(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 116561824 A
(43)申请公布日 2023.08.08

(21)申请号 202310488155.3
(22)申请日 2023.04.28
(71)申请人 南方科技大学
地址 518000 广东省深圳市南山区学苑大道1088号
申请人 支付宝(杭州)信息技术有限公司
(72)发明人 张锋巍 张一鸣 胡煜鑫 黄浩洋 闫守孟 何征宇
(74)专利代理机构 北京亿腾知识产权代理事务所(普通合伙) 11309
专利代理师 陈霁 周良玉
(51)Int.Cl.
G06F 21/78(2013.01)
G06F 21/72(2013.01)
G06F 21/74(2013.01)
G06F 21/60(2013.01)

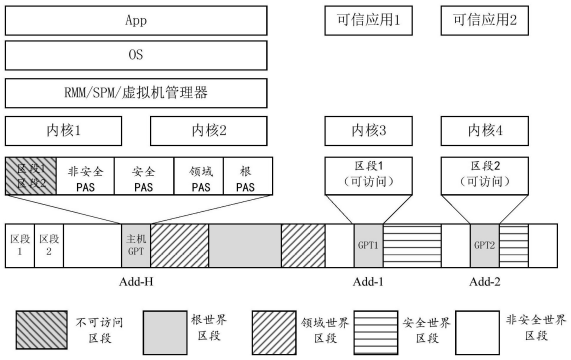
权利要求书2页 说明书9页 附图3页

(54)发明名称

在机密计算架构中管理内存的方法和装置

(57)摘要

本说明书实施例提供一种在机密计算架构中管理内存的方法和装置。机密计算架构包括，安全世界，领域世界，非安全世界，根世界；对应方法包括：非安全世界的操作系统在非安全世界的内存中，为非安全世界的第一可信应用分配第一内存区段。根世界中的根监视器更新总颗粒度保护表GPT，使得在更新后的总GPT中，所述第一内存区段的访问权限被设置为不可访问。此外，根监视器还针对第一可信应用创建第一颗粒度保护表GPT，在所述第一GPT中，所述第一内存区段的访问权限被设置为可访问的非安全内存。



CN 116561824 A

1.一种在机密计算架构中管理内存的方法,所述机密计算架构包括,安全世界,领域世界,非安全世界,根世界;所述方法包括:

非安全世界的操作系统在非安全世界的内存中,为非安全世界的第一可信应用分配第一内存区段;

根世界中的根监视器更新总颗粒度保护表GPT,使得在更新后的总GPT中,所述第一内存区段的访问权限被设置为不可访问;

所述根监视器针对所述第一可信应用创建第一颗粒度保护表GPT,在所述第一GPT中,所述第一内存区段的访问权限被设置为可访问的非安全内存。

2.根据权利要求1所述的方法,其中,为非安全世界的第一可信应用分配第一内存区段包括:

所述操作系统从预先分配的内存池中确定出所述第一内存区段;所述内存池是采用连续内存分配器CMA分配的一段内存页物理地址连续的内存。

3.根据权利要求1所述的方法,其中,在所述第一GPT中,所述安全世界,领域世界和根世界的内存区段,以及非安全世界中分配给其他可信应用的内存区段,均被设置为不可访问。

4.根据权利要求1所述的方法,还包括:

所述根监视器对所述第一内存区段进行有效性验证,所述有效性验证包括,验证是否与其他已分配内存重叠。

5.根据权利要求1所述的方法,其中,所述总GPT和所述第一GPT存储在内存的根世界部分中。

6.根据权利要求1所述的方法,还包括:

响应于CPU发出内存访问请求,所述根监视器根据CPU当前运行的应用,从已维护的GPT集中确定目标GPT,并将目标GPT设置为对所述内存访问请求进行颗粒度保护检查的基础;所述已维护的GPT集包括所述总GPT和所述第一GPT。

7.根据权利要求6所述的方法,其中,所述根监视器根据CPU当前运行的应用,从已维护的GPT集中确定目标GPT,包括:

若CPU当前运行的应用为所述第一可信应用,所述根监视器确定所述目标GPT为所述第一GPT;

若CPU当前运行的应用不是用户态可信应用,所述根监视器确定所述目标GPT为所述总GPT。

8.根据权利要求6所述的方法,其中,将目标GPT设置为对所述内存访问请求进行颗粒度保护检查的基础,包括:

获取所述目标GPT在内存中的物理地址作为目标基地址;

将所述CPU的GPT基地址寄存器设置为所述目标基地址。

9.根据权利要求6所述的方法,其中,所述CPU包括第一内核和第二内核;所述根监视器根据CPU当前运行的应用,从已维护的GPT集中确定目标GPT,包括:

根据第一内核当前运行的第一应用,确定第一内核对应的第一目标GPT;

根据第二内核当前运行的第二应用,确定第二内核对应的第二目标GPT。

10.根据权利要求9所述的方法,其中,

所述第一应用为所述第一可信应用,所述第一目标GPT为所述第一GPT;

所述第二应用不属于用户态可信应用,所述第二目标GPT为所述总GPT。

11.根据权利要求9所述的方法,其中,所述已维护的GPT集还包括针对第二可信应用的第二GPT;

所述第一应用为所述第一可信应用,所述第一目标GPT为所述第一GPT;

所述第二应用为所述第二可信应用,所述第二目标GPT为所述第二GPT。

12.根据权利要求9所述的方法,其中,所述第一内核和第二内核之间禁用TLB共享功能。

13.根据权利要求1所述的方法,还包括,

响应于移除所述第一可信应用的指令,所述根监视器清空所述第一内存区段中的内容,在内存中清除所述第一GPT表,并在所述总GPT表中,将第一内存区段设置为常规的非安全世界内存。

14.一种机密计算架构中的根监视器,所述机密计算架构包括,安全世界,领域世界,非安全世界和根世界;所述根监视器位于所述根世界中,并包括内存管理模块,所述内存管理模块配置用于:

