(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 109213573 A
(43)申请公布日 2019.01.15

(21)申请号 201811073054.5

(22)申请日 2018.09.14

(71)申请人 珠海国芯云科技有限公司
地址 519000 广东省珠海市香洲区吉大景山路莲山巷8号正方云创园九层

(72)发明人 杨立群

(74)专利代理机构 广州嘉权专利商标事务所有限公司 44205
代理人 俞梁清

(51)Int.Cl.
G06F 9/455(2006.01)

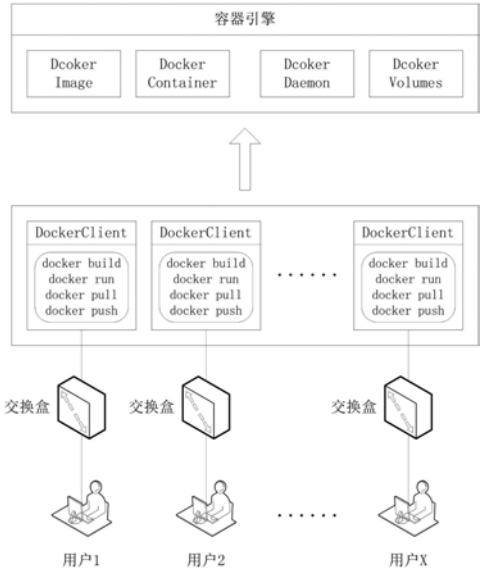
权利要求书2页 说明书10页 附图7页

(54)发明名称

基于容器的虚拟桌面的设备隔离方法及装置

(57)摘要

一种基于容器的虚拟桌面的设备隔离方法，包括以下步骤：针对新添加到服务器的硬件资源，为所述硬件资源标识其至少一个能力的资源属性；利用所标识的资源属性，将所述硬件资源与多个硬件资源池中的一个硬件资源池相关联，使得硬件资源池中的每一个硬件资源与多个资源属性中的一个相关联；分别为多个硬件资源池中的每一个硬件资源分配一个唯一识别的ID；通过容器引擎，依据每一个硬件资源的ID将硬件资源池中硬件资源分配到相应的容器和从相应的容器回收硬件资源到硬件资源池中。本申请还公开了一种对应的用于基于容器的虚拟桌面的设备隔离装置。



CN 109213573 A

1.一种用于基于容器的虚拟桌面的设备隔离方法,适用于在服务器内建立的多个彼此隔离的容器,其中多个所述容器中的每一个分别创建一个独立的虚拟桌面实例,并通过每个容器的虚拟桌面实例为对应的容器配置独立的文件管理结构,所述方法包括以下步骤:

S100)针对新添加到服务器的硬件资源,为所述硬件资源标识其至少一个能力的资源属性;

S200)利用所标识的资源属性,将所述硬件资源与多个硬件资源池中的一个硬件资源池相关联,使得硬件资源池中的每一个硬件资源与多个资源属性中的一个相关联;

S300)分别为多个硬件资源池中的每一个硬件资源分配一个唯一识别的ID;以及

S400)通过容器引擎,依据每一个硬件资源的ID将硬件资源池中硬件资源分配到相应的容器和从相应的容器回收硬件资源到硬件资源池中。

2.根据权利要求1所述的设备隔离方法,其特征在于,步骤S400还包括如下子步骤:

S401)依据硬件资源的ID将对应的硬件资源分配到相应的容器;

S402)根据硬件资源的ID上锁以分配的硬件资源,从而禁止其他容器访问对应ID的硬件资源;

S403)从容器回收硬件资源到硬件资源池中,并将根据硬件资源的ID解锁对应的硬件资源。

3.根据权利要求1所述的设备隔离方法,其特征在于,步骤S400还包括如下子步骤:

S404)为各个硬件资源池设置相关联的触发事件;

S405)确认分配给各个容器的硬件资源所对应的硬件资源池,从而基于所述硬件资源池的触发事件监控各个容器内的对应硬件资源;

S406)当检测到触发事件发生时,从容器中回收相应的硬件资源到对应的硬件资源池中。

4.根据权利要求3所述的设备隔离方法,其特征在于,所述触发事件是硬件资源的使用频率低于预设的阈值,其中所述阈值是在设置触发事件时被初始化的,并根据硬件资源的平均使用频率和/或使用频率上限自适应地调整。

5.根据权利要求3所述的设备隔离方法,其特征在于,所述触发事件是申请硬件资源的应用程序的进程结束。

6.根据权利要求1所述的设备隔离方法,其特征在于,在容器中运行的应用程序通过分配给该容器的硬件资源的ID访问该硬件资源。

7.根据权利要求6所述的设备隔离方法,其特征在于,在容器中运行的应用程序通过分配给该容器的硬件资源的ID访问该硬件资源包括以下子步骤:

S601)从硬件资源池分配硬件资源到相应的容器,并将所述硬件资源的ID记录在容器的硬件资源表中;

S602)搜索硬件资源表,当硬件资源表中存在相应资源属性时,容器中运行的应用程序根据硬件资源的ID访问对应的硬件资源,否则通过容器引擎向与该资源属性相关联的硬件资源池申请;

S603)当硬件资源从容器回收到相应的硬件资源池时,删除硬件资源表中对应的ID。

8.根据权利要求1所述的设备隔离方法,其特征在于,还包括以下步骤:

S500)按照预设的频率定期检测各个硬件资源池,并将检测到已被移除的硬件资源从

对应的硬件资源池以及对应的容器中移除。

9.一种用于基于容器的虚拟桌面的设备隔离装置，适用于在服务器内建立的多个彼此隔离的容器，其中多个所述容器中的每一个分别创建一个独立的虚拟桌面实例，并通过每个容器的虚拟桌面实例为对应的容器配置独立的文件管理结构，所述装置包括以下模块：

标识模块，用于针对新添加到服务器的硬件资源，为所述硬件资源标识其至少一个能力的资源属性；

关联模块，用于利用所标识的资源属性，将所述硬件资源与多个硬件资源池中的一个硬件资源池相关联，使得硬件资源池中的每一个硬件资源与多个资源属性中的一个相关联；

分配模块，用于分别为多个硬件资源池中的每一个硬件资源分配一个唯一识别的ID；以及

管理模块，用于通过容器引擎，依据每一个硬件资源的ID将硬件资源池中硬件资源分配到相应的容器和从相应的容器回收硬件资源到硬件资源池中。

10.一种计算机可读存储介质，其上存储有计算机指令，该指令所执行方法适用于在服务器内建立的多个彼此隔离的容器，其中多个所述容器中的每一个分别创建一个独立的虚拟桌面实例，并通过每个容器的虚拟桌面实例为对应的容器配置独立的文件管理结构，其特征在于该指令被处理器执行时实现如权利要求1至8中任一项所述的方法的步骤。

基于容器的虚拟桌面的设备隔离方法及装置

技术领域

[0001] 本发明涉及计算机技术领域，尤其涉及一种基于容器的虚拟桌面的设备隔离方法及装置。

背景技术

[0002] 操作系统虚拟化通过对真实的计算机硬件设备进行模拟，使得多名用户可以分别通过虚拟化后的操作系统共享使用硬件资源，从而可以高效率地利用硬件资源。同时，由于操作系统虚拟化使得各应用程序及其关联性被隔离，从而每位用户如同在个人计算机上操作独立的操作系统。

[0003] 例如，作为操作系统虚拟化的典型例子，虚拟机通常运行在诸如服务器等底层硬件的主机设备上，并通过虚拟机管理程序分配内存和CPU等硬件资源到多台虚拟机上。因此，可以根据当前各台虚拟机所请求的计算机硬件资源，动态地进行分配。然而，为了给各位用户提供一个完整独立的应用程序运行环境，虚拟机需要同时包含完整的虚拟硬件堆栈(包括虚拟的网络适配器、存储以及CPU等)。这意味着其自身也拥有完整的操作系统，并针对硬件资源抽象化而进行隔离，从而需要将部分由主机设备提供的共享资源占据为自身的专用资源而降低主机设备的总体性能。

[0004] 相反，作为操作系统虚拟化的另一个例子，容器是一种轻量级的操作系统虚拟化技术。各个容器通过共享主机设备上的系统内核以实现自身的轻量化，并利用进程访问控制隔离技术和进程组管理控制技术等方式隔离各自的用户空间，从而使得多套容器能够并行地运行在同一主机设备上。可是，在现有的技术方案中，由于多个容器之间共享同一套系统内核，使得可以通过将硬件资源的相关驱动程序安装在主机设备上的系统，从而方便地供各个容器使用。因此，容器间的隔离只是通过路径的转换或者访问权限控制策略等方式实现，并没有真正地相互隔离，从而使得容器之间的相互隔离非常薄弱。这使得容器之间存在明显的安全漏洞和较大的攻击面。在极端的情况下，具有高权限的用户甚至可以向系统内核发出系统调用，突破容器之间的隔离而任意调用分配到其他容器内的硬件资源(例如访问其他容器的摄像头并进行拍摄)。

