DATA PROCESSING METHOD, COMPUTER SYSTEM, STORAGE MEDIUM, AND PROGRAM PRODUCT

The present disclosure claims priority to Chinese Patent Application No. 202410375097.8, filed with the China National Intellectual Property Administration on March 28, 2024 and entitled "DATA PROCESSING METHOD, COMPUTER SYSTEM, STORAGE MEDIUM, AND PROGRAM PRODUCT," which is incorporated herein by reference in its entirety.

TECHNICAL FIELD

Embodiments of the present disclosure relate to the field of computer technology, and in particular, to a data processing method, a computer system, a storage medium, and a program product.

BACKGROUND

In a computer system, multiple service programs running in a user mode often have data access requirements with one another, for example, an application service transfers data to a storage service to implement data storage, etc. However, data access between service programs often requires participation of the kernel. Still taking the example of an application service transferring data to a storage service, the application service needs to copy data to a kernel service via a system call, and then the kernel service copies the data to the storage service. This results in a large number of data copy operations during the data access process, thereby affecting data processing efficiency.

SUMMARY

Embodiments of the present disclosure provide a data processing method, a computer system, a storage medium, and a program product, to solve the technical problem of slow data processing efficiency in the prior art.

In a first aspect, an embodiment of the present disclosure provides a data processing method applied to a kernel program running in a kernel mode, where a shared memory region is constructed between the kernel program and a first service program running in a user mode, and the shared memory region includes a service data storage region and a control command storage region, where the control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program; the

method includes:

obtaining a first control command transferred by a second service program, and determining a physical address region;

allocating a virtual address region from a second virtual address space to which the service data storage region is mapped, and generating a second control command based on the virtual address region;

storing the second control command in the control command storage region;

establishing a mapping relationship between the virtual address region and the physical address region, to enable the first service program to access the physical address region based on the virtual address region in the second control command obtained from the control command storage region.

In a second aspect, an embodiment of the present disclosure provides a data processing method applied to a first service program running in a user mode, where a shared memory region is constructed between the first service program and a kernel program, and the shared memory region includes a service data storage region and a control command storage region, where the control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program, and the service data storage region is mapped to a second virtual address space of the first service program; the method includes:

accessing the control command storage region based on the first virtual address space to obtain a second control command, where the second control command is generated by the kernel program obtaining a first control command transferred by a second service program, determining a physical address region, allocating a virtual address region from the second virtual address space to which the service data storage region is mapped, and performing generation based on the virtual address region;

determining the virtual address region in the second control command;

executing an access operation based on the virtual address region to access the physical address region based on a mapping relationship.

In a third aspect, an embodiment of the present disclosure provides a data processing method applied to a second service program running in a user mode, where virtual memory corresponding to the second service program includes a command buffer region and a data buffer region; the method includes:

determining the data buffer region;

sending, based on the data buffer region, a first control command for a first service program to a kernel program, where a shared memory region is constructed between the first service

program and the kernel program, and the shared memory region includes a service data storage region and a control command storage region, where the control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program; the first control command is used to determine a physical address region corresponding to the data buffer region, and the physical address region is used to establish a mapping relationship with a virtual address region allocated from a second virtual address space to which the service data storage region is mapped, where the second virtual address space is used to generate a second control command which is then stored in the command buffer region, for the first service program to obtain and initiate an access operation based on the second virtual address space.

In a fourth aspect, an embodiment of the present disclosure provides a data processing apparatus applied to a kernel program running in a kernel mode, where a shared memory region is constructed between the kernel program and a first service program running in a user mode, and the shared memory region includes a service data storage region and a control command storage region, where the control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program, and the service data storage region is mapped to a second virtual address space of the first service program; the apparatus includes:

a first obtaining module, configured to obtain a first control command transferred by a second service program, and determine a physical address region;

an address allocation module, configured to allocate a virtual address region from the second virtual address space to which the service data storage region is mapped, and generate a second control command based on the virtual address region;

a command storage module, configured to store the second control command in the control command storage region;

a relationship establishment module, configured to establish a mapping relationship between the virtual address region and the physical address region, to enable the first service program to access the physical address region based on the virtual address region in the second control command obtained from the control command storage region.

In a fifth aspect, an embodiment of the present disclosure provides a data processing apparatus applied to a first service program running in a user mode, where a shared memory region is constructed between the first service program and a kernel program, and the shared memory region includes a service data storage region and a control command storage region, where the control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program; the apparatus includes:

a first access module, configured to access the control command storage region based on the first virtual address space to obtain a second control command, where the second control command is generated by the kernel program obtaining a first control command transferred by a second service program, determining a physical address region, allocating a virtual address region from a second virtual address space to which the service data storage region is mapped, and performing generation based on the virtual address region;

a first determination module, configured to determine the virtual address region in the second control command;

a second access module, configured to execute an access operation based on the virtual address region to access the physical address region based on a mapping relationship.

In a sixth aspect, an embodiment of the present disclosure provides a data processing apparatus applied to a second service program running in a user mode, where virtual memory corresponding to the second service program includes a command buffer region and a data buffer region; the apparatus includes:

a second determination module, configured to determine the data buffer region;

a command sending module, configured to send, based on the data buffer region, a first control command for a first service program to a kernel program, where a shared memory region is constructed between the first service program and the kernel program, and the shared memory region includes a service data storage region and a control command storage region, where the control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program; the first control command is used to determine a physical address region corresponding to the data buffer region, and the physical address region is used to establish a mapping relationship with a virtual address region allocated from a second virtual address space to which the service data storage region is mapped, where the second virtual address space is used to generate a second control command which is then stored in the command buffer region, for the first service program to obtain and initiate an access operation based on the second virtual address space.

In a seventh aspect, an embodiment of the present disclosure provides a computer system including a processing component and a storage component;

where the storage component stores one or more computer instructions, and the one or more computer instructions are configured to be invoked and executed by the processing component to: run a kernel program in a kernel mode to implement the data processing method according to the first aspect; and run a service program in a user mode to implement the data processing method according to the second aspect and the third aspect.

In an eighth aspect, an embodiment of the present disclosure provides a computer-readable

storage medium having a computer program stored thereon, where when the computer program is executed by a processing component, the data processing method according to the first aspect is implemented, or the data processing method according to the second aspect is implemented, or the data processing method according to the third aspect is implemented.

In a ninth aspect, an embodiment of the present disclosure provides a computer program product including a computer program/computer instructions, where when the computer program/computer instructions are executed by a processing component, the data processing method according to the first aspect is implemented, or the data processing method according to the second aspect is implemented, or the data processing method according to the third aspect is implemented.

In the embodiments of the present disclosure, a shared memory region is constructed between the kernel program and the first service program, and the shared memory region includes a service data storage region and a control command storage region. The control command storage region corresponds to physical memory allocated by the kernel program and is mapped to the first virtual address space of the first service program. The second service program only needs to transfer the first control command. Based on the physical address region carried in the first control command, a virtual address region can be allocated from the second virtual address space to which the service data storage region is mapped, and a second control command can be generated based on the virtual address region. The second control command is then stored in the control command storage region. The kernel program can also establish a mapping relationship between the virtual address region and the physical address region. The first service program can then obtain the second control command from the control command storage region and initiate an access operation based on the virtual address region in the second control command, so that the first service program can access the physical address region. In the embodiments of the present disclosure, by way of shared memory and dynamic memory mapping, zero-copy data transfer between the first service program and the second service program is achieved, and high-speed data transmission between the first service program and the second service program is achieved by separating the control link and the data link.

These and other aspects of the present disclosure will be more clearly and readily understood from the following description of embodiments.