响应于非安全世界的操作系统在非安全世界的内存中,为非安全世界的第一可信应用分配第一内存区段,更新总颗粒度保护表GPT,使得在更新后的总GPT中,所述第一内存区段的访问权限被设置为不可访问;

针对所述第一可信应用创建第一颗粒度保护表GPT,在所述第一GPT中,所述第一内存区段的访问权限被设置为可访问的非安全内存。

15.根据权利要求14所述的根监视器,还包括内存隔离模块,配置用于:

响应于CPU发出内存访问请求,根据CPU当前运行的应用,从已维护的GPT集中确定目标GPT,并将目标GPT设置为对所述内存访问请求进行颗粒度保护检查的基础;所述已维护的GPT集包括所述总GPT和所述第一GPT。

16.根据权利要求14或15所述的根监视器,所述根监视器实现为安全固件。

17.一种计算设备,包括存储器和处理器,所述计算设备形成机密计算架构,所述机密计算架构包括,安全世界,领域世界,非安全世界和根世界;所述根世界包括权利要求14-16任一项所述的根监视器。

在机密计算架构中管理内存的方法和装置

技术领域

[0001] 本说明书一个或多个实施例涉及机密计算框架，尤其涉及一种在机密计算框架中管理内存的方法及装置。

背景技术

[0002] 随着各行业计算技术的发展，以及云端和终端用户的增加，人们将大量数据存储在各种计算机设备中。在行业发展的同时，人们对于设备和数据安全的关注也在日益增加。为了确保设备和数据的安全性，各个架构厂商也分别提出了各自的解决方案，如ARM提出了可信区技术(TrustZone)，AMD提出了安全虚拟机加密技术(SEV)，英特尔提出了软件防护扩展(SGX)技术，等等。这些解决方案为用户提供一个安全的可信执行环境，用于机密地保存和处理数据，使其免受不可信的内核与传统应用程序的损害。以Arm可信区技术为例，它将传统内核和应用程序的运行环境视作为非安全世界，并创建了一个隔离的安全世界，以及定义了具有最高权限的安全层用于世界切换。非安全世界将无法直接访问安全世界，需要经过安全层的固件验证才能访问特定的资源。

[0003] 在Arm可信区技术的框架下，用户态的应用一般运行于非安全世界中，因此，用户态应用只能处于安全级别相对较低的状态。存在对此进行改进的需求。

发明内容

[0004] 本说明书一个或多个实施例描述了一种在机密计算架构中管理内存的方法及装置，能够基于已有的机密计算架构的硬件特性，在非安全世界的用户态空间中，为用户态应用部署机密计算环境。

[0005] 根据第一方面，提供一种在机密计算架构中管理内存的方法，所述机密计算架构包括，安全世界，领域世界，非安全世界，根世界；所述方法包括：

[0006] 非安全世界的操作系统在非安全世界的内存中，为非安全世界的第一可信应用分配第一内存区段；

[0007] 根世界中的根监视器更新总颗粒度保护表GPT，使得在更新后的总GPT中，所述第一内存区段的访问权限被设置为不可访问；

[0008] 所述根监视器针对所述第一可信应用创建第一颗粒度保护表GPT，在所述第一GPT中，所述第一内存区段的访问权限被设置为可访问的非安全内存。

[0009] 在一种实现方式中，所述操作系统从预先分配的内存池中确定出所述第一内存区段；所述内存池是采用连续内存分配器CMA分配的一段内存页物理地址连续的内存。

[0010] 根据一个实施例，在所述第一GPT中，所述安全世界，领域世界和根世界的内存区段，以及非安全世界中分配给其他可信应用的内存区段，均被设置为不可访问。

[0011] 根据一种实施方式，上述方法还包括：所述根监视器对所述第一内存区段进行有效性验证，所述有效性验证包括，验证是否与其他已分配内存重叠。

[0012] 在一个实施例中，所述总GPT和所述第一GPT存储在内存的根世界部分中。

[0013] 根据一种实施方式，上述方法还包括：

[0014] 响应于CPU发出内存访问请求，所述根监视器根据CPU当前运行的应用，从已维护的GPT集中确定目标GPT，并将目标GPT设置为对所述内存访问请求进行颗粒度保护检查的基础；所述已维护的GPT集包括所述总GPT和所述第一GPT。

[0015] 进一步的，在一个实施例中，根监视器根据CPU当前运行的应用，从已维护的GPT集中确定目标GPT，可以包括：若CPU当前运行的应用为所述第一可信应用，所述根监视器确定所述目标GPT为所述第一GPT；若CPU当前运行的应用不是用户态可信应用，所述根监视器确定所述目标GPT为所述总GPT。

[0016] 在一个实施例中，将目标GPT设置为对所述内存访问请求进行颗粒度保护检查的基础，具体包括：获取所述目标GPT在内存中的物理地址作为目标基地址；将所述CPU的GPT基地址寄存器设置为所述目标基地址。

[0017] 在一种实现方式中，所述CPU为多核CPU，其中包括第一内核和第二内核；所述根监视器根据CPU当前运行的应用，从已维护的GPT集中确定目标GPT，包括：

[0018] 根据第一内核当前运行的第一应用，确定第一内核对应的第一目标GPT；

[0019] 根据第二内核当前运行的第二应用，确定第二内核对应的第二目标GPT。

[0020] 在一个示例中，所述第一应用为所述第一可信应用，所述第一目标GPT为所述第一GPT；所述第二应用不属于用户态可信应用，所述第二目标GPT为所述总GPT。

[0021] 在另一示例中，所述第一应用为所述第一可信应用，所述第一目标GPT为所述第一GPT；所述第二应用为第二可信应用，所述第二目标GPT为针对第二可信应用创建的第二GPT。

[0022] 在一个实施例中，所述第一内核和第二内核之间禁用TLB共享功能。

[0023] 根据一种实施方式，上述方法还包括：

[0024] 响应于移除所述第一可信应用的指令，所述根监视器清空所述第一内存区段中的内容，在内存中清除所述第一GPT表，并在所述总GPT表中，将第一内存区段设置为常规的非安全世界内存。