发明内容

[0005] 本申请的目的是解决现有技术的不足，提供一种用于基于容器的虚拟桌面的设备隔离方法及装置，通过利用对各个硬件资源上标识唯一识别的ID，能够获得实现容器自身的轻量化同时，避免不同容器之间的硬件资源被恶意调用的效果。

[0006] 为了实现上述目的，本申请采用以下的技术方案。

[0007] 首先，本申请提出一种用于基于容器的虚拟桌面的设备隔离方法。该设备隔离方法适用于在服务器内建立的多个彼此隔离的容器。其中，多个容器中的每一个分别创建一个独立的虚拟桌面实例，并通过每个容器的虚拟桌面实例为对应的容器配置独立的文件管理结构。上述设备隔离方法包括以下步骤：

[0008] S100)针对新添加到服务器的硬件资源,为所述硬件资源标识其至少一个能力的资源属性;

[0009] S200)利用所标识的资源属性,将所述硬件资源与多个硬件资源池中的一个硬件资源池相关联,使得硬件资源池中的每一个硬件资源与多个资源属性中的一个相关联;

[0010] S300)分别为多个硬件资源池中的每一个硬件资源分配一个唯一识别的ID;以及

[0011] S400)通过容器引擎,依据每一个硬件资源的ID将硬件资源池中硬件资源分配到相应的容器和从相应的容器回收硬件资源到硬件资源池中。

[0012] 进一步地,在本申请的上述方法中,步骤S400还包括如下子步骤:

[0013] S401)依据硬件资源的ID将对应的硬件资源分配到相应的容器;

[0014] S402)根据硬件资源的ID上锁以分配的硬件资源,从而禁止其他容器访问对应ID的硬件资源;

[0015] S403)从容器回收硬件资源到硬件资源池中,并将根据硬件资源的ID解锁对应的硬件资源。

[0016] 可替代地,在本申请的上述方法中,步骤S400)可包括如下子步骤:

[0017] S404)为各个硬件资源池设置相关联的触发事件;

[0018] S405)确认分配给各个容器的硬件资源所对应的硬件资源池,从而基于所述硬件资源池的触发事件监控各个容器内的对应硬件资源;

[0019] S406)当检测到触发事件发生时,从容器中回收相应的硬件资源到对应的硬件资源池中。

[0020] 再进一步地,在本申请的上述方法中,该触发事件是硬件资源的使用频率低于预设的阈值,其中该阈值是在设置触发事件时被初始化的,并根据硬件资源的平均使用频率和/或使用频率上限自适应地调整。

[0021] 可替代地,在本申请的上述方法中,该触发事件是申请硬件资源的应用程序的进程结束。

[0022] 进一步地,在本申请的上述方法中,在容器中运行的应用程序通过分配给该容器的硬件资源的ID访问该硬件资源。

[0023] 再进一步地,在本申请的上述方法中,在容器中运行的应用程序通过分配给该容器的硬件资源的ID访问该硬件资源包括以下子步骤:

[0024] S601)从硬件资源池分配硬件资源到相应的容器,并将所述硬件资源的ID记录在容器的硬件资源表中;

[0025] S602)搜索硬件资源表,当硬件资源表中存在相应资源属性时,容器中运行的应用程序根据硬件资源的ID访问对应的硬件资源,否则通过容器引擎向与该资源属性相关联的硬件资源池申请;

[0026] S603)当硬件资源从容器回收到相应的硬件资源池时,删除硬件资源表中对应的ID。

[0027] 进一步地,在本申请的上述方法还包括以下步骤:S500)按照预设的频率定期检测各个硬件资源池,并将检测到已被移除的硬件资源从对应的硬件资源池以及对应的容器中移除。

[0028] 其次,本申请还提出一种用于基于容器的虚拟桌面的设备隔离装置。该设备隔离

装置适用于在服务器内建立的多个彼此隔离的容器。其中，多个容器中的每一个分别创建一个独立的虚拟桌面实例，并通过每个容器的虚拟桌面实例为对应的容器配置独立的文件管理结构。上述设备隔离装置包括以下模块：标识模块，用于针对新添加到服务器的硬件资源，为该硬件资源标识其至少一个能力的资源属性；关联模块，用于利用所标识的资源属性，将该硬件资源与多个硬件资源池中的一个硬件资源池相关联，使得硬件资源池中的每一个硬件资源与多个资源属性中的一个相关联；分配模块，用于分别为多个硬件资源池中的每一个硬件资源分配一个唯一识别的ID；以及管理模块，用于通过容器引擎，依据每一个硬件资源的ID将硬件资源池中硬件资源分配到相应的容器和从相应的容器回收硬件资源到硬件资源池中。

[0029] 进一步地，在本申请的上述装置中，管理模块还包括如下子模块：资源模块，用于依据硬件资源的ID将对应的硬件资源分配到相应的容器；加锁模块，用于根据硬件资源的ID上锁以分配的硬件资源，从而禁止其他容器访问对应ID的硬件资源；解锁模块，用于从容器回收硬件资源到硬件资源池中，并将根据硬件资源的ID解锁对应的硬件资源。

[0030] 可替代地，在本申请的上述装置中，管理模块还包括如下子模块：设置模块，用于为各个硬件资源池设置相关联的触发事件；监控模块，用于确认分配给各个容器的硬件资源所对应的硬件资源池，从而基于该硬件资源池的触发事件监控各个容器内的对应硬件资源；回收模块，用于当检测到触发事件发生时，从容器中回收相应的硬件资源到对应的硬件资源池中。

[0031] 再进一步地，在本申请的上述装置中，该触发事件是硬件资源的使用频率低于预设的阈值，其中该阈值是在设置触发事件时被初始化的，并根据硬件资源的平均使用频率和/或使用频率上限自适应地调整。

[0032] 可替代地，在本申请的上述装置中，该触发事件是申请硬件资源的应用程序的进程结束。

[0033] 进一步地，在本申请的上述装置中，在容器中运行的应用程序通过分配给该容器的硬件资源的ID访问该硬件资源。

[0034] 再进一步地，在本申请的上述装置中，在容器中运行的应用程序通过分配给该容器的硬件资源的ID访问该硬件资源包括以下子模块：记录模块，用于从硬件资源池分配硬件资源到相应的容器，并将该硬件资源的ID记录在容器的硬件资源表中；搜索模块，用于搜索硬件资源表，当硬件资源表中存在相应资源属性时，容器中运行的应用程序根据硬件资源的ID访问对应的硬件资源，否则通过容器引擎向与该资源属性相关联的硬件资源池申请；删除模块，用于当硬件资源从容器回收到相应的硬件资源池时，删除硬件资源表中对应的ID。

[0035] 进一步地，在本申请的上述装置还包括以下模块：移除模块，用于按照预设的频率定期检测各个硬件资源池，并将检测到已被移除的硬件资源从对应的硬件资源池以及对应的容器中移除。

[0036] 最后，本申请还提出一种计算机可读存储介质，其上存储有计算机指令。该计算机指令所执行方法适用于在服务器内建立的多个彼此隔离的容器。其中，多个容器中的每一个分别创建一个独立的虚拟桌面实例，并通过每个容器的虚拟桌面实例为对应的容器配置独立的文件管理结构。该指令被处理器执行时实现以下方法的步骤：

[0037] S100)针对新添加到服务器的硬件资源,为所述硬件资源标识其至少一个能力的资源属性;

[0038] S200)利用所标识的资源属性,将所述硬件资源与多个硬件资源池中的一个硬件资源池相关联,使得硬件资源池中的每一个硬件资源与多个资源属性中的一个相关联;

[0039] S300)分别为多个硬件资源池中的每一个硬件资源分配一个唯一识别的ID;以及

[0040] S400)通过容器引擎,依据每一个硬件资源的ID将硬件资源池中硬件资源分配到相应的容器和从相应的容器回收硬件资源到硬件资源池中。

[0041] 进一步地,在本申请的上述指令被处理器执行的过程中,步骤S400还包括如下子步骤:

[0042] S401)依据硬件资源的ID将对应的硬件资源分配到相应的容器;

[0043] S402)根据硬件资源的ID上锁以分配的硬件资源,从而禁止其他容器访问对应ID的硬件资源;