BRIEF DESCRIPTION OF THE DRAWINGS

To more clearly illustrate the technical solutions in the embodiments of the present disclosure or in the prior art, the following briefly introduces the accompanying drawings required to be used

in the description of the embodiments or the prior art. Obviously, the accompanying drawings described below are some embodiments of the present disclosure, and those of ordinary skill in the art can derive other drawings from these accompanying drawings without creative effort.

FIG. 1 shows a schematic diagram of data interaction according to the present disclosure.

FIG. 2 is a flowchart of an embodiment of a data processing method according to an embodiment of the present disclosure.

FIG. 3 is a flowchart of an embodiment of a data processing method according to another embodiment of the present disclosure.

FIG. 4 is a flowchart of an embodiment of a data processing method according to another embodiment of the present disclosure.

FIG. 5 shows a schematic diagram of internal data interaction of a computer system in a practical application according to an embodiment of the present disclosure.

FIG. 6 is a schematic structural diagram of a shared memory region in a practical application according to an embodiment of the present disclosure.

FIG. 7 is a structural diagram of a data processing apparatus according to an embodiment of the present disclosure.

FIG. 8 is a structural diagram of a data processing apparatus according to an embodiment of the present disclosure.

FIG. 9 is a structural diagram of a data processing apparatus according to an embodiment of the present disclosure.

FIG. 10 is a structural diagram of a computer system according to an embodiment of the present disclosure.

DETAILED DESCRIPTION

To enable those skilled in the art to better understand the solutions of the present disclosure, the technical solutions in the embodiments of the present disclosure are clearly and completely described below with reference to the accompanying drawings.

Some procedures described in the specification and claims of the present disclosure and in the accompanying drawings include multiple operations appearing in a specific order. However, it should be clearly understood that these operations may not be executed in the order in which they appear herein, or may be executed in parallel. The sequence numbers of the operations, such as 101, 102, etc., are merely used to distinguish the individual operations from one another, and the sequence numbers themselves do not represent any execution order. In addition, these procedures may include more or fewer operations, and these operations may be executed

sequentially or in parallel. It should be noted that descriptions such as “first,” “second,” and the like herein are used to distinguish different messages, devices, modules, and the like, and do not denote any sequential order, nor do they limit the “first” and the “second” to being of different types.

To facilitate an understanding of the technical solutions of the present disclosure, explanations of technical terms involved in the present disclosure are first provided in the following.

Kernel program: refers to the kernel (kernel) of an operating system, which provides the most basic functions of the operating system and is the foundation of operation of the operating system. The kernel program is a special high-privilege system program that can access all hardware devices, such as network cards and memory devices.

Service program: a low-privilege application program or system program that has limited access to certain memory space and has no access to hardware devices. Service programs may include, for example, storage services that provide persistent storage using external storage devices, network services, application services, multimedia services, file synchronization services, etc.

Kernel mode (Kernel Mode): an operating mode of the operating system, which is the running state of the kernel program of the operating system and has the highest privilege level. In the kernel mode, the kernel program is able to access and control all system resources, including the CPU (processor), memory, external devices, and all other hardware resources. The kernel program is responsible for performing core functions such as process scheduling, memory management, device driver programs, and system call services.

User mode (User Mode): an operating mode of the operating system, which is the default state in which service programs run and has a limited privilege level. In the user mode, the service programs cannot directly access system resources and can only request services from the kernel program through system calls. Their access to memory and hardware resources is strictly limited to prevent erroneous operations or malicious behaviors from causing system crashes. For example, when a service program needs to perform an IO (Input/Output, input/output) operation or the like, the kernel program is required to participate. In the embodiments of the present disclosure, data transfer between service programs involves IO operations. When an interrupt event (such as a hardware interrupt or a software exception) occurs, the CPU will automatically switch from the user mode to the kernel mode so that the kernel program is able to perform interrupt processing.

System call (System Call): an interface provided by the kernel program to service programs. When a service program needs to perform operations involving underlying hardware, such as reading and writing data, creating processes, and network communication, the service program has

to switch to the kernel mode through a system call. The system call allows the service program to request kernel services under strict rules without directly manipulating hardware. The system call is a bridge for interaction between the user mode and the kernel mode, and is also a key mechanism for implementing resource sharing, permission control, and system service calling between service programs and the kernel program of the operating system.

Process: a process is the basic unit of resource allocation and management in the operating system, and is an instance of a program which is running. When a program is executed, the operating system will create a process for the program, allocate resources for the process, and performs execution according to the instruction sequence of the program. Simply put, a process may be deemed as a single execution of a program. A program may run multiple times, and each run will generate an independent process. The process includes various information such as program code, data, a stack, a heap, open file descriptors, etc., and each process has its own memory space and independent execution context. The programs running in the embodiments of the present disclosure are processes.

Physical memory (Physical Memory): also referred to as main memory, is a temporary storage component in a computer system, and is configured to store programs currently being executed and data thereof. The size of the physical memory directly affects the ability of the system to run multiple programs simultaneously.

Virtual memory (Virtual Memory): a memory management technique that allows each process to have an independent address space that is far larger than the actual physical memory capacity. By temporarily saving a portion of the address space of a process to a disk (typically a swap file or a page file) and loading it into the physical memory when needed, the virtual memory can effectively expand the number and size of programs that can run.

Virtual address space: the virtual address space is a memory abstraction exclusive to each process, and is a logically contiguous address range allocated by the operating system for each process. The operating system implements the mapping and management of the virtual address space to the physical memory through a virtual memory management technique such as paging, segmentation, swapping, and memory mapping. In this way, isolation between processes is achieved, while processes are also allowed to access memory resources exceeding the size of the actual physical memory.

Virtual page (Virtual Page): the virtual address space is divided into fixed-size blocks, and these blocks are referred to as virtual pages. Each virtual page may be mapped to a certain physical page in the physical memory, or may be temporarily unmapped. A virtual address region described in the embodiments of the present disclosure may refer to a virtual page.

Physical page (Physical Page): the physical memory is also divided into pages of the same

size, and a physical page is a real hardware storage unit. After a virtual address is translated, the content of the corresponding virtual page will be read from or written to the physical page. A physical address region described in the embodiments of the present disclosure may refer to a physical page.

Memory mapping: refers to associating the physical memory with the virtual address space of a process, such that the process is able to access the physical memory by way of memory access.

Memory sharing: refers to allowing multiple processes to access the same physical address region, thereby achieving the purpose of data sharing.

Page fault (Page Fault) interrupt: refers to a special type of interrupt triggered during the execution of a computer system, where a virtual memory page accessed by a process has not yet been mapped to the physical memory, and the hardware detects this condition and triggers the interrupt. This interrupt will suspend the normal execution procedure of the current service program and notify the kernel program to perform processing.

External device: refers to a device other than the host in a computer system, including an input device (such as a keyboard, a mouse, etc.), an output device (such as a display, a printer, etc.), an external storage device (such as a hard disk, a floppy disk, an optical disk), etc. External devices may be classified, according to their different working modes, into device types such as character devices (character devices), block devices (block devices), and network communication devices.

Ring buffer (Ring Buffer): a data structure, whose characteristic is that its head and tail are connected to form a ring-shaped structure. Since data access between service programs typically requires participation of the kernel, a traditional approach requires a large number of data copy operations. As shown in FIG. 1, taking data transfer from a second service program 101 to a first service program 102 as an example, a schematic diagram of data interaction is illustrated. The second service program 101 and the first service program 102 are in the user mode, and a kernel program 103 is in the kernel mode.

During data transfer, the second service program 101 copies data from its memory to the memory of the kernel program 103. Then, the kernel program 103 copies the data from its memory to the memory of the first service program 102, thereby completing the data transfer between the second service program 101 and the first service program 102. It can be seen that this process involves a large number of data copy operations, thereby affecting data processing efficiency.

