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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
14 May 2015 (14.05.2015)

WIPO | PCT

(10) International Publication Number
WO 2015/069280 A1

(51) International Patent Classification:
G06F 12/00 (2006.01) *G06F 13/14* (2006.01)

(21) International Application Number:
PCT/US2013/069238

(22) International Filing Date:
8 November 2013 (08.11.2013)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **EMPIRE TECHNOLOGY DEVELOPMENT, LLC** [US/US]; 2711 Centerville Road, Suite 400, Wilmington, DE 19808 (US).

(72) Inventor: **KRUGLICK, Ezekiel**; 13842 Deergrass Ct, Poway, CA 92064 (US).

(74) Agent: **TURK, Carl, K.**; Turk IP Law, LLC, 2885 Sanford Ave. S.W. #23998, Grandville, MI 49418 (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, IR, 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, SA, 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).

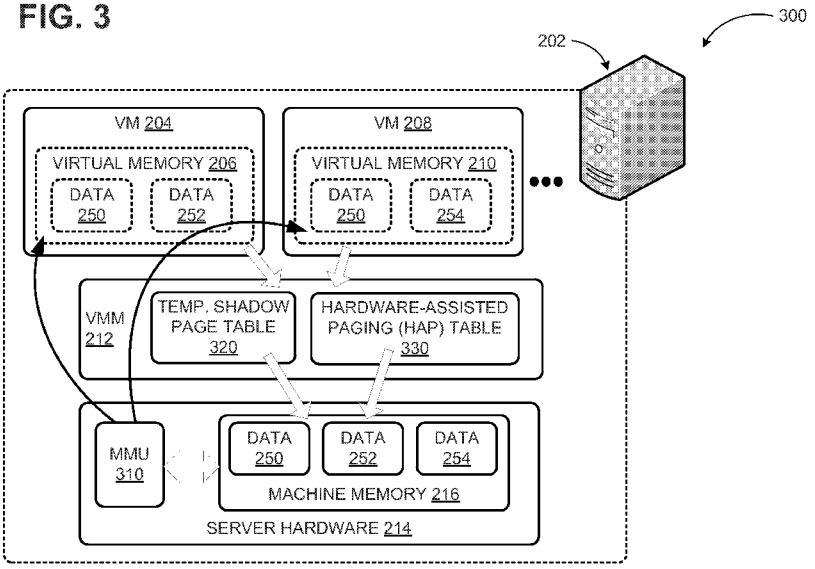
Published:

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

WO 2015/069280 A1

(54) Title: MEMORY DEDUPLICATION MASKING

FIG. 3



(57) Abstract: Technologies are generally described for virtual machine memory deduplication protection through memory deduplication masking. In some examples, a virtual machine manager (VMM) that receives data to be written to memory may initially write the data to entries in a temporary shadow paging table and then subsequently opportunistically update the memory and an associated hardware-assisted paging (HAP) table. Upon receiving a read request for the received data before the data has been written to the memory, the VMM may check the HAP table and determine that the requested data is not available from the memory. Upon determining that the received data is not available from the memory, the VMM may respond to the read request by sending the received data stored in the shadow paging table entries.

MEMORY DEDUPLICATION MASKING

BACKGROUND

[0001] Unless otherwise indicated herein, the materials described in this section are not prior art to the claims in this application and are not admitted to be prior art by inclusion in this section.

[0002] Data deduplication refers to the practice of using hashes or other semi-unique identifiers to identify portions of identical data and replacing those portions with pointers to a single stored master copy of the data portion. By collapsing such duplicated data into a single copy and pointers to that single copy, large and even substantial amounts of memory space may be freed. In network or cloud-based applications, deduplication may allow an operating system or application to be served as multiple instances to multiple customers while using about as much memory as a single instance of the operating system or application. As a result, a given amount of memory may serve substantially more customers when data deduplication is used compared to when data deduplication is not used.

SUMMARY

[0003] The present disclosure generally describes techniques for masking memory deduplication in datacenters.

[0004] According to some examples, methods may be provided to mask memory deduplication in a datacenter. An example method may include writing a first data to an entry in a shadow paging table in response to receiving a write request to write the first data and receiving a read request for the first data from a datacenter customer. The method may further include determining that the first data is not available from a memory based on a hardware-assisted paging (HAP) table and sending the first data from the entry in the shadow paging table to the datacenter customer in response to determination that the first data is not available from the memory.

[0005] According to other examples, a virtual machine manager (VMM) may be provided to mask memory deduplication in a datacenter. The VMM may include a processing module and

a memory configured to store a shadow paging table and a hardware-assisted paging (HAP) table and coupled to the processing module. The processing module may be configured to receive a write request to write a first data, write the first data to an entry in the shadow paging table, receive a read request for the first data from a datacenter customer, and determine, based on the HAP table, that the first data is not available from a memory. The processing module may be further configured to respond to the read request by sending the first data from the entry in the shadow paging table to the datacenter customer in response to determination that the first data is not available from the memory.

[0006] According to further examples, a system may be provided to mask memory deduplication. The system may include at least one virtual machine (VM) operable to be executed on one or more physical machines, a memory configured to store deduplicated data, a memory configured to store a shadow paging table and a hardware-assisted paging (HAP) table, and a controller. The controller may be configured to receive a write request to write a first data to the memory, write the first data to an entry in the shadow paging table, receive a read request for the first data from a customer, and determine, based on the HAP table, that the first data is not available from the memory. The controller may further be configured to respond to the read request by sending the first data from the entry in the shadow paging table to the customer in response to determination that the first data is not available from the memory.

[0007] According to yet further examples, a computer readable medium may store instructions for masking memory deduplication in a datacenter. The instructions may include writing a first data to an entry in a shadow paging table in response to receiving a write request to write the first data and receiving a read request for the first data from a datacenter customer. The instructions may further include determining that the first data is not available from a memory based on a hardware-assisted paging (HAP) table and sending the first data from the entry in the shadow paging table to the datacenter customer in response to determination that the first data is not available from the memory.

[0008] The foregoing summary is illustrative only and is not intended to be in any way limiting. In addition to the illustrative aspects, embodiments, and features described above, further aspects, embodiments, and features will become apparent by reference to the drawings and the following detailed description.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] The foregoing and other features of this disclosure will become more fully apparent from the following description and appended claims, taken in conjunction with the accompanying drawings. Understanding that these drawings depict only several embodiments in accordance with the disclosure and are, therefore, not to be considered limiting of its scope, the disclosure will be described with additional specificity and detail through use of the accompanying drawings, in which:

FIG. 1 illustrates an example datacenter-based system where masking of memory deduplication may be implemented;

FIG. 2 illustrates an example system at a datacenter where memory deduplication may be implemented;

FIG. 3 illustrates an example system at a datacenter where masking of memory deduplication may be implemented;

FIG. 4 illustrates how the system of FIG. 3 may be used to mask memory deduplication at the datacenter;

FIG. 5 illustrates a general purpose computing device, which may be used to provide memory deduplication masking in a datacenter;

FIG. 6 is a flow diagram illustrating an example method for masking memory deduplication at a datacenter that may be performed by a computing device such as the computing device in FIG. 5; and

FIG. 7 illustrates a block diagram of an example computer program product,

all arranged in accordance with at least some embodiments described herein.

DETAILED DESCRIPTION

[0010] In the following detailed description, reference is made to the accompanying drawings, which form a part hereof. In the drawings, similar symbols typically identify similar components, unless context dictates otherwise. The illustrative embodiments described in the detailed description, drawings, and claims are not meant to be limiting. Other embodiments may be utilized, and other changes may be made, without departing from the spirit or scope of the subject matter presented herein. It will be readily understood that the aspects of the present

disclosure, as generally described herein, and illustrated in the Figures, can be arranged, substituted, combined, separated, and designed in a wide variety of different configurations, all of which are explicitly contemplated herein.

[0011] This disclosure is generally drawn, *inter alia*, to methods, apparatus, systems, devices, and/or computer program products related to protection of virtual machine memory deduplication from side channel attacks through memory deduplication masking at datacenters.

[0012] Briefly stated, technologies are generally described for virtual machine memory deduplication protection through memory deduplication masking. In some examples, a virtual machine manager (VMM) that receives data to be written to memory may initially write the data to entries in a temporary shadow paging table and then subsequently opportunistically update the memory and an associated hardware-assisted paging (HAP) table. Upon receiving a read request for the received data before the data has been written to the memory, the VMM may check the HAP table and determine that the requested data is not available from the memory. Upon determining that the received data is not available from the memory, the VMM may respond to the read request by sending the received data stored in the shadow paging table entries.

[0013] While some embodiments are discussed herein in conjunction with datacenters for illustration purposes, embodiments are not limited to datacenter-based applications. Some embodiments may be implemented in single-machine configurations such as individual computing devices or even smart phones, where virtualization may be employed. For example, micro-VMs may be employed in individual computing devices for security purposes, where embodiments may be implemented to enhance security through memory deduplication masking.

[0014] A datacenter as used herein refers to an entity that hosts services and applications for customers through one or more physical server installations and one or more virtual machines executed in those server installations. Customers of the datacenter, also referred to as tenants, may be organizations that provide access to their services for multiple users. One example configuration may include an online retail service that provides retail sale services to consumers (users). The retail service may employ multiple applications (e.g., presentation of retail goods, purchase management, shipping management, inventory management, etc.), which may be hosted by one or more datacenters. Thus, a consumer may communicate with those applications of the retail service through a client application such as a browser over one or more networks and receive the provided service without realizing where the individual applications are actually

executed. This scenario contrasts with configurations where each service provider would execute their applications and have their users access those applications on the retail service's own servers physically located on retail service premises. One result of the networked approach described herein is that applications and virtual machines on a datacenter may be vulnerable to side channel attacks that take advantage of memory deduplication to preserve memory resources.

[0015] FIG. 1 illustrates an example datacenter-based system where masking of memory deduplication may be implemented, arranged in accordance with at least some embodiments described herein.

