

虚拟化管理器的升级方法、设备及存储介质

本公开要求于 2023 年 09 月 18 日提交中国专利局的申请号为 202311206139.7、申请名称为“虚拟化管理器的升级方法、设备及存储介质”的优先权，其全部内容通过引用结合在本公开中。

技术领域

本公开涉及计算机技术领域，尤其涉及一种虚拟化管理器的升级方法、设备及存储介质。

背景技术

在云计算的场景中，租户作为云的使用者期望虚拟机能够不间断、稳定地运行自身应用。基于内核的虚拟化管理组件（Kernel-based Virtual Machine, KVM）作为虚拟化核心组件，经常由于修复安全漏洞、功能问题等原因发布新的版本，这就要求云厂商具备热升级该组件的能力，即在 KVM 升级期间不能终止虚拟机的运行，同时不能长时间暂停虚拟机，避免给用户服务带来明显的抖动。

在传统 KVM 热升级的过程中，暂停虚拟机运行的时间较长，即产生较大的虚拟机暂停时长（Downtime）。

发明内容

本公开的多个方面提供一种虚拟化管理器的升级方法、设备及存储介质，用以降低在虚拟化管理器热升级期间虚拟实例的暂停时长。

本公开实施例提供一种虚拟化管理器的升级方法，包括：

第一虚拟化管理器响应于虚拟化管理器升级事件，暂停宿主机上虚拟实例的运行；

所述第一虚拟化管理器将所述第一虚拟化管理器中的虚拟实例状态信息保存至所述宿主机的目标内存中；所述虚拟实例状态信息为所述虚拟实例的状态信息；

所述第一虚拟化管理器将所述目标内存的地址信息提供给第二虚拟化管理器；所述第二虚拟化管理器用于对所述第一虚拟化管理器进行升级；

所述第二虚拟化管理器根据所述内存地址信息，获取所述虚拟实例状态信息；并根据所述虚拟实例状态信息，在所述第二虚拟化管理器中恢复所述虚拟实例的状态。

本公开实施例还提供一种虚拟化管理器的升级方法，宿主机的内核态运行有第一虚拟化管理组件及对第一虚拟化管理组件进行升级的虚拟化管理器的第二虚拟化管理组件；所述方法包括：

运行于所述宿主机的用户态的第一虚拟化管理进程，响应于虚拟化管理器升级事

件，在所述用户态创建第二虚拟化管理进程；所述第一虚拟化管理进程与所述第一虚拟化管理组件协作；所述第二虚拟化管理进程与所述第二虚拟化管理组件协作；

所述第一虚拟化管理组件响应于所述第一虚拟化管理进程发送的虚拟实例状态保存请求，将所述第一虚拟化管理组件中的虚拟实例状态信息保存至所述内核态的目标
5 内存中；并将所述虚拟实例状态信息的内存地址信息提供给所述第一虚拟化管理进程；

所述第一虚拟化管理进程将所述虚拟实例状态信息的内存地址信息提供给所述第二虚拟化管理进程；

所述第二虚拟化管理组件响应于所述第二虚拟化管理进程发送的虚拟实例状态恢复请求，根据所述虚拟实例状态恢复请求包含的所述虚拟实例状态信息的内存地址信息，获取所述虚拟实例状态信息；并根据所述虚拟实例状态信息，在所述第二虚拟化管理组件中恢复所述虚拟实例的状态。
10

本公开实施例还提供一种虚拟化管理器的升级方法，宿主机的内核态运行有第一虚拟化管理组件及对第一虚拟化管理组件进行升级的虚拟化管理器的第二虚拟化管理组件；所述宿主机的用户态运行有与所述第一虚拟化管理组件协作的第一虚拟化管理
15 进程；所述方法包括：

所述第一虚拟化管理进程响应于虚拟化管理器升级事件，在所述用户态创建第二虚拟化管理进程；所述第二虚拟化管理进程与所述第二虚拟化管理组件协作；

所述第一虚拟化管理进程暂停所述宿主机上的虚拟实例的运行；

所述第一虚拟化管理组件响应于所述第一虚拟化管理进程发送的虚拟实例状态保存请求，将所述第一虚拟化管理组件中的虚拟实例状态信息保存至所述内核态的目标
20 内存中；

所述第一虚拟化管理组件将所述虚拟实例状态信息的内存地址信息提供给所述第一虚拟化管理进程，以供所述第一虚拟化管理进程将所述内存地址信息提供给所述第二虚拟化管理进程；

所述第二虚拟化管理组件根据所述第二虚拟化管理进程发送的虚拟实例状态恢复请求中的所述内存地址信息，获取所述虚拟实例状态信息；

所述第二虚拟化管理组件根据所述虚拟实例状态信息，在所述第二虚拟化管理组件中恢复所述虚拟实例的状态。

本公开实施例还提供一种计算设备，所述计算设备可实现为宿主机；所述计算设备包括：存储器和处理器；其中，所述存储器，用于存储所述第一虚拟化管理器及第二虚拟化管理器对应的计算机程序，以及其它计算机程序；
30

所述处理器耦合至所述存储器，用于执行所述存储器中存储的计算机程序以用于执行上述虚拟化管理器的升级方法中的步骤。

本公开实施例还提供一种存储有计算机指令的计算机可读存储介质，当所述计算机指令被一个或多个处理器执行时，致使所述一个或多个处理器执行上述虚拟化管理
35

器的升级方法中的步骤。

本公开实施例在虚拟化管理器升级过程中，在虚拟实例状态保存和恢复时，原虚拟化管理器和升级后的新的虚拟化管理器之间传递的为虚拟实例状态信息的内存地址，相较于传统方案中虚拟实例状态信息需要在用户态的虚拟化管理进程和内核态的虚拟化管理器之间来回复制的方式，减少了原虚拟化管理器和新的虚拟化管理器之间的复制的数据量，而降低虚拟实例状态恢复过程中的时间开销，有助于降低虚拟化管理器升级产生的虚拟实例的暂停时间。

附图说明

此处所说明的附图用来提供对本公开的进一步理解，构成本公开的一部分，本公开的示意性实施例及其说明用于解释本公开，并不构成对本公开的不当限定。在附图中：

图 1 为传统 KVM 热升级方案进行 KVM 热升级的过程示意图；

图 2a 为本公开实施例提供的宿主机的内部结构示意图及虚拟化管理器的升级过程示意图；

图 2b 为本公开实施例提供的宿主机的另一内部结构示意图及虚拟化管理器的升级过程示意图；

图 3 为本公开实施例提供的虚拟化管理器的升级过程的详细过程示意图；

图 4a 和图 4b 为本公开实施例提供的虚拟化管理器的升级方法的流程示意图；

图 5 为本公开实施例提供的计算设备的结构示意图。

具体实施方式

为使本公开的目的、技术方案和优点更加清楚，下面将结合本公开具体实施例及相应的附图对本公开技术方案进行清楚、完整地描述。显然，所描述的实施例仅是本公开一部分实施例，而不是全部的实施例。基于本公开中的实施例，本领域普通技术人员在没有做出创造性劳动前提下所获得的所有其他实施例，都属于本公开保护的范畴。

虚拟机管理器（Virtual Machine Monitor，VMM）为运行在宿主机（Host）系统上的实现虚拟化管理功能的计算机程序。VMM 一般包括：模拟器 QEMU 进程和基于内核的虚拟话管理组件（Kernel-Based Virtual Machine，KVM）。

其中，KVM 是一种用于内核中的虚拟化组件。KVM 的主要功能是实现中央处理器（Central Processing Unit，CPU）虚拟化、内存虚拟化及中断虚拟化等。操作系统加载了 KVM 模块后，因用户无法直接控制内核模块的操作，需进一步利用其它工具创建虚拟机。

QEMU 是运行在用户态的进程。虚拟机跑在 QEMU 进程中，虚拟机的一个虚拟

CPU (Virtual CPU, VCPU) 对应 QEMU 进程的一个线程。虚拟机使用的内存是 QEMU 进程在宿主机上申请的。除此之外, QEMU 进程还负责虚拟机设备(例如网卡、键盘、鼠标)的模拟。

然而, QEMU 独立运行时性能较差, 利用 KVM 通过一些硬件虚拟化功能提供的接口, 可以实现运行时的高性能。QEMU 通过 `open()` 命令或关闭 `close()` 来打开或关闭 `/dev/kvm` 设备接口文件, 并通过设备输入输出控制 (Input/Output Control, `ioctl`) 接口 `ioctl()` 来调用 KVM 提供的接口函数, 以此来应用 KVM 基于硬件辅助虚拟化技术实现的 CPU 虚拟化、内存虚拟化及 IO 虚拟化等功能。

虚拟机管理器热升级, 分为 QEMU 热升级和 KVM 热升级。本公开重点针对 KVM 热升级进行介绍。KVM 热升级是指在 KVM 升级过程中不终止虚拟机的运行, 也不长时间暂停虚拟机, 避免中断虚拟机上部署的用户服务, 进而避免给虚拟机上部署的用户服务带来明显的抖动。

如图 1 示出的传统 KVM 热升级方案通常采用本地热迁移的思路。其中, QEMU 进程运行在宿主机的用户态, KVM 运行在宿主机的内核态。新 KVM 是对原 KVM 进行升级的 KVM, 一般为 KVM 运维人员加载到内核态的。其中, 内核态和用户态为操作系统的两种不同的运行状态。在内核态下, 可运行操作系统程序, 操作硬件; 而在用户态下只能运行应用程序。

如图 1 所示, 传统 KVM 热升级方案主要包括以下步骤:

步骤 1、原 QEMU 进程可响应于 KVM 热升级指令, 创建新 QEMU 进程; 并在新 QEMU 进程中加载新 QEMU 的二进制文件, 实现 QEMU 热升级。

其中, KVM 热升级指令一般为为用户态中的热升级管理组件发出的。热升级管理组件可由运维人员触发, 向原 QEMU 进程发出 KVM 热升级指令。或者, 热升级管理组件可周期性地向原 QEMU 进程发出 KVM 热升级指令。

步骤 2、新 QEMU 进程打开新 KVM, 并在新 KVM 创建虚拟机 (Virtual Machine, VM) 相关的结构体对象, 即在新 KVM 创建新的 VM。结构体对象为用于存储 VM 的状态信息的数据结构。如 `kvm`、`kvm_vcpu`、`kvm_device` 等。该步骤 2 中的结构体对象为初始状态, 为存储 VM 的状态信息。