During implementation of the concept of the present disclosure, the inventors found that in the process from receiving data from a source service program to copying the data to a destination service program, the kernel program typically will not process the data but merely performs certain operations such as interface conversion and protocol conversion. A large amount of data copying

is performed in order to implement operations such as interface conversion and protocol conversion, which not only affects data processing efficiency but will also degrade system performance.

To address the above problems, the inventors analyzed the above process and found that the content copied from the second service program to the kernel program includes control commands and data, and the kernel program primarily performs protocol conversion and interface conversion processing on the control commands sent by the source program, rather than processing the data. Therefore, the content copied from the second service program may be classified into two types, namely a control link and a data link. For control commands, since their data volume is relatively small, the transmission may be performed by way of copying. For the data link, whether a zero-copy transmission mode is applicable is to be explored.

To this end, the inventors proposed the technical solutions of the embodiments of the present disclosure through a series of research. In the embodiments of the present disclosure, by way of shared memory and dynamic memory mapping, physical memory for IO operations is dynamically mapped to a designated virtual address region, thereby achieving zero-copy data transfer between a first service program and a second service program. High-speed data transmission between the first service program and the second service program is achieved by separating the control link and the data link.

It should be noted that the embodiments of the present disclosure may involve the use of user data. In practical applications, user-specific personal data may be used in the solutions described herein within the scope permitted by applicable laws and regulations of the relevant jurisdictions, under the condition that the requirements of the applicable laws and regulations are met (for example, with explicit user consent, effective user notification, etc.).

It should be noted that user information (including but not limited to user device information, user personal information, etc.) and data (including but not limited to data used for analysis, stored data, displayed data, etc.) involved in the present disclosure are all information and data authorized by the user or fully authorized by all relevant parties. The collection, use, and processing of the relevant data need to comply with the relevant laws, regulations, and standards of the relevant jurisdictions, and corresponding operation entries are provided for the user to choose to authorize or refuse.

The technical solutions in the embodiments of the present disclosure will be clearly and completely described below with reference to the accompanying drawings in the embodiments of the present disclosure. It is apparent that the described embodiments are merely some of the embodiments of the present disclosure, rather than all embodiments. Based on the embodiments of the present disclosure, all other embodiments obtained by those skilled in the art without

creative effort fall within the scope of protection of the present disclosure.

FIG. 2 is a flowchart of an embodiment of a data processing method according to an embodiment of the present disclosure. The technical solution of this embodiment may be performed by a kernel program in a computer system.

In one implementation, a kernel mode and a user mode may refer to a kernel mode and a user mode of a virtual machine created by the computer system. Of course, the kernel mode and the user mode may also refer to a kernel mode and a user mode provided by a physical host of the computer system.

In a case where the technical solution of the embodiment of the present disclosure are applied in a cloud computing scenario, the computer system is a hardware device provided by a cloud computing provider, and a created virtual machine may for example serve as a cloud server or the like. One or more service programs for performing different processing operations or the like may run in the virtual machine, and these service programs may also include cloud applications and the like.

The kernel program runs in the kernel mode, and the service programs run in the user mode.

In the embodiment of the present disclosure, a shared memory region is constructed between the kernel program and a first service program. The shared memory region includes a service data storage region and a control command storage region. The control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program. The service data storage region is mapped to a second virtual address space of the first service program.

The shared memory region may be pre-constructed, etc.

The data processing method in the embodiment shown in FIG. 2 may include the following steps.

Step 201: obtain a first control command transferred by a second service program, and determine a physical address region.

The first control command may be used to indicate a data processing operation, such as data storage or data reading.

In one practical application of the present disclosure, the first service program may be implemented as a storage service, and the second service program may be implemented as a network service, an application service, a multimedia service, a file synchronization service, or the like, for example. For a control command indicating data storage, the control command is used to utilize the first service program to achieve persistent storage of data. For a control command indicating data reading, the control command is used to utilize the first service program to obtain required data.

Depending on service types, the data referred to herein refers to file data, log data, configuration data, sensor data, or the like, for example, and the present disclosure is not limited thereto.

After the second service program is started, the kernel program allocates a virtual address space for the second service program and maps the virtual address space to corresponding physical memory. Virtual memory corresponding to the virtual address space in the second service program may be divided into a command buffer region and a data buffer region, corresponding to different virtual address regions, respectively. The command buffer region stores control commands, and the data buffer region stores actual data.

The first control command may carry the physical address region to which the virtual address region allocated for the data buffer region is mapped. Of course, alternatively, the first control command may carry the virtual address region allocated for the data buffer region, and the kernel program determines the physical address region to which the virtual address region is mapped.

In one practical application, the virtual address region described herein may refer to a virtual page corresponding to a virtual address range. In one practical application, the physical address region may refer to a physical page corresponding to a physical address range.

Step 202: allocate a virtual address region from the second virtual address space to which the service data storage region is mapped, and generate a second control command based on the virtual address region.

The second control command may be stored in the control command storage region.

Step 203: establish a mapping relationship between the virtual address region and the physical address region, to enable the first service program to access the physical address region based on the virtual address region in the second control command obtained from the control command storage region.

The second virtual address space to which the service data storage region is mapped may be divided into a plurality of virtual address regions, i.e., a plurality of virtual pages. The kernel program may allocate a free virtual address region from the second virtual address space for the first service program in response to the first control command.

The kernel program may then generate the second control command based on the virtual address region. The virtual address region may replace the physical address region in the first control command. In addition, corresponding interface conversion and/or protocol conversion, etc., may be performed on the first control command according to actual demands, thereby generating the second control command.

After generating the second control command, the kernel program may store the second control command in the control command storage region. Because the control command storage

region is located in the shared memory region between the kernel program and the first service program, the first service program can access the control command storage region based on the first virtual address space. Accordingly, the first service program can obtain the second control command by accessing the control command storage region, and can read the virtual address region in the second control command.

The mapping relationship may be established between the virtual address region and the physical address region, thereby achieving a mapping from the virtual memory to the physical memory. By establishing the mapping relationship between the virtual virtual address and the physical address region, when the first service program accesses the virtual address region, the physical address region corresponding to the virtual address region can be found based on the mapping relationship, and the physical address region is further mapped to the data buffer region of the second service program, which can thus ultimately implement the access operation on data in the data buffer region. The data in the data buffer region does not need to be copied, thereby achieving zero-copy data access, improving data access efficiency, and improving data processing efficiency.

It should be noted that when the service data storage region is mapped to the second virtual address space, the kernel program does not allocate physical memory for the service data storage region. Instead, during data processing, memory mapping is performed dynamically, to establish the mapping relationship between the virtual address region allocated for the first control command and the physical address region.

In the above process, the first control command may be transferred by the second service program to the kernel program and then by the kernel program to the first service program by way of data copying, thereby forming a control link. Data that needs to be transferred is separated from the control link and does not need to be transferred from the second service program to the kernel program and then to the first service program, or from the first service program to the kernel program and then to the second service program. Zero-copy data transfer is thus achieved, and thus the IO volume can be reduced, thereby improving data processing efficiency, and saving system resources and memory bandwidth.

In the embodiment of the present disclosure, a shared memory region is constructed between the kernel program and the first service program, and the shared memory region includes a service data storage region and a control command storage region. The control command storage region corresponds to physical memory allocated by the kernel program and is mapped to the first virtual address space of the first service program. The second service program only needs to transfer the first control command. Based on the physical address region carried in the first control command, a virtual address region can be allocated from the second virtual address space to which the service

data storage region is mapped, and a second control command can be generated based on the virtual address region. The second control command is then stored in the control command storage region. The kernel program can also establish a mapping relationship between the virtual address region and the physical address region. The first service program can then obtain the second control command from the control command storage region and initiate an access operation based on the virtual address region in the second control command, so that the first service program can access the physical address region. In the embodiment of the present disclosure, by way of shared memory and dynamic memory mapping, zero-copy data transfer between the first service program and the second service program is achieved, and high-speed data transmission between the first service program and the second service program is achieved by separating the control link and the data link.