[0016] As shown in a diagram 100, a physical datacenter 102 may include one or more physical servers 110, 111, and 113, each of which may be configured to provide one or more virtual machines 104. For example, the physical servers 111 and 113 may be configured to provide four virtual machines and two virtual machines, respectively. In some embodiments, one or more virtual machines may be combined into one or more virtual datacenters. For example, the four virtual machines provided by the server 111 may be combined into a virtual datacenter 112. The virtual machines 104 and/or the virtual datacenter 112 may be configured to provide cloud-related data/computing services such as various applications, data storage, data processing, or comparable ones to a group of customers 108, such as individual users or enterprise customers, via a cloud 106.

[0017] Network or cloud-based applications may use data deduplication to provide multiple instances of an operating system or application to multiple users from a single instance of the operating system and/or application, thereby serving more users with the same memory. For example, an instance of an operating system or application may use a particular file, library, or application programming interface (API), but may not modify that data during execution. Multiple instances of the operating system/application may then be served using pointers to a single copy of the data. Deduplication may be used for both files (i.e., data stored on a hard disk drive, solid-state drive, tape drive, and the like) and data in memory (i.e., data used by currently-executing operating systems or applications, typically stored in random-access memory and the like). In the present disclosure, masking of memory deduplication is described in the context of data in memory, but the same techniques may be applicable to files.

[0018] FIG. 2 illustrates an example system at a datacenter where memory deduplication may be implemented, arranged in accordance with at least some embodiments described herein.

[0019] According to a diagram 200, a physical server 202 (similar to the physical servers 110, 111, and 113) includes server hardware 214 and a machine memory 216 for storing deduplicated data, and may be configured to execute a first virtual machine (VM) 204 and a second VM 208. The VMs 204 and 208 may each execute one or more operating systems and/or applications, each of which may store or retrieve data from memory (e.g., the machine memory 216). For example, the VM 204 may have an associated virtual memory 206 which stores a data 250 and a data 252, each of which may be associated with an operating system or application executing on the VM 204. Similarly, the VM 208 may have an associated virtual memory 210 that stores the data 250 and a data 254, each of which may be associated with an operating system or application executing on the VM 208. The data 250 and 252 associated with the VM 204, while appearing to be stored in the virtual memory 206, may actually be data pointers that refer to a single copy of the data 250 and the data 252 stored at the machine memory 216. Similarly, the data 250 and 254 in the virtual memory 210 may actually be data pointers that refer to the copies of the data 250 and the data 254 stored at the machine memory 216. As a result, data that is common to both VMs, such as the data 250, can be stored as a single copy of the data 250 at the machine memory 216 and pointers to that copy of the data 250 at each of the virtual memories 206 and 210.

[0020] Virtual memory may be supported by operating systems so that an application or process executing on the operating system can access the entire available memory space. For example, an operating system may maintain a group of page tables that map virtual memory addresses to memory addresses for each application or process. In a non-virtualized environment (i.e., where a computer executes a single operating system), memory addresses may correspond to actual machine memory addresses (i.e., addresses of the memory on the computer hardware).

[0021] On the other hand, in a virtualized environment as depicted in the diagram 200, each of the VMs 204 and 208 may execute a different, "guest" operating system instance. Each guest operating system instance may maintain a group of page tables that map guest virtual memory addresses to guest memory addresses, as described above. However, since each guest operating system actually executes in a guest virtual machine (e.g., the VMs 204/208), the guest memory addresses may not correspond to actual machine memory addresses, in contrast to the non-virtualized example described above. Instead, the memory seen by each operating system instance may be virtualized and further mapped to memory addresses at the machine memory

216. For example, the virtual memory 206 associated with the VM 204 may appear to store the data 250 and 252. An application executing at the VM 204 may access the data 250 and/or 252 using guest virtual memory addresses, which an operating system executing at the VM 204 may translate to guest memory addresses. Subsequently, a virtual machine manager (VMM) 212 may either translate the guest memory addresses or the original guest virtual memory addresses to machine memory addresses at the machine memory 216, where the data 250 and 252 are actually stored.

[0022] As described above, memory deduplication may allow an operating system or application to be served as multiple instances to multiple users while using significantly less memory than a system in which deduplication is not used. As such, memory deduplication may be desirable for network or cloud service providers, such as datacenters. However, using memory deduplication may make datacenters susceptible to side channel attacks based on data access timing. For example, data that is deduplicated may exist as a single copy that serves multiple VMs. When a particular VM attempts to write to that data, another copy of that data may first be generated before the write is performed (also known as "copy-on-write"). As a result, the write access time for deduplicated data may be greater than the write access time for data that is not deduplicated due to the additional time required to generate the copy in the copy-on-write process. An attacker may then use this write access time difference to determine whether some particular data was deduplicated by another VM. This type of attack may be used for cloud cartography to determine what tasks are running where, co-location detection to determine if a specific target user is co-located with an attacker VM, and even for direct side channel attacks. In one specific scenario, an attacker may email a corporate target a graphics file with a unique binary representation and use attack VMs scattered through a datacenter to detect deduplication of the image and thus locate machines where the target receives or opens the image, allowing a highly-targeted attack.

[0023] FIG. 3 illustrates an example system at a datacenter where masking of memory deduplication may be implemented, arranged in accordance with at least some embodiments described herein.

[0024] A diagram 300 of FIG. 3 shares some similarities with the diagram 200 in FIG. 2, with similarly-numbered elements behaving similarly. According to the diagram 300, the server hardware 214 may include a memory management unit (MMU) 310 that manages the mapping

of guest virtual memory addresses to machine memory addresses. In some embodiments, the MMU 310 may manage this mapping in conjunction with the VMM 212. The VMM 212 may perform memory virtualization based on techniques such as para-virtualization, shadow paging-based full virtualization, and hardware-assisted full virtualization. In para-virtualization, the guest operating system may be modified to be able to communicate directly with the VMM 212 and thereby access machine memory. Para-virtualization may need the guest operating system to be modified to be aware of its virtualization, and as such may introduce additional difficulties in the virtualization process. In shadow paging-based full virtualization, the VMM 212 may maintain a shadow page table that maps guest virtual addresses to machine addresses while the guest operating system may maintain its own virtual-to-physical address translation table. Since the VMM 212 may store what is essentially an additional set of full page tables, shadow paging-based virtualization may be memory-intensive. Moreover, in hardware-assisted full virtualization, the MMU may maintain a guest page table that translates between virtual addresses and physical addresses, while specialized hardware at the server hardware 214 maintains page tables that translate between physical addresses and machine addresses. These tables may be collectively known as a hardware-assisted paging (HAP) table, and may effectively map virtual addresses to machine addresses. Using hardware-assisted virtualization and HAP tables may allow guest VMs to handle their own page faults or errors without triggering VM exits, as well as avoid the memory-intensive nature of shadow paging-based virtualization.

[0025] In order to defeat side channel attacks using write access timing as described above, the VMM 212 may combine memory virtualization techniques. For example, the VMM 212 may maintain a temporary shadow page table 320 and a HAP table 330. The temporary shadow page table 320 may be used to respond to requests for just-written data, and the HAP table 330 may be used to respond to other data requests, as described below in more detail in FIG. 4.

[0026] FIG. 4 illustrates how the system of FIG. 3 may be used to mask memory deduplication at the datacenter, arranged in accordance with at least some embodiments described herein.

[0027] As depicted in a diagram 400, the physical server 202 may include a machine memory 216 (as part of the server hardware 214), a temporary shadow page table 320 maintained by the VMM 212, and a hardware-assisted paging (HAP) table 330, also maintained by the

VMM 212. A VM associated with a customer (e.g., of the datacenter) may request a write 410 to the machine memory 216. For example, the write 410 may contain a modified version of deduplicated data stored at the machine memory 216. In some embodiments, the deduplicated data may be associated with an operating system and/or an application. Upon receiving the write 410, the VMM 212 may create a new entry at the temporary shadow paging table 320 and store the write 410 in the newly-created entry. Subsequently, the VMM 212 may update the machine memory 216 and the HAP table 330 based on entries in the temporary shadow page table 320. For example, the VMM 212 may create a copy of the original, deduplicated data at the machine memory 216, write the modified data to the copy, and then update the HAP table 330 based on the copy. In some embodiments the VMM 212 may perform an opportunistic updating 412 of the machine memory 216 and the HAP table 330. For example, the VMM 212 may decide to perform the opportunistic updating 412 upon determination that it has sufficient time or processing capability to perform the updating or if the temporary shadow page table 320 has reached a particular size. After performing the opportunistic updating 412, the VMM 212 may then delete the corresponding entries in the temporary shadow page table 320, thus keeping the size of the temporary shadow page table 320 relatively small.

[0028] A VM associated with the same or a different customer may also request a read 414 of data stored at the physical server 202. Upon receiving the read 414, the VMM 212 may use the HAP table 330 to look up the requested data from the machine memory 216. In response to determination that the data is present in the machine memory 216, the VMM 212 may then use the HAP table 330 to first retrieve the data and then send it back from the machine memory 216 to the requesting VM in a return 418.

[0029] In some embodiments, a VM associated with the same or a different customer may request a read 430 of data that has just been sent to the VMM 212 in a previous write. For example, a VM may request the read 430 for the modified version of the data stored at the machine memory 216 written by the write 410. In this situation, the VMM 212 may not have performed the opportunistic updating 412 of the machine memory 216 and the HAP table 330. As a result, the modified data in the write 410 may still reside in an entry in the temporary shadow page table 320, and the machine memory 216 may still only contain the original deduplicated data. When the VMM 212 attempts to use the HAP table 330 to look up and retrieve the modified data in the write 410, the attempt may trigger a fault 432, because the

modified data in the write 410 has not yet been written to the machine memory 216. In response to the fault 432, the VMM 212 may then look up and retrieve the data in the write 410 from its entry in the temporary shadow page table 320, then send the data in the write 410 to the requesting VM in a return 434. By sending the data from the temporary shadow page table 320, the VMM 212 may not have to create and modify a copy of the original deduplicated data in response to the read 430 and incur the write access delay associated with the copy-on-write process. As a result, a requesting VM may not observe a write access time delay for the data that was just written, and any access time delays due to data deduplication may be masked.

