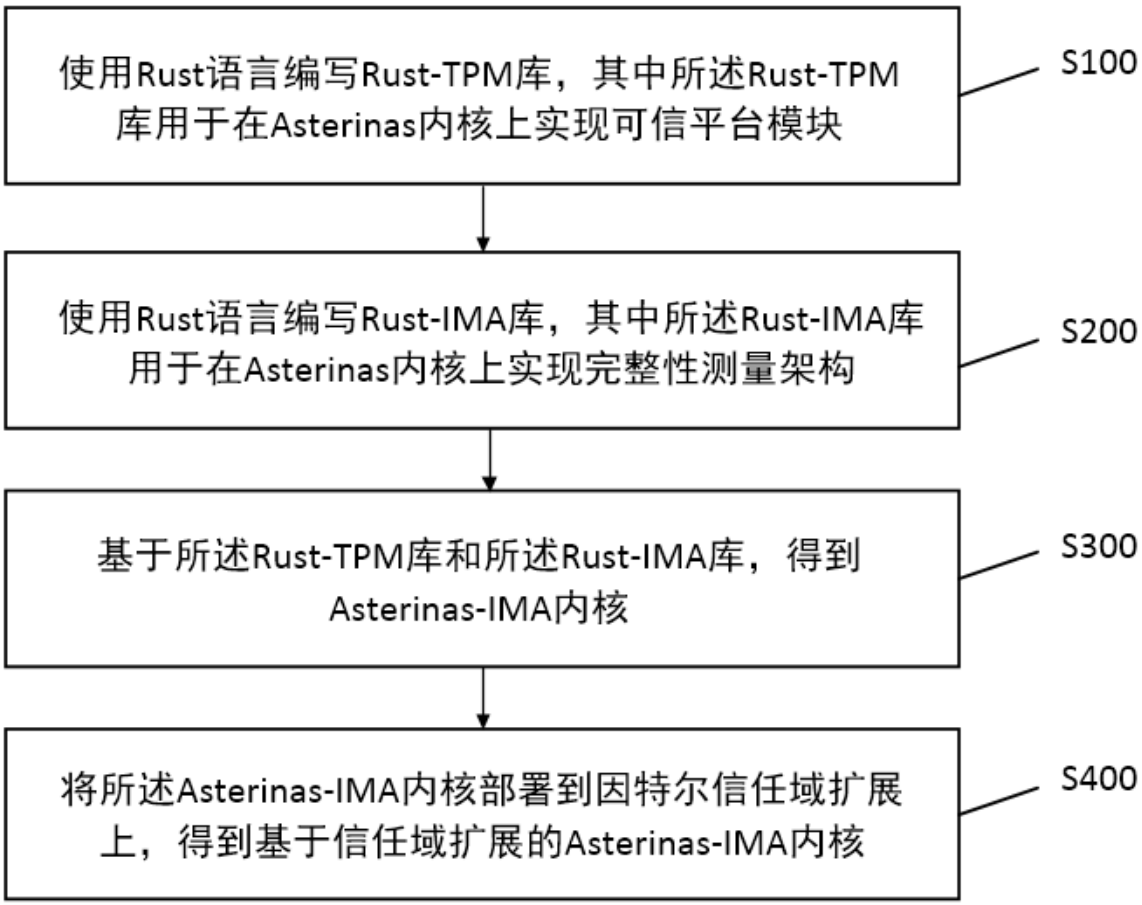
说明书摘要

本发明公开了一种基于信任域扩展的隐私保护方法和对应的操作系统内核设计方法及装置，所述方法包括：使用 Rust 语言实现 Rust-TPM 功能和 Rust-IMA 功能；利用 Rust 自身的安全属性，减少安全漏洞且提高代码的可靠性；基于 Rust-TPM 功能和 Rust-IMA 功能，得到 Asterinas-IMA 内核，通过 Rust 实现的系统内核将能够有效地减少安全漏洞；将上述内核部署到因特尔信任域扩展上，得到基于信任域扩展的内核，借助于因特尔信任域扩展的隔离作用，实现隐私计算功能和隐私保护效果，增强了系统的安全性，保护了用户的隐私数据。本发明利用 Rust 和信任域扩展的内存安全特性重新实现完整性测量架构功能，从而大幅提升了操作系统的安全性。同时，因特尔信任域扩展提供的隐私保护和隐私计算能力，使得系统内核在进行完整性测量和数据处理时，可以在隔离的受信环境中执行，避免敏感数据的泄露，从而进一步提升了操作系统的安全性和隐私保护能力。