新 QEMU 创建新的 VM 完成之后, 向原 QEMU 回复虚拟机创建完成消息。

步骤 3、原 QEMU 进程响应虚拟机创建完成消息, 暂停宿主机上虚拟机的运行。

步骤 4、原 KVM 通过大量的设备输入输出控制 (Input/Output Control, `IOCTL`) 指令调用, 将原 KVM 中的虚拟机状态信息同步到原 QEMU 进程。虚拟机状态信息包括: CPU 寄存器状态、CPU 所在主板上的设备的状态等。CPU 所在主板上的设备包括但不限于: 中断控制器、中断相关的设备及其它设备等。

其中, 一个 CPU 寄存器记录的状态信息的同步, 对应一次 `IOCTL` 调用。一个设备的状态的同步, 也对应一次 `IOCTL` 调用。CPU 寄存器和设备是大量的, 将虚拟机状

态信息同步到原 QEMU，也就需要大量的 IOCTL 调用。

步骤 5、原 QEMU 进程对虚拟机状态信息及虚拟机的模拟设备（如网卡、键盘、鼠标等）的状态信息进行序列化处理，并将序列化的状态信息保存在用户态的内存空间。新 QEMU 进程可从用户态的内存空间获取序列化的状态信息；并对序列化的状态信息

进行反序列化处理，得到上述虚拟机状态信息及虚拟机的模拟设备的状态信息。

步骤 6、新 QEMU 进程通过大量的 IOCTL 调用，将虚拟机状态信息传输给新 KVM。新 KVM 根据虚拟机状态信息，恢复虚拟机的状态，即恢复新 KVM 中各结构体对象的状态。

步骤 7、新 QEMU 进程恢复虚拟机的运行。原 QEMU 释放占用的资源并退出。

根据上述传统 KVM 热升级的流程可知：传统 KVM 热升级的步骤 4-6 均是在虚拟机暂停运行的情况下完成的。上述步骤 4-6 中涉及大量的 IOCTL 调用及多次数据复制，这些操作增加了虚拟机的暂停运行的时长，使得 KVM 热升级的停机时长(Downtime)较大，导致虚拟机部署的用户应用的抖动较大。其中，KVM 热升级的停机时长是指虚拟机在 KVM 热升级期间暂停运行的时长。

为了降低 KVM 热升级过程中虚拟机的暂停时长，在本公开一些实施例中，在虚拟化管理器升级过程中，在虚拟实例状态保存和恢复时，原虚拟化管理器和升级后的新的虚拟化管理器之间传递的为虚拟实例状态信息的内存地址，相较于传统方案中虚拟实例状态信息需要在用户态的虚拟化管理进程和内核态的虚拟化管理器之间来回复制的方式，减少了原虚拟化管理器和新的虚拟化管理器之间的复制的数据量，而降低虚拟实例状态恢复过程中的时间开销，有助于降低虚拟化管理器升级产生的虚拟实例的暂停时间。

以下结合附图，详细说明本公开各实施例提供的技术方案。

应注意到：相同的标号在下面的附图以及实施例中表示同一物体，因此，一旦某一物体在一个附图或实施例中定义，则在随后的附图和实施例中不需要对其进行进一步讨论。

图 2a 为本公开实施例提供的宿主机的内部结构示意图。宿主机可实现为任何具有计算、存储等功能的计算机设备，如服务器或终端设备等。终端设备可为电脑、工作站或手机等。

宿主机上部署有虚拟化管理器。虚拟化管理器是指对虚拟化实例进行管理的计算机程序，可实现为软件功能模块或插件等。虚拟化实例是指利用虚拟机技术，在一台宿主机上部署的多台虚拟服务器，可提供不同的服务。虚拟化实例可为虚拟机（Virtual Machine, VM）、容器或容器组（如 Pod 等）。虚拟化实例也可称为客户端（Guest）。宿主机操作系统负责在多个虚拟化实例之间分配硬件资源，让多个虚拟化实例彼此独立。虚拟化实例的操作系统可通过虚拟化管理器与硬件进行通信。虚拟化实例的实现形式不同，虚拟化管理器不同。

例如，在虚拟化实例为容器或容器组的实施例中，可使用操作系统虚拟化架构将宿主机单个的操作系统划分为多个容器，使用容器管理器进行管理。多个是指 2 个或 2 个以上。相应地，虚拟化管理器实现为容器管理器。

又例如，在虚拟化实例为虚拟机的实施例中，不再对底层的硬件资源进行划分，而是在宿主机的操作系统上加装了虚拟机管理器（Virtual Machine Monitor, VMM），虚拟层作为应用级别的软件而存在，不涉及操作系统内核。虚拟层会给每个虚拟机模拟一套独立的硬件设备，包含处理器、内存、主板、显卡、网卡等硬件资源，在其上安装客户端操作系统。相应地，虚拟化管理器实现为 VMM。

虚拟化管理器经常由于修复安全漏洞、功能问题等原因发布新的版本，这就需要
对虚拟化管理器进行热升级。虚拟化管的热升级是指在虚拟化管
理器的升级过程中，不终止虚拟实例的运行，同时在虚拟化管
理器升级期间虚拟实例不产生较大的虚拟机暂停时长（Downtime），避免给虚拟实例上部署的用户应用带来明显的抖动。虚拟实例的暂停时长是指虚拟化管
理器升级期间虚拟实例的暂停运行时长。

下面对本公开实施例提供的虚拟化管
理器的升级过程进行示例性说明。

如图 2a 所示，在虚拟化管
理器需要升级时，可将
对宿主机中当前运行的虚拟化管
理器进行升级的新的虚拟化管
理器对应的计算机程序加载至宿主机，得到新的虚拟化管
理器。在本公开实施例中，为了便于描述和区分，将宿主机原先运行的虚拟化管
理器，定义为第一虚拟化管
理器 S10；将对第一虚拟化管
理器进行升级的新的虚拟化管
理器，定义为第二虚拟化管
理器 S20。

在虚拟化管
理器升级过程中，需要将第一虚拟化管
理器 S10 中的虚拟实例状态信
息，同步到第二虚拟化管
理器 S20，之后，由第二虚拟化管
理器 S20 对宿主机上的虚
拟实例进行管理。虚拟实例状态信息是指宿主机上的虚拟实例的状态信息，包括但不
局限于：虚拟实例对应的虚拟处理器的状态信息（如虚拟处理器对应的各寄存器记录
的状态信息）及主板上设备的状态信息（如中断控制器、中断相关的其它设备的状态
信息等）等。

在虚拟化管
理器升级过程中，在第二虚拟化管
理器中恢复虚拟实例的状态越快，
虚拟实例的暂停时间越短，虚拟实例上的用户应用的抖动越小。在本公开实施例中，
为了降低虚拟实例的暂停时间，第一虚拟化管
理器 S10，可响应于虚拟化管
理器升级事件，暂停宿主机上的虚拟实例的运行（对应图 2a 步骤 1）。

虚拟化管
理器升级事件可为运行在用户态的升级管理组件（图 2a 中未示出）触发的。升级管理组件可实现对虚拟化管
理升级进行管理的软件功能模块或插件等。升级
管理组件可检测宿主机是否有新的虚拟化管
理器；并在检测到新的虚拟化管
理器的情况下，触发虚拟化管
理器升级事件，并向第一虚拟化管
理器 S10 发出进程创建指令（图
2a 中未示出）。或者，运维人员在将第二虚拟化管
理器加载至宿主机之后，可利用升
级管理组件提供的调用接口，调用升级管理组件来触发虚拟化管
理器升级事件。升级

管理组件响应于虚拟化管理器升级事件，向第一虚拟化管理器发出进程创建指令。

虚拟实例暂停运行之后，可将虚拟实例的状态迁移至第二虚拟化管理器 S20，在第二虚拟化管理器 S20 恢复虚拟实例的运行，实现虚拟化管理器的热升级。基于此，在本实施例中，为了将虚拟实例的状态迁移至第二虚拟化管理器 S20，将第一虚拟化
5 管理器 S10 中的虚拟实例状态信息保存至宿主机的目标内存中（对应图 2a 步骤 2）。具体地，第一虚拟化管理器 S10 可在宿主机中，申请目标内存；并将第一虚拟化管理器 S10 中的虚拟实例状态信息保存至目标内存中。进一步，第一虚拟化管理器 S10 可将虚拟实例状态信息的内存地址信息提供给第二虚拟化管理器 S20（对应图 2a 步骤 3）。虚拟实例状态信息的内存地址信息可为虚拟实例状态信息在内核态内存空间的起始地址及长度等。
10

相应地，第二虚拟化管理器 S20 可获取目标内存的内存地址信息，并根据目标内存的内存地址信息，从目标内存中获取虚拟实例状态信息。进一步，第二虚拟化管理器 S20 可根据虚拟实例状态信息，在第二虚拟化管理组件 S20 中恢复虚拟实例的状态，实现虚拟化管理器的热升级（对应图 2a 步骤 4）。

15 传统方案中虚拟实例状态保存和恢复过程中，虚拟实例状态信息需要在原虚拟化管理器 and 新的虚拟化管理器之间来回复制，还需要虚拟实例状态信息需要在用户态的虚拟化管理进程和内核态的虚拟化管理器之间来回复制，大量的数据复制耗时较长。本实施例中，虚拟实例状态保存和恢复过程中，原虚拟化管理器 and 新的虚拟化管理器之间传递的为虚拟实例状态信息的内存地址，减少了原虚拟化管理器 and 新的虚拟化
20 管理器之间的复制的数据量，降低虚拟化管理器升级产生的虚拟实例暂停时间，进而降低虚拟实例上用户应用的抖动，保障服务性能。

在本公开实施例中，如图 2b 所示，宿主机划分为用户态和内核态。内核态和用户态为操作系统的两种不同的运行状态。在内核态下，可运行操作系统程序，并操作硬件；而在用户态下只能运行应用程序。相应地，虚拟化管理器包括：部署于用户态的
25 虚拟化管理进程 10 及部署于内核态的虚拟化管理组件 20。