[0044] S403)从容器回收硬件资源到硬件资源池中,并将根据硬件资源的ID解锁对应的硬件资源。

[0045] 可替代地,在本申请的上述指令被处理器执行的过程中,步骤S400)可包括如下子步骤:

[0046] S404)为各个硬件资源池设置相关联的触发事件;

[0047] S405)确认分配给各个容器的硬件资源所对应的硬件资源池,从而基于所述硬件资源池的触发事件监控各个容器内的对应硬件资源;

[0048] S406)当检测到触发事件发生时,从容器中回收相应的硬件资源到对应的硬件资源池中。

[0049] 再进一步地,在本申请的上述指令被处理器执行的过程中,该触发事件是硬件资源的使用频率低于预设的阈值,其中该阈值是在设置触发事件时被初始化的,并根据硬件资源的平均使用频率和/或使用频率上限自适应地调整。

[0050] 可替代地,在本申请的上述指令被处理器执行的过程中,该触发事件是申请硬件资源的应用程序的进程结束。

[0051] 进一步地,在本申请的上述指令被处理器执行的过程中,在容器中运行的应用程序通过分配给该容器的硬件资源的ID访问该硬件资源。

[0052] 再进一步地,在本申请的上述指令被处理器执行的过程中,在容器中运行的应用程序通过分配给该容器的硬件资源的ID访问该硬件资源包括以下子步骤:

[0053] S601)从硬件资源池分配硬件资源到相应的容器,并将所述硬件资源的ID记录在容器的硬件资源表中;

[0054] S602)搜索硬件资源表,当硬件资源表中存在相应资源属性时,容器中运行的应用程序根据硬件资源的ID访问对应的硬件资源,否则通过容器引擎向与该资源属性相关联的硬件资源池申请;

[0055] S603)当硬件资源从容器回收到相应的硬件资源池时,删除硬件资源表中对应的ID。

[0056] 进一步地,在本申请的上述指令被处理器执行的过程中还包括以下步骤:S500)按照预设的频率定期检测各个硬件资源池,并将检测到已被移除的硬件资源从对应的硬件资

源池以及对应的容器中移除。

[0057] 本申请的有益效果为:通过利用对各个硬件资源上标识唯一识别的ID,能够获得实现容器自身的轻量化同时,保障不同容器之间的硬件资源安全性的效果。

附图说明

[0058] 图1所示为现有的虚拟机和容器结构示意图;

[0059] 图2所示为本申请所公开实施例中基于容器的虚拟桌面的架构示意图;

[0060] 图3所示为本申请所公开实施例中基于容器的虚拟桌面的用例图;

[0061] 图4所示为本申请所公开的用于基于容器的虚拟桌面的设备隔离方法的流程图;

[0062] 图5所示为本申请所公开的第一实施例中访问硬件资源的子方法流程图;

[0063] 图6所示为图5所示实施例中访问硬件资源过程的示意图;

[0064] 图7所示为本申请所公开的第二实施例中访问硬件资源的子方法流程图;

[0065] 图8所示为本申请所公开的第三实施例中访问硬件资源过程的示意图;

[0066] 图9所示为图8所示实施例中访问硬件资源的子方法流程图;

[0067] 图10所示为本申请所公开的用于基于容器的虚拟桌面的设备隔离装置的结构图。

具体实施方式

[0068] 以下将结合实施例和附图对本申请的构思、具体结构及产生的技术效果进行清楚、完整的描述,以充分地理解本申请的目的、方案和效果。需要说明的是,在不冲突的情况下,本申请中的实施例及实施例中的特征可以相互组合。附图中各处使用的相同的附图标记指示相同或相似的部分。

[0069] 参考图1所示的现有的虚拟机和容器结构示意图。作为虚拟化技术的两个典型例子,虚拟机和容器包含应用程序及运行该应用程序的所必需的组件(例如系统的二进制文件及库),并分别通过运行在物理计算机上的虚拟机监控器(Hypervisor)和容器引擎(Docker Engine)申请位于系统底层的计算机硬件资源。对于虚拟机和容器,底层的单个计算机硬件资源(例如服务器、存储设备、中央处理器、I/O端口和网络端口等)由于被虚拟化而可以充当多个可被共享的逻辑资源。同时,主机操作系统(Operation System,OS)亦可以为上述虚拟机监控器和容器引擎与底层计算机硬件资源的交互提供进一步优化。例如,主机OS可支持多个空间上相互隔离的用户通过虚拟机或容器共享底层的计算机硬件资源。其中主机OS的一些示例可以是Windows®、Unix®和Linux®。然而,正如前文所指出,虚拟机由于其自身操作系统需要专门占用更多的资源并包含更多的处理开销,从而降低了主机设备的总体性能。虽然容器通过共享主机OS的内核而实现自身的轻量化,但是容器之间的并没有实现真正的隔离,从而为用户留下安全隐患。

[0070] 因此,为了充分利用容器的轻量化优势,并提升容器之间的隔离程度,避免不同容器之间的硬件资源被错误调用,本申请提出了一种用于基于容器的虚拟桌面的设备隔离方法。该方法适用于如图2所示的基于容器的虚拟桌面。在该基于容器的虚拟桌面中,服务器内建立的多个彼此隔离的容器。其中,多个容器中的每一个分别创建一个独立的虚拟桌面实例,并通过每个容器的虚拟桌面实例为对应的容器配置独立的文件管理结构。在每个容器内运行的应用程序通过容器引擎由主机的硬件资源管理器从计算机硬件资源池中获取

相应的计算机硬件资源。在向对应的容器发出文件操作请求时，应用程序通过虚拟桌面实例调用文件管理结构对该文件操作请求涉及的文件进行操作。对于容器的各位用户，基于容器所提供的虚拟桌面实例就像一个独立的个人操作系统，而不仅仅是从主机操作系统中划分出来的，作为主机操作系统一部分而隔离出来的操作执行环境。进一步地，参照图3所示的用例图，本地服务器通过容器引擎为每个容器配置相应的系统环境、个人文件以及虚拟界面的配置文件，并利用容器的DockerClient形成虚拟桌面实例。对于容器的各位用户，容器就像一个独立的个人操作系统，而不仅仅是从主机操作系统中划分出来的，作为主机操作系统一部分而隔离出来的操作执行环境。进一步地，上述配置文件可进一步地设置针对主机OS的内核的差异文件，并在各个容器上形成个性化的虚拟操作系统环境，而实现对主机OS的内核重定向。此外，对于容器内的应用程序，其运行时所必需的计算机硬件资源都是通过容器引擎申请。因此，一方面相对于虚拟机所必需占用专用资源以供虚拟机自身的操作系统运行，上述技术方案中的容器更为轻量，从而可以在同一服务器上为更多的用户创建容器；另一方面，由于应用程序需要通过容器引擎共享底层的计算机硬件资源，并且当其发出文件操作请求时，必需通过虚拟桌面实例调用容器的文件管理结构以对所述文件操作请求涉及的文件进行操作，从而保证了容器之间隔离程度，提高了容器的安全水平。进一步地，容器的客户端可以设置在异地。如图3所示，用户的客户端与本地服务器之间设置有交换盒，该交换盒上设置有嵌入式系统以执行相关的图像生成和文件传输协议，从而在用户端上形成相应的图形操作界面。

[0071] 具体地，参照图4所示的方法流程图，上述设备隔离方法包括如下步骤：

[0072] S100)针对新添加到服务器的硬件资源，为所述硬件资源标识其至少一个能力的资源属性；

[0073] S200)利用所标识的资源属性，将所述硬件资源与多个硬件资源池中的一个硬件资源池相关联，使得硬件资源池中的每一个硬件资源与多个资源属性中的一个相关联；

[0074] S300)分别为多个硬件资源池中的每一个硬件资源分配一个唯一识别的ID；以及

[0075] S400)通过容器引擎，依据每一个硬件资源的ID将硬件资源池中硬件资源分配到相应的容器和从相应的容器回收硬件资源到硬件资源池中。

[0076] 其中，资源属性可以是根据硬件资源的功能进行划分，例如但是不限于用于计算的计算硬件资源池(例如CPU和GPU)、用于存储的存储硬件资源池(例如各级缓存和非暂时性存储介质)以及用于通讯的网络硬件资源池(例如带宽)。此时，对于容器内的应用程序，其运行时所必需的硬件资源都是通过容器引擎申请。容器引擎根据硬件资源的ID对硬件资源池中的硬件资源进行过滤，确保在任何时刻每个硬件资源都仅被分配到多个共享该硬件资源的容器中的一个(例如位于办公室内被多个用户所共享的打印机或扫描仪)，或在任意时刻仅可被分配到一个指定的容器(例如个人笔记本上的摄像头)。此外，本领域技术人员可以通过本领域惯用技术手段获取并标识硬件资源属性(例如图2所示实施例中被关联到计算硬件资源池中的CPU、关联到存储硬件资源池中磁盘或关联到网络硬件资源池中的I/O端口等)，并对硬件资源的ID设置适当的标识位，以区别硬件资源的资源属性以及上述两类(可被多个容器共享或只能被指定容器共享的)硬件资源。本申请对与硬件资源的ID具体标识方式及标识所包含的信息不予限定。此外，本领域技术人员应理解到，附图中所显示元件的数量和形状仅作为示例性的参考，不作为对本申请的限制。