[0030] FIG. 5 illustrates a general purpose computing device, which may be used to provide memory deduplication masking in a datacenter, arranged in accordance with at least some embodiments described herein.

[0031] For example, the computing device 500 may be used to mask memory deduplication in a datacenter as described herein. In an example basic configuration 502, the computing device 500 may include one or more processors 504 and a system memory 506. A memory bus 508 may be used to communicate between the processor 504 and the system memory 506. The basic configuration 502 is illustrated in FIG. 5 by those components within the inner dashed line.

[0032] Depending on the desired configuration, the processor 504 may be of any type, including but not limited to a microprocessor (μP), a microcontroller (μC), a digital signal processor (DSP), or any combination thereof. The processor 504 may include one more levels of caching, such as a level cache memory 512, a processor core 514, and registers 516. The example processor core 514 may include an arithmetic logic unit (ALU), a floating point unit (FPU), a digital signal processing core (DSP Core), or any combination thereof. An example memory controller 518 may also be used with the processor 504, or in some implementations the memory controller 518 may be an internal part of the processor 504.

[0033] Depending on the desired configuration, the system memory 506 may be of any type including but not limited to volatile memory (such as RAM), non-volatile memory (such as ROM, flash memory, etc.) or any combination thereof. The system memory 506 may include an operating system 520, a virtual machine manager 522, and program data 524. The program data 524 may include, among other data, a shadow paging table 526, a hardware-assisted paging table 528, deduplicated data 530, or the like, as described herein.

[0034] The computing device 500 may have additional features or functionality, and additional interfaces to facilitate communications between the basic configuration 502 and any desired devices and interfaces. For example, a bus/interface controller 530 may be used to facilitate communications between the basic configuration 502 and one or more data storage devices 532 via a storage interface bus 534. The data storage devices 532 may be one or more removable storage devices 536, one or more non-removable storage devices 538, or a combination thereof. Examples of the removable storage and the non-removable storage devices include magnetic disk devices such as flexible disk drives and hard-disk drives (HDD), optical disk drives such as compact disk (CD) drives or digital versatile disk (DVD) drives, solid state drives (SSD), and tape drives to name a few. Example computer storage media may include volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information, such as computer readable instructions, data structures, program modules, or other data.

[0035] The system memory 506, the removable storage devices 536 and the non-removable storage devices 538 are examples of computer storage media. Computer storage media includes, but is not limited to, RAM, ROM, EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD), solid state drives, or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which may be used to store the desired information and which may be accessed by the computing device 500. Any such computer storage media may be part of the computing device 500.

[0036] The computing device 500 may also include an interface bus 540 for facilitating communication from various interface devices (e.g., one or more output devices 542, one or more peripheral interfaces 544, and one or more communication devices 566) to the basic configuration 502 via the bus/interface controller 530. Some of the example output devices 542 include a graphics processing unit 548 and an audio processing unit 550, which may be configured to communicate to various external devices such as a display or speakers via one or more A/V ports 552. One or more example peripheral interfaces 544 may include a serial interface controller 554 or a parallel interface controller 556, which may be configured to communicate with external devices such as input devices (e.g., keyboard, mouse, pen, voice input device, touch input device, etc.) or other peripheral devices (e.g., printer, scanner, etc.) via

one or more I/O ports 558. An example communication device 566 includes a network controller 560, which may be arranged to facilitate communications with one or more other computing devices 562 over a network communication link via one or more communication ports 564. The one or more other computing devices 562 may include servers at a datacenter, customer equipment, and comparable devices.

[0037] The network communication link may be one example of a communication media. Communication media may be embodied by computer readable instructions, data structures, program modules, or other data in a modulated data signal, such as a carrier wave or other transport mechanism, and may include any information delivery media. A "modulated data signal" may be a signal that has one or more of its characteristics set or changed in such a manner as to encode information in the signal. By way of example, and not limitation, communication media may include wired media such as a wired network or direct-wired connection, and wireless media such as acoustic, radio frequency (RF), microwave, infrared (IR) and other wireless media. The term computer readable media as used herein may include both storage media and communication media.

[0038] The computing device 500 may be implemented as a part of a general purpose or specialized server, mainframe, or similar computer that includes any of the above functions. The computing device 500 may also be implemented as a personal computer including both laptop computer and non-laptop computer configurations.

[0039] FIG. 6 is a flow diagram illustrating an example method for masking memory deduplication at a datacenter that may be performed by a computing device such as the computing device in FIG. 5, arranged in accordance with at least some embodiments described herein.

[0040] Example methods may include one or more operations, functions or actions as illustrated by one or more of blocks 622, 624, 626, and/or 628, and may in some embodiments be performed by a computing device such as the computing device 500 in FIG. 5. The operations described in the blocks 622-628 may also be stored as computer-executable instructions in a computer-readable medium such as a computer-readable medium 620 of a computing device 610.

[0041] An example process for masking memory deduplication at a datacenter may begin with block 622, "WRITE A FIRST DATA TO AN ENTRY IN A SHADOW PAGING TABLE

IN RESPONSE TO RECEIVING A WRITE REQUEST TO WRITE THE FIRST DATA TO A MEMORY", where a VMM (e.g., the VMM 212) may write a received first data to an entry in a shadow paging table before writing it to a memory, as described above. In some embodiments, the first data may be a modified version of data already stored in the memory.

[0042] Block 622 may be followed by block 624, "RECEIVE A READ REQUEST FOR THE FIRST DATA FROM A DATACENTER CUSTOMER", where the VMM may receive a read request for the first data from a datacenter customer (e.g., a VM at the datacenter). The VMM may receive the read request before the VMM has had a chance to write the first data from the shadow paging table to the memory, as described above.

[0043] Block 624 may be followed by block 626, "DETERMINE, BASED ON A HARDWARE-ASSISTED PAGING (HAP) TABLE, THAT THE FIRST DATA IS NOT AVAILABLE FROM THE MEMORY", where the VMM may use a HAP table to determine whether the first data is available from the memory, as described above. For example, the VMM may not have had the opportunity to write the first data from the shadow paging table to the memory. As a result, the HAP table may not include a reference to the first data, and the VMM may determine based on the HAP table that that first data is not available from the memory.

[0044] Finally, block 626 may be followed by block 628, "RESPOND TO THE READ REQUEST BY SENDING THE FIRST DATA FROM THE ENTRY IN THE SHADOW PAGING TABLE TO THE DATACENTER CUSTOMER IN RESPONSE TO DETERMINATION THAT THE FIRST DATA IS NOT AVAILABLE FROM THE MEMORY", where the VMM, upon the determination that the first data is not available from the memory performed in block 626, instead sends the first data from the entry in the shadow paging table, as described above.

[0045] FIG. 7 illustrates a block diagram of an example computer program product, arranged in accordance with at least some embodiments described herein.

[0046] In some examples, as shown in FIG. 7, a computer program product 700 may include a signal bearing medium 702 that may also include one or more machine readable instructions 704 that, when executed by, for example, a processor may provide the functionality described herein. Thus, for example, referring to the processor 504 in FIG. 5, the virtual machine manager 522 may undertake one or more of the tasks shown in FIG. 7 in response to the instructions 704 conveyed to the processor 504 by the medium 702 to perform actions associated

with masking memory deduplication as described herein. Some of those instructions may include, for example, to write a first data to an entry in a shadow paging table in response to receiving a write request to write the first data to a memory, to receive a read request for the first data from a datacenter customer, to determine, based on a hardware-assisted paging (HAP) table, that the first data is not available from the memory, and/or to respond to the read request by sending the first data from the entry in the shadow paging table to the datacenter customer in response to determination that the first data is not available from the memory, according to some embodiments described herein.

[0047] In some implementations, the signal bearing media 702 depicted in FIG. 7 may encompass computer-readable media 706, such as, but not limited to, a hard disk drive, a solid state drive, a Compact Disc (CD), a Digital Versatile Disk (DVD), a digital tape, memory, etc. In some implementations, the signal bearing media 702 may encompass recordable media 707, such as, but not limited to, memory, read/write (R/W) CDs, R/W DVDs, etc. In some implementations, the signal bearing media 702 may encompass communications media 710, such as, but not limited to, a digital and/or an analog communication medium (e.g., a fiber optic cable, a waveguide, a wired communications link, a wireless communication link, etc.). Thus, for example, the program product 700 may be conveyed to one or more modules of the processor 504 by an RF signal bearing medium, where the signal bearing media 702 is conveyed by the wireless communications media 710 (e.g., a wireless communications medium conforming with the IEEE 802.11 standard).

[0048] According to some examples, method may be provided to mask memory deduplication in a datacenter. An example method may include writing a first data to an entry in a shadow paging table in response to receiving a write request to write the first data and receiving a read request for the first data from a datacenter customer. The method may further include determining that the first data is not available from a memory based on a hardware-assisted paging (HAP) table and sending the first data from the entry in the shadow paging table to the datacenter customer in response to determination that the first data is not available from the memory.

[0049] According to some embodiments, the HAP table may be configured to map to the memory and the memory may be configured to store deduplicated data. The method may further include opportunistically updating the HAP table and the memory to include the first data and/or

deleting the entry in the shadow table after the updating. The method may further include receiving another read request for a second data from another datacenter customer, determining that the second data is available from the memory based on the HAP table, and responding to the other read request by sending the second data from the memory to the other datacenter customer in response to determination that the second data is available from the memory. The method may further include using the HAP table to retrieve the second data from the memory before sending the second data.