其中，部署于内核态的虚拟化管理组件 20 是一种用于内核中的虚拟化组件，主要功能是实现处理器虚拟化、内存虚拟化及中断虚拟化等。操作系统加载了虚拟化管理组件 20 后，因用户无法直接控制内核模块的操作，需进一步利用其它工具创建虚拟机。

虚拟化管理进程 10 是运行在用户态的进程。虚拟机跑在虚拟化管理进程 10 中，
30 虚拟机的一个虚拟处理器对应虚拟化管理进程的一个线程。虚拟机使用的内存是虚拟化管理进程在宿主机上申请的。除此之外，虚拟化管理进程还负责虚拟机设备（例如网卡、键盘、鼠标）的模拟。

然而，虚拟化管理进程独立运行时性能较差，可利用虚拟化管理组件 20 通过一些硬件虚拟化功能提供的接口，可以实现运行时的高性能。

35 在一些实施例中，虚拟化管理组件 20 可为 KVM。相应地，虚拟化管理进程 10

可为 QEMU 进程。QEMU 通过 open()命令或关闭 close() 来打开或关闭/dev/kvm 设备接口文件，并通过设备输入输出控制 (Input/Output Control, ioctl) 接口 ioctl() 来调用 KVM 提供的接口函数，以此来应用 KVM 基于硬件辅助虚拟化技术实现的 CPU 虚拟化、内存虚拟化及 IO 虚拟化等功能。即用户态的虚拟化管理进程 10 与内核态的虚拟化管理组件 20 共同协作，实现对宿主机上的虚拟实例的管理。

虚拟化管理器经常由于修复安全漏洞、功能问题等原因发布新的版本，这就需要对虚拟化管理器进行热升级。虚拟化管理器的热升级分为虚拟化管理进程的热升级和部署于内核态的虚拟化管理组件的热升级。本公开重点针对部署于内核态的虚拟化管理组件的热升级进行介绍。

在运行在内核态的虚拟化管理组件 20 需要升级时，可将对内核态当前运行的虚拟化管理组件 20 进行升级的新的虚拟化管理组件对应的计算机程序加载至内核态，得到部署在内核态的新的虚拟化管理组件。在本公开实施例中，为了便于描述和区分，将宿主机的内核态原先运行的虚拟化管理组件，定义为第一虚拟化管理组件 201；将对第一虚拟化管理组件进行升级的新的虚拟化管理组件，定义为第二虚拟化管理组件 202。

在内核态的虚拟化管理组件升级过程中，需要将第一虚拟化管理组件 201 中的虚拟实例状态信息，同步到第二虚拟化管理组件 202，之后，由第二虚拟化管理组件 202 对宿主机上的虚拟实例进行管理。虚拟实例状态信息是指宿主机上的虚拟实例的状态信息，包括但不限于：虚拟实例对应的虚拟处理器的状态信息（如虚拟处理器对应的各寄存器记录的状态信息）及主板上设备的状态信息（如中断控制器、中断相关的其它设备的状态信息等）等。

在内核态的虚拟化管理组件 20 升级过程中，在第二虚拟化管理组件 202 中恢复虚拟实例的状态越快，虚拟实例的暂停时间越短，虚拟实例上的用户应用的抖动越小。在本公开实施例中，为了降低虚拟实例的暂停时间，结合图 2b 和图 3，用户态的原虚拟化管理进程，可响应于虚拟化管理器升级事件，在用户态创建新的虚拟化管理进程（对应图 2b 中步骤 1 及图 3 中步骤 1-3）。在本公开实施例中，为了便于描述和区分，将用户态的原虚拟化管理进程，定义为第一虚拟化管理进程 101；并将第一虚拟化管理进程 101 创建的新的虚拟化管理进程，定义为第二虚拟化管理进程 102。

如图 3 所示，虚拟化管理器升级事件可为运行在用户态的升级管理组件 30 触发的。升级管理组件 30 可实现对虚拟化管理升级进行管理的软件功能模块或插件等。升级管理组件 30 可检测宿主机的内核态是否有新的虚拟化管理组件；并在检测到新的虚拟化管理组件的情况下，触发虚拟化管理器升级事件（对应图 3 中步骤 1），并向第一虚拟化管理进程 101 发出进程创建指令（对应图 3 中步骤 2）。或者，运维人员在将第二虚拟化管理组件 202 加载至内核态之后，可利用升级管理组件 30 提供的调用接口，调用升级管理组件 30 来触发虚拟化管理器升级事件。升级管理组件 30 响应于虚拟化

管理器升级事件，向第一虚拟化管理进程 101 发出进程创建指令。

针对第一虚拟化管理进程 101，虚拟化管理器升级事件实现为接收到升级管理组件 30 发出的进程创建指令。进一步，第一虚拟化管理进程 101，响应于该进程创建指令，在用户态创建第二虚拟化管理进程 102（对应图 3 中步骤 3）。其中，第二虚拟化管理进程 102 与第二虚拟化管理组件 202 协作，对宿主机上的虚拟实例进行管理。

在一些实施例中，第一虚拟化管理进程 101 可响应于虚拟化管理器升级事件，在用户态创建第一虚拟化管理进程 101 的子进程；并在子进程中加载虚拟化管理进程的 5 二进制文件，以得到第二虚拟化管理进程 102（对应图 3 中步骤 3）。

具体地，如图 3 步骤 3 所示，第一虚拟化管理进程 101 可响应于虚拟化管理器升级事件，调用复刻函数（如 fork 函数）创建第一虚拟化管理进程 101 的子进程。第一虚拟化管理进程 101 为父进程，被创建的进程称为子进程。由于 fork 函数会复制父进程的内存映象，因此新进程会收到父进程地址空间的一份复制的数据。两个进程在 10 fork 函数之后，会分别在自己的内存空间执行继续执行后面的指令。

在本实施例中，fork 函数创建了调用进程（即第一虚拟化管理进程 101）的一份 15 复制进程（即子进程），但很多应用程序要求子进程执行与其父进程不同的代码。exec 函数提供了用新映象覆盖调用进程（即父进程）映象的功能。组合使用 fork-exec 函数可让子进程（用 exec 函数）执行新程序，而父进程继续执行原程序。相应地，如图 3 中步骤 3 所示，第一虚拟化管理进程 101 可调用 exec 函数在子进程中加载虚拟化管理进程的 20 二进制文件，得到第二虚拟化管理进程 102。

其中，若宿主机的用户态存在新版本的二进制文件，则在子进程中加载的虚拟化管理进程的 25 二进制文件为新版本的二进制文件，此时的第二虚拟化管理进程为对第一虚拟化管理进程升级后的虚拟化管理进程，虚拟化管理进程进行了升级。若宿主机的用户态不存在新版本的二进制文件，则在子进程中加载的虚拟化管理进程的 30 二进制文件为原虚拟化管理进程（即第一虚拟化管理进程）对应的二进制文件，则虚拟化管理进程未升级。

在本实施例中，第二虚拟化管理进程 102 与内核态的第二虚拟化管理组件 202 相互协作，对虚拟实例进行管理。具体地，升级管理组件 30 可将第二虚拟化管理组件 202 的标识携带在进程创建指令中，第一虚拟化管理进程 101 在创建第二虚拟化管理进程 102 的过程中，可将第二虚拟化管理组件 202 的标识提供给第二虚拟化管理进程 30 102。

基于第二虚拟化管理组件 202 的标识，第二虚拟化管理进程 102 可打开第二虚拟化管理组件 202。进一步，第二虚拟化管理进程 102 可在第二虚拟化管理组件 202 中创建虚拟实例关联的结构体对象（对应图 3 中步骤 4）。虚拟实例关联的结构体对象是指内核态的虚拟化管理组件 20 中用于记录虚拟实例的状态信息的数据结构，包括但不 35 局限于：用于记录虚拟实例对应的虚拟处理器关联的寄存器的状态信息的数据结构

及用于记录虚拟实例对应的虚拟设备的状态信息的数据结构等。

具体地，第二虚拟化管理进程 102 可根据第二虚拟化管理组件 202 的标识，获取第二虚拟化管理组件 202 向用户态提供的接口（如/dev/kvm 接口），第二虚拟化管理进程 102 可通过打开命令（如 open() 命令）打开第二虚拟化管理组件 202 向用户态提供的接口（如/dev/kvm 接口），来打开第二虚拟化管理组件 202。第二虚拟化管理组件 202 可向第二虚拟管理进程 102 返回文件句柄（File handle）。第二虚拟管理进程 102 可根据文件句柄，在第二虚拟化管理组件 202 中创建虚拟实例关联的结构体对象。

具体地，第二虚拟管理进程 102 可通过 IOCTL 指令，在第二虚拟化管理组件 202 中创建虚拟实例关联的结构体对象。该 IOCTL 指令包括：指示第二虚拟化管理组件 202 执行的命令及文件句柄。IOCTL 指令可为 IOCTL (fd1, cmd)。fd1 为文件句柄；cmd 为指示第二虚拟化管理组件 202 执行的命令，此处该命令为创建命令 cmd。相应地，第二虚拟化管理组件 202 可响应 IOCTL 指令，在第二虚拟化管理组件 202 中创建虚拟实例关联的结构体对象。此时创建的虚拟实例关联的结构体对象，为虚拟实例对应的虚拟处理器及设备等的初始状态，可用来运行一个新的虚拟实例。

第二虚拟管理进程 102 在虚拟实例关联的结构体对象完成之后，可向第一虚拟管理进程 101 返回结构体对象创建完成消息（对应图 3 中步骤 5）。第一虚拟管理进程 101 可响应于虚拟化管理器升级事件，暂停宿主机上的虚拟实例的运行（对应图 2b 步骤 2 及图 3 中步骤 6）具体地，第一虚拟管理进程 101 可在虚拟实例关联的结构体对象完成之后，暂停宿主机上的虚拟实例的运行（对应图 2b 步骤 2 及图 3 中步骤 6）。具体地，第一虚拟管理进程 101 可在接收到结构体对象创建完成消息之后，暂停宿主机上的虚拟实例的运行。