[0077] 参照图5所示的子方法流程图，在本申请的一个或多个实施例中，步骤S400)还包括如下子步骤：

[0078] S401)依据硬件资源的ID将对应的硬件资源分配到相应的容器；

[0079] S402)根据硬件资源的ID上锁以分配的硬件资源，从而禁止其他容器访问对应ID的硬件资源；

[0080] S403)从容器回收硬件资源到硬件资源池中，并将根据硬件资源的ID解锁对应的硬件资源。

[0081] 此时，参照图6所示的访问硬件资源过程的示意图，对于容器的用户，相关硬件资源如同用户的个人计算机一部分而被包含在其容器中。在容器内部运行的应用程序可以通过容器引擎访问该硬件资源。然而对于其他容器的应用程序而言，由于该硬件资源被上锁，所以其无法基于硬件资源的ID向容器引擎申请该硬件资源，从而提高了容器之间硬件资源的安全性。

[0082] 参照图7所示的子方法流程图，在本申请的一个或多个实施例中，步骤S400还包括如下子步骤：

[0083] S404)为各个硬件资源池设置相关联的触发事件；

[0084] S405)确认分配给各个容器的硬件资源所对应的硬件资源池，从而基于所述硬件资源池的触发事件监控各个容器内的对应硬件资源；

[0085] S406)当检测到触发事件发生时，从容器中回收相应的硬件资源到对应的硬件资源池中。具体地，触发事件是申请硬件资源的应用程序的进程结束，即当应用程序的进程结束时，立即回收相关的硬件资源。可替代地，为了避免频繁地分配/回收硬件资源所带来的额外开销，该触发事件可以是硬件资源的使用频率低于预设的阈值。该使用频率例如但是不限于数据处理器利用率、存储器利用率、数据存储利用率和网络利用率。相应地，阈值是与该使用频率具有相同单位的值。其中，该阈值是在设置触发事件时被初始化，并根据硬件资源的平均使用频率和/或使用频率上限自适应地调整。例如，当硬件资源的平均使用频率是使用频率上限的80％时，将当前阈值上调10％。本领域技术人员可以根据具体的硬件资源设置该初始化值和该阈值的自适应调整规则，本申请对此不予限定。

[0086] 参照图2所示的基于容器的虚拟桌面的架构示意图和图8所示的访问硬件资源过程的示意图，在本申请的一个或多个实施例中，在容器中运行的应用程序通过分配给该容器的硬件资源的ID访问该硬件资源。具体地，每个容器的虚拟桌面实例中都维护有一份硬件资源表，容器中的应用程序(例如图2中在容器1内部运行的应用程序A1和应用程序A2)通过该硬件资源表向容器引擎申请相应的硬件资源。进一步地，参照图9所示的子方法流程图，在本申请的上述一个或多个实施例中，在容器中运行的应用程序通过分配给该容器的硬件资源的ID访问该硬件资源包括以下子步骤：

[0087] S601)从硬件资源池分配硬件资源到相应的容器，并将所述硬件资源的ID记录在容器的硬件资源表中；

[0088] S602)搜索硬件资源表，当硬件资源表中存在相应资源属性时，容器中运行的应用程序根据硬件资源的ID访问对应的硬件资源，否则通过容器引擎向与该资源属性相关联的硬件资源池申请；

[0089] S603)当硬件资源从容器回收到相应的硬件资源池时，删除硬件资源表中对应的

ID。在本申请的一个或多个实施例中，上述设备隔离方法还包括以下步骤以及时更新硬件资源池中实际可用的硬件资源：

[0090] S500)按照预设的频率定期检测各个硬件资源池，并将检测到已被移除的硬件资源从对应的硬件资源池以及对应的容器中移除。

[0091] 类似地，本领域技术人员可以根据具体的硬件资源初始化频率值和设置该频率值的自适应调整规则，本申请对此不予限定。

[0092] 相应地，参照图10所示的模块结构图，本申请所公开用于基于容器的虚拟桌面的设备隔离装置包括以下模块：标识模块，用于针对新添加到服务器的硬件资源，为该硬件资源标识其至少一个能力的资源属性；关联模块，用于利用所标识的资源属性，将该硬件资源与多个硬件资源池中的一个硬件资源池相关联，使得硬件资源池中的每一个硬件资源与多个资源属性中的一个相关联；分配模块，用于分别为多个硬件资源池中的每一个硬件资源分配一个唯一识别的ID；以及管理模块，用于通过容器引擎，依据每一个硬件资源的ID将硬件资源池中硬件资源分配到相应的容器和从相应的容器回收硬件资源到硬件资源池中。此时，对于容器内的应用程序，其运行时所必需的硬件资源都是通过容器引擎申请。容器引擎根据硬件资源的ID对硬件资源池中的硬件资源进行过滤，确保在任何时刻每个硬件资源都仅被分配到多个共享该硬件资源的容器中的一个(例如位于办公室内被多个用户所共享的打印机或扫描仪)，或在任意时刻仅可被分配到一个指定的容器(例如个人笔记本上的摄像头)。此外，本领域技术人员可以通过本领域惯用技术手段获取并标识硬件资源属性(例如图2所示实施例中被关联到计算硬件资源池中的CPU、关联到存储硬件资源池中磁盘或关联到网络硬件资源池中的I/O端口等)，并对硬件资源的ID设置适当的标识位，以区别硬件资源的资源属性以及上述两类(可被多个容器共享或只能被指定容器共享的)硬件资源。本申请对与硬件资源的ID具体标识方式及标识所包含的信息不予限定。此外，本领域技术人员应理解到，附图中所显示元件的数量和形状仅作为示例性的参考，不作为对本申请的限制。

[0093] 在本申请的一个或多个实施例中，管理模块还包括如下子模块：资源模块，用于依据硬件资源的ID将对应的硬件资源分配到相应的容器；加锁模块，用于根据硬件资源的ID上锁以分配的硬件资源，从而禁止其他容器访问对应ID的硬件资源；解锁模块，用于从容器回收硬件资源到硬件资源池中，并将根据硬件资源的ID解锁对应的硬件资源。此时，参照图6所示的访问硬件资源过程的示意图，对于容器的用户，相关硬件资源如同用户的个人计算机一部分而被包含在其容器中。在容器内部运行的应用程序可以通过容器引擎访问该硬件资源。然而对于其他容器的应用程序而言，由于该硬件资源被上锁，所以其无法基于硬件资源的ID向容器引擎申请该硬件资源，从而提高了容器之间硬件资源的安全性。

[0094] 在本申请的一个或多个实施例中，管理模块还包括如下子模块：设置模块，用于为各个硬件资源池设置相关联的触发事件；监控模块，用于确认分配给各个容器的硬件资源所对应的硬件资源池，从而基于该硬件资源池的触发事件监控各个容器内的对应硬件资源；回收模块，用于当检测到触发事件发生时，从容器中回收相应的硬件资源到对应的硬件资源池中。具体地，触发事件是申请硬件资源的应用程序的进程结束，即当应用程序的进程结束时，立即回收相关的硬件资源。可替代地，为了避免频繁地分配/回收硬件资源所带来的额外开销，该触发事件可以是硬件资源的使用频率低于预设的阈值。该使用频率例如但是不限于数据处理器利用率、存储器利用率、数据存储利用率和网络利用率。相应地，阈值

是与该使用频率具有相同单位的值。其中，该阈值是在设置触发事件时被初始化，并根据硬件资源的平均使用频率和/或使用频率上限自适应地调整。例如，当硬件资源的平均使用频率是使用频率上限的80％时，将当前阈值上调10％。本领域技术人员可以根据具体的硬件资源设置该初始化值和该阈值的自适应调整规则，本申请对此不予限定。