Because the virtual address region is dynamically mapped to the physical address region in the embodiment of the present disclosure, a situation may arise where the virtual address region has not been mapped to the physical address region when the first service program initiates the access operation based on the virtual address region. In such a situation, a page fault interrupt will be generated, resulting in access failure.

Therefore, in some embodiments, the establishing the mapping relationship between the virtual address region and the physical address region may be: establishing the mapping relationship between the virtual address region and the physical address region in response to a page fault interrupt. That is, the kernel program may establish the mapping relationship between the virtual address region and the physical address region when processing the page fault interrupt. The page fault interrupt refers to a special type of interrupt triggered during execution of the computer system, where a virtual memory page accessed by a process has not yet been mapped to the physical memory, and the hardware detects this condition and triggers the interrupt. This interrupt will suspend the normal execution procedure of the current service program and notify the kernel program to perform processing.

After the page fault interrupt, the access operation of the first service program will be suspended. Therefore, in some embodiments, after the kernel program establishes the mapping relationship between the virtual address region and the physical address region in response to the page fault interrupt, the method may further include:

sending a resumption instruction to the first service program, to enable the first service program to access, in response to the resumption instruction, the physical address region based on the mapping relationship.

After the mapping relationship between the virtual address region and the physical address region is established, the first service program can access the physical address region based on the

created mapping relationship. In this situation, the kernel program may send a resumption instruction to the first service program. The resumption instruction may be used to resume the access procedure previously suspended by the kernel program.

In response to the resumption instruction, the first service program may continue to execute the access operation.

In some embodiments, after allocating the virtual address region from the second virtual address space to which the service data storage region is mapped, the method may further include:

saving a first correspondence between the virtual address region and the physical address region.

In some embodiments, the establishing the mapping relationship between the virtual address region and the physical address region may specifically be implemented as:

searching for the first correspondence based on the virtual address region to determine the physical address region; and establishing the mapping relationship between the virtual address region and the physical address region.

After the virtual address region is allocated from the second virtual address space, the first correspondence between the virtual address region and the physical address region may be saved at first, to maintain the correspondence between the virtual address region and the physical address region.

The mapping relationship between the virtual address region and the physical address region may be created after the kernel program stores the second control command in the control command storage region and then the first service program initiates the access operation to the shared memory region based on the virtual address region obtained from the second control command.

In some embodiments, the data processing method may further include:

saving a second correspondence between the virtual address region and the physical address region.

The second correspondence may be the same as the first correspondence described above, but is not limited thereto. The second correspondence may also be saved after the mapping relationship between the virtual address region and the physical address region is established.

In some embodiments, the allocating the virtual address region from the second virtual address space to which the service data storage region is mapped, and generating the second control command based on the virtual address region includes:

searching for the second correspondence to determine whether a virtual address region to which the physical address region is mapped exists;

if no virtual address region to which the physical address region is mapped exists, allocating

a virtual address region from the second virtual address space to which the service data storage region is mapped, and generating the second control command based on the virtual address region;

if the virtual address region to which the physical address region is mapped exists, generating the second control command based on the virtual address region that is found.

During practical applications, for some physical address regions, mapping relationships with virtual address regions may have already been created previously, and the kernel program may save these mapping relationships, for example, saving them as second correspondences.

In order to avoid frequent page fault interrupts and further ensure data processing efficiency, after receiving the first control command, the kernel program may first search for the second correspondence to determine whether a virtual address region corresponding to the physical address region exists. If the physical address region already has a corresponding virtual address region, the mapping relationship may directly be reused without re-allocating a corresponding virtual address region for the physical address region, so that when the first service program initiates the access operation based on the virtual address region, the access can succeed because the virtual address region is mapped to the physical address region, and no page fault interrupt will be generated, thereby reducing page fault interrupts.

If it is determined by searching for the second correspondence that no virtual address region to which the physical address region is mapped exists, the step of allocating the virtual address region for the physical address region may continue to be executed.

In addition, in order to efficiently manage the correspondence between the virtual address region and the physical address region, the correspondence between the virtual address region and the physical address region may be saved as a radix tree (Radix Tree).

The radix tree is a multi-way tree, where each node includes a prefix and a plurality of child nodes. When constructing a mapping relationship from a virtual address to a physical address, tree nodes may be established level by level according to bits of the virtual address, until a leaf node stores the corresponding physical address.

In some embodiments, the saving the first correspondence between the virtual address region and the physical address region may specifically be implemented as:

constructing a first radix tree using the virtual address region as a key and the corresponding physical address region as a value.

In some embodiments, the saving the second correspondence between the virtual address region and the physical address region may specifically be implemented as:

constructing a second radix tree using the physical address region as a key and the corresponding virtual address region as a value.

Different approaches may be adopted to construct radix trees for different application

scenarios.

For example, in a scenario where it is needed to search for a corresponding physical address region using a virtual address region, a first radix tree may be constructed using the virtual address region as a key and the corresponding physical address region as a value.

For example, in a scenario where it is needed to search for a corresponding virtual address region using a physical address region, a second radix tree may be constructed using the physical address region as a key and the corresponding virtual address region as a value.

In the above embodiments, the first correspondence and the second correspondence may be saved in a form of a radix tree. By using the radix tree, fast querying can be achieved to further improve data processing efficiency. Of course, saving structures of the first correspondence and the second correspondence may not be limited to the radix tree, and may also be other tree forms such as a binary tree.

Before allocating the virtual address region for the physical address region, the second radix tree may first be searched based on the physical address region to determine whether a virtual address region corresponding to the physical address region exists in the second radix tree.

Because the second virtual address space can only be divided into a finite number of virtual address regions, after multiple mappings from virtual address regions to physical address regions are performed, the virtual address regions may be exhausted, that is, all virtual address regions are mapped to corresponding physical address regions. In this case, an eviction mechanism may further be established to release virtual address region resources.

Therefore, in some embodiments, the method may further include: releasing a mapping relationship corresponding to a virtual address region in the second virtual address space that satisfies a release condition.

In a possible implementation, the virtual address region may be released based on an establishment time of the mapping relationship. For example, the release condition may be that the establishment time is earlier than a preset time, and the mapping relationship whose establishment time is earlier than the preset time may be released, thereby releasing the virtual address region in the mapping relationship. As another example, the release condition may also be that the establishment time is the earliest, and the mapping relationship with the earliest establishment time among a plurality of mapping relationships may be released.

In addition, an access frequency may also be used for determination. For example, the release condition may be that the access frequency is lower than a preset frequency, etc.

The allocating the virtual address region from the second virtual address space to which the service data storage region is mapped may include:

determining whether a free virtual address region exists in the second virtual address space;

if so, randomly allocating a free virtual address region; if not, releasing the mapping relationship corresponding to the virtual address region in the second virtual address space that satisfies the release condition.

In some embodiments, command retrieval may be facilitated. The control command storage region may include a command region and a description region, where the command region may be implemented using a ring buffer, the ring buffer storing the second control commands in a form of a ring queue, and the description region is used to save a write pointer and a read pointer of the command region.

The ring buffer is a commonly used data structure, formed by a contiguous memory region with two ends connected to form a closed loop. New data may be written from one end and read out from the other end. When one full round is written, writing continues from the starting point again, forming a cycle.

The write pointer may be updated by the kernel program, and the read pointer may be updated by the first service program.

After the second control command is generated, the second control command may be stored in the control command storage region. The storing the second control command in the control command storage region may include: writing the second control command to the command region based on the write pointer and the read pointer, and updating the write pointer in the description region.

The read pointer is configured to be updated after the first service program obtains the second control command from the command region.