[0025] 根据第二方面，提供了一种机密计算架构中的根监视器，所述机密计算架构包括，安全世界，领域世界，非安全世界和根世界；所述根监视器位于所述根世界中，并包括内存管理模块，所述内存管理模块配置用于：

[0026] 响应于非安全世界的操作系统在非安全世界的内存中，为非安全世界的第一可信应用分配第一内存区段，更新总颗粒度保护表GPT，使得在更新后的总GPT中，所述第一内存区段的访问权限被设置为不可访问；

[0027] 针对所述第一可信应用创建第一颗粒度保护表GPT，在所述第一GPT中，所述第一内存区段的访问权限被设置为可访问的非安全内存。

[0028] 根据第三方面，提供了一种计算设备，包括存储器和处理器，所述计算设备形成机密计算架构，所述机密计算架构包括，安全世界，领域世界，非安全世界和根世界；所述根世界包括如第二方面所述的根监视器。

[0029] 在本说明书实施例提供的方案中，提出改进的机密计算架构，其中包括安全世界，非安全世界，领域世界和根世界。在根世界中提供根监视器，根监视器通过配置、管理和切换多个颗粒度保护GPT表，实现对用户态可信应用的内存隔离管理，从而为用户态可信应用

创建机密、隔离的计算环境。

附图说明

[0030] 为了更清楚地说明本发明实施例的技术方案，下面将对实施例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的前提下，还可以根据这些附图获得其它的附图。

[0031] 图1示出Arm机密计算架构的示意图；

[0032] 图2示出机密计算架构中各个世界对物理地址空间的访问权限控制；

[0033] 图3示出根据一个实施例改进的机密计算架构的示意图；

[0034] 图4示出根据一个实施例中在机密计算架构中管理内存的方法；

[0035] 图5示出多核多GPT表的配置示例。

具体实施方式

[0036] 下面结合附图，对本说明书提供的方案进行描述。

[0037] 为了保证数据的安全性，ARM提供了TrustZone可信区技术。在该技术中，将传统内核和应用程序的运行环境视为非安全世界(Normal World)，在此之外创建了一个隔离的安全世界(Secure World)，并定义了具有最高权限的安全层用于世界切换。

[0038] 具体的，在Armv8-A架构中，CPU核基于特权划分将异常划分为4个等级，EL0至EL3，其中，EL0表示应用级，EL1用于系统内核(kernal)，EL2表示虚拟机管理器(hypervisor)，EL3表示安全层监视器。这四个等级也可用于表示运行环境的权限等级。在TrustZone可信区技术中，CPU安全状态被划分为非安全(Normal)状态和安全状态。EL0和EL1可以运行于任意状态，例如，可以在非安全世界的EL1中执行非可信的操作系统OS(untrusted OS)，在安全世界的EL1中执行可信OS。EL2可用于安全状态。EL3即安全层监视器，永远存在于安全世界，用于执行安全状态的切换。

[0039] 在该架构下，非安全世界无法直接访问安全世界，需要经过安全层监视器的验证才能访问特定的资源。敏感或机密的数据，以及高权限的软件应用运行于安全世界，从而为这些机密数据提供一种可信执行环境TEE。

[0040] 在以上的TrustZone基础架构基础上，ARM近来又发布了改进的Arm机密计算架构CCA(Confidential Compute Architecture)。Arm机密计算架构是Armv9-A架构的一部分，其在原本的TrustZone架构基础上引入了领域管理扩展，该扩展在可信区技术中已经存在的非安全世界与安全世界之外，额外引入领域(Realm)世界和根(Root)世界。为了支持不同世界的隔离，CCA架构在硬件层提供领域管理扩展RME(Realm Management Extension)组件，来扩展隔离模式。

[0041] 图1示出Arm机密计算架构的示意图。如图1所示，在Arm机密计算机构CCA中，运行环境被划分为四个世界：安全世界，非安全世界，领域世界和根世界。根世界中运行着拥有最高权限的根世界监视器，负责世界之间的隔离和通信。领域世界用于为虚拟机提供名为机密领域的受保护的虚拟机机密计算环境。领域世界中运行着领域管理监视器RMM，负责管理领域虚拟机的执行以及与非安全世界的交互。用户可以将虚拟机作为领域虚拟机放入机

密领域中,隔离来自外界软件的非法访问。具体的,用户可以通过非安全世界中的虚拟机管理器创建虚拟机,通过领域管理监视器RMM将其转入到领域世界,成为领域虚拟机。领域管理监视器RMM会负责与机密领域安全相关的检查和保护。领域虚拟机之间使用虚拟化技术相互隔离,领域管理监视器会负责管理不同领域虚拟机的可访问地址空间。领域虚拟机不需要相信非安全世界和安全世界,只需要相信领域管理监视器和根世界监视器。

[0042] 相应的,Arm机密计算架构CCA将内存的物理地址空间PAS(Physical address spaces)也划分为四个世界。图2示出机密计算架构中各个世界的安全状态对物理地址空间的访问权限控制。如图2所示,根世界具有最高的访问权限,可以访问所有四个世界的地址空间。非安全世界具有最低的访问权限,仅可以访问非安全世界的地址空间。安全世界和领域世界,则可以访问非安全世界的地址空间,以及属于自己世界的地址空间。

[0043] 在Arm机密计算架构中,不同世界的地址空间访问控制,通过构建颗粒度保护表GPT(Granule Protection Table),并基于该GPT表进行颗粒度保护检查(Granule Protection Check)而实现。具体的,机密计算架构CCA在内存中维护一个颗粒度保护表GPT,其中记录细粒度的每段物理内存的安全状态。典型的,记录的粒度是以内存页(4KB大小的区段)为单位。如此,GPT表中记录每个内存页的安全状态和访问权限。当内存页的分配在不同世界发生迁移和变更,则可以动态更新GPT中的条目。