[0095] 参照图2所示的基于容器的虚拟桌面的架构示意图和图8所示的访问硬件资源过程的示意图，在本申请的一个或多个实施例中，在容器中运行的应用程序通过分配给该容器的硬件资源的ID访问该硬件资源。具体地，每个容器的虚拟桌面实例中都维护有一份硬件资源表，容器中的应用程序(例如图2中在容器1内部运行的应用程序A1和应用程序A2)通过该硬件资源表向容器引擎申请相应的硬件资源。进一步地，在容器中运行的应用程序通过分配给该容器的硬件资源的ID访问该硬件资源包括以下子模块：记录模块，用于从硬件资源池分配硬件资源到相应的容器，并将该硬件资源的ID记录在容器的硬件资源表中；搜索模块，用于搜索硬件资源表，当硬件资源表中存在相应资源属性时，容器中运行的应用程序根据硬件资源的ID访问对应的硬件资源，否则通过容器引擎向与该资源属性相关联的硬件资源池申请；删除模块，用于当硬件资源从容器回收到相应的硬件资源池时，删除硬件资源表中对应的ID，从而实现动态地维护更新每个容器内的硬件资源表。

[0096] 在本申请的一个或多个实施例中，上述设备隔离装置还包括以下模块以及时更新硬件资源池中实际可用的硬件资源：移除模块，用于按照预设的频率定期检测各个硬件资源池，并将检测到已被移除的硬件资源从对应的硬件资源池以及对应的容器中移除。类似地，本领域技术人员可以根据具体的硬件资源初始化频率值和设置该频率值的自适应调整规则，本申请对此不予限定。

[0097] 应当认识到，本申请的实施例可以由计算机硬件、硬件和软件的组合、或者通过存储在非暂时性计算机可读存储器中的计算机指令来实现或实施。该方法可以使用标准编程技术-包括配置有计算机程序的非暂时性计算机可读存储介质在计算机程序中实现，其中如此配置的存储介质使得计算机以特定和预定义的方式操作-根据在具体实施例中描述的方法和附图。每个程序可以以高级过程或面向对象的编程语言来实现以与计算机系统通信。然而，若需要，该程序可以以汇编或机器语言实现。在任何情况下，该语言可以是编译或解释的语言。此外，为此目的该程序能够在编程的专用集成电路上运行。

[0098] 进一步地，该方法可以在可操作地连接至合适的任何类型的计算平台中实现，包括但不限于个人电脑、迷你计算机、主框架、工作站、网络或分布式计算环境、单独的或集成的计算机平台、或者与带电粒子工具或其它成像装置通信等等。本申请的各方面可以以存储在非暂时性存储介质或设备上的机器可读代码来实现，无论是可移动的还是集成至计算平台，如硬盘、光学读取和/或写入存储介质、RAM、ROM等，使得其可由可编程计算机读取，当存储介质或设备由计算机读取时可用于配置和操作计算机以执行在此所描述的过程。此外，机器可读代码，或其部分可以通过有线或无线网络传输。当此类媒体包括结合微处理器或其他数据处理器实现上文该步骤的指令或程序时，本文所述的申请包括这些和其他不同类型的非暂时性计算机可读存储介质。当根据本申请所述的方法和技术编程时，本申请还包括计算机本身。

[0099] 计算机程序能够应用于输入数据以执行本文所述的功能，从而转换输入数据以生成存储至非易失性存储器的输出数据。输出信息还可以应用于一个或多个输出设备如显示

器。在本申请优选的实施例中，转换的数据表示物理和有形的对象，包括显示器上产生的物理和有形对象的特定视觉描绘。

[0100] 因此，应以说明性意义而不是限制性意义来理解本说明书和附图。然而，将明显的是：在不脱离如权利要求书中阐述的本申请的更宽广精神和范围的情况下，可以对本申请做出各种修改和改变。

[0101] 其他变型在本申请的精神内。因此，尽管所公开的技术可容许各种修改和替代构造，但在附图中已示出并且在上文中详细描述所示的其某些实施例。然而，应当理解，并不意图将本申请局限于所公开的一种或多种具体形式；相反，其意图涵盖如所附权利要求书中所限定落在本申请的精神和范围内的所有修改、替代构造和等效物。

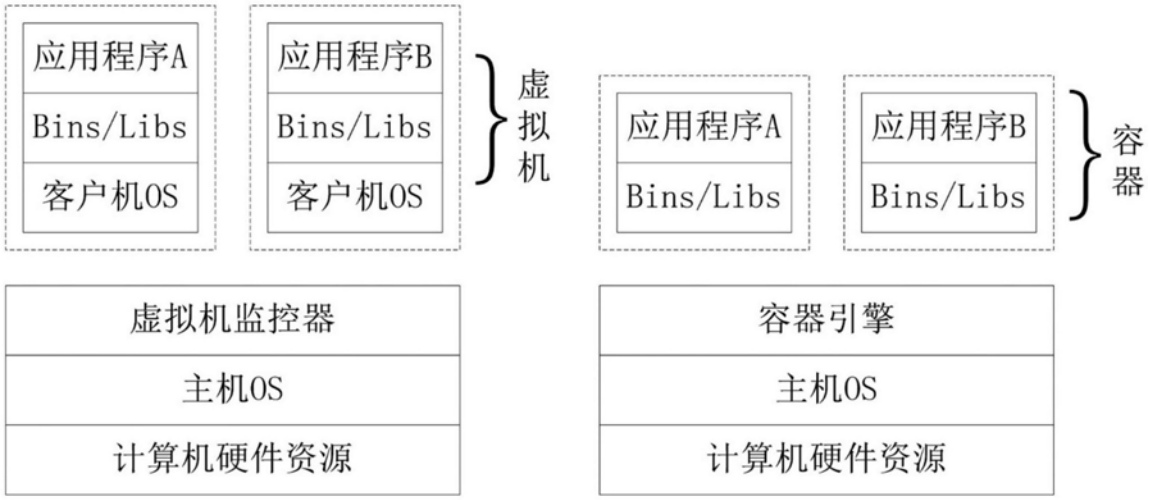

应用程序A
Bins/Libs
客户机OS
应用程序B
Bins/Libs
客户机OS
虚拟机
应用程序A
Bins/Libs
应用程序B
Bins/Libs
容器
虚拟机监控器
主机OS
计算机硬件资源
容器引擎
主机OS
计算机硬件资源