The write pointer (or referred to as a head pointer), is used to track a next position to which data is to be written, and represents an offset of the current write position relative to a starting address of the ring buffer. Each time the kernel program writes one second control command, the write pointer will be incremented.

The read pointer (or referred to as a tail pointer) is used to track a next position from which data is to be read, and represents an offset of the current read position relative to the starting address of the ring buffer. Each time the first service program reads one second control command, the read pointer will also be incremented.

Whether an unread second control command exists in the command region may be determined based on whether the write pointer and the read pointer are the same. Therefore, the kernel program may write the second control command when the write pointer and the read pointer are the same, that is, when the command region is empty. The first service program may read the second control command from the command region and update the read pointer until the write pointer and the read pointer are the same, when the write pointer and the read pointer are different.

In a possible implementation, the first service program may query the description region in a polling manner to determine whether there is a second control command that needs to be processed.

In another possible implementation, after the kernel program fills the second control command in the command region and adds the write pointer in the description region, the kernel program may send a notification message to the first service program to notify the first service program to obtain the second control command.

In some embodiments, to ensure accurate transfer of the control command, the obtaining the first control command transferred by the second service program may specifically be implemented as:

obtaining the first control command copied by the second service program through a system call.

As described above, the control command may be transferred from the second service program to the kernel program, and then from the kernel program to the first service program, thereby forming a control link.

During transmission of the control command, the system call approach may be used to copy the control command from the second service program to the kernel program, and then copy the control command from the kernel program to the first service program.

Since the data volume of the control command is typically small, transferring the control command by copying will not introduce significant IO overhead or system resource consumption.

In some embodiments, the data processing method may further include:

constructing the shared memory region;

allocating the first virtual address space for the control command storage region and allocating the second virtual address space for the data storage partition from a virtual address space corresponding to the first service program;

allocating physical memory for the control command storage region, and establishing a mapping relationship between the physical memory and the first virtual address space;

feeding back the first virtual address space and the second virtual address space to the first service program, to enable the first service program to access the control command storage region based on the first virtual address space.

The shared memory region may be created based on a sharing request from the first service program, but it is not limited thereto. The shared memory region may also be proactively triggered by the kernel program. For example, the kernel program may create the shared memory region upon receiving the first control command sent from the second service program to the first service program, etc.

After the shared memory region is created, a shared memory identifier may be generated for the shared memory region, where the shared memory identifier may be used to uniquely identify the shared memory region. Then, the kernel program may feed the shared memory identifier back to the first service program.

Correspondingly, the kernel program may also release the shared memory region. For example, the kernel program may release the shared memory region in a case where the running of the first service program terminates.

The first service program may send a mapping request to the kernel program through a system call, so that the kernel program allocates the first virtual address space for the control command storage region and allocates the second virtual address space for the data storage partition from the virtual address space corresponding to the first service program, allocates the physical memory for the control command storage region, establishes the mapping relationship between the physical memory and the first virtual address space, and feeds back the first virtual address space and the second virtual address space to the first service program, to enable the first service program to access the control command storage region based on the first virtual address space.

FIG. 3 is a flowchart of an embodiment of a data processing method according to another embodiment of the present disclosure. The data processing method may be applied to a first service program running in a user mode. A shared memory region is constructed between the first service program and a kernel program. The shared memory region includes a service data storage region and a control command storage region. The control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program. The service data storage region is mapped to a second virtual address space of the first service program.

The data processing method in the embodiment shown in FIG. 3 may include the following steps.

Step 301: access the control command storage region based on the first virtual address space to obtain a second control command, where the second control command is generated by the kernel program obtaining a first control command transferred from a second service program, determining a physical address region, allocating a virtual address region from the second virtual address space to which the service data storage region is mapped, and performing generation based on the virtual address region.

Step 302: determine the virtual address region in the second control command.

Step 303: execute an access operation based on the virtual address region to access the physical address region based on a mapping relationship.

The second service program may include a service program that needs to perform data

transfer with the first service program, for example. The first control command may be used to instruct the kernel program to perform a data transfer operation with the first service program.

In one embodiment of the present disclosure, the first service program may be implemented as a storage service, and the second service program may be implemented as a network service, an application service, a multimedia service, a file synchronization service, or the like, for example. Through the first control command, the kernel program may be requested to transfer data to the storage service to achieve persistent storage of the data.

The data that needs to be transferred between the first service program and the second service program may include file data, log data, configuration data, sensor data, and the like.

The first control command transferred by the second service program to the kernel program may carry a physical address region, where the physical address region may be a storage address of data that needs to be transferred to the first service program. The second service program may write the physical address region to the first control command to inform the kernel program of the physical address range in which the data to be operated is stored.

The virtual address region may refer to a virtual page.

The second virtual address space to which the service data storage region is mapped may be divided into a plurality of virtual address regions, i.e., a plurality of virtual pages.

In response to the first control command, the kernel program may allocate one or more virtual pages for the first service program from the plurality of virtual pages in the second virtual address space.

The kernel program may then generate the second control command based on the virtual address region.

After generating the second control command, the kernel program may store the second control command in the control command storage region. Since the control command storage region is in the shared memory region between the kernel program and the first service program, the first service program can access the control command storage region. Accordingly, the first service program can obtain the second control command by accessing the control command storage region, and can read the virtual page contained in the second control command.

For the virtual page in the second virtual address space, a mapping relationship with a physical page in the physical memory, i.e., a physical address region, may be constructed and generated, thereby achieving a mapping from the virtual memory to the physical memory. By establishing the mapping relationship between the virtual page and the physical page, when a service program accesses the virtual page, the operating system can find the corresponding physical page based on this virtual page, thereby enabling access to the physical page based on the virtual page.

Based on this, after the kernel program establishes the mapping relationship between the virtual address region and the physical address region, the first service program can access the physical address region through the virtual address region, and thus read the data stored in the physical address region that needs to be transferred to the first service program.

In some embodiments, the control command storage region includes a command region and a description region, where the description region is used to save a write pointer and a read pointer of a cache buffer region.

In some embodiments, the accessing the control command storage region based on the first virtual address space to obtain the second control command may specifically be implemented as:

accessing the description region and the command region based on the first virtual address space;

reading the second control command from the command region based on the read pointer and the write pointer in the description region;

updating the read pointer in the description region.

The write pointer may be updated by the kernel program, and the read pointer may be updated by the first service program.

The second control command may be written to the command region of the control command storage region. Each time the kernel program writes the second control command to the command region, the write pointer in the description region may be updated.

The first service program may obtain the second control command from the command region of the control command storage region. Each time the first service program obtains the second control command from the command region, the read pointer in the description region may be updated.

When the write pointer and the read pointer are different, the first service program may read the second control command from the command region and update the read pointer until the write pointer and the read pointer are the same.

In a practical application, the first service program may be a storage service that provides persistent storage, where the storage service may write data to or read data from a persistent storage medium.

FIG. 4 is a flowchart of an embodiment of a data processing method according to another embodiment of the present disclosure.

The data processing method may be applied to a second service program running in a user mode, and virtual memory corresponding to the second service program includes a command buffer region and a data buffer region.

The data processing method in the embodiment shown in FIG. 4 may include the following

steps.

Step 401: determine the data buffer region.

Step 402: send, based on the data buffer region, a first control command for a first service program to a kernel program, where a shared memory region is constructed between the first service program and the kernel program, and the shared memory region includes a service data storage region and a control command storage region, where the control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program; the first control command is used to determine a physical address region corresponding to the data buffer region, and the physical address region is used to establish a mapping relationship with a virtual address region allocated from a second virtual address space to which the service data storage region is mapped, where the second virtual address space is used to generate a second control command which is then stored in the command buffer region, for the first service program to obtain and initiate an access operation based on the second virtual address space.