[0044] 当处理器访问内存时,硬件层中的前述RME组件执行颗粒度保护检查GPC。在检查中,获取当前CPU的安全状态,并通过读取GPT表获取请求访问的内存页的安全状态,检查二者是否匹配。如果没有通过GPC检查(例如,如果非安全世界的主机OS请求访问领域世界的内存),则会发出颗粒度保护异常信号,从而拒绝此次内存访问,由此保证世界之间的隔离。并且,由于颗粒度保护检查GPC会在内存和缓存访问之前进行,因此,即使内存信息已经被提前加载到缓存中,该检查也能在读取缓存内容之前中止非法内存访问操作。

[0045] 通过以上的隔离机制,Arm机密计算架构进一步为领域世界的领域虚拟机提供了隔离的机密计算环境。然而,多数第三方开发的普通应用只能运行在非安全世界的用户态环境中。在许多情况下,同样希望保证用户态应用的运行安全。上述已有的Arm机密计算框架并不支持为用户态的应用或进程提供隔离的计算环境。

[0046] 有鉴于此,在本说明书的实施例中提出一种方案,基于Arm机密计算架构的硬件特性,在不影响Arm机密计算架构原有功能的基础上,提供部署用户态机密计算环境的服务,以此扩展Arm机密计算架构的功能。

[0047] 图3示出根据一个实施例改进的机密计算架构的示意图。如图3所示,该改进的机密计算架构利用已有的Arm机密计算架构中引入的RME硬件原语,来运行和驻留根世界的监视器,后续又称为根监视器。根监视器运行在最高的权限级别(即EL3级),以提供隔离机制。该根监视器提供有限的一些接口API,供用户在非安全世界(Normal world)的用户态空间中部署可信应用。每个可信应用与其他可信应用、不可信OS/虚拟机管理器hypervisor以及特权软件(例如,可信OS,领域管理监视器RMM,安全划分管理器SPM等)均保持隔离。

[0048] 显然,上述改进的机密计算架构相比于现有的Arm机密计算架构的不同之处在于,其在非安全世界中为用户态进程提供机密计算环境,并且用户态进程只需要相信位于根世界的根监视器的安全性。为了实现部署用户空间机密计算环境的功能,改进的机密计算框架在根监视器中额外实现了以下模块:内存隔离模块,内存管理模块,和生命周期管理模

块。

[0049] 内存隔离模块负责用户态机密计算环境与外界的内存隔离。为了实现这样的隔离,在本说明书的实施例中,复用已有的颗粒度保护表GPT机制,但是不同的是,为了实现可信应用级的隔离,配置和管理多个GPT表。具体来说,内存隔离模块维护一个与已有的GPT类似的总颗粒度保护表GPT,在该总GPT表中,分配给安全世界、根世界、领域世界的内存区域如常记录,但是分配给可信应用的内存区段的访问权限被设置为不可访问。此外,内存隔离模块还针对每个可信应用维护一张专属的GPT表,在专属的GPT表中,分配给对应可信应用的内存区段被设置为可访问的非安全内存,其他内存区段被设置为不可访问。内存隔离模块可以在多个GPT表之间进行切换。通过上述切换,在运行可信应用时,利用该可信应用专属的GPT表来进行颗粒度保护检查GPC,使得可信应用可以正常运行,但是不能访问其他内存区域。当运行其他软件时,利用总GPT表来进行GPC检查。由于总GPT表中分配给可信应用的内存区段的访问权限被设置为不可访问,如此,其他外界软件均不能访问分配给可信应用的内存区段,从而为可信应用构建了阻止外界软件非法访问的隔离机密环境。

[0050] 内存管理模块负责用户态机密计算环境的内存管理的调度,进行内存分配转发和结果安全检查。在针对某个可信应用创建用户态机密计算环境时,会由主机负责机密计算环境的内存分配。之后,根监视器中的内存管理模块会负责验证主机所分配的内存是否合法,并配置颗粒度保护表以施加内存隔离。具体的,内存管理模块配置总GPT表,使得分配给该可信应用的内存区段的访问权限被设置为不可访问。此外,还创建并初始化该可信应用专属的GPT表,在其中将分配给对应可信应用的内存区段设置为可访问的非安全内存,将其他内存区段设置为不可访问。

[0051] 生命周期管理模块负责用户态机密计算环境的生命周期管理,包括管理用户态机密计算环境的创建、运行和销毁。在创建用户态机密计算环境时,根监视器会为其初始化一张专属的颗粒度保护表,并将用户态机密计算环境信息记录在结构体中用来管理。在每次执行可信应用之前,根监视器可以通过配置颗粒度保护表基址寄存器,让当前核使用对应的专属颗粒度保护表进行内存访问检查,并在退出用户态机密计算环境时更改使用的颗粒度保护表。在销毁用户态机密计算环境时,根监视器会回收其颗粒度保护表和结构体,并且清空机密计算环境信息,避免泄露机密数据。

[0052] 下面结合单个可信应用的生命周期,描述为其部署和维护隔离计算环境的过程。

[0053] 图4示出根据一个实施例中在机密计算架构中管理内存的方法。如前所述,该机密计算架构包括安全世界,领域世界,非安全世界,根世界;根世界中具有根监视器。上述根监视器实现为安全固件的形式。

[0054] 当用户想要在非安全世界中创建一个可信应用时,用户可以利用特定的引导工具发出创建请求,其中该特定的引导工具基于根监视器提供的接口API构建。如此,根监视器可以确定用户将要创建的应用是一个可信应用,需要为其部署隔离的机密计算环境。下文将该可信应用称为第一可信应用。

[0055] 为了确保可信计算基TCB(Trusted Computing Base)尽量小,在本说明书的实施例中,要求根监视器只负责内存隔离等安全机制,而不负责诸如系统调用处理,内存分配等非安全职责。因此,创建第一可信应用时所要求的内存分配,不必由根监视器执行,而是由非安全世界中的主机操作系统执行。

[0056] 相应的，如图4所示，在创建第一可信应用时，响应于用户的创建请求，在步骤41，非安全世界的操作系统OS在非安全世界的内存中，为非安全世界的第一可信应用分配第一内存区段。然后，该操作系统OS将第一内存区段的指示信息传递给根监视器。

