(19)国家知识产权局

(12)发明专利

(10)授权公告号 CN 115858101 B
(45)授权公告日 2023.06.23

(21)申请号 202310048002.7
(22)申请日 2023.01.31
(65)同一申请的已公布的文献号
申请公布号 CN 115858101 A
(43)申请公布日 2023.03.28
(73)专利权人 天翼云科技有限公司
地址 100093 北京市海淀区西山赢府商务中心E座4层
(72)发明人 陈长春
(74)专利代理机构 北京润泽恒知识产权代理有限公司 11319
专利代理师 任亚娟
(51)Int.Cl.
G06F 9/455(2018.01)
G06F 16/16(2019.01)

审查员 王思文

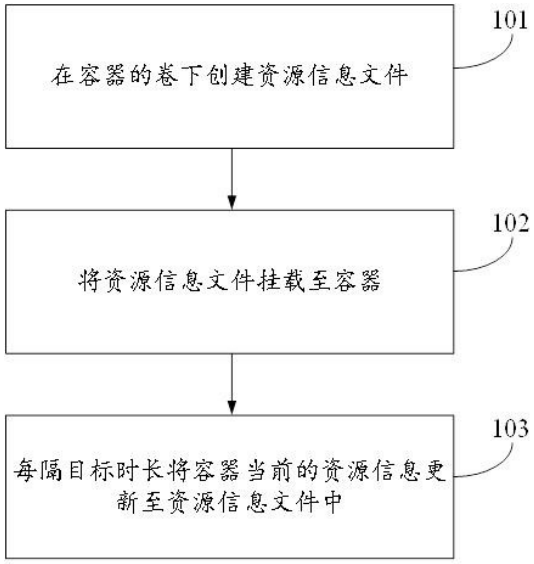
权利要求书2页 说明书7页 附图2页

(54)发明名称

容器资源视图隔离的方法、装置以及电子设备

(57)摘要

本发明实施例提供了一种容器资源视图隔离的方法、装置以及电子设备，该方法包括：在容器的卷下创建资源信息文件；将所述资源信息文件挂载至所述容器；每隔目标时长将所述容器当前的资源信息更新至所述资源信息文件中。本发明利用在容器的卷下创建的资源信息文件维护容器的资源信息，不再直接依赖于lxcfs创建的proc文件，降低了lxcfs的影响，即使lxcfs发生异常，依旧可以读取到容器的资源信息。



CN 115858101 B

1.一种容器资源视图隔离的方法,其特征在于,所述方法包括:

在容器的卷下创建资源信息文件;

将所述资源信息文件挂载至所述容器;

每隔目标时长将所述容器当前的资源信息更新至所述资源信息文件中;

在容器的卷下创建资源信息文件之后,所述方法还包括:

获取所述容器的资源配置信息;

将所述资源配置信息写入所述资源信息文件;

所述每隔目标时长将所述容器当前的资源信息更新至所述资源信息文件中,包括:

每隔目标时长通过lxcfs守护进程查询目标文件,得到所述容器当前的资源信息;其中,所述目标文件为lxcfs生成的proc文件;

采用所述容器当前的资源信息更新所述资源信息文件中的资源信息。

2.根据权利要求1所述的方法,其特征在于,所述在容器的卷下创建资源信息文件,包括:

采用所述容器的卷插件在所述卷下创建所述资源信息文件。

3.根据权利要求1所述的方法,其特征在于,在每隔目标时长将所述容器当前的资源信息更新至所述资源信息文件中之后,所述方法还包括:

在接收到用户的目标输入的情况下,删除所述容器的卷。

4.根据权利要求1所述的方法,其特征在于,所述资源信息包括:内存信息、中央处理器信息、磁盘统计信息、系统统计信息、交换区信息、系统运行时间及负载信息中的至少一个。

5.根据权利要求1所述的方法,其特征在于,所述资源信息文件的存储路径中包含有所述容器的身份标识。

6.一种容器资源视图隔离的装置,其特征在于,所述装置包括:

文件模块,用于在容器的卷下创建资源信息文件;

挂载模块,用于将所述资源信息文件挂载至所述容器;

更新模块,用于每隔目标时长将所述容器当前的资源信息更新至所述资源信息文件中;

所述装置还包括:

获取模块,用于获取所述容器的资源配置信息;

写入模块,用于将所述资源配置信息写入所述资源信息文件;

所述更新模块,包括:

查询单元,用于每隔目标时长通过lxcfs守护进程查询目标文件,得到所述容器当前的资源信息;其中,所述目标文件为lxcfs生成的proc文件;

更新单元,用于采用所述容器当前的资源信息更新所述资源信息文件中的资源信息。

7.根据权利要求6所述的装置,其特征在于,所述文件模块,具体用于采用所述容器的卷插件在所述卷下创建所述资源信息文件。

8.根据权利要求6所述的装置,其特征在于,所述装置还包括:

移除模块,用于在接收到用户的目标输入的情况下,删除所述容器的卷。

9.根据权利要求6所述的装置,其特征在于,所述资源信息包括:内存信息、中央处理器信息、磁盘统计信息、系统统计信息、交换区信息、系统运行时间及负载信息中的至少一个。

10.根据权利要求6所述的装置,其特征在于,所述资源信息文件的存储路径中包含有所述容器的身份标识。

11.一种电子设备,其特征在于,包括:处理器、存储器以及存储在所述存储器上并可在所述处理器上运行的计算机程序,所述处理器执行所述程序时实现如权利要求1-5中任一所述的容器资源视图隔离的方法。

容器资源视图隔离的方法、装置以及电子设备

技术领域

[0001] 本发明涉及容器技术领域，尤其涉及一种容器资源视图隔离的方法、装置以及电子设备。

背景技术

[0002] 容器和虚拟机都依赖于宿主机才能运行。其中，容器技术提供了不同于传统虚拟机技术的隔离方式，相较于传统的虚拟化技术，容器不是模拟一个完整的操作系统，而是对进程进行隔离。容器相比虚拟机虽然有更多优势，但也导致了容器资源视图无法隔离的问题。

[0003] 目前，为解决容器资源视图无法隔离的问题，大多会引入lxcfs。该lxcfs是一个开源的用户态文件系统，其作为一个常驻服务，启动以后会在指定目录中自行维护一proc文件。容器内的应用读取资源信息时，lxcfs会从其维护的proc文件中读取实际的资源信息，而不是整个宿主机的状态，从而实现了容器资源视图的隔离。

[0004] 然而，但上述方法存在一个稳定性问题，即在lxcfs服务异常退出后再访问proc文件时会出现异常，无法读取到容器的资源信息。

发明内容

[0005] 鉴于上述问题，本发明实施例提供一种容器资源视图隔离的方法、装置以及电子设备，以解决现有技术中lxcfs服务异常退出后无法读取到容器的资源信息的问题。