图1

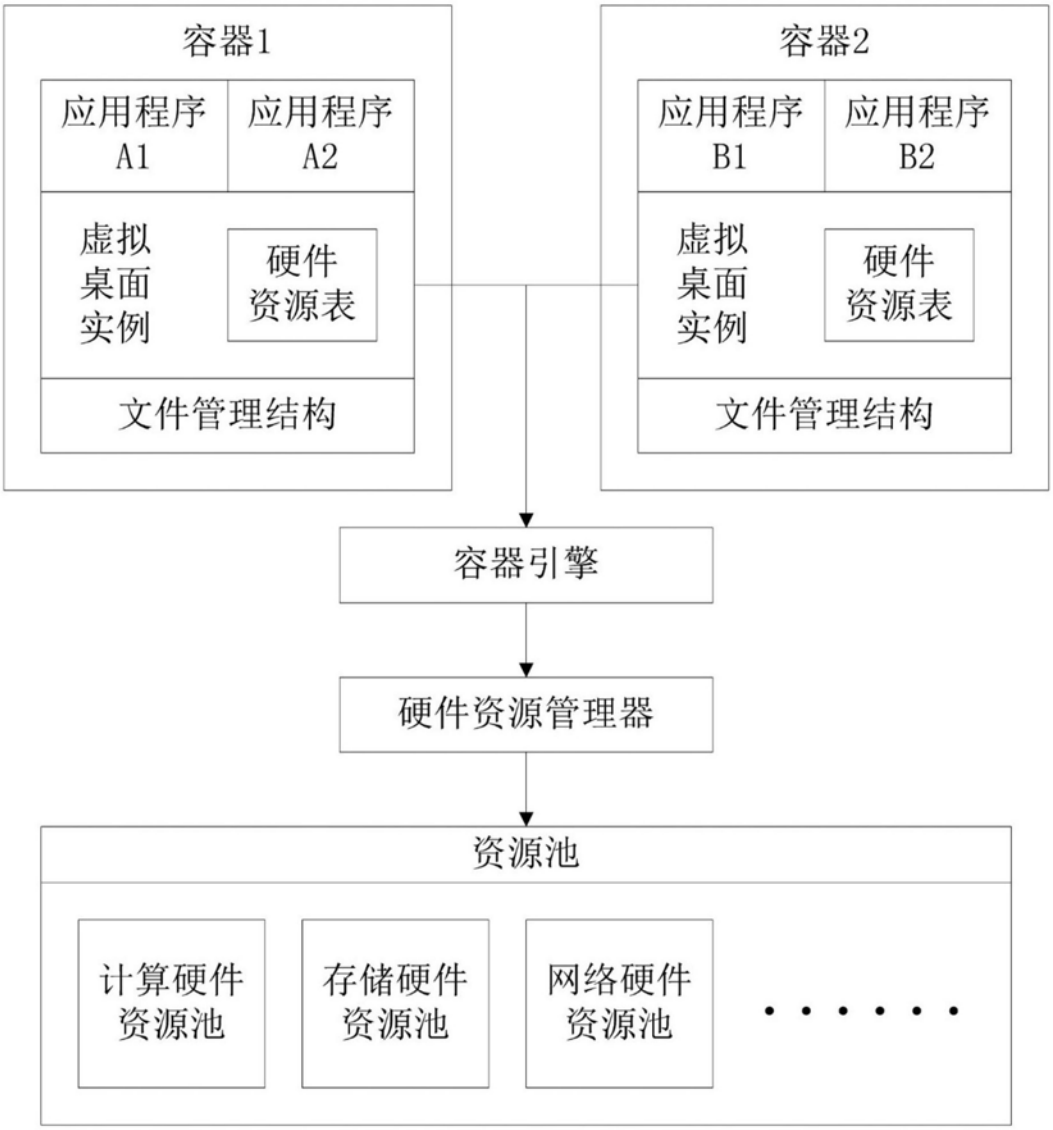

容器1
应用程序 A1
应用程序 A2
虚拟 桌面 实例
硬件 资源表
文件管理结构
容器2
应用程序 B1
应用程序 B2
虚拟 桌面 实例
硬件 资源表
文件管理结构
容器引擎
硬件资源管理器
资源池
计算硬件 资源池
存储硬件 资源池
网络硬件 资源池

图2

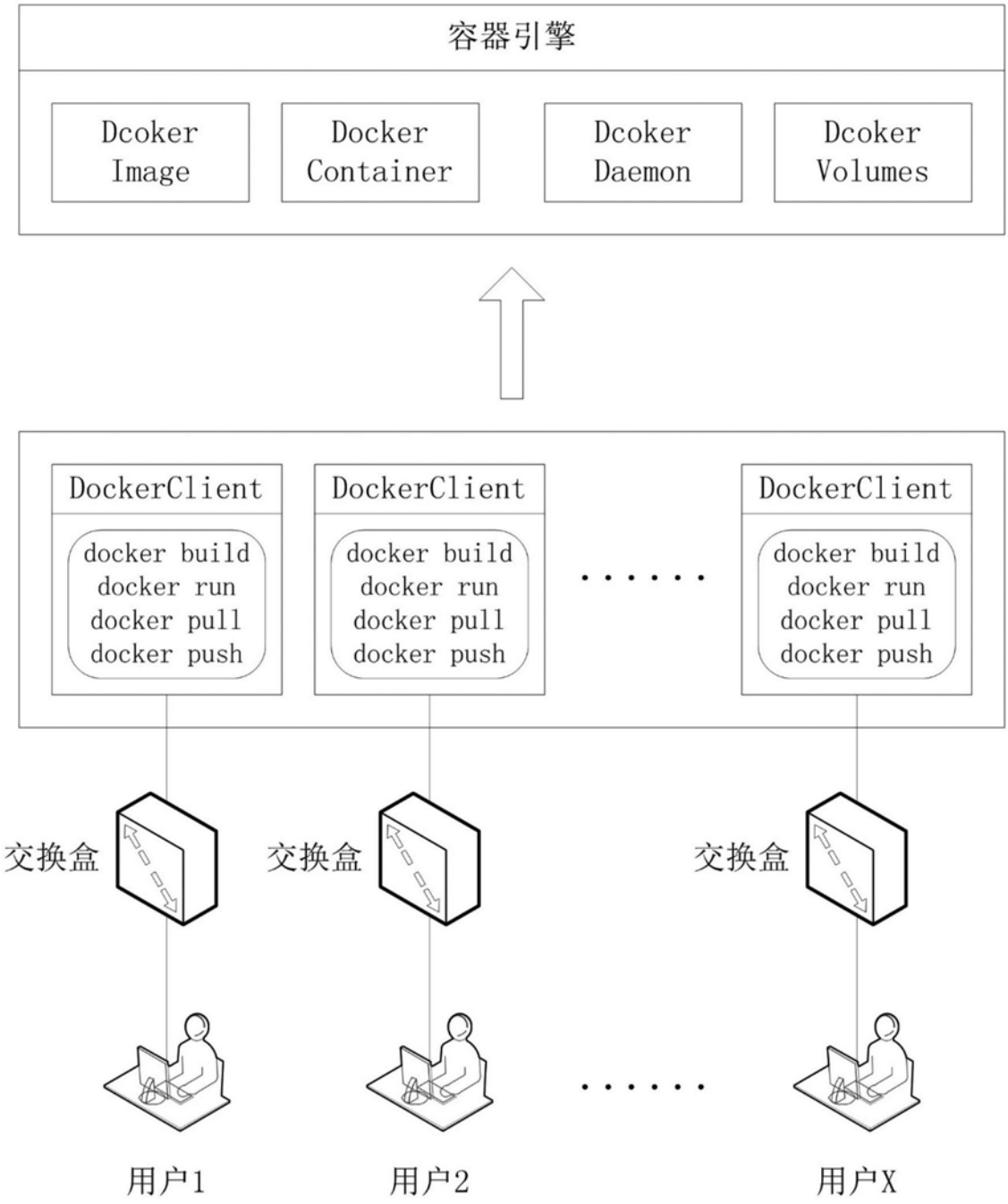

容器引擎
Dcoker Image
Docker Container
Dcoker Daemon
Dcoker Volumes
DockerClient
docker build
docker run
docker pull
docker push
DockerClient
docker build
docker run
docker pull
docker push
DockerClient
docker build
docker run
docker pull
docker push
交换盒
交换盒
交换盒
用户1
用户2
用户X

图3

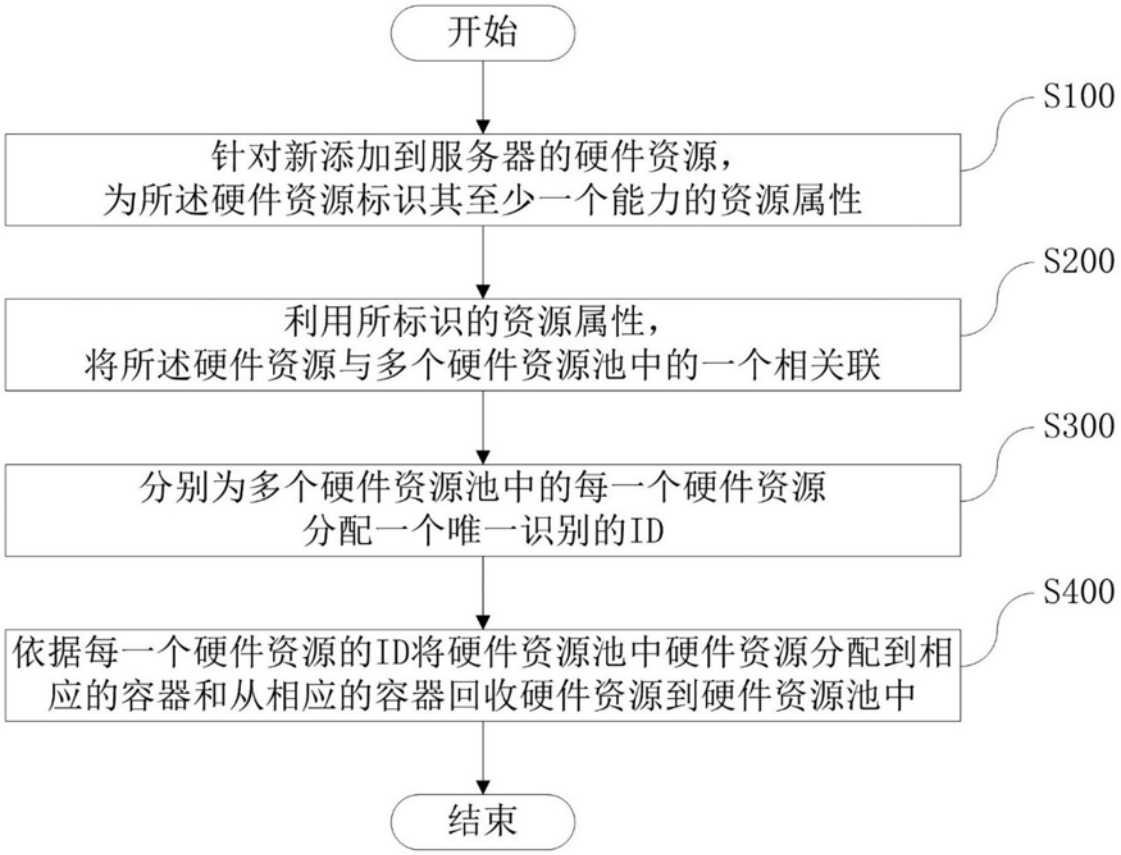

开始
S100
针对新添加到服务器的硬件资源，
为所述硬件资源标识其至少一个能力的资源属性
S200
利用所标识的资源属性，
将所述硬件资源与多个硬件资源池中的一个相关联
S300
分别为多个硬件资源池中的每一个硬件资源
分配一个唯一识别的ID
S400
依据每一个硬件资源的ID将硬件资源池中硬件资源分配到相
应的容器和从相应的容器回收硬件资源到硬件资源池中
结束