虚拟实例暂停运行之后，可将虚拟实例的状态迁移至第二虚拟化管理组件 202，在第二虚拟化管理组件 202 恢复虚拟实例的运行，实现内核态的虚拟化管理组件的热升级。基于此，在本实施例中，为了将虚拟实例的状态迁移至第二虚拟化管理组件 202，第一虚拟化管理进程 101 可在虚拟实例暂停运行之后，请求第一虚拟化管理组件 201 保存虚拟实例的状态信息。具体地，第一虚拟化管理进程 101 可在虚拟实例暂停运行之后，向第一虚拟化管理组件 201 发送虚拟实例状态保存请求（对应图 2b 步骤 3 及图 3 中步骤 7）。

相应地，第一虚拟化管理组件 201 响应于虚拟实例状态保存请求，将第一虚拟化管理组件 201 中的虚拟实例状态信息保存至内核态的目标内存中（对应图 2b 步骤 4 及图 3 中步骤 8）。具体地，第一虚拟化管理组件 201 响应于虚拟实例状态保存请求，在内核态申请目标内存；并将第一虚拟化管理组件 201 中的虚拟实例状态信息保存至内核态的目标内存中。进一步，第一虚拟化管理组件 201 可将虚拟实例状态信息的内存地址信息提供给第一虚拟化管理进程 101（对应图 2b 步骤 5 及图 3 中步骤 9）。虚拟实例状态信息的内存地址信息可为虚拟实例状态信息在内核态内存空间的起始地址

及长度等。

在一些实施例中，第一虚拟化管理进程 101 可在虚拟实例暂停运行之后，可通过 IOCTL 指令向第一虚拟化管理组件 201 发送虚拟实例状态保存请求。相应地，虚拟实例状态保存请求可实现为 IOCTL 指令，该 IOCTL 指令用于请求第一虚拟化管理组件 201 保存虚拟实例状态信息。IOCTL 指令可包括：指示第一虚拟化管理组件 201 保存虚拟实例状态信息的保存命令，如 save 命令，还可包括第一虚拟化管理组件 201 提供的文件句柄 fd0 等。相应地，第一虚拟化管理组件 201 可响应该 IOCTL 指令，将第一虚拟化管理组件 201 中的虚拟实例状态信息保存至内核态的目标内存中；并将虚拟实例状态信息的内存地址信息作为 IOCTL 指令的调用结果，返回给第一虚拟化管理进程 101。

可选地，第一虚拟化管理组件 201 可响应该 IOCTL 指令，对第一虚拟化管理组件 201 中的虚拟实例状态信息进行序列化，以得到序列化的虚拟实例状态信息（对应图 3 中步骤 8）。在本实施例中，对第一虚拟化管理组件 201 中的虚拟实例状态信息进行序列化，主要是指将虚拟实例状态信息转换为结构体对象与各结构体对象的状态信息之间的对应关系，这样在将虚拟实例状态迁移到第二虚拟化管理组件 202 时，第二虚拟化管理组件 202 可根据序列化的虚拟实例状态信息，确定哪个状态信息对应虚拟实例的结构体对象的哪个字段。可选地，结构体对象与各结构体对象的状态信息之间的对应关系，可实现为键值对（Key-Value，KV）结构。其中，结构体对象的字段标识为键（Key），该结构体对象的字段对应的状态信息为值（Value）。

相应地，第一虚拟化管理组件 201 可将序列化的虚拟实例状态信息的内存地址信息作为 IOCTL 指令的调用结果，返回给第一虚拟化管理进程 101（对应图 3 中步骤 9）。

进一步，第一虚拟化管理进程 101 可将虚拟实例状态信息的内存地址信息提供给第二虚拟化管理进程 102（对应图 2b 步骤 6 及图 3 中步骤 10）。第二虚拟化管理进程 102 可基于虚拟实例状态信息的内存地址信息，请求第二虚拟化管理组件 202 恢复虚拟实例状态。具体地，第二虚拟化管理进程 102 可向第二虚拟化管理组件 202 发送虚拟实例状态恢复请求，该虚拟实例状态恢复请求包含虚拟实例状态信息的内存地址信息（对应图 2b 步骤 7 及图 3 中步骤 11）。

相应地，第二虚拟化管理组件 202 可响应于该虚拟实例状态恢复请求，根据该请求包含的虚拟实例状态信息的内存地址信息，从内核态的目标内存中获取虚拟实例状态信息（对应图 2b 步骤 7 及图 3 中步骤 12）。

在一些实施例中，第二虚拟化管理进程 102 可通过 IOCTL 指令向第二虚拟化管理组件 202 发送虚拟实例状态恢复请求。相应地，虚拟实例状态恢复请求可实现为 IOCTL 指令，该 IOCTL 指令用于请求第二虚拟化管理组件 202 恢复虚拟实例状态信息。IOCTL 指令可包括：指示第二虚拟化管理组件 202 恢复虚拟实例状态信息的恢复命令，如 restore 命令，还可包括第二虚拟化管理组件 202 提供的文件句柄 fd1 及虚拟实例状态

信息的内存地址信息等。

相应地，第二虚拟化管理组件 202 可响应该 IOCTL 指令，获取虚拟实例状态信息的内存地址信息，并根据该请求包含的虚拟实例状态信息的内存地址信息，从内核态的目标内存中获取虚拟实例状态信息。

- 5 对于上述虚拟实例状态信息以序列化的形式存储在内核态的目标内存中的实施例，第二虚拟化管理组件 202 可根据序列化的虚拟实例状态信息的内存地址信息，从内核态的目标内存中获取序列化的虚拟实例状态信息；并对序列化的虚拟实例状态信息进行反序列化处理，以得到虚拟实例状态信息（对应图 3 中步骤 12）。

- 10 在本实施例中，由内核态的虚拟化管理组件可直接在内核态，对虚拟实例状态信息直接进行序列化和反序列化处理，跳过了用户态的虚拟化管理进程在用户态对虚拟实例状态信息直接进行序列化和反序列化处理的过程，减少了虚拟实例状态信息的处理时长，可降低内核态的虚拟化管理组件热升级过程中虚拟实例的暂停时长，可降低虚拟实例上用户应用的抖动，保障服务性能。

- 15 进一步，第二虚拟化管理组件 202 可根据虚拟实例状态信息，在第二虚拟化管理组件 202 中恢复虚拟实例的状态，实现内核态的虚拟化管理组件的热升级（对应图 2b 步骤 8 及图 3 中步骤 13）。

具体地，第二虚拟化管理组件 202 可根据虚拟实例状态信息，对第二虚拟化管理组件 202 中的结构体对象进行赋值，从而在第二虚拟化管理组件 202 中恢复虚拟实例的状态。

- 20 在本实施例中，在内核态的虚拟化管理组件升级过程中，用户态的原虚拟化管理进程向原虚拟化管理组件发送虚拟实例状态保存请求，原虚拟化管理组件响应该请求，将虚拟实例状态信息保存至内核态的内存空间，并返回虚拟实例状态信息的内存地址，实现了原虚拟化管理进程通过一次系统调用保存虚拟实例状态。相较于传统方案中通过大量 IOCTL 类型的系统调用实现虚拟实例状态的保存方式，本实施例可减少虚拟实例状态保存过程中系统调用次数，从而降低虚拟实例状态保存过程中系统调用的时间开销，有助于降低内核态的虚拟化管理组件升级产生的虚拟实例暂停时间，可降低虚拟实例上用户应用的抖动，保障服务性能。
- 25

- 30 另一方面，在本实施例中，用户态的新虚拟化管理进程向新虚拟化管理组件发送包含虚拟实例状态信息的内存地址的虚拟实例状态恢复请求，新虚拟化管理组件可根据虚拟实例状态信息的内存地址，获取虚拟实例状态信息，并在新虚拟化管理组件中恢复虚拟实例状态，实现了新虚拟化管理进程通过一次系统调用恢复虚拟实例状态。相较于传统方案中通过大量 IOCTL 类型的系统调用实现恢复的方式，本实施例可减少虚拟实例状态恢复过程中系统调用次数，从而降低虚拟实例状态恢复过程中系统调用的时间开销，有助于降低虚拟化管理组件升级产生的虚拟实例暂停时间，可进一步降低虚拟实例上用户应用的抖动，保障服务性能。
- 35

再者，传统方案中虚拟实例状态保存和恢复过程中，虚拟实例状态信息需要在用户态的虚拟化管理进程和内核态的虚拟化管理组件之间来回复制，大量的数据复制耗时较长。本实施例中，虚拟实例状态保存和恢复过程中，用户态的虚拟化管理进程和内核态的虚拟化管理组件之间传递的为虚拟实例状态信息的内存地址，减少了用户态的虚拟化管理进程和内核态的虚拟化管理组件之间的复制的数据量，可进一步降低虚拟化管理组件升级产生的虚拟实例暂停时间，可降低虚拟实例上用户应用的抖动，保障服务性能。

除了上述内核态的虚拟化管理组件热升级之外，本公开实施例还提供用户态的虚拟化管理进程升级与内核态的虚拟化管理组件升级解耦，可在内核态的虚拟化管理组件热升级时，灵活选择是否升级用户态的虚拟化管理进程。例如，可向用户提供用户态的虚拟化管理进程是否升级的选择接口，由用户自主选择是否升级虚拟化管理进程。

在本实施例中，第二虚拟化管理组件 202 在恢复虚拟实例的状态之后，可向第二虚拟化管理进程 102 返回恢复完成消息。可选地，第二虚拟化管理组件 202 可将恢复完成消息，作为上述用于请求第二虚拟化管理组件 202 恢复虚拟实例状态信息的 IOCTL 指令的调用结果，返回给第二虚拟化管理进程 102。对于选择升级用户态的虚拟化管理进程的实施例，第二虚拟化管理进程 102 可响应于恢复完成消息，恢复宿主机上虚拟实例的运行。

对于选择不升级用户态的虚拟化管理进程的实施例，由于升级后的第二虚拟化管理组件 202 的文件句柄是第二虚拟化管理进程 102 打开的，第一虚拟化管理进程 101 需要获得第二虚拟化管理组件 202 的操作句柄，才可操作升级后的第二虚拟化管理组件 202。基于此，第二虚拟化管理进程 102 可响应于恢复完成消息，将其打开的目标文件句柄提供给第一虚拟化管理进程 101（对图 3 中步骤 14），并释放第二虚拟化管理进程 102 占用的资源。其中，第二虚拟化管理进程 102 打开的目标文件句柄，为打开的第二虚拟化管理组件 202 提供的文件句柄，包括上述打开第二虚拟化管理组件 202 过程中的文件句柄 fd1，也包括对第二虚拟化管理组件 202 操作的其它文件句柄。