[0006] 第一方面，本发明实施例提供了一种容器资源视图隔离的方法，所述方法包括：

[0007] 在容器的卷下创建资源信息文件；

[0008] 将所述资源信息文件挂载至所述容器；

[0009] 每隔目标时长将所述容器当前的资源信息更新至所述资源信息文件中。

[0010] 可选地，在容器的卷下创建资源信息文件之后，所述方法还包括：

[0011] 获取所述容器的资源配置信息；

[0012] 将所述资源配置信息写入所述资源信息文件。

[0013] 可选地，所述每隔目标时长将所述容器当前的资源信息更新至所述资源信息文件中，包括：

[0014] 每隔目标时长通过lxcfs守护进程查询目标文件，得到所述容器当前的资源信息；其中，所述目标文件为lxcfs生成的proc文件；

[0015] 采用所述容器当前的资源信息更新所述资源信息文件中的资源信息。

[0016] 可选地，所述在容器的卷下创建资源信息文件，包括：

[0017] 采用所述容器的卷插件在所述卷下创建所述资源信息文件。

[0018] 可选地，在每隔目标时长将所述容器当前的资源信息更新至所述资源信息文件中之后，所述方法还包括：

[0019] 在接收到用户的目标输入的情况下，删除所述容器的卷。

[0020] 可选地，所述资源信息包括：内存信息、中央处理器信息、磁盘统计信息、系统统计信息、交换区信息、系统运行时间及负载信息中的至少一个。

[0021] 可选地，所述资源信息文件的存储路径中包含有所述容器的身份标识。

[0022] 第二方面，本发明实施例提供了一种容器资源视图隔离的装置，所述装置包括：

[0023] 文件模块，用于在容器的卷下创建资源信息文件；

[0024] 挂载模块，用于将所述资源信息文件挂载至所述容器；

[0025] 更新模块，用于每隔目标时长将所述容器当前的资源信息更新至所述资源信息文件中。

[0026] 可选地，所述装置还包括：

[0027] 获取模块，用于获取所述容器的资源配置信息；

[0028] 写入模块，用于将所述资源配置信息写入所述资源信息文件。

[0029] 可选地，所述更新模块，包括：

[0030] 查询单元，用于每隔目标时长通过lxcfs守护进程查询目标文件，得到所述容器当前的资源信息；其中，所述目标文件为lxcfs生成的proc文件；

[0031] 更新单元，用于采用所述容器当前的资源信息更新所述资源信息文件中的资源信息。

[0032] 可选地，所述文件模块，具体用于采用所述容器的卷插件在所述卷下创建所述资源信息文件。

[0033] 可选地，所述装置还包括：

[0034] 移除模块，用于在接收到用户的目标输入的情况下，删除所述容器的卷。

[0035] 可选地，所述资源信息包括：内存信息、中央处理器信息、磁盘统计信息、系统统计信息、交换区信息、系统运行时间及负载信息中的至少一个。

[0036] 可选地，所述资源信息文件的存储路径中包含有所述容器的身份标识。

[0037] 第三方面，本发明实施例提供了一种电子设备，包括：处理器、存储器以及存储在所述存储器上并可在所述处理器上运行的计算机程序，所述处理器执行所述程序时实现如上所述的容器资源视图隔离的方法。

[0038] 第四方面，本发明实施例提供了一种可读存储介质，当所述存储介质中的指令由电子设备的处理器执行时，使得电子设备能够执行如上所述的容器资源视图隔离的方法。

[0039] 在本发明实施例中，可以在容器的卷下创建资源信息文件，进而将该资源信息文件挂载至容器，最后每隔目标时长将所述容器当前的资源信息更新至所述资源信息文件中，利用在容器的卷下创建的资源信息文件维护容器的资源信息，不再直接依赖于lxcfs创建的proc文件，降低了lxcfs的影响，即使lxcfs发生异常，依旧可以读取到容器的资源信息。

附图说明

[0040] 为了更清楚地说明本发明实施例的技术方案，下面将对本发明实施例的描述中所需要使用的附图作简单地介绍，显而易见地，下面描述中的附图仅仅是本发明的一些实施例，对于本领域普通技术人员来讲，在不付出创造性劳动性的前提下，还可以根据这些附图获得其他的附图。

[0041] 图1为本发明实施例提供的一种容器资源视图隔离的方法的步骤流程图；

[0042] 图2为本发明实施例提供的容器资源视图隔离的方法的应用架构图；

[0043] 图3为本发明实施例提供的一种容器资源视图隔离的装置的结构框图；

[0044] 图4为本发明实施例提供的电子设备的结构框图。

具体实施方式

[0045] 下面将结合本发明实施例中的附图，对本发明实施例中的技术方案进行清楚、完整地描述，显然，所描述的实施例是本发明一部分实施例，而不是全部的实施例。基于本发明中的实施例，本领域普通技术人员在没有作出创造性劳动前提下所获得的所有其他实施例，都属于本发明保护的范围。

[0046] 应理解，说明书通篇中提到的“一个实施例”或“一实施例”意味着与实施例有关的特定特征、结构或特性包括在本发明的至少一个实施例中。因此，在整个说明书各处出现的“在一个实施例中”或“在一实施例中”未必一定指相同的实施例。此外，这些特定的特征、结构或特性可以任意适合的方式结合在一个或多个实施例中。

[0047] 在本发明的各种实施例中，应理解，下述各过程的序号的大小并不意味着执行顺序的先后，各过程的执行顺序应以其功能和内在逻辑确定，而不应对本发明实施例的实施过程构成任何限定。

[0048] 参见图1，本发明实施例提供了一种容器资源视图隔离的方法，该方法包括：

[0049] 步骤101：在容器的卷下创建资源信息文件。

[0050] 应当说明的是，容器可以为利用容器技术在宿主机上创建的隔离空间。例如该容器可以为docker容器(container)。卷即为容器下的volume，也可以称之为数据卷，其用于持久化容器的一些数据。资源信息文件为用来维护容器的资源信息的普通文件。其中，该资源信息为容器中各资源的任意信息，例如各资源的状态信息、使用情况等。

[0051] 在该容器为docker容器的情况下，可以先利用docker容器的卷插件(volume plugin)的Create接口创建volume，进而在volume下创建资源文件信息。其中，涉及的参数包括volume的名称，容器的资源配置信息等。volume的名称可以与容器的身份标识相同。