[0057] 在一个实施例中，为了进一步加快执行性能，使用主机操作系统的连续内存分配器CMA(Contiguous Memory Allocator)提前为用户态机密计算环境分配一个内存池，该内存池中内存页物理地址连续。这样，主机操作系统可以在一次请求中将内存分配信息(基地址和长度)传输给根监视器。当可信应用申请内存时，会先查找内存池，避免每次内存申请都需要操作系统重新申请。因此，在一个示例中，在步骤41，操作系统通过查找预先分配的内存池，从中确定出第一内存区段。

[0058] 根监视器获取到第一内存区段的指示信息后，可以对其进行验证。具体的，根监视器可以验证第一内存区段与分配给其他应用的内存区段有没有重叠；如有重叠，则验证不通过。在验证通过之后，根监视器初始化第一可信应用的元数据，例如包括第一内存区段的地址和大小，其中页表的基地址，线程ID，上下文等。根监视器还可以校验页表的映射是否有效，以确保统一有效的地址映射。

[0059] 为了确保第一可信应用的内存隔离，在步骤43，根世界中的根监视器更新总颗粒度保护表GPT，使得在更新后的总GPT中，第一内存区段的访问权限被设置为不可访问。

[0060] 可以理解，该总GPT表与已有Arm机密计算框架中的GPT表具有相似的形式，其中记录各个内存页所属的世界和访问权限。在该总GPT表中，分配给安全世界、根世界、领域世界的内存区域如常记录，但是通过步骤43，将属于非安全世界的第一内存区段的访问权限设置为不可访问。

[0061] 此外，在步骤45，根监视器针对第一可信应用创建第一颗粒度保护表GPT，在该第一GPT中，第一内存区段的访问权限被设置为可访问的非安全内存。

[0062] 更具体的，该第一GPT也可以具有与已有GPT表相似的形式。在该第一GPT表中，分配给安全世界、根世界、领域世界的内存区域可以如常记录，或者统一设置为不可访问。但是，第一内存区段的访问权限需设置为可访问的非安全内存。此外，其他可信应用(如果有的话)的内存区段需要设置为不可访问。

[0063] 为了加快执行，在一个实施例中，根监视器根据预先设置的GPT模板，快速创建上述第一GPT。在GPT模板中，可以预先填入常规记录的其他信息，在创建第一GPT时，只需要针对第一内存区段进行信息补入即可。

[0064] 需要说明的是，根监视器将上述总GPT表和第一GPT表都存储在根世界内存中进行维护。

[0065] 通过以上过程，在创建第一可信应用阶段，为其分配了第一内存区段，并相应配置了GPT表。于是，可以将第一可信应用相关的代码和数据加载到该第一内存区段，使其处于可执行状态。

[0066] 需要说明的是，上述第一可信应用是任意一个可信应用。在改进的机密计算架构中，可以为多个可信应用各自分配内存区段，各自创建专属的GPT表，后续独立地执行各个可信应用。

[0067] 在应用执行阶段，运行某一应用的CPU通过内存管理单元MMU的内存映射，确定运行该应用需访问的内存的物理地址，基于该物理地址发出内存访问请求。硬件层中的RME基

于颗粒度保护表GPT对该内存访问请求进行颗粒度保护检查GPC。然而，如前所述，在本说明书的改进的机密计算框架中，根监视器配置和维护了多个GPT表。如此，在应用执行阶段，由根监视器负责进行GPT表的切换，使得RME可以基于正确的GPT表执行GPC检查。

[0068] 具体的，根监视器针对CPU的内存访问请求，根据该CPU当前运行的应用，确定对应的GPT表。若当前运行的应用为某个可信应用，则确定对应的GPT表为该可信应用专属的GPT表。例如，若当前运行的应用为第一可信应用，则确定对应的GPT表为前述第一GPT表。若当前运行的应用不是可信应用，则确定对应的GPT表为前述总GPT表。

[0069] 如此，对于不是可信应用的任何其他软件/应用，RME基于总GPT表执行GPC。如前所述，在总GPT表中，针对可信应用分配的内存区段被设置为不可访问。因此，该任何其他软件/应用发起的内存访问请求如果请求访问针对可信应用分配的内存区段，将会产生颗粒度保护异常，该异常指示将会汇报给根监视器。如此，任何其他软件/应用，包括特权软件(例如，SPM，RMM)均无法访问可信应用的内存区段。

[0070] 对于可信应用发起的内存访问请求，RME将会基于其专属的GPT表执行GPC。例如，对于第一可信应用发起的访问请求，根监视器将会切换到第一GPT表，使得RME基于该第一GPT表执行GPC检查。如前所述，在该第一GPT表中，为第一可信应用分配的第一内存区段被设置为可访问的非安全内存，因此，第一可信应用可以顺利访问第一内存区段，从而正常执行。同时，在该第一GPT表中，根世界、安全世界和领域世界的内存区段，以及非安全世界的其他可信应用的内存区段均设置为不可访问，因此，第一可信应用无法访问根世界、安全世界和领域世界的内存区段，也无法访问其他可信应用的内存区段。如此，各个可信应用的内存区段仅可由自身访问，任何其他软件/应用(包括特权软件，以及其他可信应用)均无法实现访问，从而实现内存和计算环境的隔离。

[0071] 在一个实施例中，根监视器通过配置CPU中GPT基址寄存器，实现GPT表的切换。具体地，如前所述，根监视器在内存的根世界区段中存储并维护各个GPT表。因此，各个GPT表可以通过其在内存中存储位置的基地址来标识和区分。另一方面，可以在CPU中设置一寄存器，用于存储当前使用的GPT表的基地址。该寄存器即可称为GPT基址寄存器。如此，当CPU发出内存访问请求，根监视器在确定对应的GPT表后，就将GPT基址寄存器配置为存储该GPT表的基地址。从而使得，RME根据GPT基址寄存器的存储内容，寻址定位出当前对应的GPT表，基于该GPT表执行GPC检查。

[0072] 在一些实现方式中，改进的机密计算架构可以基于多核CPU。在这样的情况下，根监视器可以针对每个内核(core)，配置对应的GPT表，使其进行GPC检查。