图4

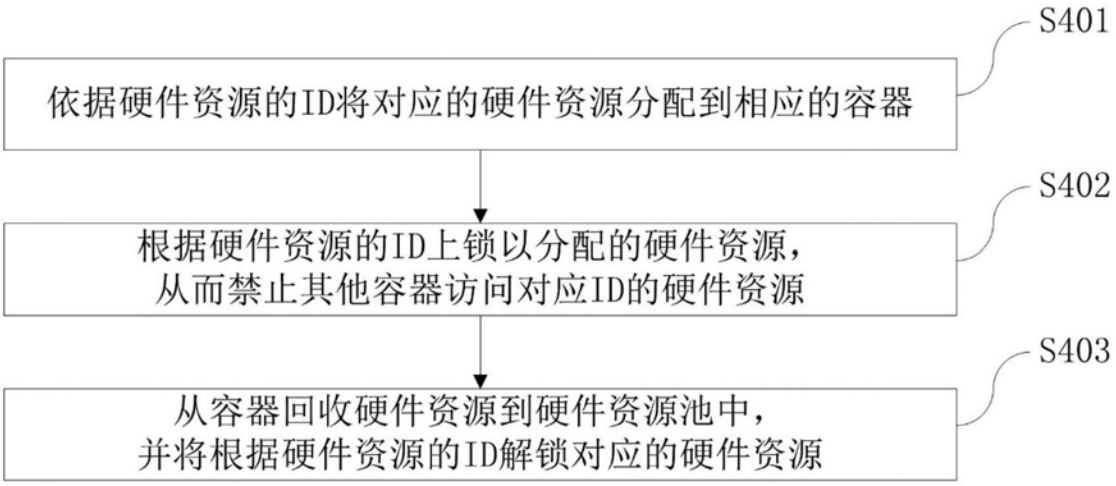

S401
依据硬件资源的ID将对应的硬件资源分配到相应的容器
S402
根据硬件资源的ID上锁以分配的硬件资源，
从而禁止其他容器访问对应ID的硬件资源
S403
从容器回收硬件资源到硬件资源池中，
并将根据硬件资源的ID解锁对应的硬件资源

图5

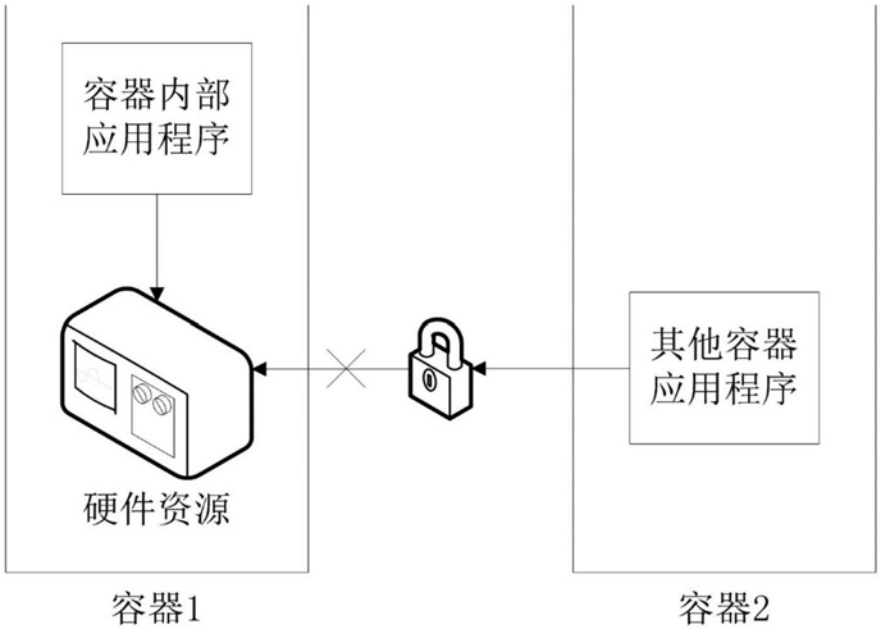

容器内部
应用程序
其他容器
应用程序
硬件资源
容器1
容器2

图6

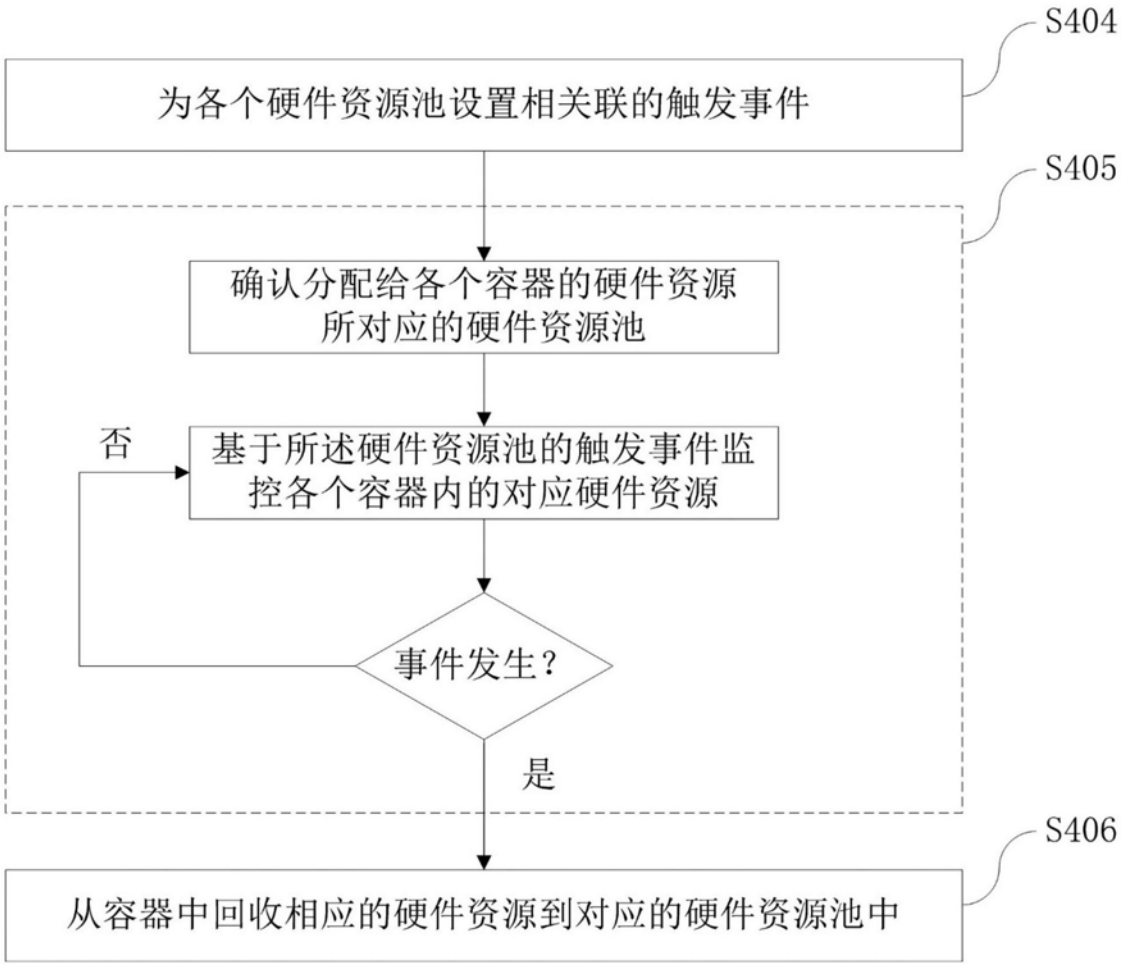

S404
为各个硬件资源池设置相关联的触发事件
S405
确认分配给各个容器的硬件资源
所对应的硬件资源池
否
基于所述硬件资源池的触发事件监
控各个容器内的对应硬件资源
事件发生？
是
S406
从容器中回收相应的硬件资源到对应的硬件资源池中