[0052] 步骤102：将资源信息文件挂载至容器。

[0053] 应当说明的是，将资源信息文件挂载至容器，即为将资源信息文件挂载至容器内部的proc文件系统。挂载资源信息文件之后，可以利用容器内部的proc文件系统访问到资源信息文件，进而可以实现读取该资源信息文件中的数据的功能，也可以实现向资源信息文件中写入数据的功能。

[0054] 步骤103：每隔目标时长将容器当前的资源信息更新至资源信息文件中。

[0055] 应当说明的是，目标时长为预先设置的任意时长，例如，该目标时长可以为一分钟、一小时、一天或一周，但不限于此。可以理解的是，使用容器当前的资源信息更新资源信息文件，即为将器当前的资源信息同步到资源信息文件中，使得资源信息文件始终记录有容器最新的资源信息。因此，目标时长也可以理解为同步频率。较佳地，可以在启动容器的过程中执行步骤101、步骤102，从而只需创建一次资源信息文件，且挂载一次该资源信息文件。同时，在容器的运行过程中始终执行步骤103，以保证资源信息文件中资源信息的准确性。

[0056] 在本发明实施例中,可以在容器的卷下创建资源信息文件,进而将该资源信息文件挂载至容器,最后每隔目标时长将容器当前的资源信息更新至资源信息文件中,利用在容器的卷下创建的资源信息文件维护容器的资源信息,不再直接依赖于lxcfs创建的proc文件,降低了lxcfs的影响,即使lxcfs发生异常,依旧可以读取到容器的资源信息。

[0057] 可选地,在容器的卷下创建资源信息文件之后,该方法还包括:

[0058] 获取容器的资源配置信息;

[0059] 将资源配置信息写入资源信息文件。

[0060] 应当说明的是,容器的资源配置信息,可以表征创建容器时,对该容器所能够使用的宿主机上的资源的限额情况。例如,资源配置信息可以包括:限制容器所能够使用的内存上限为512兆,所能够使用的中央处理器的数量为2个。可以理解的是,对该容器所能够使用的宿主机上的资源的进行限额时,可以在启动参数中增加命令:--enable-lxcfs,具体的,该命令可以为:dockerrun -d --enable-lxcfs --memory 512M --cpus 2 nginx,可以限制容器所能够使用的内存上限为512兆,所能够使用的中央处理器的数量为2个。

[0061] 这里,可以采用任意方式获取容器的资源配置信息。获取到的资源配置信息可以作为资源信息文件中初始的资源信息。具体的,资源信息文件的初始内容可以通过容器的资源限额信息结合宿主机的资源信息生成。

[0062] 本申请实施例中,将容器的资源配置信息作为资源信息文件的初始内容,写入资源信息文件,可以避免由于资源信息文件未来得及同步容器当前的资源信息导致的异常问题。

[0063] 可选地,每隔目标时长将容器当前的资源信息更新至资源信息文件中,包括:

[0064] 每隔目标时长通过lxcfs守护进程查询目标文件,得到容器当前的资源信息;其中,目标文件为lxcfs生成的proc文件;

[0065] 采用容器当前的资源信息更新资源信息文件中的资源信息。

[0066] 应当说明的是,lxcfs守护进程,即为lxcfs daemon。lxcfs生成的proc文件,即为lxcfs维护的用于读取容器的资源信息的文件。通过lxcfs守护进程查询目标文件,即为利用lxcfs从容器对应的cgroup中读取到容器实际的资源信息。具体的,可以利用nsenter命令进入容器的pid namespace(进程控制符命名空间)中,去cat lxcfs维护的proc文件。

[0067] 这里,每获取一次容器当前的资源信息,就会利用最新获取的资源信息更新一次资源信息文件。较佳地,可以利用卷插件实现定时更新资源信息文件的功能。也就是说,卷插件定时通过lxcfs daemon去同步容器的实时资源信息到卷插件创建的资源信息文件中。

[0068] 本发明实施例中,利用lxcfs可以直接获取容器当前的资源信息,进而将其同步到资源信息文件中,便于实现,

[0069] 可选地,在容器的卷下创建资源信息文件,包括:

[0070] 采用容器的卷插件在卷下创建资源信息文件。

[0071] 应当说明的是,卷插件即为volume plugin,其利用了docker plugin机制,实现容器下的卷,使得容器访问存储都使用统一的接口(文件接口),对于容器中的进程来说,volume就是一个已挂载的目录,容器内进程使用该目录就与普通目录一样。容器内访问资源信息文件也就像访问普通目录一样。