[0073] 图5示出多核多GPT表的配置示例。在图5的例子中，内存被划分为非安全世界、安全世界、领域世界和根世界，这四个世界的内存在图5中分别以不同的图样填充以示区别。通过前述的可信应用创建过程，假定已经创建了可信应用1和可信应用2，其中，为可信应用1分配了内存区段1，为可信应用2分配了内存区段2。内存区段1和内存区段2均属于非安全世界的内存区段。

[0074] 根监视器已经配置并维护有主机GPT(即总GPT表)，以及针对可信应用1的GPT1，和针对可信应用2的GPT2。在主机GPT中，记录有各个内存页所属的世界和访问权限，在图中示出为非安全PAS，安全PAS，领域PAS，根PAS。此外，对于分配给可信应用的内存区段1和内存区段2，主机GPT中将其设置为不可访问。而在GPT1中，内存区段1被设置为可访问的非安全

世界内存,在GPT2中,内存区段2被设置为可访问的非安全世界内存。

[0075] 主机GPT,GPT1和GPT2均存储于内存的根世界区段中。更具体的,主机GPT存储位置的基地址为Add-H,GPT1存储位置的基地址为Add-1,GPT2存储位置的基地址为Add-2。

[0076] 在图示例子中,CPU具有4个CPU内核。在当前状态下,内核1和内核2运行其他应用(可以是安全世界的应用,领域虚拟机中的应用,或者非安全世界的普通应用),内核3运行可信应用1,内核4运行可信应用2。

[0077] 则根监视器针对各个内核,分别设置其使用的GPT表。具体的,根监视器将内核1和内核2中GPT基址寄存器均设置为Add-H。如此,对于内核1和内核2发出的内存访问请求,均使用主机GPT进行GPC检查。由于在主机GPT中,内存区段1和内存区段2均被设置为不可访问,因此,内核1和内核2上运行的其他应用均不能访问可信应用对应的内存区段1和区段2。

[0078] 对于运行可信应用1的内核3,根监视器将其中GPT基址寄存器设置为Add-1。如此,对于内核3发出的内存访问请求,使用Add-1指向的GPT1进行GPC检查。由于在GPT1中,内存区段1被设置为可以访问,因此,可信应用1可以访问内存区段1,从而正常执行应用功能。但是可信应用1不可以访问其他可信应用,以及其他世界的内存区段。

[0079] 对于运行可信应用2的内核4,根监视器将其中GPT基址寄存器设置为Add-2。如此,对于内核4发出的内存访问请求,使用Add-2指向的GPT2进行GPC检查。由于在GPT2中,内存区段2被设置为可以访问,因此,可信应用2可以访问内存区段2,从而正常执行应用功能。但是根据可信应用2不可以访问其他可信应用,以及其他世界的内存区段。

[0080] 应用的运行有可能动态地在不同内核之间切换。当发生了应用的切换,根监视器相应地动态更改对应内核的GPT基地址寄存器,从而对应切换GPT表。如此,通过根监视器的配置和管理,针对各个内核,根据其当前运行的应用使用对应的GPT表进行GPC检查,确保可信应用的内存隔离。

[0081] 进一步的,在一个实施例中,为了避免应用在内核之间切换时,通过共享TLB发生的数据泄露,可以禁用内核之间的TLB共享功能。

[0082] 当确定需要卸载或移除某个可信应用时,根监视器会销毁针对该可信应用构建的用户态机密计算环境。例如,在需要卸载或清除前述的第一可信应用时,根监视器首先会清空第一内存区段中的内容,在内存中清除第一GPT表,还删除第一可信应用的元数据,避免泄露机密数据。然后,根监视器才将第一内存区段返还给操作系统OS。相应的,根监视器在总GPT表中,将第一内存区段设置为常规的非安全世界内存。通过以上操作,销毁针对第一可信应用构建的机密计算环境,回收相应的内存资源。

[0083] 回顾以上可信应用的生命周期全过程,可以看到,根监视器通过配置多个GPT表,实现对用户态可信应用的内存隔离管理,从而为用户态可信应用创建机密、隔离的计算环境。

[0084] 另一方面,与上述方法过程相对应的,本说明书实施例还披露一种机密计算架构中的根监视器,所述机密计算架构包括,安全世界,领域世界,非安全世界和根世界;所述根监视器位于所述根世界中。根监视器可以包括内存管理模块,所述内存管理模块配置用于:

[0085] 响应于非安全世界的操作系统在非安全世界的内存中,为非安全世界的第一可信应用分配第一内存区段,更新总颗粒度保护表GPT,使得在更新后的总GPT中,所述第一内存区段的访问权限被设置为不可访问;

[0086] 针对所述第一可信应用创建第一颗粒度保护表GPT,在所述第一GPT中,所述第一内存区段的访问权限被设置为可访问的非安全内存。

[0087] 在具体实现方式中,根监视器还可以包括内存隔离模块,配置用于:响应于CPU发出内存访问请求,根据CPU当前运行的应用,从已维护的GPT集中确定目标GPT,并将目标GPT设置为对所述内存访问请求进行颗粒度保护检查的基础;所述已维护的GPT集包括所述总GPT和所述第一GPT。

[0088] 在典型实施例中,所述根监视器实现为安全固件。

[0089] 根据再又方面的实施例,还提供一种计算设备,包括存储器和处理器,所述计算设备形成机密计算架构,所述机密计算架构包括,安全世界,领域世界,非安全世界和根世界;所述根世界包括前述的根监视器。

[0090] 本领域技术人员应该可以意识到,在上述一个或多个示例中,本发明所描述的功能可以用硬件、软件、固件或它们的任意组合来实现。当使用软件实现时,可以将这些功能存储在计算机可读介质中或者作为计算机可读介质上的一个或多个指令或代码进行传输。

[0091] 以上所述的具体实施方式,对本发明的目的、技术方案和有益效果进行了进一步详细说明,所应理解的是,以上所述仅为本发明的具体实施方式而已,并不用于限定本发明的保护范围,凡在本发明的技术方案的基础之上,所做的任何修改、等同替换、改进等,均应包括在本发明的保护范围之内。