图7

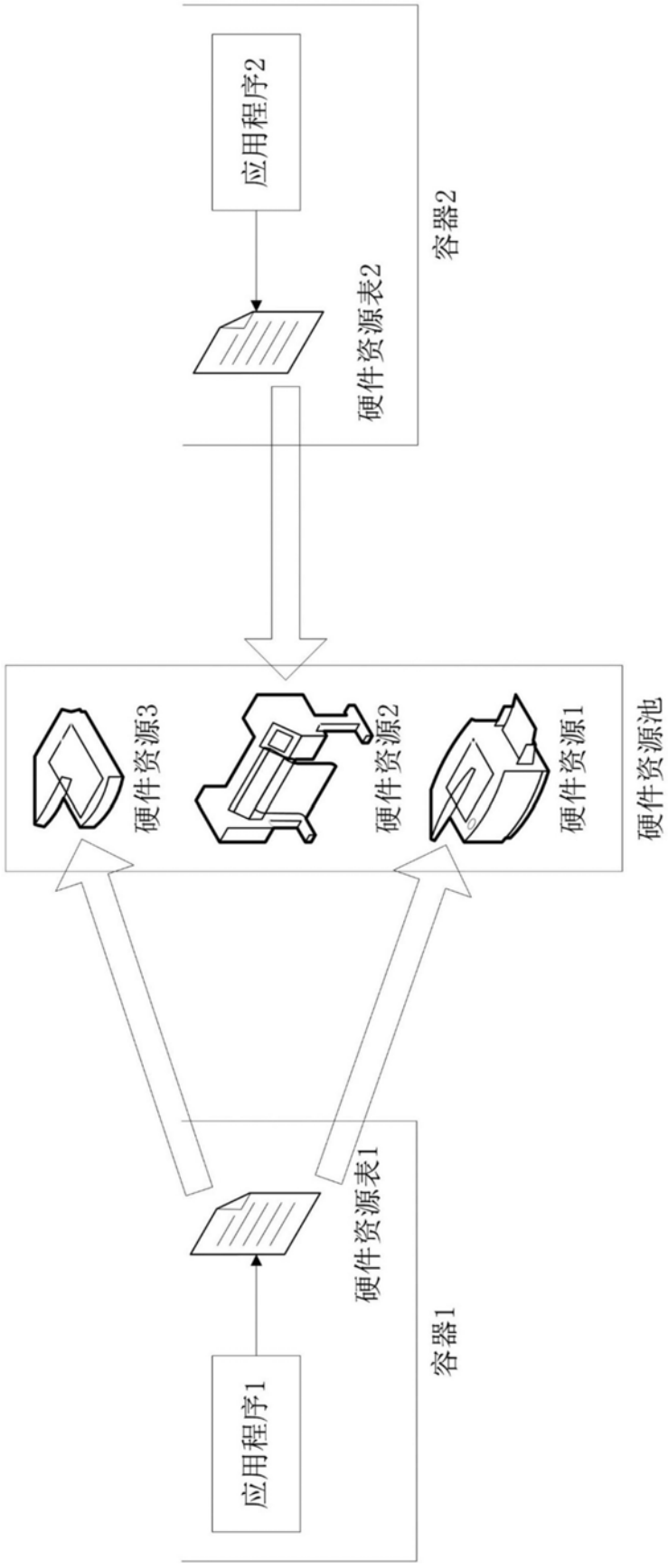


图8

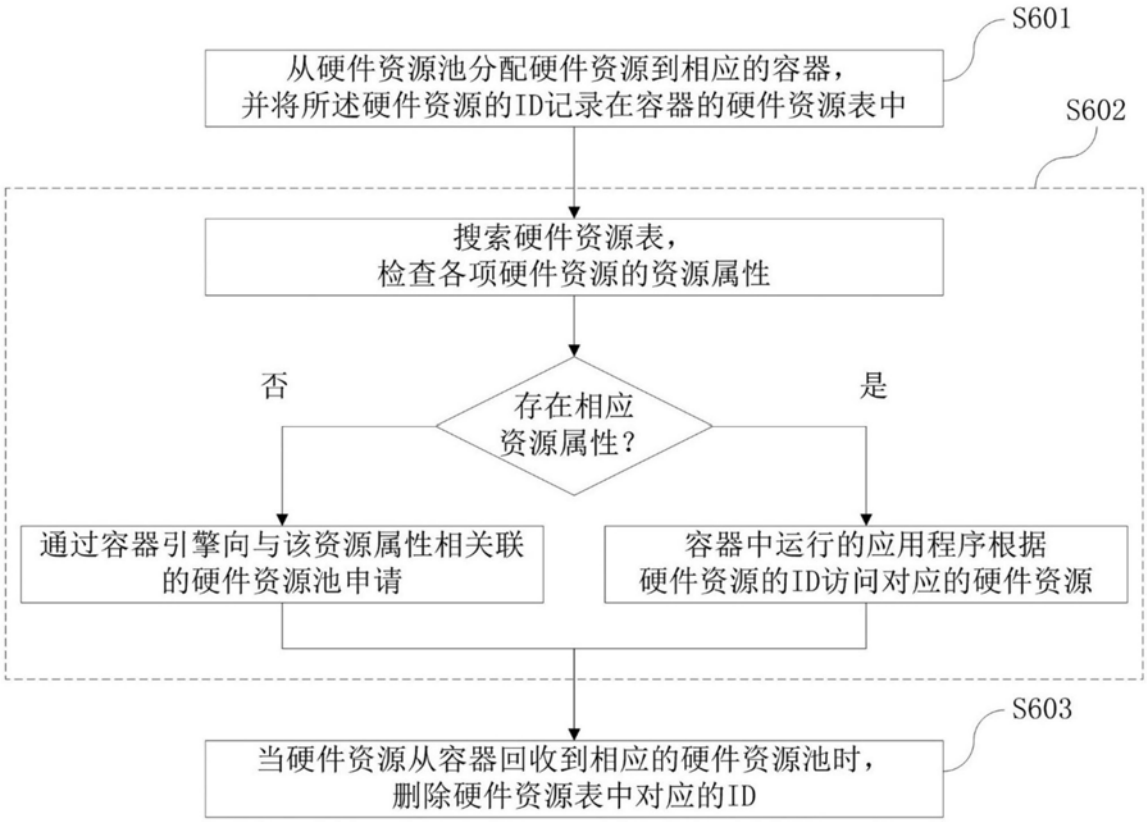

S601
从硬件资源池分配硬件资源到相应的容器，
并将所述硬件资源的ID记录在容器的硬件资源表中
S602
搜索硬件资源表，
检查各项硬件资源的资源属性
否
存在相应
资源属性？
是
通过容器引擎向与该资源属性相关联
的硬件资源池申请
容器中运行的应用程序根据
硬件资源的ID访问对应的硬件资源
S603
当硬件资源从容器回收到相应的硬件资源池时，
删除硬件资源表中对应的ID

图9

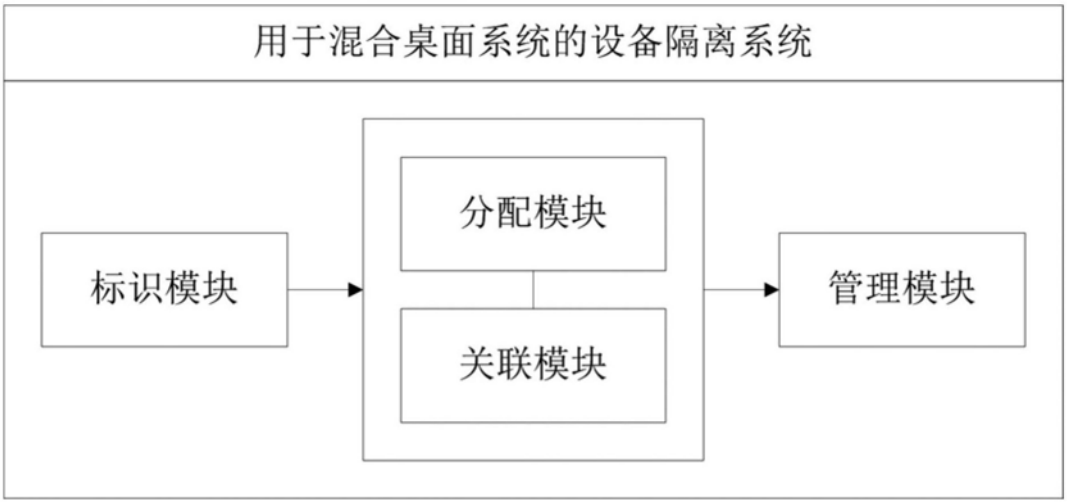

用于混合桌面系统的设备隔离系统
标识模块
分配模块
关联模块
管理模块

图10