The second service program may include a service program that needs to perform data transfer with the first service program, for example. The first control command may be used to instruct the kernel program to perform a data transfer operation with the first service program.

In one embodiment of the present disclosure, the first service program may be implemented as a storage service, and the second service program may be implemented as a network service, an application service, a multimedia service, a file synchronization service, or the like, for example. Through the first control command, the kernel program may be requested to transfer data to the storage service to achieve persistent storage of the data.

The data that needs to be transferred between the first service program and the second service program may include file data, log data, configuration data, sensor data, and the like.

The first control command transferred by the second service program to the kernel program may carry a physical address region, where the physical address region may be a storage address of data that needs to be transferred to the first service program. The second service program may write the physical address region to the first control command to inform the kernel program of the physical address range in which the data to be operated is stored.

The virtual address region may refer to a virtual page.

The second virtual address space to which the service data storage region is mapped may be divided into a plurality of virtual address regions, that is, a plurality of virtual pages.

In response to the first control command, the kernel program may allocate one or more virtual pages for the first service program from the plurality of virtual pages in the second virtual address space.

The kernel program may then generate the second control command based on the virtual address region.

After generating the second control command, the kernel program may store the second control command in the control command storage region. Since the control command storage region is in the shared memory region between the kernel program and the first service program, the first service program can access the control command storage region. Accordingly, the first service program can obtain the second control command by accessing the control command storage region, and can read the virtual page in the second control command.

For the virtual page in the second virtual address space, a mapping relationship with a physical page in the physical memory, i.e., a physical address region, may be constructed and generated, thereby achieving a mapping from the virtual memory to the physical memory. By establishing the mapping relationship between the virtual page and the physical page, when a service program accesses the virtual page, the operating system can find the corresponding physical page based on this virtual page, thereby enabling access to the physical page based on the virtual page.

Based on this, after the kernel program establishes the mapping relationship between the virtual address region and the physical address region, the first service program can access the physical address region through the virtual address region, and thus read the data stored in the physical address region that needs to be transferred to the first service program.

For ease of understanding, the technical solutions of the embodiments of the present disclosure are described below by taking an example in which the first service program is a storage service and the second service program is an application service. As shown in FIG. 5, a schematic diagram of internal data interaction of a computer system in a practical application according to an embodiment of the present disclosure is illustrated.

As shown in FIG. 5, an application service 501 and a storage service 502 are in a user mode, and a kernel program 503 is in a kernel mode. The application service has a data access requirement for the storage service, such as data storage or data reading.

To implement data access, the kernel program 503 may construct a shared memory region 5031 with the storage service 502. The shared memory region 5031 may be created according to a sharing request from the storage service 502, but it is not limited thereto. The shared memory region 5031 may also be created by being proactively triggered by the kernel program 503.

The shared memory region 5031 may include a command region 5032, a description region 5033, and a service data storage region 5034, where the command region may be implemented using a ring buffer, although the present disclosure is not limited thereto.

The description region 5033 is used to save a write pointer and a read pointer of the command

region 5032, where the write pointer may be updated by the kernel program 503, and the read pointer may be updated by the storage service 502.

The command region 5032 may be used to store control commands.

A second virtual address space to which the service data storage region 5034 is mapped may be divided into a plurality of virtual address regions, that is, a plurality of virtual pages.

When the application service 501 has an access requirement, a first control command may be sent from a command buffer region 5011 of the application service 501 to a command buffer region 5035 of the kernel program 503.

The first control command may carry a physical address region, where the physical address region may be a physical address region 504 corresponding to a data buffer region 5012 of the application service 501.

After receiving the first control command, the kernel program 503 may obtain the physical address region from the first control command, and then the kernel program 503 may first determine whether a virtual address region having a mapping relationship with the physical address region exists. If such a virtual address region exists, the kernel program 503 may write the corresponding virtual address region to the first control command to generate a second control command, and store the second control command in the command region 5032. If no virtual address region having a mapping relationship with the physical address region exists, the kernel program 503 may allocate a free virtual address region from the plurality of virtual address regions of the second virtual address space to which the service data storage region 5034 is mapped, and then save a first correspondence between the physical address region and the virtual address region, write the virtual address region to the first control command to generate the second control command, and store the second control command in the command region 5032.

After the second control command is stored in the command region 5032, the storage service 502 may obtain the second control command and the virtual address region written in the second control command from the command region 5032, and initiate a data access operation using the virtual address region.

In one case, where a mapping relationship has been established between the virtual address region and the physical address region, the first service program 502 may directly access the physical address region based on the virtual address region to obtain data.

In another case, where the virtual address region is newly allocated by the kernel program 503, the kernel program 503 has only saved the correspondence between the virtual address region and the physical address region but has not yet established a mapping relationship between the virtual address region and the physical address region, which will cause a page fault interrupt. In response to the page fault interrupt, the kernel program 503 may establish the mapping relationship

between the virtual address region and the physical address region, and then the storage service 502 may access the physical memory region based on the mapping relationship to obtain data.

FIG. 6 is a schematic structural diagram of a shared memory region in a practical application according to an embodiment of the present disclosure.

As shown in FIG. 6, the shared memory region includes a description region, a command region, and a service data storage region. To facilitate storage, a plurality of command lines may be stored in the command region, and each command line may use an IO vector (IO vector) to describe the corresponding service data storage region. A data address of the IO vector may be used to point to a certain offset address of the service data storage region, that is, a certain virtual address region, whereby the first service program may directly read data by accessing the offset address.

The service data storage region may be formed by a plurality of virtual address regions, that is, virtual pages, which have no allocated actual physical pages during an initialization phase, and are dynamically mapped to physical pages during data processing.

Through the technical solutions of the embodiments of the present disclosure, zero-copy data transfer between the first service program and the second service program is achieved, and high-speed data transmission between the first service program and the second service program is achieved by separating the control link and the data link.

FIG. 7 is a structural diagram of a data processing apparatus according to an embodiment of the present disclosure. The data processing apparatus may be applied to a kernel program running in a kernel mode. A shared memory region is constructed between the kernel program and a first service program running in a user mode. The shared memory region includes a service data storage region and a control command storage region. The control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program. The service data storage region is mapped to a second virtual address space of the first service program. The apparatus includes:

a first obtaining module 701, configured to obtain a first control command transferred by a second service program, and determine a physical address region;

an address allocation module 702, configured to allocate a virtual address region from the second virtual address space to which the service data storage region is mapped, and generate a second control command based on the virtual address region;

a command storage module 703, configured to store the second control command in the control command storage region;

a relationship establishment module 704, configured to establish a mapping relationship between the virtual address region and the physical address region, to enable the first service

program to access the physical address region based on the virtual address region in the second control command obtained from the control command storage region.

In some embodiments, the data processing apparatus further includes:

a first saving module, configured to save a first correspondence between the virtual address region and the physical address region;

in some embodiments, the relationship establishment module 704 includes:

a relationship search submodule, configured to search for the first correspondence based on the virtual address region to determine the physical address region;

a relationship establishment submodule, configured to establish the mapping relationship between the virtual address region and the physical address region.

In some embodiments, the data processing apparatus further includes:

the first saving module, configured to save a second correspondence between the virtual address region and the physical address region;

in some embodiments, the address allocation module 702 includes:

a first search module, configured to search for the second correspondence to determine whether a virtual address region to which the physical address region is mapped exists;

a first allocation submodule, configured to, in a case where no virtual address region to which the physical address region is mapped exists, allocate a virtual address region from the second virtual address space to which the service data storage region is mapped, and generate the second control command based on the virtual address region;

a command generation submodule, configured to, in a case where the virtual address region to which the physical address region is mapped exists, generate the second control command based on the virtual address region.