[0072] 本发明实施例中,基于docker plugin机制,使用了卷插件创建资源信息文件,从

而使得容器内访问资源信息文件也就像访问普通目录一样简单方便。

[0073] 可选地，在每隔目标时长将容器当前的资源信息更新至资源信息文件中之后，该方法还包括：

[0074] 在接收到用户的目标输入的情况下，删除容器的卷。

[0075] 应当说明的是，目标输入可以为针对显示屏的点击、滑动、长按等操作，但不限于此。目标输入还可以为针对实体按键的按压操作。可以理解的是，在不需要使用容器的情况下，通常需要将该容器删除，而作为容器的相关文件的资源信息文件应当与容器保持同步。因此，在删除容器时，只需删除容器的卷，就可以将资源信息文件与容器同步被删除。具体的，在容器删除过程中，调用volume plugin的Remove接口，删除卷插件创建的资源信息文件，完成清理操作。

[0076] 本发明实施例中，可以在删除容器的同时，将容器的资源信息文件同步删除，避免无用的资源信息文件占用宿主机的存储空间。

[0077] 可选地，资源信息包括：内存信息、中央处理器信息、磁盘统计信息、系统统计信息、交换区信息、系统运行时间及负载信息中的至少一个。

[0078] 应当说明的是，在资源信息涉及多个资源的情况下，可以针对每个资源创建一个资源信息文件。具体的，这里的资源信息可以包括：/proc/cpuinfo(cpu信息)，/proc/meminfo(内存信息)，/proc/diskstats(磁盘统计信息)，/proc/stat(系统统计信息)，/proc/swaps(交换区信息)，/proc/uptime(系统运行时间及负载信息)。例如在资源信息涉及内存和中央处理器时，创建如下资源信息文件分别维护内存信息和中央处理器信息：/opts/lxcfs-plugin/{id}/cpuinfo;/opts/lxcfs-plugin/{id}/meminfo。其中，内存信息包括与内存相关的信息，例如内存的限额、内存当前的使用情况、使用状态等。

[0079] 可选地，资源信息文件的存储路径中包含有容器的身份标识。

[0080] 应当说明的是，在宿主机上的容器数量较多时，为便于区分不同容器的资源信息文件，可以在资源信息文件的存储路径中添加容器的身份标识。例如创建如下资源信息文件分别维护内存信息和中央处理器信息：/opts/lxcfs-plugin/{id}/cpuinfo;/opts/lxcfs-plugin/{id}/meminfo。其中，两个存储路径中的id即为容器的ID(身份标识)。

[0081] 本发明实施例中，将容器的身份标识添加到存储路径中，便于区分不同容器资源信息文件，避免不同容器的资源信息文件发生混淆。

[0082] 参见图2，本发明实施例提供了一种容器资源视图隔离的方法的架构示意图，具体的，在宿主机上分别部署运行docker守护进程、lxcfs守护进程以及卷插件，其中，lxcfs守护进程以及卷插件由systemd管理。其中，docker守护进程支持容器创建/启动增加参数--enable-lxcfs使能容器资源视图隔离。

[0083] 在创建宿主机上创建容器时，利用卷插件的Create接口创建卷，涉及的参数包括卷名(值等于容器ID)，容器的资源配置信息(中央处理器和内存的限额)。

[0084] 在启动容器时，调用卷插件的Mount接口在宿主机上创建该容器的proc普通文件(即图2中的资源信息文件)，如/opts/lxcfs-plugin/{id}/cpuinfo，/opts/lxcfs-plugin/{id}/meminfo等，其中id为容器ID，区分不同容器实例，资源信息文件初始内容通过容器的资源限额值结合宿主机的资源信息生成，确保cpuinfo，meminfo等文件内容为该容器的资源信息，然后将卷插件创建的proc普通文件挂载到容器的proc系统。在创建容器实例之

后可以自由访问到proc普通文件。这里，可以在容器的启动参数中增加--enable-lxcfs，如docker run -d--enable-lxcfs --memory 512M --cpus 2 nginx，启动过程中自动挂载卷插件创建的proc普通文件(相当于上述发明实施例中的资源信息文件)。docker exec进入容器，执行top，free等运维工具，输出的结果为该容器对应的资源信息，非主机资源信息。