对于第一虚拟化管理进程 102 在获取目标文件句柄之后，可释放原第一虚拟化管理组件 201 的文件句柄，并基于第二虚拟化管理组件 202 的目标文件句柄继续运行。具体地，第一虚拟化管理进程 102 可基于目标文件句柄，在第二虚拟化管理组件 202 中恢复宿主机上虚拟实例的运行（对应图 2b 步骤 9 及图 3 中步骤 15）。

具体地，第一虚拟化管理进程 102 基于目标文件句柄，向第二虚拟化管理组件发送请求运行虚拟实例的 IOCTL 指令。该 IOCTL 指令可包括请求运行虚拟实例的命令，如 run 命令。第二虚拟化管理组件 202 响应该 IOCTL 指令，在第二虚拟化管理组件 202 中恢复宿主机上虚拟实例的运行。

在本实施例中，通过将新虚拟化管理进程打开的文件句柄返回给原虚拟化管理进程，可实现仅升级内核态的虚拟化管理组件，而保持用户态的原虚拟化管理进程，提

高了虚拟化管理器运维的灵活性。

值得说明的是，上述虚拟化管理组件的升级过程，可适用于单个虚拟化管理组件进行升级，也适用于同时对多个虚拟化管理组件进行升级。多个是指2个或2个以上。对多个虚拟化管理组件同时升级的过程，是对每个虚拟化管理组件单独升级的，升级过程与单个虚拟化管理组件升级过程相同。多个虚拟化管理组件同时升级时，每个原虚拟化管理组件（即第一虚拟化管理组件）对应的新的虚拟化管理组件（即第二虚拟化管理组件）可以相同，也可不同，具体由运维人员根据实际需求灵活设置。

除了上述实施例提供的虚拟化管理器升级系统之外，本公开实施例还提供虚拟化管理器升级方法。下面对本公开实施例提供的虚拟化管理器的升级方法进行示例性说明。

图4a为本公开实施例提供的虚拟化管理器的升级方法的流程示意图。如图4a所示，该方法包括：

41、第一虚拟化管理器响应于虚拟化管理器升级事件，暂停宿主机上虚拟实例的运行。

42、第一虚拟化管理器将第一虚拟化管理器中的虚拟实例状态信息保存至宿主机的目标内存中；虚拟实例状态信息为虚拟实例的状态信息。

43、第一虚拟化管理器将目标内存的地址信息提供给第二虚拟化管理器；第二虚拟化管理器用于对第一虚拟化管理器进行升级。

44、第二虚拟化管理器根据内存地址信息，获取虚拟实例状态信息。

45、第二虚拟化管理器根据虚拟实例状态信息，在第二虚拟化管理器中恢复虚拟实例的状态。

虚拟化管理器经常由于修复安全漏洞、功能问题等原因发布新的版本，这就需要对虚拟化管理器进行热升级。在虚拟化管理器需要升级时，可将对宿主机中当前运行的虚拟化管理器进行升级的新的虚拟化管理器对应的计算机程序加载至宿主机，得到新的虚拟化管理器。在本公开实施例中，为了便于描述和区分，将宿主机原先运行的虚拟化管理器，定义为第一虚拟化管理器；将对第一虚拟化管理器进行升级的新的虚拟化管理器，定义为第二虚拟化管理器。

在虚拟化管理器升级过程中，需要将第一虚拟化管理器中的虚拟实例状态信息，同步到第二虚拟化管理器，之后，由第二虚拟化管理器对宿主机上的虚拟实例进行管理。关于虚拟实例状态信息的描述可参见上述实施例的相关内容，在此不再赘述。

在虚拟化管理器升级过程中，在第二虚拟化管理器中恢复虚拟实例的状态越快，虚拟实例的暂停时间越短，虚拟实例上的用户应用的抖动越小。在本公开实施例中，为了降低虚拟实例的暂停时间，在步骤41中，第一虚拟化管理器可响应于虚拟化管理器升级事件，暂停宿主机上的虚拟实例的运行。关于虚拟化管理器升级事件的描述，可参见上述实施例的相关内容，在此不再赘述。

虚拟实例暂停运行之后，可将虚拟实例的状态迁移至第二虚拟化管理器，在第二虚拟化管理器恢复虚拟实例的运行，实现虚拟化管理器的热升级。基于此，在本实施例中，为了将虚拟实例的状态迁移至第二虚拟化管理器，在步骤42中，第一虚拟化管理器可将第一虚拟化管理器中的虚拟实例状态信息保存至宿主机的目标内存中。具体地，第一虚拟化管理器可在宿主机中申请目标内存；并将第一虚拟化管理器中的虚拟实例状态信息保存至目标内存中。进一步，在步骤43中，第一虚拟化管理器可将虚拟实例状态信息的内存地址信息提供给第二虚拟化管理器。虚拟实例状态信息的内存地址信息可为虚拟实例状态信息在内核态内存空间的起始地址及长度等。

相应地，在步骤44中，第二虚拟化管理器可获取目标内存的内存地址信息，并根据目标内存的内存地址信息，从目标内存中获取虚拟实例状态信息。进一步，在步骤45中，第二虚拟化管理器可根据虚拟实例状态信息，在第二虚拟化管理组件中恢复虚拟实例的状态，实现虚拟化管理器的热升级。

传统方案中虚拟实例状态保存和恢复过程中，虚拟实例状态信息需要在原虚拟化管理器和新的虚拟化管理器之间来回复制，还需要虚拟实例状态信息需要在用户态的虚拟化管理进程和内核态的虚拟化管理器之间来回复制，大量的数据复制耗时较长。本实施例中，虚拟实例状态保存和恢复过程中，原虚拟化管理器和新的虚拟化管理器之间传递的为虚拟实例状态信息的内存地址，减少了原虚拟化管理器和新的虚拟化管理器之间的复制的数据量，降低虚拟化管理器升级产生的虚拟实例暂停时间，进而降低虚拟实例上用户应用的抖动，保障服务性能。

在本公开实施例中，宿主机划分为用户态和内核态。相应地，虚拟化管理器包括：部署于用户态的虚拟化管理进程及部署于内核态的虚拟化管理组件。虚拟化管理器的热升级分为虚拟化管理进程的热升级和部署于内核态的虚拟化管理组件的热升级。下面重点针对部署于内核态的虚拟化管理组件的热升级进行介绍。

图4b为本公开实施例提供的另一虚拟化管理器的升级方法的流程示意图。如图4b所示，该方法主要包括：

401、第一虚拟化管理进程响应于虚拟化管理器升级事件，在用户态创建第二虚拟化管理进程；第二虚拟化管理进程与第二虚拟化管理组件协作。

402、第一虚拟化管理进程暂停宿主机上的虚拟实例的运行。

403、第一虚拟化管理组件响应于第一虚拟化管理进程发送的虚拟实例状态保存请求，将第一虚拟化管理组件中的虚拟实例状态信息保存至内核态的目标内存中。

404、第一虚拟化管理组件将目标内存的内存地址信息提供给第一虚拟化管理进程，以供第一虚拟化管理进程将内存地址信息提供给第二虚拟化管理进程。

405、第二虚拟化管理组件根据第二虚拟化管理进程发送的虚拟实例状态恢复请求中的内存地址信息，获取虚拟实例状态信息。

406、第二虚拟化管理组件根据虚拟实例状态信息，在第二虚拟化管理组件中恢复

虚拟实例的状态。

在运行在内核态的虚拟化管理器需要升级时，可将对内核态当前运行的虚拟化管
理组件进行升级的新的虚拟化组件对应的计算机程序加载至内核态，得到部署在
5 内核态的新的虚拟化组件。在本公开实施例中，为了便于描述和区分，将宿主机的
的内核态原先运行的虚拟化组件，定义为第一虚拟化组件；将对第一虚拟化
管理组件进行升级的新的虚拟化组件，定义为第二虚拟化组件。

在内核态的虚拟化组件升级过程中，需要将第一虚拟化组件中的虚拟实
例状态信息，同步到第二虚拟化组件，之后，由第二虚拟化组件对宿主机上
的虚拟实例进行管理。

10 在本公开实施例中，为了降低虚拟实例的暂停时间，在步骤 401 中，用户态的原
虚拟化进程响应于虚拟化组件升级事件，在用户态创建新的虚拟化进程。
在本公开实施例中，为了便于描述和区分，将用户态的原虚拟化进程，定义为第
一虚拟化进程；并将第一虚拟化进程创建的新的虚拟化进程，定义为第
二虚拟化进程。其中，第二虚拟化进程与第二虚拟化组件协作，对宿主
15 机上的虚拟实例进行管理。

在一些实施例中，第一虚拟化进程响应于虚拟化管理器升级事件，在用户态
创建第一虚拟化进程的子进程；并在子进程中加载虚拟化进程的二进制文件，
以得到第二虚拟化进程。

具体地，第一虚拟化进程响应于虚拟化管理器升级事件，调用复刻函数（如
20 fork 函数）创建第一虚拟化进程的子进程；并控制第一虚拟化进程调用 exec
函数在子进程中加载虚拟化进程的二进制文件，得到第二虚拟化进程。

其中，若宿主机的用户态存在新版本的二进制文件，则在子进程中加载的虚拟化
管理进程的二进制文件为新版本的二进制文件，此时的第二虚拟化进程为对第
一虚拟化进程升级后的虚拟化进程，虚拟化进程进行了升级。若宿主机的
25 用户态不存在新版本的二进制文件，则在子进程中加载的虚拟化进程的
二进制文件为原虚拟化进程（即第一虚拟化进程）对应的二进制文件，则虚拟化
进程未升级。

在本实施例中，第二虚拟化进程与内核态的第二虚拟化组件相互协作，
对虚拟实例进行管理。具体地，第二虚拟化进程打开第二虚拟化组件。进一
30 步，第二虚拟化进程在第二虚拟化组件中创建虚拟实例关联的结构体对象。