[0050] According to other embodiments, the first data may be a modified version of deduplicated data available from the memory. Receiving the write request and/or receiving the read request may include receiving the write request or the read request from a virtual machine associated with the datacenter customer. Receiving the read request may include receiving the read request following receipt of the write request. The first data may be data associated with an operating system and/or data associated with an application.

[0051] According to other examples, a virtual machine manager (VMM) may be provided to mask memory deduplication in a datacenter. The VMM may include a processing module and a memory configured to store a shadow paging table and a hardware-assisted paging (HAP) table and coupled to the processing module. The processing module may be configured to receive a write request to write a first data, write the first data to an entry in the shadow paging table, receive a read request for the first data from a datacenter customer, and determine, based on the HAP table, that the first data is not available from a memory. The processing module may be further configured to respond to the read request by sending the first data from the entry in the shadow paging table to the datacenter customer in response to determination that the first data is not available from the memory.

[0052] According to some embodiments, the HAP table may be configured to map to the memory and the memory may be configured to store deduplicated data. The processing module may be further configured to opportunistically update the HAP table and the memory to include the first data and/or delete the entry in the shadow table after the updating.

[0053] According to further examples, a system may be provided to mask memory deduplication. The system may include at least one virtual machine (VM) operable to be executed on one or more physical machines, a memory configured to store deduplicated data, a memory configured to store a shadow paging table and a hardware-assisted paging (HAP) table,

and a controller. The controller may be configured to receive a write request to write a first data to the memory, write the first data to an entry in the shadow paging table, receive a read request for the first data from a customer, and determine, based on the HAP table, that the first data is not available from the memory. The controller may further be configured to respond to the read request by sending the first data from the entry in the shadow paging table to the customer in response to determination that the first data is not available from the memory.

[0054] According to some embodiments, the HAP table may be configured to map to the memory. The controller may be further configured to opportunistically update the HAP table and the memory to include the first data and/or delete the entry in the shadow table after the updating. The controller may be further configured to receive another read request for a second data from another customer, determine that the second data is available from the memory based on the HAP table, and respond to the other read request by sending the second data from the memory to the other customer in response to determination that the second data is available from the memory. The controller may be further configured to use the HAP table to retrieve the second data from the memory before sending the second data.

[0055] According to other embodiments, the first data may be a modified version of deduplicated data available from the memory. Receiving the write request and/or receiving the read request may include receiving the write request or the read request from a virtual machine associated with the customer. Receiving the read request may include receiving the read request following receipt of the write request. The first data may be data associated with an operating system and/or data associated with an application.

[0056] According to yet further examples, a computer readable medium may store instructions for masking memory deduplication in a datacenter. The instructions may include writing a first data to an entry in a shadow paging table in response to receiving a write request to write the first data and receiving a read request for the first data from a datacenter customer. The instructions may further include determining that the first data is not available from a memory based on a hardware-assisted paging (HAP) table and sending the first data from the entry in the shadow paging table to the datacenter customer in response to determination that the first data is not available from the memory.

[0057] According to some embodiments, the HAP table may be configured to map to the memory and the memory may be configured to store deduplicated data. The instructions may

further include opportunistically updating the HAP table and the memory to include the first data and/or deleting the entry in the shadow table after the updating. The instructions may further include receiving another read request for a second data from another datacenter customer, determining that the second data is available from the memory based on the HAP table, and responding to the other read request by sending the second data from the memory to the other datacenter customer in response to determination that the second data is available from the memory. The instructions may further include using the HAP table to retrieve the second data from the memory before sending the second data.

[0058] According to other embodiments, the first data may be a modified version of deduplicated data available from the memory. Receiving the write request and/or receiving the read request may include receiving the write request or the read request from a virtual machine associated with the datacenter customer. Receiving the read request may include receiving the read request following receipt of the write request. The first data may be data associated with an operating system and/or data associated with an application.

[0059] There is little distinction left between hardware and software implementations of aspects of systems; the use of hardware or software is generally (but not always, in that in certain contexts the choice between hardware and software may become significant) a design choice representing cost vs. efficiency tradeoffs. There are various vehicles by which processes and/or systems and/or other technologies described herein may be effected (e.g., hardware, software, and/or firmware), and that the preferred vehicle will vary with the context in which the processes and/or systems and/or other technologies are deployed. For example, if an implementer determines that speed and accuracy are paramount, the implementer may opt for a mainly hardware and/or firmware vehicle; if flexibility is paramount, the implementer may opt for a mainly software implementation; or, yet again alternatively, the implementer may opt for some combination of hardware, software, and/or firmware.

[0060] The foregoing detailed description has set forth various embodiments of the devices and/or processes via the use of block diagrams, flowcharts, and/or examples. Insofar as such block diagrams, flowcharts, and/or examples contain one or more functions and/or operations, it will be understood by those within the art that each function and/or operation within such block diagrams, flowcharts, or examples may be implemented, individually and/or collectively, by a wide range of hardware, software, firmware, or virtually any combination thereof. In one

embodiment, several portions of the subject matter described herein may be implemented via Application Specific Integrated Circuits (ASICs), Field Programmable Gate Arrays (FPGAs), digital signal processors (DSPs), or other integrated formats. However, those skilled in the art will recognize that some aspects of the embodiments disclosed herein, in whole or in part, may be equivalently implemented in integrated circuits, as one or more computer programs executing on one or more computers (e.g., as one or more programs executing on one or more computer systems), as one or more programs executing on one or more processors (e.g., as one or more programs executing on one or more microprocessors), as firmware, or as virtually any combination thereof, and that designing the circuitry and/or writing the code for the software and or firmware would be well within the skill of one of skill in the art in light of this disclosure.

[0061] The present disclosure is not to be limited in terms of the particular embodiments described in this application, which are intended as illustrations of various aspects. Many modifications and variations can be made without departing from its spirit and scope, as will be apparent to those skilled in the art. Functionally equivalent methods and apparatuses within the scope of the disclosure, in addition to those enumerated herein, will be apparent to those skilled in the art from the foregoing descriptions. Such modifications and variations are intended to fall within the scope of the appended claims. The present disclosure is to be limited only by the terms of the appended claims, along with the full scope of equivalents to which such claims are entitled. It is also to be understood that the terminology used herein is for the purpose of describing particular embodiments only, and is not intended to be limiting.

[0062] In addition, those skilled in the art will appreciate that the mechanisms of the subject matter described herein are capable of being distributed as a program product in a variety of forms, and that an illustrative embodiment of the subject matter described herein applies regardless of the particular type of signal bearing medium used to actually carry out the distribution. Examples of a signal bearing medium include, but are not limited to, the following: a recordable type medium such as a floppy disk, a hard disk drive, a Compact Disc (CD), a Digital Versatile Disk (DVD), a digital tape, a computer memory, a solid state drive, etc.; and a transmission type medium such as a digital and/or an analog communication medium (e.g., a fiber optic cable, a waveguide, a wired communications link, a wireless communication link, etc.).

[0063] Those skilled in the art will recognize that it is common within the art to describe devices and/or processes in the fashion set forth herein, and thereafter use engineering practices to integrate such described devices and/or processes into data processing systems. That is, at least a portion of the devices and/or processes described herein may be integrated into a data processing system via a reasonable amount of experimentation. Those having skill in the art will recognize that a data processing system may include one or more of a system unit housing, a video display device, a memory such as volatile and non-volatile memory, processors such as microprocessors and digital signal processors, computational entities such as operating systems, drivers, graphical user interfaces, and applications programs, one or more interaction devices, such as a touch pad or screen, and/or control systems including feedback loops and control motors.

[0064] A data processing system may be implemented utilizing any suitable commercially available components, such as those found in data computing/communication and/or network computing/communication systems. The herein described subject matter sometimes illustrates different components contained within, or connected with, different other components. It is to be understood that such depicted architectures are merely exemplary, and that in fact many other architectures may be implemented which achieve the same functionality. In a conceptual sense, any arrangement of components to achieve the same functionality is effectively "associated" such that the desired functionality is achieved. Hence, any two components herein combined to achieve a particular functionality may be seen as "associated with" each other such that the desired functionality is achieved, irrespective of architectures or intermediate components. Likewise, any two components so associated may also be viewed as being "operably connected", or "operably coupled", to each other to achieve the desired functionality, and any two components capable of being so associated may also be viewed as being "operably couplable", to each other to achieve the desired functionality. Specific examples of operably couplable include but are not limited to physically connectable and/or physically interacting components and/or wirelessly interactable and/or wirelessly interacting components and/or logically interacting and/or logically interactable components.

[0065] With respect to the use of substantially any plural and/or singular terms herein, those having skill in the art can translate from the plural to the singular and/or from the singular

to the plural as is appropriate to the context and/or application. The various singular/plural permutations may be expressly set forth herein for sake of clarity.

[0066] It will be understood by those within the art that, in general, terms used herein, and especially in the appended claims (*e.g.*, bodies of the appended claims) are generally intended as "open" terms (*e.g.*, the term "including" should be interpreted as "including but not limited to," the term "having" should be interpreted as "having at least," the term "includes" should be interpreted as "includes but is not limited to," etc.). It will be further understood by those within the art that if a specific number of an introduced claim recitation is intended, such an intent will be explicitly recited in the claim, and in the absence of such recitation no such intent is present. For example, as an aid to understanding, the following appended claims may contain usage of the introductory phrases "at least one" and "one or more" to introduce claim recitations. However, the use of such phrases should not be construed to imply that the introduction of a claim recitation by the indefinite articles "a" or "an" limits any particular claim containing such introduced claim recitation to embodiments containing only one such recitation, even when the same claim includes the introductory phrases "one or more" or "at least one" and indefinite articles such as "a" or "an" (*e.g.*, "a" and/or "an" should be interpreted to mean "at least one" or "one or more"); the same holds true for the use of definite articles used to introduce claim recitations. In addition, even if a specific number of an introduced claim recitation is explicitly recited, those skilled in the art will recognize that such recitation should be interpreted to mean at least the recited number (*e.g.*, the bare recitation of "two recitations," without other modifiers, means at least two recitations, or two or more recitations).