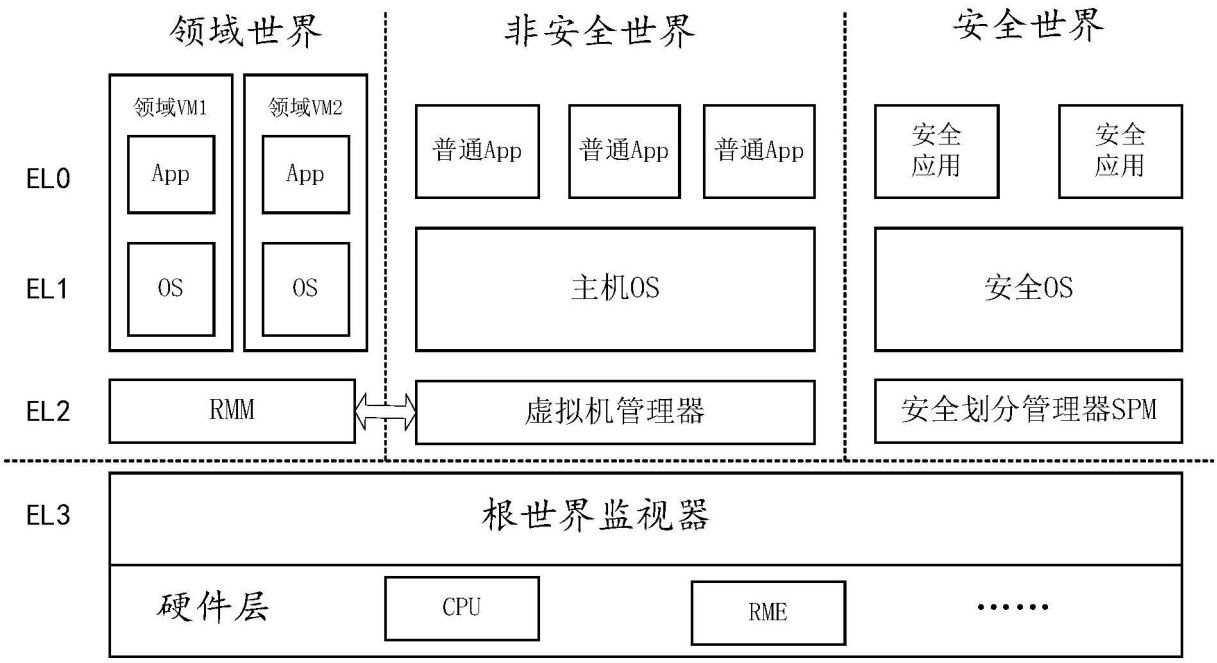


图1

安全状态	非安全 PAS	安全 PAS	领域 PAS	根 PAS
非安全	✓	×	×	×
安全	✓	✓	×	×
领域	✓	×	✓	×
根	✓	✓	✓	✓

图2

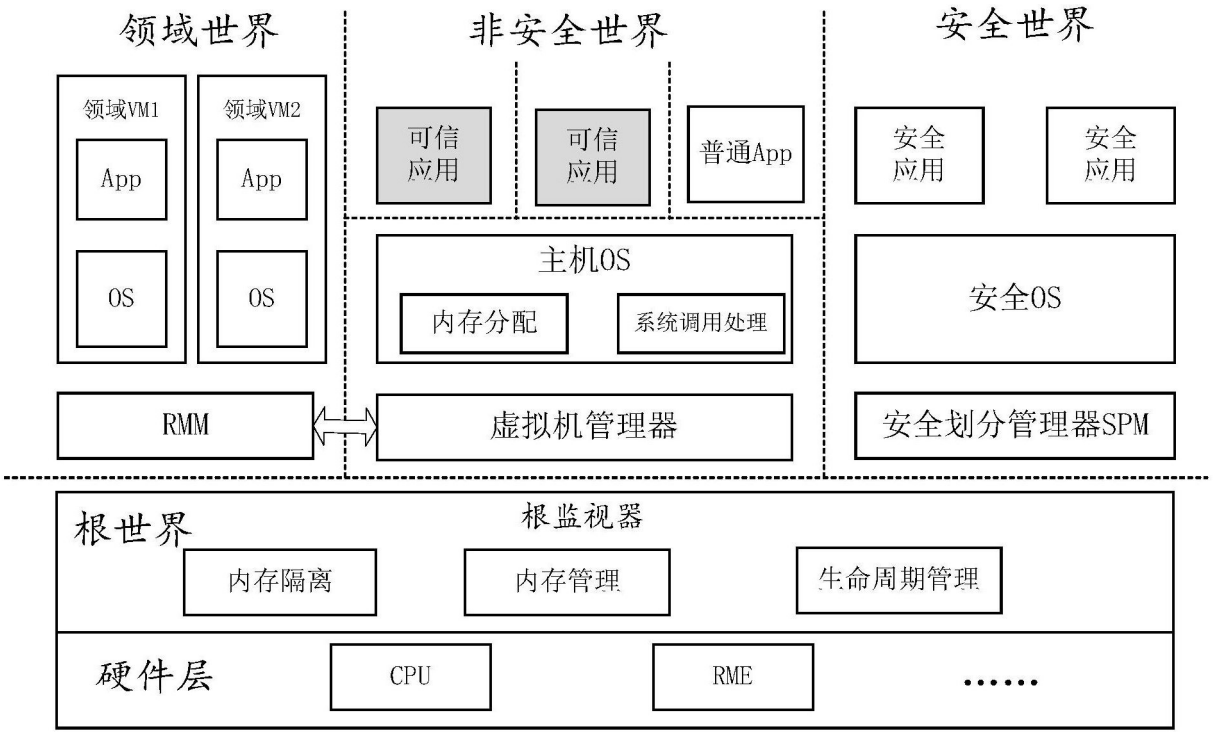

领域世界
非安全世界
安全世界
领域VM1
App
OS
领域VM2
App
OS
可信应用
可信应用
普通App
主机OS
内存分配
系统调用处理
安全应用
安全应用
安全OS
RMM
虚拟机管理器
安全划分管理器SPM
根世界
根监视器
内存隔离
内存管理
生命周期管理
硬件层
CPU
RME
……