具体地，第二虚拟化进程可根据第二虚拟化组件的标识，获取第二虚拟
化组件向用户态提供的接口（如/dev/kvm 接口）。进一步，第二虚拟化进程
通过打开命令（如 open() 命令）打开第二虚拟化组件向用户态提供的接口（如
/dev/kvm 接口），来打开第二虚拟化组件，以获取第二虚拟管理进程返回的文件
35 句柄（File handle）。进一步，第二虚拟管理进程可根据文件句柄，在第二虚拟化管理

组件中创建虚拟实例关联的结构体对象。

具体地，第二虚拟管理进程通过 IOCTL 指令，在第二虚拟化管理组件中创建虚拟实例关联的结构体对象。该 IOCTL 指令包括：指示第二虚拟化管理组件执行的用于创建数据结构对象的命令及文件句柄。相应地，第二虚拟化管理组件响应 IOCTL 指令，
5 在第二虚拟化管理组件中创建虚拟实例关联的结构体对象。此时创建的虚拟实例关联的结构体对象，为虚拟实例对应的虚拟处理器及设备等的初始状态，可用来运行一个新的虚拟实例。

第二虚拟管理进程在虚拟实例关联的结构体对象创建完成之后，可向第一虚拟管理进程返回结构体对象创建完成消息。在虚拟实例关联的结构体对象完成之后，在步骤 402 中，第一虚拟管理进程可响应于虚拟化管理器升级事件，暂停宿主机上的虚拟实例的运行。上述步骤 41 可实现为本步骤 402。

虚拟实例暂停运行之后，可将虚拟实例的状态迁移至第二虚拟化管理组件，在第二虚拟化管理组件恢复虚拟实例的运行，实现内核态的虚拟化管理组件的热升级。基于此，在本实施例中，为了将虚拟实例的状态迁移至第二虚拟化管理组件，上述步骤
15 42 可实现为：第一虚拟化管理进程可在虚拟实例暂停运行之后，请求第一虚拟化管理组件保存虚拟实例的状态信息。具体地，在第一虚拟化管理进程可在虚拟实例暂停运行之后，第一虚拟化管理进程可向第一虚拟化管理组件发送虚拟实例状态保存请求。

相应地，在步骤 403 中，第一虚拟化管理组件响应于虚拟实例状态保存请求，将第一虚拟化管理组件中的虚拟实例状态信息保存至内核态的目标内存中。具体地，第一虚拟化管理组件响应于虚拟实例状态保存请求，在内核态申请目标内存；并将第一虚拟化管理组件中的虚拟实例状态信息保存至内核态的目标内存中。进一步，在步骤 404 中，第一虚拟化管理组件可将目标内存的内存地址信息提供给第一虚拟化管理进程。

在一些实施例中，在虚拟实例暂停运行之后，第一虚拟化管理进程可通过 IOCTL 指令向第一虚拟化管理组件发送虚拟实例状态保存请求。相应地，虚拟实例状态保存请求可实现为 IOCTL 指令，该 IOCTL 指令用于请求第一虚拟化管理组件保存虚拟实例状态信息。IOCTL 指令可包括：指示第一虚拟化管理组件保存虚拟实例状态信息的保存命令，如 save 命令，还可包括第一虚拟化管理组件提供的文件句柄 fd0 等。相应地，第一虚拟化管理组件响应该 IOCTL 指令，将第一虚拟化管理组件中的虚拟实例状态信息保存至内核态的目标内存中；并将虚拟实例状态信息的内存地址信息作为
30 IOCTL 指令的调用结果，返回给第一虚拟化管理进程。

可选地，第一虚拟化管理组件响应该 IOCTL 指令，对第一虚拟化管理组件中的虚拟实例状态信息进行序列化，以得到序列化的虚拟实例状态信息。进一步，第一虚拟化管理组件将序列化的虚拟实例状态信息的内存地址信息作为 IOCTL 指令的调用结果，返回给第一虚拟化管理进程。
35

进一步，第一虚拟化管理进程将目标内存的内存地址信息提供给第二虚拟化管理进程。第二虚拟化管理进程可基于目标内存的内存地址信息，请求第二虚拟化管理组件恢复虚拟实例状态。具体地，第二虚拟化管理进程可向第二虚拟化管理组件发送虚拟实例状态恢复请求，该虚拟实例状态恢复请求包含虚拟实例状态信息的内存地址信息。

相应地，在步骤 405 中，第二虚拟化管理组件响应于该虚拟实例状态恢复请求，根据该请求包含的目标内存的内存地址信息，从内核态的目标内存中获取虚拟实例状态信息。

在一些实施例中，第二虚拟化管理进程通过 IOCTL 指令向第二虚拟化管理组件发送虚拟实例状态恢复请求。相应地，虚拟实例状态恢复请求可实现为 IOCTL 指令，该 IOCTL 指令用于请求第二虚拟化管理组件恢复虚拟实例状态信息。IOCTL 指令可包括：指示第二虚拟化管理组件恢复虚拟实例状态信息的恢复命令，如 restore 命令，还可包括第二虚拟化管理组件提供的文件句柄 fd1 及虚拟实例状态信息的内存地址信息等。

相应地，第二虚拟化管理组件响应该 IOCTL 指令，获取目标内存的内存地址信息，并根据该请求包含的目标内存的内存地址信息，从内核态的目标内存中获取虚拟实例状态信息。

对于上述虚拟实例状态信息以序列化的形式存储在目标内存中的实施例，可控制第二虚拟化管理组件根据序列化的虚拟实例状态信息的内存地址信息，从内核态的目标内存中获取序列化的虚拟实例状态信息；并对序列化的虚拟实例状态信息进行反序列化处理，以得到虚拟实例状态信息。

在本实施例中，由内核态的虚拟化管理组件可直接在内核态，对虚拟实例状态信息直接进行序列化和反序列化处理，跳过了用户态的虚拟化管理进程在用户态对虚拟实例状态信息直接进行序列化和反序列化处理的过程，减少了虚拟实例状态信息的处理时长，可降低内核态的虚拟化管理组件热升级过程中虚拟实例的暂停时长，可降低虚拟实例上用户应用的抖动，保障服务性能。

进一步，在步骤 406 中，第二虚拟化管理组件可根据虚拟实例状态信息，在第二虚拟化管理组件中恢复虚拟实例的状态，实现内核态的虚拟化管理组件的热升级。

具体地，第二虚拟化管理组件根据虚拟实例状态信息，对第二虚拟化管理组件中的结构体对象进行赋值，从而在第二虚拟化管理组件中恢复虚拟实例的状态。

在本实施例中，在内核态的虚拟化管理组件升级过程中，用户态的原虚拟化管理进程向原虚拟化管理组件发送虚拟实例状态保存请求，原虚拟化管理组件响应该请求，将虚拟实例状态信息保存至内核态的内存空间，并返回虚拟实例状态信息的内存地址，实现了原虚拟化管理进程通过一次系统调用保存虚拟实例状态。相较于传统方案中通过大量 IOCTL 类型的系统调用实现虚拟实例状态的保存方式，本实施例可减少虚拟实例状态保存过程中系统调用次数，从而降低虚拟实例状态保存过程中系统调用的时间。

开销，有助于降低虚拟化管理组件升级产生的虚拟实例暂停时间，可降低虚拟实例上用户应用的抖动，保障服务性能。

另一方面，在本实施例中，用户态的新虚拟化管理进程向新虚拟化管理组件发送包含虚拟实例状态信息的内存地址的虚拟实例状态恢复请求，新虚拟化管理组件可根据虚拟实例状态信息的内存地址，获取虚拟实例状态信息，并在新虚拟化管理组件中恢复虚拟实例状态，实现了新虚拟化管理进程通过一次系统调用恢复虚拟实例状态。相较于传统方案中通过大量 IOCTL 类型的系统调用实现恢复的方式，本实施例可减少虚拟实例状态恢复过程中系统调用次数，从而降低虚拟实例状态恢复过程中系统调用的时间开销，有助于降低虚拟化管理组件升级产生的虚拟实例暂停时间，可进一步降低虚拟实例上用户应用的抖动，保障服务性能。

再者，传统方案中虚拟实例状态保存和恢复过程中，虚拟实例状态信息需要在用户态的虚拟化管理进程和内核态的虚拟化管理组件之间来回复制，大量的数据复制耗时较长。本实施例中，虚拟实例状态保存和恢复过程中，用户态的虚拟化管理进程和内核态的虚拟化管理组件之间传递的为虚拟实例状态信息的内存地址，减少了用户态的虚拟化管理进程和内核态的虚拟化管理器之间的复制的数据量，可进一步降低虚拟化管理组件升级产生的虚拟实例暂停时间，可降低虚拟实例上用户应用的抖动，保障服务性能。

除了上述内核态的虚拟化管理组件热升级之外，本公开实施例还提供用户态的虚拟化管理进程升级与内核态的虚拟化管理组件升级解耦，可在内核态的虚拟化管理组件热升级时，灵活选择是否升级用户态的虚拟化管理进程。例如，可向用户提供用户态的虚拟化管理进程是否升级的选择接口，由用户自主选择是否升级虚拟化管理进程。

在本实施例中，在恢复虚拟实例的状态之后，第二虚拟化管理组件可向第二虚拟化管理进程返回恢复完成消息。可选地，第二虚拟化管理组件将恢复完成消息，作为上述用于请求第二虚拟化管理组件恢复虚拟实例状态信息的 IOCTL 指令的调用结果，返回给第二虚拟化管理进程。对于选择升级用户态的虚拟化管理进程的实施例，第二虚拟化管理进程可响应于恢复完成消息，恢复宿主机上虚拟实例的运行。

对于选择不升级用户态的虚拟化管理进程的实施例，由于升级后的第二虚拟化管理组件的文件句柄是第二虚拟化管理进程打开的，原第一虚拟化管理进程需要获得第二虚拟化管理组件的操作句柄，才可操作升级后的第二虚拟化管理组件。基于此，第二虚拟化管理进程响应于恢复完成消息，将其打开的目标文件句柄提供给第一虚拟化管理进程，并释放第二虚拟化管理进程占用的资源。

在第一虚拟化管理进程在获取目标文件句柄之后，第一虚拟化管理进程释放第一虚拟化管理组件的文件句柄，并基于第二虚拟化管理组件的目标文件句柄继续运行。具体地，第一虚拟化管理进程基于目标文件句柄，在第二虚拟化管理组件中恢复宿主机上虚拟实例的运行。