[0067] Furthermore, in those instances where a convention analogous to "at least one of A, B, and C, etc." is used, in general such a construction is intended in the sense one having skill in the art would understand the convention (*e.g.*, " a system having at least one of A, B, and C" would include but not be limited to systems that have A alone, B alone, C alone, A and B together, A and C together, B and C together, and/or A, B, and C together, etc.). It will be further understood by those within the art that virtually any disjunctive word and/or phrase presenting two or more alternative terms, whether in the description, claims, or drawings, should be understood to contemplate the possibilities of including one of the terms, either of the terms, or both terms. For example, the phrase "A or B" will be understood to include the possibilities of "A" or "B" or "A and B."

[0068] As will be understood by one skilled in the art, for any and all purposes, such as in terms of providing a written description, all ranges disclosed herein also encompass any and all possible subranges and combinations of subranges thereof. Any listed range can be easily recognized as sufficiently describing and enabling the same range being broken down into at least equal halves, thirds, quarters, fifths, tenths, etc. As a non-limiting example, each range discussed herein can be readily broken down into a lower third, middle third and upper third, etc. As will also be understood by one skilled in the art all language such as "up to," "at least," "greater than," "less than," and the like include the number recited and refer to ranges which can be subsequently broken down into subranges as discussed above. Finally, as will be understood by one skilled in the art, a range includes each individual member. Thus, for example, a group having 1-3 cells refers to groups having 1, 2, or 3 cells. Similarly, a group having 1-5 cells refers to groups having 1, 2, 3, 4, or 5 cells, and so forth.

[0069] While various aspects and embodiments have been disclosed herein, other aspects and embodiments will be apparent to those skilled in the art. The various aspects and embodiments disclosed herein are for purposes of illustration and are not intended to be limiting, with the true scope and spirit being indicated by the following claims.

CLAIMS

WHAT IS CLAIMED IS:

1. A method to mask memory deduplication in a datacenter, the method comprising:

in response to receiving a write request to write a first data to a memory, writing the first data to an entry in a shadow paging table;

receiving a read request for the first data from a datacenter customer;

determining, based on a hardware-assisted paging (HAP) table, that the first data is not available from the memory; and

in response to determination that the first data is not available from the memory, responding to the read request by sending the first data from the entry in the shadow paging table to the datacenter customer.

2. The method of claim 1, wherein:

the HAP table is configured to map to the memory; and

the memory is configured to store deduplicated data.

3. The method of claim 1, further comprising opportunistically updating the HAP table and the memory to include the first data.

4. The method of claim 3, further comprising deleting the entry in the shadow table after the updating.

5. The method of claim 1, further comprising:

receiving another read request for a second data from another datacenter customer;

determining, based on the HAP table, that the second data is available from the memory; and

in response to determination that the second data is available from the memory, responding to the other read request by sending the second data from the memory to the other datacenter customer.

6. The method of claim 5, further comprising using the HAP table to retrieve the second data from the memory before sending the second data.

7. The method of claim 1, wherein the first data is a modified version of deduplicated data available from the memory.

8. The method of claim 1, wherein at least one of receiving the write request and receiving the read request comprises receiving the write request or the read request from a virtual machine associated with the datacenter customer.

9. The method of claim 1, wherein receiving the read request comprises receiving the read request following receipt of the write request.

10. The method of claim 1, wherein the first data is one of data associated with an operating system and data associated with an application.

11. A virtual machine manager (VMM) configured to mask memory deduplication in a datacenter, the VMM comprising:

a memory configured to store a shadow paging table and a hardware-assisted paging (HAP) table; and

a processing module coupled to the memory and configured to:

receive a write request to write a first data to a memory;

write the first data to an entry in the shadow paging table;

receive a read request for the first data from a datacenter customer;

determine, based on the HAP table, that the first data is not available from the memory; and

in response to determination that the first data is not available from the memory, respond to the read request by sending the first data from the entry in the shadow paging table to the datacenter customer.

12. The VMM of claim 11, wherein:

the HAP table is configured to map to the memory; and

the memory is configured to store deduplicated data.

13. The VMM of claim 12, wherein the processing module is further configured to opportunistically update the HAP table and the memory to include the first data.

14. The VMM of claim 13, wherein the processing module is further configured to delete the entry in the shadow table after the updating.

15. A system configured to mask memory deduplication, the system comprising:

at least one virtual machine (VM) operable to be executed on one or more physical machines;

a memory configured to store deduplicated data;

a memory configured to store a shadow paging table and a hardware-assisted paging (HAP) table; and

a controller configured to:

receive a write request to write a first data to the memory;

write the first data to an entry in the shadow paging table;

receive a read request for the first data from a customer;

determine, based on the HAP table, that the first data is not available from the memory; and

in response to determination that the first data is not available from the memory, respond to the read request by sending the first data from the entry in the shadow paging table to the customer.

16. The system of claim 15, wherein the HAP table maps to the memory.

17. The system of claim 16, wherein the controller is further configured to opportunistically update the HAP table and the memory to include the first data.

18. The system of claim 17, wherein the controller is further configured to delete the entry in the shadow table after the updating.

19. The system of claim 15, wherein the controller is further configured to:

receive another read request for a second data from another customer;

determine, based on the HAP table, that the second data is available from the memory; and

in response to determination that the second data is available from the memory, respond to the other read request by sending the second data from the memory to the other customer.

20. The system of claim 19, wherein the controller is further configured to use the HAP table to retrieve the second data from the memory before sending the second data.

21. The system of claim 15, wherein the first data is a modified version of a deduplicated data available from the memory.

22. The system of claim 15, wherein at least one of the write request and the read request are received from the at least one virtual machine associated with the customer.

23. The system of claim 15, wherein the read request is received shortly after the write request is received.

24. The system of claim 15, wherein the first data is one of a data associated with an operating system and a data associated with an application.

25. A computer readable storage medium with instructions stored thereon, which when executed on one or more computing devices perform a method to mask memory deduplication, wherein the method includes action of claims 1 through 10.