[0085] 模拟lxcfs守护进程故障重启，经验证可以正常读取到容器的资源信息。

[0086] 若需要删除该容器，则删除创建的容器实例，利用卷插件删除其创建的卷。proc普通文件将跟随卷同步被删除。

[0087] 本发明实施例中，基于lxcfs实现容器资源视图隔离的基础，不直接挂载lxcfs管理的proc文件，通过挂载卷插件创建的proc普通文件，可以实现在lxcfs服务异常重启后无需重启容器和重新挂载，即可读取到容器的资源信息，增强了资源视图隔离的稳定性，同时提高业务的稳定性。

[0088] 以上介绍了本发明实施例提供的容器资源视图隔离的方法，下面将结合附图介绍本发明实施例提供的容器资源视图隔离的装置。

[0089] 如图3所示，本发明实施例提供了一种容器资源视图隔离的装置，该装置包括：

[0090] 文件模块31，用于在容器的卷下创建资源信息文件；

[0091] 挂载模块32，用于将资源信息文件挂载至容器；

[0092] 更新模块33，用于每隔目标时长将容器当前的资源信息更新至资源信息文件中。

[0093] 可选地，该装置还包括：

[0094] 获取模块，用于获取容器的资源配置信息；

[0095] 写入模块，用于将资源配置信息写入资源信息文件。

[0096] 可选地，更新模块33，包括：

[0097] 查询单元，用于每隔目标时长通过lxcfs守护进程查询目标文件，得到容器当前的资源信息；其中，目标文件为lxcfs生成的proc文件；

[0098] 更新单元，用于采用容器当前的资源信息更新资源信息文件中的资源信息。

[0099] 可选地，文件模块31，具体用于采用容器的卷插件在卷下创建资源信息文件。

[0100] 可选地，该装置还包括：

[0101] 移除模块，用于在接收到用户的目标输入的情况下，删除容器的卷。

[0102] 可选地，资源信息包括：内存信息、中央处理器信息、磁盘统计信息、系统统计信息、交换区信息、系统运行时间及负载信息中的至少一个。

[0103] 可选地，资源信息文件的存储路径中包含有容器的身份标识。

[0104] 本发明实施例中，可以在容器的卷下创建资源信息文件，进而将该资源信息文件挂载至容器，最后每隔目标时长将容器当前的资源信息更新至资源信息文件中，利用在容器的卷下创建的资源信息文件维护容器的资源信息，不再直接依赖于lxcfs创建的proc文件，降低了lxcfs的影响，即使lxcfs发生异常，依旧可以读取到容器的资源信息。

[0105] 本申请实施例提供的容器资源视图隔离的装置能够实现图1和图2的方法实施例实现的各个过程，实现相同的技术效果，为避免重复，这里不再赘述。

[0106] 另一方面，本发明实施例还提供了一种电子设备，包括存储器、处理器、总线以及存储在存储器上并可在处理器上运行的计算机程序，所述处理器执行所述程序时实现上述容器资源视图隔离的方法中的步骤。

[0107] 举个例子如下,图4示出了一种电子设备的实体结构示意图。

[0108] 如图4所示,该电子设备可以包括:处理器(processor)410、通信接口(CommunicationsInterface)420、存储器(memory)430和通信总线440,其中,处理器410,通信接口420,存储器430通过通信总线440完成相互间的通信。处理器410可以调用存储器430中的逻辑指令,以执行如下方法:

[0109] 在容器的卷下创建资源信息文件;

[0110] 将资源信息文件挂载至容器;

[0111] 每隔目标时长将容器当前的资源信息更新至资源信息文件中。

[0112] 此外,上述的存储器430中的逻辑指令可以通过软件功能单元的形式实现并作为独立的产品销售或使用时,可以存储在一个计算机可读取存储介质中。基于这样的理解,本发明的技术方案本质上或者说对现有技术做出贡献的部分或者该技术方案的部分可以以软件产品的形式体现出来,该计算机软件产品存储在一个存储介质中,包括若干指令用以使得一台计算机设备(可以是个人计算机,服务器,或者网络设备等)执行本发明各个实施例所述方法的全部或部分步骤。而前述的存储介质包括:U盘、移动硬盘、只读存储器(ROM,Read-Only Memory)、随机存取存储器(RAM,Random Access Memory)、磁碟或者光盘等各种可以存储程序代码的介质。