In some embodiments, the relationship establishment module 704 includes:

the relationship establishment submodule, configured to establish the mapping relationship between the virtual address region and the physical address region in response to a page fault interrupt, where the page fault interrupt is generated in a case where an operation of the solution fails.

In some embodiments, the first saving module includes:

a first saving submodule, configured to construct a first radix tree using the virtual address region as a key and the corresponding physical address region as a value.

In some embodiments, the first saving module includes:

a second saving submodule, configured to construct a second radix tree using the physical address region as a key and the corresponding virtual address region as a value.

In some embodiments, the control command storage region includes a command region and

a description region, where the command region may be implemented using a ring buffer, and the description region is used to save a write pointer and a read pointer of the command region.

In some embodiments, the command storage module 703 includes:

a command writing submodule, configured to write the second control command to the command region based on the write pointer and the read pointer, and update the write pointer in the description region;

where the read pointer is configured to be updated after the first service program obtains the second control command from the command region based on the write pointer.

In some embodiments, the first obtaining module 701 includes:

a command obtaining submodule, configured to obtain the first control command copied by the second service program through a system call.

In some embodiments, the data processing apparatus further includes:

an access resumption module, configured to send a resumption instruction to the first service program, to enable the first service program to access, in response to the resumption instruction, the physical address region based on the mapping relationship.

In some embodiments, the data processing apparatus further includes:

a region construction module, configured to construct the shared memory region;

a first space allocation module, configured to allocate the first virtual address space for the control command storage region and allocate the second virtual address space for the data storage partition from a virtual address space corresponding to the first service program;

a second space allocation module, configured to allocate physical memory for the control command storage region, and establish a mapping relationship between a physical address region of the physical memory and the first virtual address space;

a feedback module, configured to feed back the first virtual address space and the second virtual address space to the first service program, to enable the first service program to access the control command storage region based on the first virtual address space.

In some embodiments, the data processing apparatus further includes:

a release module, configured to release a mapping relationship corresponding to a virtual address region in the second virtual address space that satisfies a release condition.

The data processing apparatus of FIG. 7 is capable of performing the data processing method of the embodiment shown in FIG. 2, and the implementation principles and technical effects thereof are not repeated again. For the data processing apparatus in the foregoing embodiments, specific manners in which the modules and units perform operations have been described in detail in the embodiments related to the methods, and are not elaborated further herein.

FIG. 8 is a structural diagram of a data processing apparatus according to an embodiment of

the present disclosure, applied to a first service program running in a user mode. A shared memory region is constructed between the first service program and a kernel program. The shared memory region includes a service data storage region and a control command storage region. The control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program. The apparatus includes:

a first access module 801, configured to access the control command storage region based on the first virtual address space to obtain a second control command, where the second control command is generated by the kernel program obtaining a first control command transferred by a second service program, determining a physical address region, allocating a virtual address region from a second virtual address space to which the service data storage region is mapped, and performing generation based on the virtual address region;

a first determination module 802, configured to determine the virtual address region in the second control command;

a second access module 803, configured to execute an access operation based on the virtual address region to access the physical address region based on a mapping relationship.

In some embodiments, the control command storage region includes a command region and a description region, where the description region is used to save a write pointer and a read pointer of the command region.

In some embodiments, the first access module 801 includes:

a first access submodule, configured to access the description region and the control command storage region based on the first virtual address space;

a command reading submodule, configured to read the second control command from the control command storage region based on the write pointer in the description region;

an update submodule, configured to update the read pointer in the description region.

In some embodiments, the first service program is a storage service.

The data processing apparatus of FIG. 8 is capable of performing the data processing method of the embodiment shown in FIG. 3, and implementation principles and technical effects thereof are not repeated herein. For the data processing apparatus in the foregoing embodiments, specific manners in which the modules and units perform operations have been described in detail in the embodiments related to the methods, and are not elaborated further herein.

FIG. 9 is a structural diagram of a data processing apparatus according to an embodiment of the present disclosure, applied to a second service program running in a user mode. Virtual memory corresponding to the second service program includes a command buffer region and a data buffer region. The apparatus includes:

a second determining module 901, configured to determine the data buffer region;

a command sending module 902, configured to send, based on the data buffer region, a first control command for a first service program to a kernel program, where a shared memory region is constructed between the first service program and the kernel program, and the shared memory region includes a service data storage region and a control command storage region, where the control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program; the first control command is used to determine a physical address region corresponding to the data buffer region, and the physical address region is used to establish a mapping relationship with a virtual address region allocated from a second virtual address space to which the service data storage region is mapped, where the second virtual address space is used to generate a second control command which is then stored in the command buffer region, for the first service program to obtain and initiate an access operation based on the second virtual address space.

The data processing apparatus of FIG. 9 is capable of performing the data processing method of the embodiment shown in FIG. 4, and implementation principles and technical effects thereof are not repeated herein. For the data processing apparatus in the foregoing embodiments, specific manners in which the modules and units perform operations have been described in detail in the embodiments related to the methods, and are not elaborated further herein.

In addition, an embodiment of the present disclosure further provides a computer system. As shown in FIG. 10, the computer system may include a storage component 1001 and a processing component 1002;

the storage component 1001 stores one or more computer instructions; the one or more computer instructions are configured to be invoked and executed by the processing component 1002 to: run a kernel program in a kernel mode to implement the data processing method of the embodiment shown in FIG. 2, and run a service program in a user mode to implement the data processing method of the embodiment shown in FIG. 3 and the data processing method of the embodiment shown in FIG. 4.

Of course, the computer system may certainly further include other components, such as an input/output interface, a display component, a communication component, and the like.

The input/output interface provides an interface between the processing component and a peripheral interface module, where the peripheral interface module may be an output device, an input device, or the like. The communication component is configured to facilitate wired or wireless communication between a computing device and other devices, etc.

The processing component may include one or more processors to execute the computer instructions to complete all or some of the steps in the foregoing methods. Of course, the processing component may also be implemented by one or more application specific integrated

circuits (ASICs), digital signal processors (DSPs), digital signal processing devices (DSPDs), programmable logic devices (PLDs), field programmable gate arrays (FPGAs), controllers, microcontrollers, microprocessors, or other electronic components, to perform the foregoing methods.

The storage component is configured to store various types of data to support operations at a terminal. The storage component may be implemented by any type of volatile or non-volatile storage device or a combination thereof, such as static random access memory (SRAM), electrically erasable programmable read-only memory (EEPROM), erasable programmable read-only memory (EPROM), programmable read-only memory (PROM), read-only memory (ROM), magnetic storage, flash memory, a magnetic disk, or an optical disk.

The display component may be an electroluminescent (EL) element, a liquid crystal display, a micro-display having a similar structure, a direct retinal display, or a similar laser-scanning display.

It should be noted that the foregoing computing device may be a physical device or an elastic computing host provided by a cloud computing platform, etc. The computing device may be implemented as a distributed cluster formed by a plurality of servers or terminal devices, or may be implemented as a single server or a single terminal device.

An embodiment of the present disclosure further provides a computer-readable storage medium storing a computer program, where when the computer program is executed by a computer, the data processing methods of the embodiments shown in FIG. 2, FIG. 3, and FIG. 4 may be implemented. The computer-readable medium may be included in the electronic device described in the foregoing embodiments, or exists independently without being assembled into the electronic device.

An embodiment of the present disclosure further provides a computer program product including a computer program carried on a computer-readable storage medium, where when the computer program is executed by a computer, the data processing methods of the embodiments shown in FIG. 2, FIG. 3, and FIG. 4 may be implemented. In such an embodiment, the computer program may be downloaded and installed from a network, and/or installed from a removable medium. When the computer program is executed by a processor, various functions defined in the system of the present disclosure are performed.