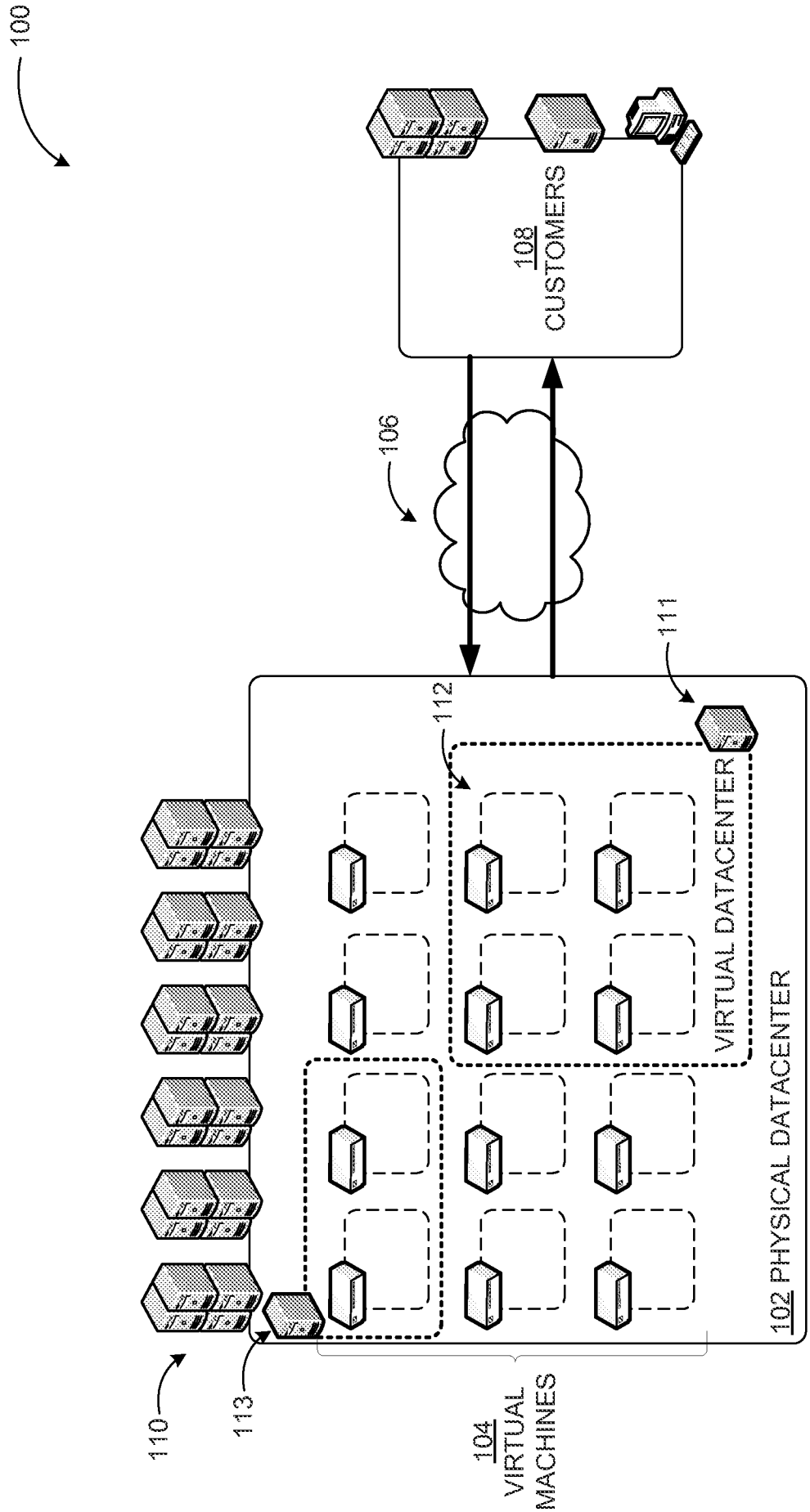

100
108 CUSTOMERS
106
111
112
VIRTUAL DATACENTER
102 PHYSICAL DATACENTER
110
113
104 VIRTUAL MACHINES

FIG. 1

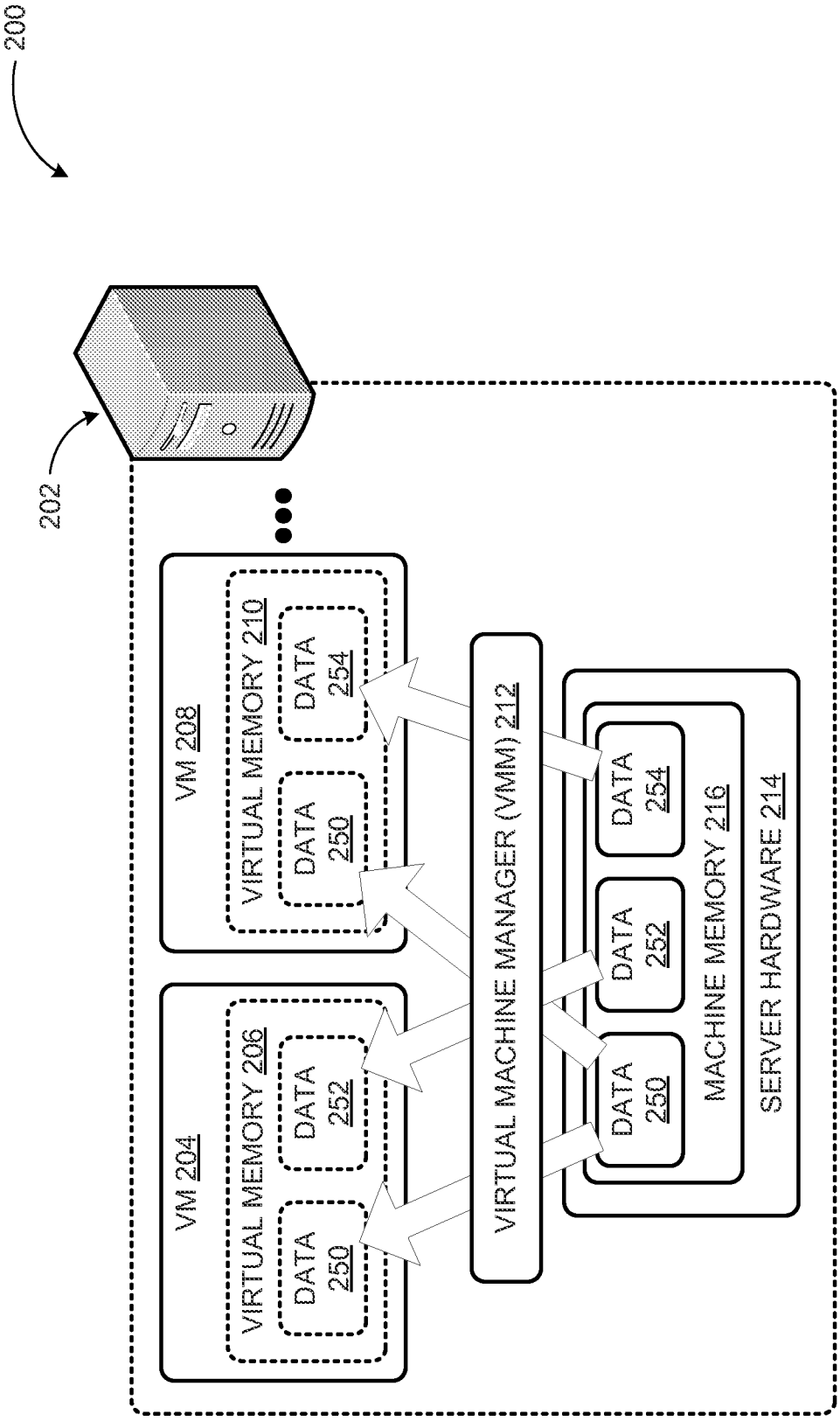

200
202
VM 204
VIRTUAL MEMORY 206
DATA 250
DATA 252
VM 208
VIRTUAL MEMORY 210
DATA 250
DATA 254
VIRTUAL MACHINE MANAGER (VMM) 212
SERVER HARDWARE 214
MACHINE MEMORY 216
DATA 250
DATA 252
DATA 254

FIG. 2

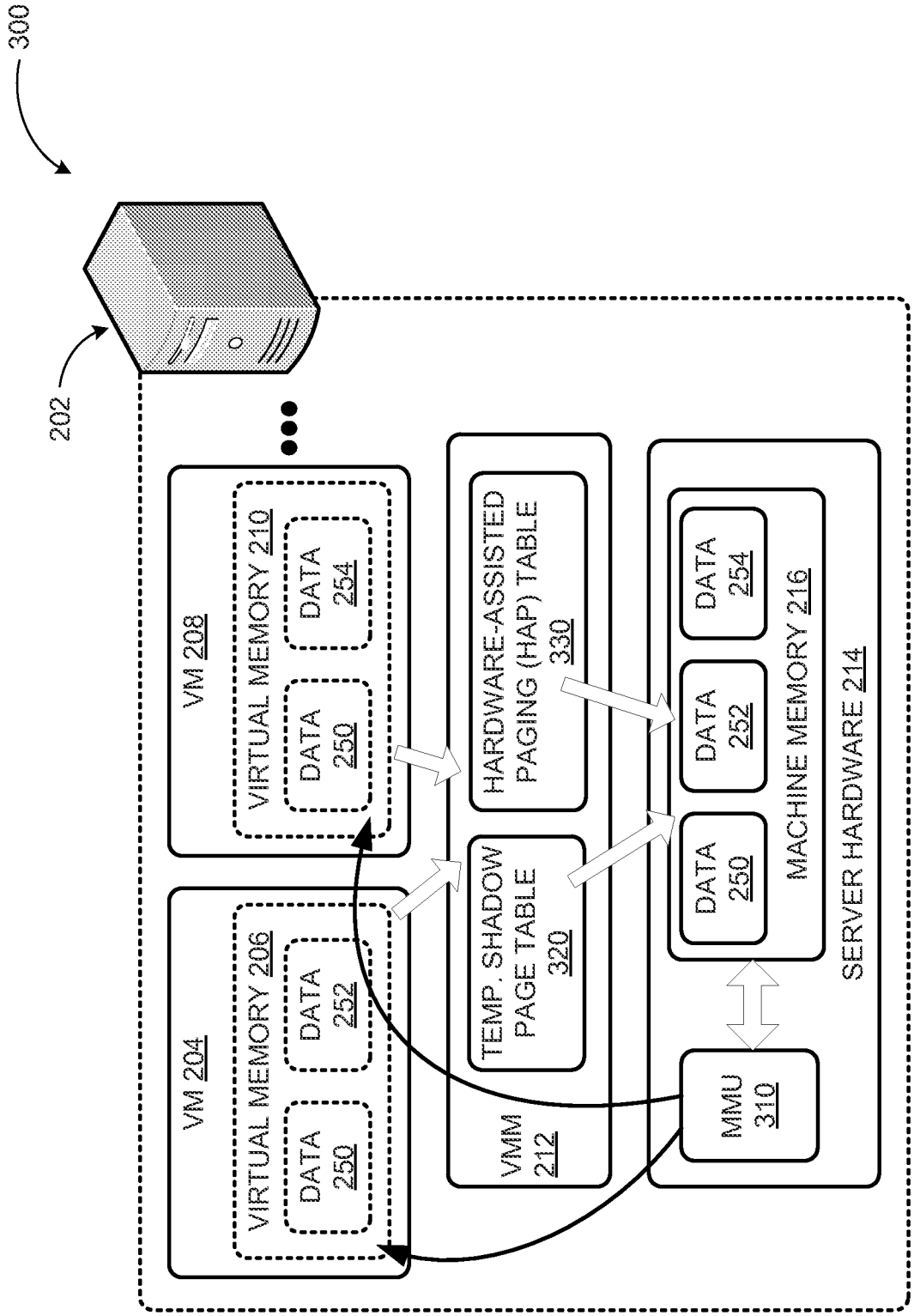

300
202
SERVER HARDWARE 214
VM 204
VIRTUAL MEMORY 206
DATA 250
DATA 252
VM 208
VIRTUAL MEMORY 210
DATA 250
DATA 254
VMM 212
TEMP. SHADOW PAGE TABLE 320
HARDWARE-ASSISTED PAGING (HAP) TABLE 330
MMU 310
MACHINE MEMORY 216
DATA 250
DATA 252
DATA 254