[0113] 再一方面,本发明实施例还提供了一种计算机可读存储介质,其上存储有计算机程序,该计算机程序被处理器执行时实现以执行上述各实施例提供的容器资源视图隔离的方法,例如包括:

[0114] 在容器的卷下创建资源信息文件;

[0115] 将资源信息文件挂载至容器;

[0116] 每隔目标时长将容器当前的资源信息更新至资源信息文件中。

[0117] 以上所描述的装置实施例仅仅是示意性的,其中所述作为分离部件说明的单元可以是或者也可以不是物理上分开的,作为单元显示的部件可以是或者也可以不是物理单元,即可以位于一个地方,或者也可以分布到多个网络单元上。可以根据实际的需要选择其中的部分或者全部模块来实现本实施例方案的目的。本领域普通技术人员在不付出创造性的劳动的情况下,即可以理解并实施。

[0118] 通过以上的实施方式的描述,本领域的技术人员可以清楚地了解到各实施方式可借助软件加必需的通用硬件平台的方式来实现,当然也可以通过硬件。基于这样的理解,上述技术方案本质上或者说对现有技术做出贡献的部分可以以软件产品的形式体现出来,该计算机软件产品可以存储在计算机可读存储介质中,如ROM/RAM、磁碟、光盘等,包括若干指令用以使得一台计算机设备(可以是个人计算机,服务器,或者网络设备等)执行各个实施例或者实施例的某些部分所述的方法。

[0119] 最后应说明的是:以上实施例仅用以说明本发明的技术方案,而非对其限制;尽管参照前述实施例对本发明进行了详细的说明,本领域的普通技术人员应当理解:其依然可以对前述各实施例所记载的技术方案进行修改,或者对其中部分技术特征进行等同替换;而这些修改或者替换,并不使相应技术方案的本质脱离本发明各实施例技术方案的精神和范围。