Those skilled in the art can clearly understand that, for convenience and brevity of description, specific working processes of the systems, apparatuses, and units described above may be found in the corresponding processes in the foregoing method embodiments, and are not repeated herein.

The apparatus embodiments described above are merely illustrative, where units described as

separate components may be or may be not physically separated, and components displayed as units may be or may be not physical units, that is, they may be located in one place, or may be distributed on a plurality of network units. Some or all of the modules may be selected according to actual needs to achieve the objectives of the solutions of the embodiments. Those skilled in the art can understand and implement the solutions without creative effort.

From the description of the foregoing implementations, those skilled in the art can clearly understand that the implementations may be implemented by means of software plus a necessary general-purpose hardware platform, or alternatively by hardware. Based on such understanding, the technical solutions in essence, or the part contributing to the prior art, may be embodied in the form of a software product, where the computer software product may be stored in a computer-readable storage medium, such as a ROM/RAM, a magnetic disk, an optical disk, or the like, and include a plurality of instructions for causing a computer device (which may be a personal computer, a server, or a network device, etc.) to perform the methods described in the various embodiments or some parts of the embodiments.

It should be noted that the foregoing embodiments are merely used to illustrate the technical solutions of the present disclosure, rather than to limit them; although the present disclosure has been described in detail with reference to the foregoing embodiments, those skilled in the art should understand that modifications may still be made to the technical solutions described in the foregoing embodiments, or equivalent substitutions may be made to some technical features thereof; and these modifications or substitutions do not cause the essence of the corresponding technical solutions to depart from the spirit and scope of the technical solutions of the embodiments of the present disclosure.

CLAIMS

1. A data processing method, applied to a kernel program, wherein a shared memory region is constructed between the kernel program and a first service program, and the shared memory region comprises a service data storage region and a control command storage region, wherein the control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program, and the service data storage region is mapped to a second virtual address space of the first service program; the method comprises:

obtaining a first control command transferred by a second service program, and determining a physical address region;

allocating a virtual address region from the second virtual address space to which the service data storage region is mapped, and generating a second control command based on the virtual address region;

establishing a mapping relationship between the virtual address region and the physical address region, to enable the first service program to access the physical address region based on the virtual address region in the second control command obtained from the control command storage region.

2. The method of claim 1, wherein after allocating the virtual address region from the second virtual address space to which the service data storage region is mapped, the method further comprises:

saving a first correspondence between the virtual address region and the physical address region;

the establishing the mapping relationship between the virtual address region and the physical address region comprises:

searching for the first correspondence based on the virtual address region to determine the physical address region;

establishing the mapping relationship between the virtual address region and the physical address region.

3. The method of claim 1 or 2, further comprising:

saving a second correspondence between the virtual address region and the physical address region;

the allocating the virtual address region from the second virtual address space to which the service data storage region is mapped, and generating the second control command based on the virtual address region comprises:

searching for the second correspondence to determine whether a virtual address region to which the physical address region is mapped exists;

if not, allocating a virtual address region from the second virtual address space to which the service data storage region is mapped, and generating the second control command based on the virtual address region;

if so, generating the second control command based on the virtual address region.

4. The method of any one of claims 1 to 3, wherein the establishing the mapping relationship between the virtual address region and the physical address region comprises:

establishing the mapping relationship between the virtual address region and the physical address region in response to a page fault interrupt, wherein the page fault interrupt is generated in a case where an access operation fails.

5. The method of claim 2, wherein the saving the first correspondence between the virtual address region and the physical address region comprises:

constructing a first radix tree using the virtual address region as a key and the corresponding physical address region as a value.

6. The method of claim 3, wherein the saving the second correspondence between the virtual address region and the physical address region comprises:

constructing a second radix tree using the physical address region as a key and the corresponding virtual address region as a value.

7. The method of any one of claims 1 to 6, wherein the control command storage region comprises a command region and a description region, and the description region is used to save a write pointer and a read pointer of the command region; the method further comprises:

writing the second control command to the command region based on the write pointer and the read pointer, and updating the write pointer in the description region;

wherein the read pointer is configured to be updated after the first service program obtains the second control command from the command region.

8. The method of any one of claims 1 to 7, wherein the obtaining the first control command transferred by the second service program comprises:

obtaining the first control command copied by the second service program through a system call.

9. The method of claim 4, wherein after establishing the mapping relationship between the virtual address region and the physical address region in response to the page fault interrupt, the method further comprises:

sending a resumption instruction to the first service program, to enable the first service program to access, in response to the resumption instruction, the physical address region based on

the mapping relationship.

10. The method of any one of claims 1 to 9, further comprising:

constructing the shared memory region;

allocating the first virtual address space for the control command storage region and allocating the second virtual address space for the data storage partition from a virtual address space corresponding to the first service program;

allocating physical memory for the control command storage region, and establishing a mapping relationship between a physical address region of the physical memory and the first virtual address space;

feeding back the first virtual address space and the second virtual address space to the first service program, to enable the first service program to access the control command storage region based on the first virtual address space.

11. The method of any one of claims 1 to 10, further comprising:

releasing a mapping relationship corresponding to a virtual address region in the second virtual address space that satisfies a release condition.

12. A data processing method, applied to a first service program running in a user mode, wherein a shared memory region is constructed between the first service program and a kernel program, and the shared memory region comprises a service data storage region and a control command storage region, wherein the control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program, and the service data storage region is mapped to a second virtual address space of the first service program; the method comprises:

accessing the control command storage region based on the first virtual address space to obtain a second control command, wherein the second control command is generated by the kernel program obtaining a first control command transferred by a second service program, determining a physical address region, allocating a virtual address region from the second virtual address space to which the service data storage region is mapped, and performing generation based on the virtual address region;

determining the virtual address region in the second control command;

executing an access operation based on the virtual address region to access the physical address region based on a mapping relationship.

13. The method of claim 12, wherein the control command storage region comprises a command region and a description region, and the description region is used to save a write pointer and a read pointer of the command region;

the accessing the control command storage region based on the first virtual address space to

obtain the second control command comprises:

accessing the description region and the command region based on the first virtual address space;

reading the second control command from the command region based on the read pointer and the write pointer in the description region;

updating the read pointer in the description region.

14. The method of claim 12, wherein the first service program is a storage service.

15. A data processing method, applied to a second service program running in a user mode, wherein virtual memory corresponding to the second service program comprises a command buffer region and a data buffer region; the method comprises:

determining the data buffer region;

sending, based on the data buffer region, a first control command for a first service program to a kernel program, wherein a shared memory region is constructed between the first service program and the kernel program, and the shared memory region comprises a service data storage region and a control command storage region, wherein the control command storage region corresponds to physical memory allocated by the kernel program and is mapped to a first virtual address space of the first service program; the first control command is used to determine a physical address region corresponding to the data buffer region, and the physical address region is used to establish a mapping relationship with a virtual address region allocated from a second virtual address space to which the service data storage region is mapped, wherein the second virtual address space is used to generate a second control command which is then stored in the command buffer region, for the first service program to obtain and initiate an access operation based on the second virtual address space.

16. A computer system, comprising a processing component and a storage component;

wherein the storage component stores one or more computer instructions, and the one or more computer instructions are configured to be invoked and executed by the processing component to: run a kernel program in a kernel mode to implement the data processing method of claim 1; and run a service program in a user mode to implement the data processing method of claim 12 or the data processing method of claim 15.

17. A computer-readable storage medium having a computer program stored thereon, wherein when the computer program is executed by a processing component, the data processing method of claim 1 is implemented, or the data processing method of claim 12 is implemented, or the data processing method of claim 15 is implemented.

18. A computer program product, comprising a computer program/computer instructions, wherein when the computer program/computer instructions are executed by a processing

component, the data processing method of claim 1 is implemented, or the data processing method of claim 12 is implemented, or the data processing method of claim 15 is implemented.