FIG. 3

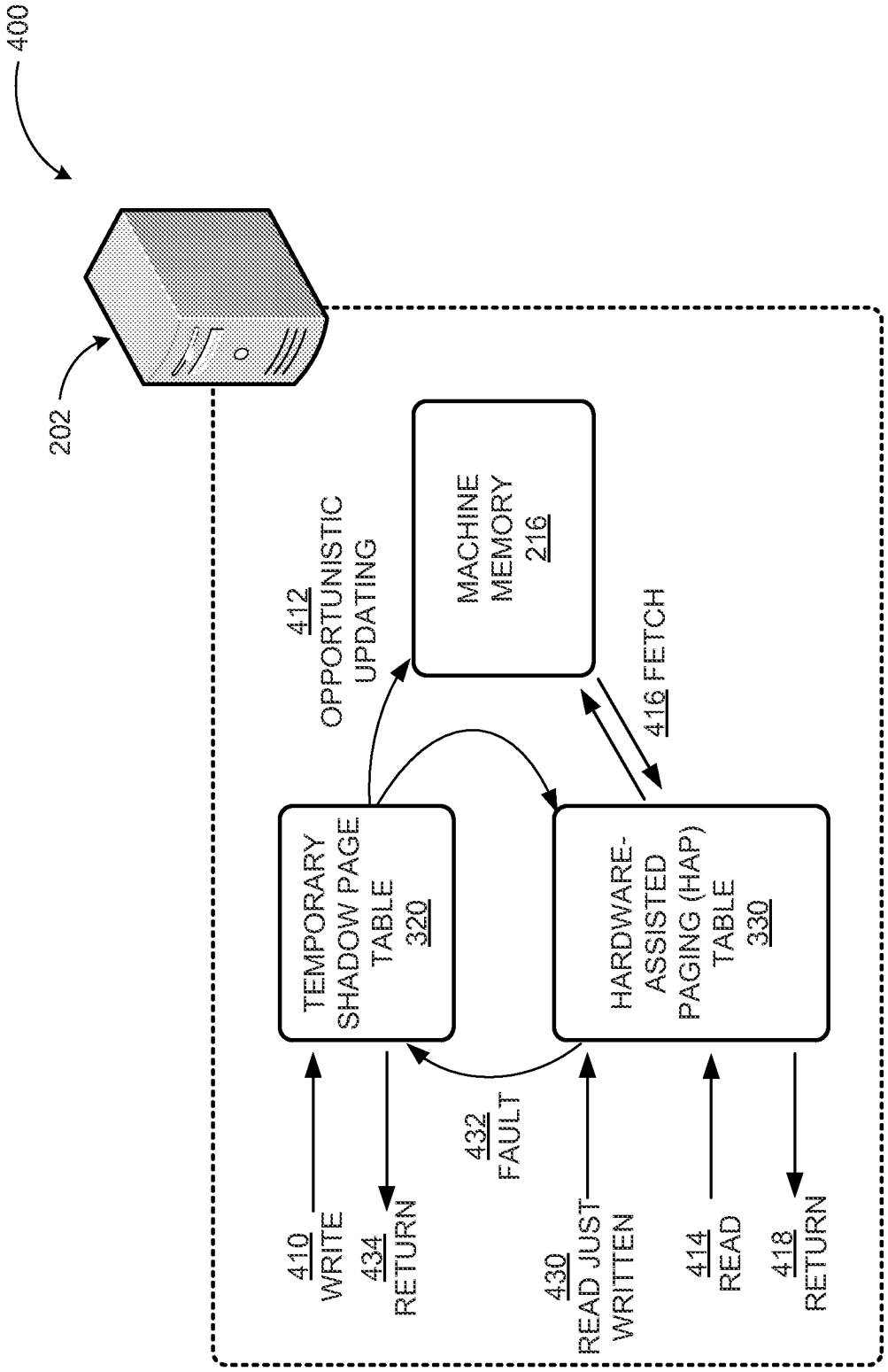

400
202
412
OPPORTUNISTIC
UPDATING
MACHINE
MEMORY
216
416 FETCH
TEMPORARY
SHADOW PAGE
TABLE
320
HARDWARE-
ASSISTED
PAGING (HAP)
TABLE
330
410
WRITE
434
RETURN
432
FAULT
430
READ JUST
WRITTEN
414
READ
418
RETURN

FIG. 4

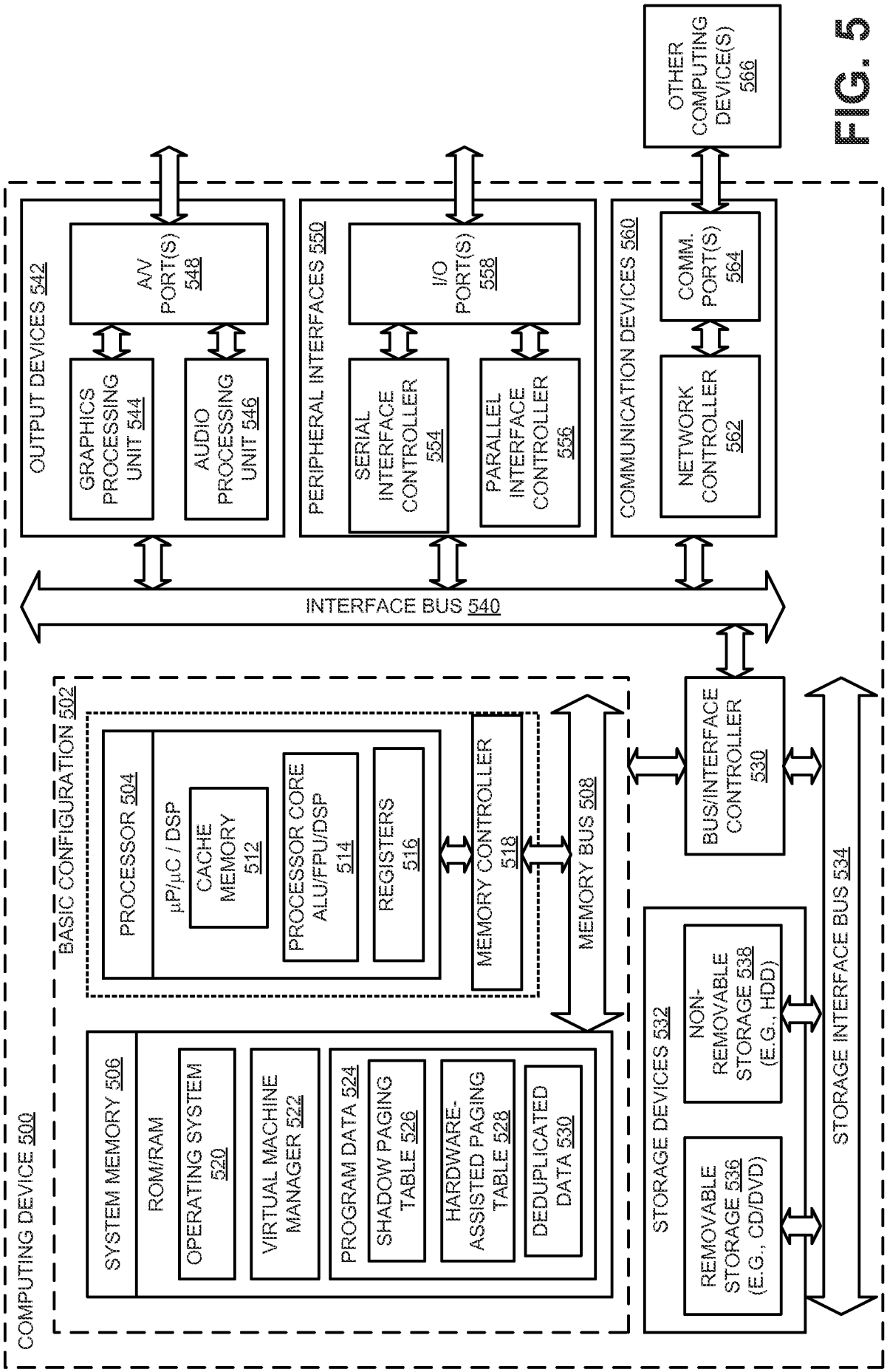

COMPUTING DEVICE 500
BASIC CONFIGURATION 502
PROCESSOR 504
µP/µC / DSP
CACHE MEMORY 512
PROCESSOR CORE ALU/FPU/DSP 514
REGISTERS 516
MEMORY CONTROLLER 518
MEMORY BUS 508
SYSTEM MEMORY 506
ROM/RAM
OPERATING SYSTEM 520
VIRTUAL MACHINE MANAGER 522
PROGRAM DATA 524
SHADOW PAGING TABLE 526
HARDWARE-ASSISTED PAGING TABLE 528
DEDUPLICATED DATA 530
BUS/INTERFACE CONTROLLER 530
STORAGE DEVICES 532
REMOVABLE STORAGE 536 (E.G., CD/DVD)
NON-REMOVABLE STORAGE 538 (E.G., HDD)
STORAGE INTERFACE BUS 534
INTERFACE BUS 540
OUTPUT DEVICES 542
GRAPHICS PROCESSING UNIT 544
AUDIO PROCESSING UNIT 546
A/V PORT(S) 548
PERIPHERAL INTERFACES 550
SERIAL INTERFACE CONTROLLER 554
PARALLEL INTERFACE CONTROLLER 556
I/O PORT(S) 558
COMMUNICATION DEVICES 560
NETWORK CONTROLLER 562
COMM. PORT(S) 564
OTHER COMPUTING DEVICE(S) 566

