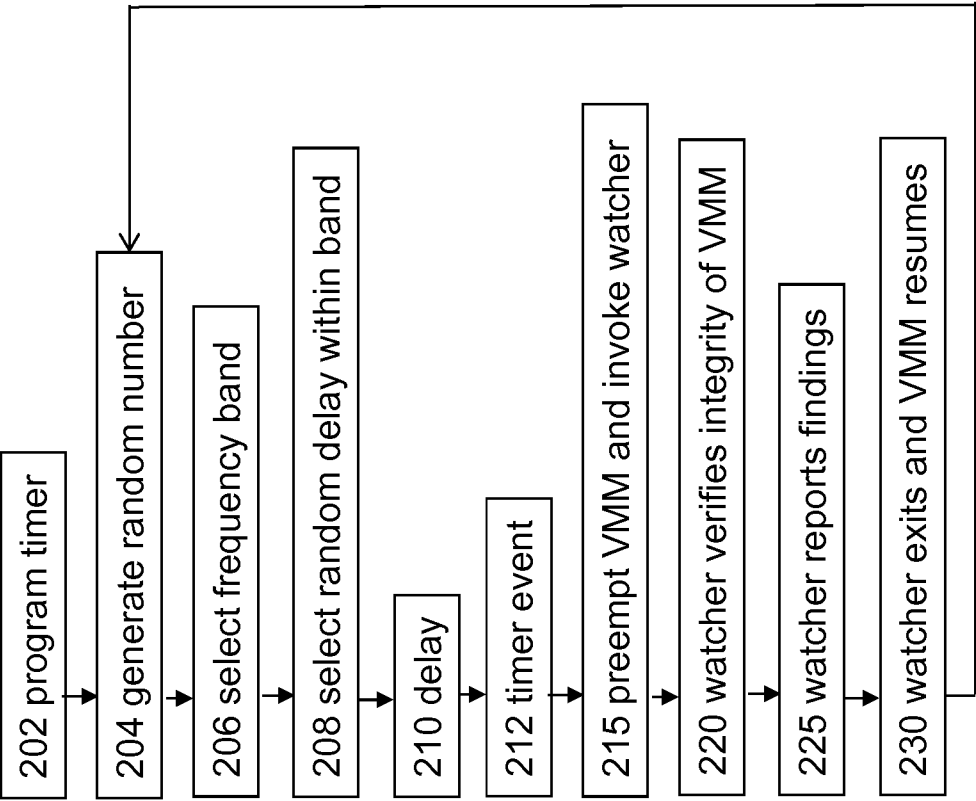


FIGURE 2



A. CLASSIFICATION OF SUBJECT MATTER**G06F 21/53(2013.01)i, G06F 21/70(2013.01)i**

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 21/53; G06F 12/02; G06F 15/173; G06F 9/455; G06F 21/00; G06F 21/70

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: virtual machine monitor (VMM), runtime integrity watcher, timer, hardware protection, integrity check

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2009-0044265 A1 (ANUP K. GHOSH et al.) 12 February 2009 See paragraphs [0032]-[0121]; and figures 1-18.	1-20
A	US 2008-0250406 A1 (TODD L. CARPENTER et al.) 09 October 2008 See paragraphs [0021]-[0055]; and figures 2-7.	1-20
A	US 2009-0164770 A1 (VINCENT J. ZIMMER et al.) 25 June 2009 See paragraphs [0010]-[0029]; and figures 1-3.	1-20
A	US 2008-0244573 A1 (RAVI SAHITA et al.) 02 October 2008 See paragraphs [0008]-[0040]; and figures 1-4.	1-20
A	WO 2011-076464 A1 (INTERNATIONAL BUSINESS MACHINES CORPORATION) 30 June 2011 See pages 7-21; and figures 1-7.	1-20



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 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

15 October 2013 (15.10.2013)

Date of mailing of the international search report

17 October 2013 (17.10.2013)

Name and mailing address of the ISA/KR


 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

KIM Tae Hoon

Telephone No. +82-42-481-8407



INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2013/046573

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2009-0044265 A1	12/02/2009	None	
US 2008-0250406 A1	09/10/2008	WO 2008-124652 A2 WO 2008-124652 A3	16/10/2008 23/06/2011
US 2009-0164770 A1	25/06/2009	US 7962738 B2	14/06/2011
US 2008-0244573 A1	02/10/2008	US 8464251 B2	11/06/2013
WO 2011-076464 A1	30/06/2011	CN 102667794 A JP 2013-515989 A TW 201137660 A	12/09/2012 09/05/2013 01/11/2011