图3

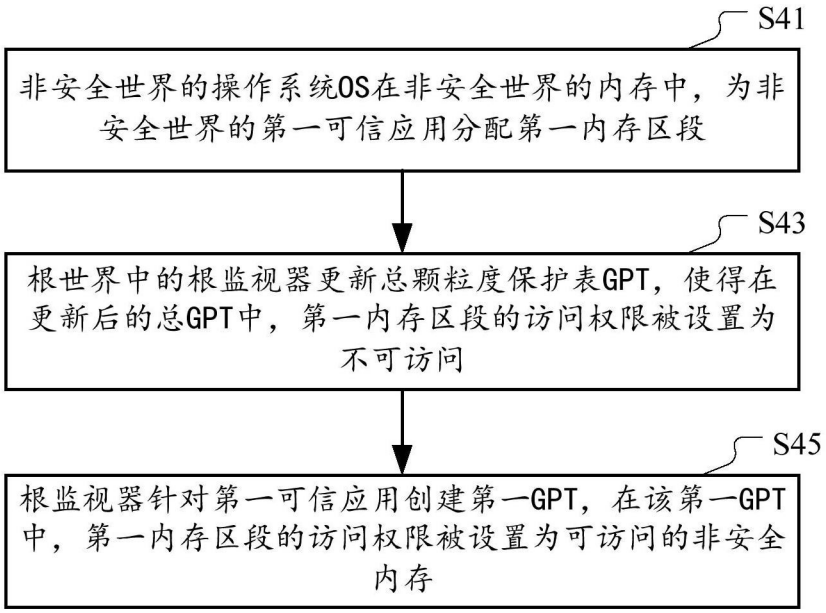

S41
非安全世界的操作系统OS在非安全世界的内存中，为非安全世界的第一可信应用分配第一内存区段
S43
根世界中的根监视器更新总颗粒度保护表GPT，使得在更新后的总GPT中，第一内存区段的访问权限被设置为不可访问
S45
根监视器针对第一可信应用创建第一GPT，在该第一GPT中，第一内存区段的访问权限被设置为可访问的非安全内存

图4

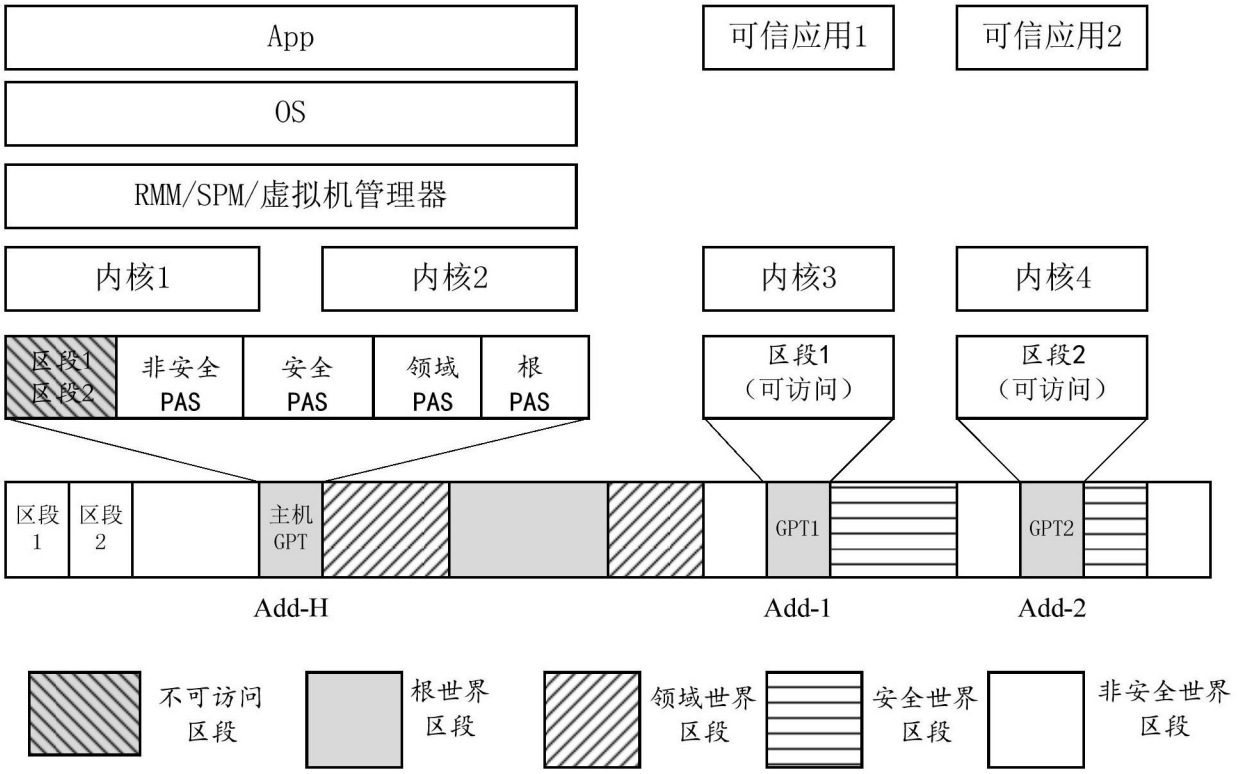

App
OS
RMM/SPM/虚拟机管理器
内核1
内核2
内核3
内核4
可信应用1
可信应用2
区段1
区段2
非安全
PAS
安全
PAS
领域
PAS
根
PAS
区段1
（可访问）
区段2
（可访问）
区段
1
区段
2
主机
GPT
GPT1
GPT2
Add-H
Add-1
Add-2
不可访问
区段
根世界
区段
领域世界
区段
安全世界
区段
非安全世界
区段

图5