具体地，第一虚拟化管理进程基于目标文件句柄，向第二虚拟化管理组件发送请求运行虚拟实例的 IOCTL 指令。该 IOCTL 指令可包括请求运行虚拟实例的命令，如 run 命令。进一步，第二虚拟化管理组件响应该 IOCTL 指令，在第二虚拟化管理组件中恢复宿主机上虚拟实例的运行。

5 在本实施例中，通过将新虚拟化管理进程打开的文件句柄返回给原虚拟化管理进程，可实现仅升级内核态的虚拟化管理组件，而保持用户态的原虚拟化管理进程，提高了虚拟化管理器运维的灵活性。

需要说明的是，上述实施例所提供方法的各步骤的执行主体均可以是同一设备，或者，该方法也由不同设备作为执行主体。比如，步骤 401 和 402 的执行主体可以为设备 A；又比如，步骤 401 的执行主体可以为设备 A，步骤 402 的执行主体可以为设备 B；等等。

另外，在上述实施例及附图中的描述的一些流程中，包含了按照特定顺序出现的多个操作，但是应该清楚了解，这些操作可以不按照其在本文中出现的顺序来执行或并行执行，操作的序号如 401、402 等，仅仅是用于区分开各个不同的操作，序号本身不代表任何的执行顺序。另外，这些流程可以包括更多或更少的操作，并且这些操作可以按顺序执行或并行执行。

相应地，本公开实施例还提供一种存储有计算机指令的计算机可读存储介质，当计算机指令被一个或多个处理器执行时，致使一个或多个处理器执行上述虚拟化管理器的升级方法中的步骤。

20 图 5 为本公开实施例提供的计算设备的结构示意图。该计算设备可实现为宿主机。如图 5 所示，计算设备可包括：存储器 50a 和处理器 50b。其中，存储器 50a 用于存储上述第一虚拟化管理器及第二虚拟化管理器对应的计算机程序，以及其它计算机程序。在本实施例中，处理器 50b 耦合至存储器 50a 用于执行上述存储器 50a 中存储的计算机程序，以用于执行上述虚拟化管理器的升级方法中的步骤。关于各方法步骤
25 详细描述可参见前述实施例，在此不在赘述。

在一些实施例中，宿主机的内核态运行有第一虚拟化管理组件及对第一虚拟化管理组件进行升级的虚拟化管理器的第二虚拟化管理组件；宿主机的用户态运行有与第一虚拟化管理组件协作的第一虚拟化管理进程。存储器 50a 用于存储上述第一虚拟化管理组件、第二虚拟化管理组件及第一虚拟化管理进程对应的计算机程序，以及其它
30 计算机程序。

在本实施例中，处理器 50b 耦合至存储器 50a 用于执行上述计算机程序，以用于执行上述图 4b 及其具体实施方式示出的虚拟化管理器的升级方法中的步骤。关于各方法步骤的详细描述可参见前述实施例，在此不在赘述。

在一些可选实施方式中，如图 5 所示，该计算设备还可以包括：通信组件 50c、
35 电源组件 50d、显示组件 50e 及音频组件 50f 等可选组件。图 5 中仅示意性给出部分组

件，并不意味着计算设备必须包含图 5 所示全部组件，也不意味着计算设备只能包括图 5 所示组件。

另外，图 5 中虚线框内的组件为可选组件，而非必选组件，具体可视电子设备的
5 产品形态而定。本实施例的计算设备可以实现为台式电脑、笔记本电脑、手机或物联网设备等终端设备；也可以是传统服务器、云服务器或服务器集群等各种服务器设备。

在本公开实施例中，存储器用于存储计算机程序，并可被配置为存储其它各种数据以支持在其所在设备上的操作。其中，处理器可执行存储器中存储的计算机程序，以实现相应控制逻辑。存储器可以由任何类型的易失性或非易失性存储设备或者它们的组合实现，如静态随机存取存储器（Static Random-Access Memory, SRAM），电可
10 擦除可编程只读存储器（Electrically Erasable Programmable Read Only Memory, EEPROM），可擦除可编程只读存储器（Electrical Programmable Read Only Memory, EPROM），可编程只读存储器（Programmable Read Only Memory, PROM），只读存储器（Read Only Memory, ROM），磁存储器，快闪存储器，磁盘或光盘。

在本公开实施例中，处理器可以为任意可执行上述方法逻辑的硬件处理设备。可
15 选地，处理器可以为中央处理器（Central Processing Unit, CPU）、图形处理器（Graphics Processing Unit, GPU）或微控制单元（Microcontroller Unit, MCU）；也可以为现场可编程门阵列（Field-Programmable Gate Array, FPGA）、可编程阵列逻辑器件

（Programmable Array Logic, PAL）、通用阵列逻辑器件（General Array Logic, GAL）、复杂可编程逻辑器件（Complex Programmable Logic Device, CPLD）等可编程器件；
20 或者为先进精简指令集（Reduced Instruction Set Compute, RISC）处理器（Advanced RISC Machines, ARM）或系统芯片（System on Chip, SoC）等等，但不限于此。

在本公开实施例中，通信组件被配置为便于其所在设备和其他设备之间有线或无线方式的通信。通信组件所在设备可以接入基于通信标准的无线网络，如无线保真（Wireless Fidelity, WiFi），2G 或 3G，4G，5G 或它们的组合。在一个示例性实施例中，通信组件经由广播信道接收来自外部广播管理系统的广播信号或广播相关信息。
25 在一个示例性实施例中，通信组件还可基于近场通信（Near Field Communication, NFC）技术、射频识别（Radio Frequency Identification, RFID）技术、红外数据协会（Infrared Data Association, IrDA）技术、超宽带（Ultra Wide Band, UWB）技术、蓝牙（Bluetooth, BT）技术或其他技术来实现。

在本公开实施例中，显示组件可以包括液晶显示器（Liquid Crystal Display, LCD）和触摸面板（Touch Panel, TP）。如果显示组件包括触摸面板，显示组件可以被实现为触摸屏，以接收来自用户的输入信号。触摸面板包括一个或多个触摸传感器以感测触摸、滑动和触摸面板上的手势。触摸传感器可以不仅感测触摸或滑动动作的边界，而且还检测与触摸或滑动操作相关的持续时间和压力。
30

在本公开实施例中，电源组件被配置为其所在设备的各种组件提供电力。电源组
35

件可以包括电源管理系统，一个或多个电源，及其他与为电源组件所在设备生成、管理和分配电力相关联的组件。

在本公开实施例中，音频组件可被配置为输出和/或输入音频信号。例如，音频组件包括一个麦克风（Microphone，MIC），当音频组件所在设备处于操作模式，如呼叫模式、记录模式和语音识别模式时，麦克风被配置为接收外部音频信号。所接收的音频信号可以被进一步存储在存储器或经由通信组件发送。在一些实施例中，音频组件还包括一个扬声器，用于输出音频信号。例如，对于具有语言交互功能的设备，可通过音频组件实现与用户的语音交互等。

需要说明的是，本公开所涉及的用户信息（包括但不限于用户设备信息、用户个人信息等）和数据（包括但不限于用于分析的数据、存储的数据、展示的数据等），均为经用户授权或者经过各方充分授权的信息和数据，并且相关数据的收集、使用和处理需要遵守相关国家和地区的相关法律法规和标准，并提供有相应的操作入口，供用户选择授权或者拒绝。

还需要说明的是，本文中的“第一”、“第二”等描述，是用于区分不同的消息、设备、模块等，不代表先后顺序，也不限定“第一”和“第二”是不同的类型。

本领域内的技术人员应明白，本公开的实施例可提供为方法、系统、或计算机程序产品。因此，本公开可采用完全硬件实施例、完全软件实施例、或结合软件和硬件方面的实施例的形式。而且，本公开可采用在一个或多个其中包含有计算机可用程序代码的计算机可用存储介质（包括但不限于磁盘存储器、只读光盘（Compact Disc Read-Only Memory，CD-ROM）、光学存储器等）上实施的计算机程序产品的形式。

本公开是参照根据本公开实施例的方法、设备（系统）、和计算机程序产品的流程图和/或方框图来描述的。应理解可由计算机程序指令实现流程图和/或方框图中的每一流程和/或方框、以及流程图和/或方框图中的流程和/或方框的结合。可提供这些计算机程序指令到通用计算机、专用计算机、嵌入式处理机或其他可编程数据处理设备的处理器以产生一个机器，使得通过计算机或其他可编程数据处理设备的处理器执行的指令产生用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的装置。

这些计算机程序指令也可存储在能引导计算机或其他可编程数据处理设备以特定方式工作的计算机可读存储器中，使得存储在该计算机可读存储器中的指令产生包括指令装置的制造品，该指令装置实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能。

这些计算机程序指令也可装载到计算机或其他可编程数据处理设备上，使得在计算机或其他可编程设备上执行一系列操作步骤以产生计算机实现的处理，从而在计算机或其他可编程设备上执行的指令提供用于实现在流程图一个流程或多个流程和/或方框图一个方框或多个方框中指定的功能的步骤。

在一个典型的配置中，计算设备包括一个或多个处理器 (CPU 等)、输入/输出接口、网络接口和内存。

内存可能包括计算机可读介质中的非永久性存储器，随机存取存储器 (Random-Access Memory, RAM) 和/或非易失性内存等形式，如只读存储器 (ROM) 或闪存(flash RAM)。内存是计算机可读介质的示例。

计算机的存储介质为可读存储介质，也可称为可读介质。可读存储介质包括永久性和非永久性、可移动和非可移动媒体可以由任何方法或技术来实现信息存储。信息可以是计算机可读指令、数据结构、程序的模块或其他数据。计算机的存储介质的例子包括，但不限于相变内存 (Phase-Change Memory, PRAM)、静态随机存取存储器 (SRAM)、动态随机存取存储器 (Dynamic Random Access Memory, DRAM)、其他类型的随机存取存储器 (RAM)、只读存储器 (ROM)、电可擦除可编程只读存储器 (EEPROM)、快闪记忆体或其他内存技术、只读光盘只读存储器 (CD-ROM)、数字多功能光盘 (Digital Video Disc, DVD) 或其他光学存储、磁盒式磁带，磁盘存储或其他磁性存储设备或任何其他非传输介质，可用于存储可以被计算设备访问的信息。按照本文中的界定，计算机可读介质不包括暂存电脑可读媒体 (transitory media)，如调制的数据信号和载波。