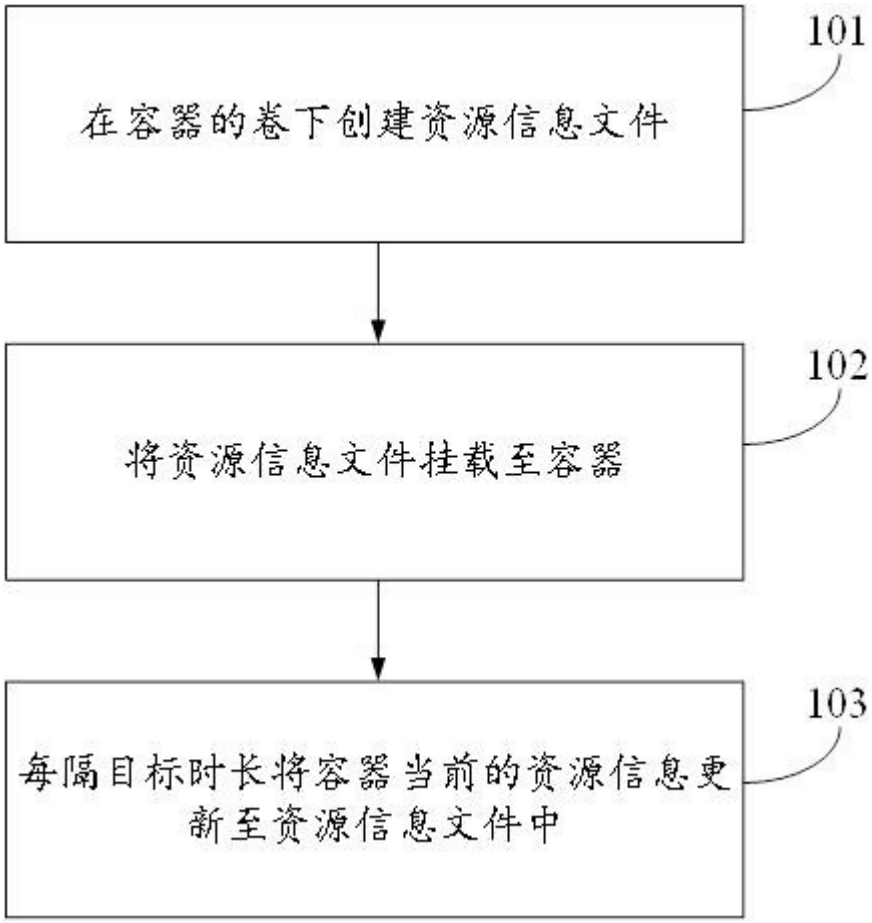

101
在容器的卷下创建资源信息文件
102
将资源信息文件挂载至容器
103
每隔目标时长将容器当前的资源信息更新至资源信息文件中

图 1

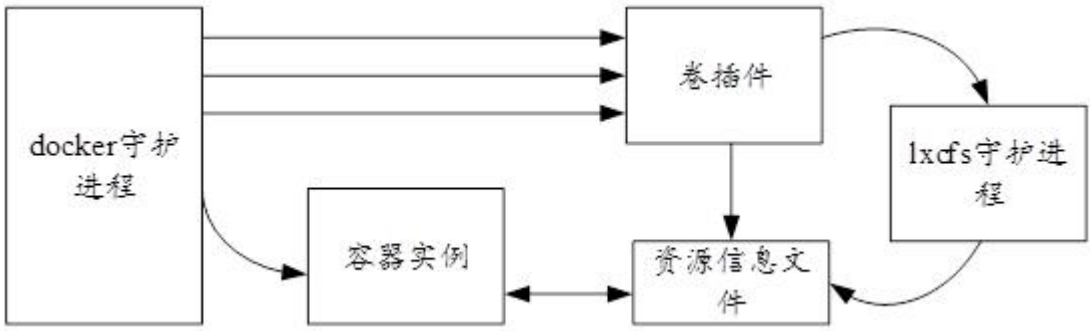

docker守护进程
卷插件
lxcfs守护进程
容器实例
资源信息文件

图 2

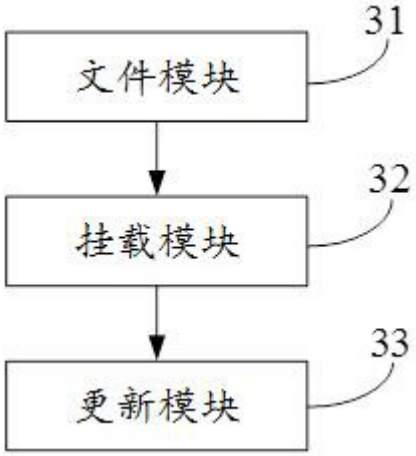


图 3

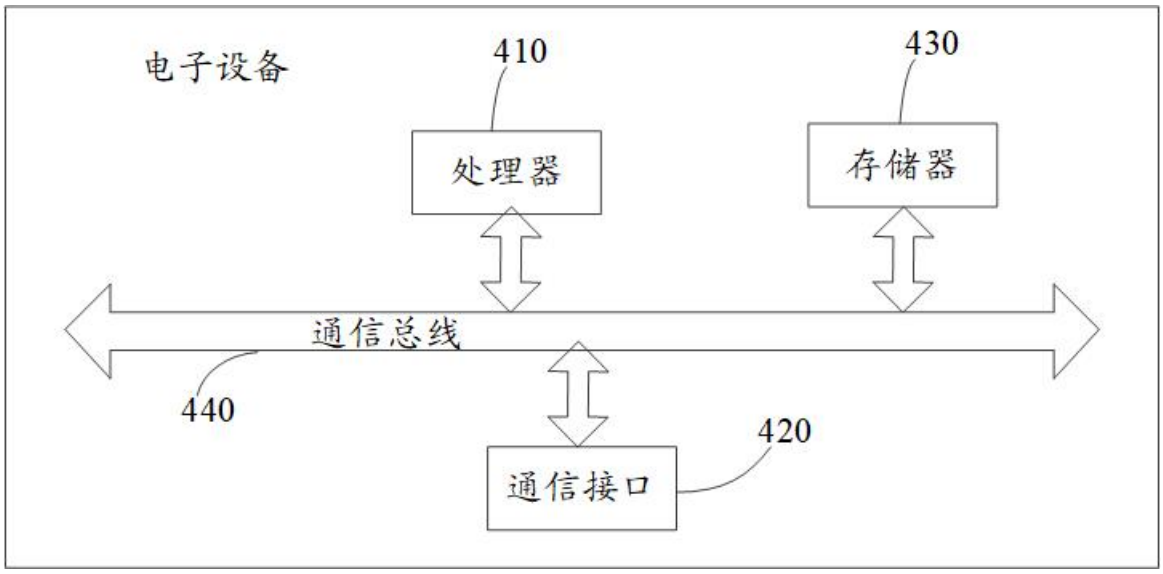


图 4