FIG. 5

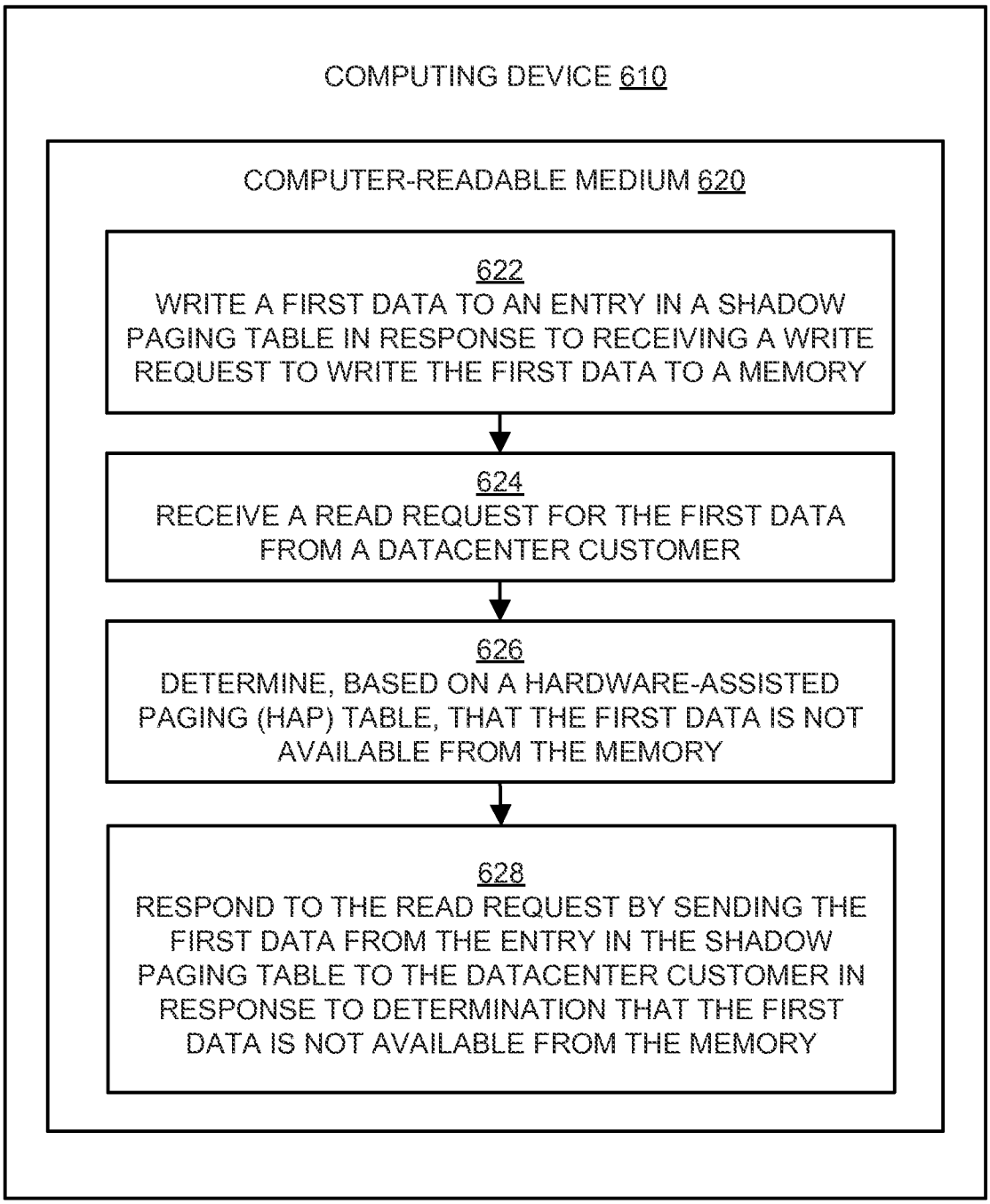

COMPUTING DEVICE 610
COMPUTER-READABLE MEDIUM 620
622
WRITE A FIRST DATA TO AN ENTRY IN A SHADOW PAGING TABLE IN RESPONSE TO RECEIVING A WRITE REQUEST TO WRITE THE FIRST DATA TO A MEMORY
624
RECEIVE A READ REQUEST FOR THE FIRST DATA FROM A DATACENTER CUSTOMER
626
DETERMINE, BASED ON A HARDWARE-ASSISTED PAGING (HAP) TABLE, THAT THE FIRST DATA IS NOT AVAILABLE FROM THE MEMORY
628
RESPOND TO THE READ REQUEST BY SENDING THE FIRST DATA FROM THE ENTRY IN THE SHADOW PAGING TABLE TO THE DATACENTER CUSTOMER IN RESPONSE TO DETERMINATION THAT THE FIRST DATA IS NOT AVAILABLE FROM THE MEMORY

FIG. 6

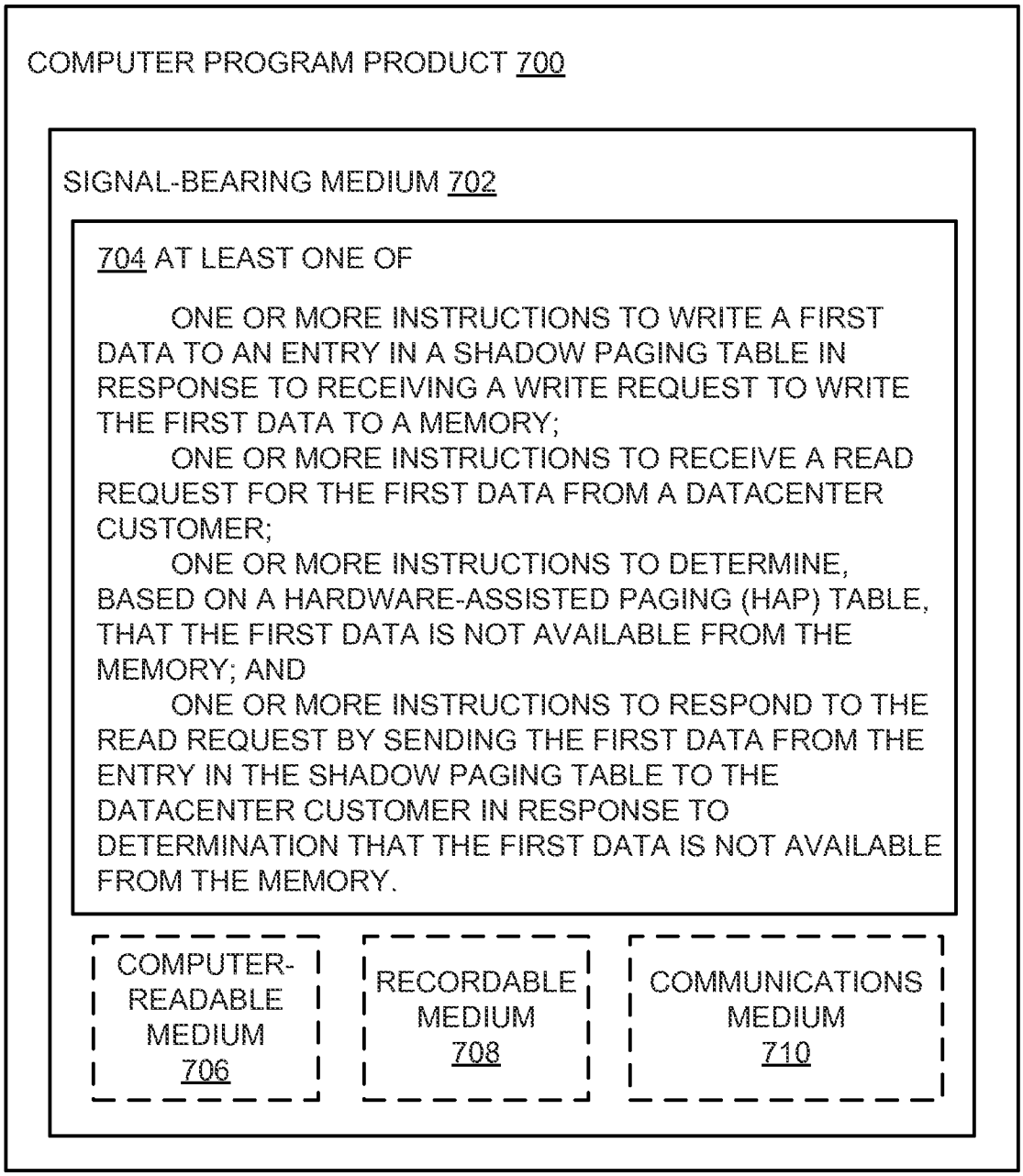

COMPUTER PROGRAM PRODUCT 700
SIGNAL-BEARING MEDIUM 702
704 AT LEAST ONE OF
ONE OR MORE INSTRUCTIONS TO WRITE A FIRST DATA TO AN ENTRY IN A SHADOW PAGING TABLE IN RESPONSE TO RECEIVING A WRITE REQUEST TO WRITE THE FIRST DATA TO A MEMORY;
ONE OR MORE INSTRUCTIONS TO RECEIVE A READ REQUEST FOR THE FIRST DATA FROM A DATACENTER CUSTOMER;
ONE OR MORE INSTRUCTIONS TO DETERMINE, BASED ON A HARDWARE-ASSISTED PAGING (HAP) TABLE, THAT THE FIRST DATA IS NOT AVAILABLE FROM THE MEMORY; AND
ONE OR MORE INSTRUCTIONS TO RESPOND TO THE READ REQUEST BY SENDING THE FIRST DATA FROM THE ENTRY IN THE SHADOW PAGING TABLE TO THE DATACENTER CUSTOMER IN RESPONSE TO DETERMINATION THAT THE FIRST DATA IS NOT AVAILABLE FROM THE MEMORY.
COMPUTER-READABLE MEDIUM 706
RECORDABLE MEDIUM 708
COMMUNICATIONS MEDIUM 710

FIG. 7

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US13/69238

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - G06F 12/00, 13/14 (2014.01)
USPC - 711/206; 707/736
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)
IPC(8): G06F 3/00, 3/01, 7/00, 12/00, 13/14, 17/30 (2014.01)
USPC: 707/615, 657, 736, 757, 781; 710/20, 49; 711/6, 100, 113, 126, 147, 150, 206, 221

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

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
MicroPatent (US-G, US-A, EP-A, EP-B, WO, JP-bib, DE-C,B, DE-A, DE-T, DE-U, GB-A, FR-A); ProQuest; IEEE/IEEEXplore; Google/Google Scholar; KEYWORDS: data, memory, paging, shadow, table, hardware, unavailable

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2013/0283004 A1 (DEVINE, S et al.) 24 October 2013; abstract; figures 3, 5, 8; paragraphs [0031]-[0033], [0040], [0044], [0047], [0049], [0050], [0056], [0064], [0066], [0076], [0078], [0079], [0081].	1-24, 25/1-25/10
A	US 2013/0132690 A1 (EPSTEIN, J) 23 May 2013; entire document.	1-24, 25/1-25/10
A	US 2007/0067435 A1 (LANDIS, J et al.) 22 March 2007; entire document.	1-24, 25/1-25/10

☐ 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 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	Date of mailing of the international search report
17 April 2014 (17.04.2014)	02 May 2014

Name and mailing address of the ISA/US	Authorized officer:
Mail Stop PCT, Attn: ISA/US, Commissioner for Patents P.O. Box 1450, Alexandria, Virginia 22313-1450 Facsimile No. 571-273-3201	Shane Thomas PCT Helpdesk: 571-272-4300 PCT OSP: 571-272-7774