还需要说明的是，术语“包括”、“包含”或者其任何其他变体意在涵盖非排他性的包含，从而使得包括一系列要素的过程、方法、商品或者设备不仅包括那些要素，而且还包括没有明确列出的其他要素，或者是还包括为这种过程、方法、商品或者设备所固有的要素。在没有更多限制的情况下，由语句“包括一个……”限定的要素，并不排除在包括上述要素的过程、方法、商品或者设备中还存在另外的相同要素。

以上内容仅为本公开的实施例而已，并不用于限制本公开。对于本领域技术人员来说，本公开可以有各种更改和变化。凡在本公开的精神和原理之内所作的任何修改、等同替换、改进等，均应包含在本公开的权利要求范围之内。

权 利 要 求 书

- 1、一种虚拟化管理器的升级方法，其中，包括：
- 第一虚拟化管理器响应于虚拟化管理器升级事件，暂停宿主机上虚拟实例的运行；
- 所述第一虚拟化管理器将所述第一虚拟化管理器中的虚拟实例状态信息保存至所述宿主机
- 5 所述宿主机的目标内存中；所述虚拟实例状态信息为所述虚拟实例的状态信息；
- 所述第一虚拟化管理器将所述目标内存的地址信息提供给第二虚拟化管理器；所述第二虚拟化管理器用于对所述第一虚拟化管理器进行升级；
- 所述第二虚拟化管理器根据所述内存地址信息，获取所述虚拟实例状态信息；并根据所述虚拟实例状态信息，在所述第二虚拟化管理器中恢复所述虚拟实例的状态。
- 10 2、根据权利要求1所述的方法，其中，所述第一虚拟化管理器包括：运行于用户态的第一虚拟化管理进程；
- 所述第一虚拟化管理器响应于虚拟化管理器升级事件，暂停宿主机上虚拟实例的运行，包括：
- 所述第一虚拟化管理进程，响应于虚拟化管理器升级事件，暂停宿主机上虚拟实例的运行。
- 15 3、根据权利要求2所述的方法，其中，所述第一虚拟化管理器还包括：运行于内核态的第一虚拟化管理组件；
- 所述第一虚拟化管理器将所述第一虚拟化管理器中的虚拟实例状态信息保存至所述宿主机的目标内存中，包括：
- 20 所述第一虚拟化管理进程向所述第一虚拟化管理组件发送虚拟实例状态保存请求；
- 所述第一虚拟化管理组件响应于所述虚拟实例状态保存请求，将所述第一虚拟化管理组件中的虚拟实例状态信息保存至所述内核态的目标内存中。
- 4、根据权利要求3所述的方法，其中，所述内核态还运行有第二虚拟化管理组件；所述第二虚拟化管理组件为对所述第一虚拟化管理组件进行升级的组件，所述第二虚拟化管理器包括：所述第二虚拟化管理组件；
- 25 所述方法还包括：
- 所述第一虚拟化管理进程，响应于虚拟化管理器升级事件，在所述用户态创建第二虚拟化管理进程；所述第二虚拟化管理器还包括所述第二虚拟化管理进程；所述第二虚拟化管理进程与所述第二虚拟化管理组件协作；
- 30 所述第一虚拟化管理组件将所述目标内存的内存地址信息提供给所述第一虚拟化管理进程；
- 所述第一虚拟化管理进程将所述目标内存的内存地址信息提供给所述第二虚拟化管理进程。
- 5、根据权利要求4所述的方法，其中，所述第二虚拟化管理器根据所述内存地址信息，获取所述虚拟实例状态信息，包括：
- 35

所述第二虚拟化管理进程向所述第二虚拟化管理组件发送虚拟实例状态恢复请求；
所述虚拟实例状态恢复请求包含所述目标内存的内存地址信息；

所述第二虚拟化管理组件响应于所述虚拟实例状态恢复请求，根据所述目标内存的内存地址信息，获取所述虚拟实例状态信息；

5 所述根据所述虚拟实例状态信息，在所述第二虚拟化管理器中恢复所述虚拟实例的状态，包括：

所述第二虚拟化管理组件根据所述虚拟实例状态信息，在所述第二虚拟化管理组件中恢复所述虚拟实例的状态。

6、根据权利要求4所述的方法，其中，所述第一虚拟化管理进程响应于虚拟化管理器升级事件，在所述用户态创建第二虚拟化管理进程，包括：

所述第一虚拟化管理进程响应于所述虚拟化管理器升级事件，在所述用户态创建所述第一虚拟化管理进程的子进程；并在所述子进程中加载虚拟化管理进程的二进制文件，以得到所述第二虚拟化管理进程。

7、根据权利要求5所述的方法，其中，还包括：

15 所述第二虚拟化管理进程打开所述第二虚拟化管理组件，并在所述第二虚拟化管理组件中创建虚拟实例关联的结构体对象；

所述第二虚拟化管理组件根据所述虚拟实例状态信息，在所述第二虚拟化管理组件中恢复所述虚拟实例的状态，包括：

20 所述第二虚拟化管理组件根据所述虚拟实例状态信息，对所述结构体对象进行赋值，以在所述第二虚拟化管理组件中恢复所述虚拟实例的状态。

8、根据权利要求7所述的方法，其中，所述第二虚拟化管理进程打开所述第二虚拟化管理组件，并在所述第二虚拟化管理组件中创建虚拟实例关联的结构体对象，包括：

25 所述第二虚拟化管理进程打开所述第二虚拟化管理组件，以获得所述第二虚拟化管理组件提供的文件句柄；

所述第二虚拟化管理进程根据所述文件句柄，在所述第二虚拟化管理组件中创建虚拟实例关联的结构体对象。

9、根据权利要求8所述的方法，其中，所述第二虚拟化管理进程根据所述文件句柄，在所述第二虚拟化管理组件中创建虚拟实例关联的结构体对象，包括：

30 所述第二虚拟化管理进程根据所述文件句柄，向所述第二虚拟化管理组件发送用于请求所述第二虚拟化管理组件创建结构体对象的第一设备输入输出控制 IOCTL 指令；

所述第二虚拟化管理组件响应第一 IOCTL 指令，创建所述虚拟实例关联的结构体对象。

10、根据权利要求5所述的方法，其中，所述第一虚拟化管理组件发送虚拟实例状态保存请求，包括：

35

所述第一虚拟化管理进程向所述第一虚拟化管理组件发送用于请求所述第一虚拟化管理组件保存虚拟实例状态信息的第二 IOCTL 指令，作为所述虚拟实例状态保存请求；

5 所述第二虚拟化管理进程向所述第二虚拟化管理组件发送虚拟实例状态恢复请求，包括：

所述第二虚拟化管理进程向所述第二虚拟化管理进程，发送用于请求所述第二虚拟化管理组件恢复所述第一虚拟化管理组件中的虚拟实例的状态的第三 IOCTL 指令，作为所述虚拟实例状态恢复请求；所述第三 IOCTL 指令包括所述虚拟实例状态信息的内存地址信息。

10 11、根据权利要求 5 所述的方法，其中，所述、第一虚拟化管理组件响应所述虚拟实例状态保存请求，将所述第一虚拟化管理组件中的虚拟实例状态信息保存至所述内核态的目标内存中，包括：

所述第一虚拟化管理组件响应所述第一虚拟化管理进程发送的虚拟实例状态保存请求，获取所述第一虚拟化管理组件中的虚拟实例状态信息；

15 所述第一虚拟化管理组件对所述第一虚拟化管理组件中的虚拟实例状态信息进行序列化处理，得到序列化的虚拟实例状态信息；以及，

所述第一虚拟化管理组件将所述序列化的虚拟实例状态信息保存至所述内核态的目标内存中。

20 12、根据权利要求 11 所述的方法，其中，所述第二虚拟化管理组件响应于所述虚拟实例状态恢复请求，根据所述目标内存的内存地址信息，获取所述虚拟实例状态信息，包括：

所述第二虚拟化管理组件响应于所述虚拟实例状态恢复请求，根据所述目标内存的内存地址信息，获取所述序列化的虚拟实例状态信息；

25 所述第二虚拟化管理组件对所述序列化的虚拟实例状态信息进行反序列化处理，以得到所述虚拟实例状态信息。

13、根据权利要求 5 或 7-12 任一项所述的方法，其中，在所述第二虚拟化管理组件根据所述虚拟实例状态信息，在所述第二虚拟化管理组件中恢复所述虚拟实例的状态之后，所述方法还包括：

30 所述第二虚拟化管理进程将其打开的目标文件句柄提供给所述第一虚拟化管理进程，并释放所述第二虚拟化管理进程占用的资源；

所述第一虚拟化管理进程基于所述目标文件句柄，在所述第二虚拟化管理组件中恢复虚拟实例的运行。

14、根据权利要求 13 所述的方法，其中，所述第一虚拟化管理进程基于所述目标文件句柄，所述第二虚拟化管理组件中恢复虚拟实例的运行，包括：

35 所述第一虚拟化管理进程基于所述目标文件句柄，向所述第二虚拟化管理组件发

送请求运行虚拟实例的第四 IOCTL 指令；

所述第二虚拟化管理组件响应所述第四 IOCTL 指令，在所述第二虚拟化管理组件中恢复虚拟实例的运行。

15、一种计算设备，其中，所述计算设备可实现为宿主机；

5 所述计算设备包括：存储器和处理器；其中，所述存储器，用于存储第一虚拟化管理器及第二虚拟化管理器对应的计算机程序，以及其它计算机程序；

所述处理器耦合至所述存储器，用于执行所述存储器中存储的计算机程序以用于执行权利要求 1-14 任一项所述方法中的步骤。

10 16、一种存储有计算机指令的计算机可读存储介质，其中，当所述计算机指令被一个或多个处理器执行时，致使所述一个或多个处理器执行权利要求 1-14 任一项所述方法中的步骤。

17、一种计算机程序，其中，包括计算机指令，所述计算机指令被处理器执行时实现权利要求 1-14 任意一项所述方法的步骤。