摘要附图


使用Rust语言编写Rust-TPM库，其中所述Rust-TPM库用于在Asterinas内核上实现可信平台模块
S100
使用Rust语言编写Rust-IMA库，其中所述Rust-IMA库用于在Asterinas内核上实现完整性测量架构
S200
基于所述Rust-TPM库和所述Rust-IMA库，得到Asterinas-IMA内核
S300
将所述Asterinas-IMA内核部署到因特尔信任域扩展上，得到基于信任域扩展的Asterinas-IMA内核
S400

权利要求书

1．一种基于信任域扩展的操作系统内核设计方法，其特征在于，所述方法包括：

使用 Rust 语言实现 Rust-TPM 功能，其中所述 Rust-TPM 功能用于为 Asterinas 内核提供可信平台所需的相关算法框架；

使用 Rust 语言实现 Rust-IMA 功能，其中所述 Rust-IMA 功能用于在 Asterinas 内核上实现完整性测量架构；

基于所述 Rust-TPM 功能和所述 Rust-IMA 功能，得到 Asterinas-IMA 内核；

将所述 Asterinas-IMA 内核部署到因特尔信任域扩展上，得到基于信任域扩展的 Asterinas-IMA 内核。

2．根据权利要求 1 所述的基于信任域扩展的操作系统内核设计方法，其特征在于，所述使用 Rust 语言实现 Rust-TPM 功能，包括：

使用 Rust 语言编写 Asterinas 内核的可信平台模块组件和硬件接口驱动程序，其中所述可信平台模块组件用于生成密钥、硬件加密、硬件解密和安全启动；

根据所述可信平台模块组件和硬件接口驱动程序，实现所述 Rust-TPM 功能。

3．根据权利要求 1 所述的基于信任域扩展的操作系统内核设计方法，其特征在于，所述使用 Rust 语言实现 Rust-IMA 功能，包括：

使用 Rust 语言编写 Asterinas 内核的系统调用拦截器组件、测量单元组件、完整性锚点组件和硬件辅助验证组件，其中所述测量单元组件用于处理测量请求、储存测量结果和验证测量数据；

根据所述系统调用拦截器组件、测量单元组件、完整性锚点组件和硬件辅助验证组件，实现所述 Rust-IMA 功能。

4．根据权利要求 3 所述的基于信任域扩展的操作系统内核设计方法，其特征在于，所述系统调用拦截器组件用于：

当获取系统调用处理程序的调用指令时，通过系统调用拦截器向所述测量单元组件发送测量请求，其中所述调用指令用于调用系统文件；

根据所述测量请求测量被调用的系统文件，得到系统文件的哈希值；

将所述系统文件的哈希值与测量参考值进行比较，得到比较结果；

若所述比较结果为所述系统文件的哈希值与预设的测量参考值一致，则根据所述调用指令调用所述系统文件；

若所述比较结果为所述系统文件的哈希值与预设的测量参考值不一致，则不执行所述调用指令。

5．根据权利要求 3 所述的基于信任域扩展的操作系统内核设计方法，其特征在于，所述完整性锚点组件用于：

获取测量单元的测量记录，并将所述测量记录与测量列表中的历史测量结果进行比较，以验证测量列表中内容的完整性，其中所述测量列表部署在完整性测量架构中用于保存历史测量结果。

6．根据权利要求 5 所述的基于信任域扩展的操作系统内核设计方法，其特征在于，所述硬件辅助验证组件用于：

在 Asterinas 内核上运行信任域扩展驱动程序，并通过所述信任域扩展驱动程序触发测量单元向所述信任域扩展发送 TDCALL 请求；

根据所述 TDCALL 请求，记录运行时测量数据；

通过 SHA384 算法计算所述运行时测量数据，得到本地运行时测量寄存器扩展数值，并将所述本地运行时测量寄存器扩展数值追加至测量列表、存储在信任域扩展的运行时测量寄存器中；

通过信任域扩展认证报告读取所述本地运行时测量寄存器扩展数值，同时计算测量列表的扩展结果，若所述本地运行时测量寄存器扩展数值与所述计算测量结果一致，则通过验证。

7．根据权利要求 6 所述的基于信任域扩展的操作系统内核设计方法，其特征在于，所述基于所述 Rust-TPM 功能和所述 Rust-IMA 功能，得到 Asterinas-IMA 内核，包括：

通过所述 Rust-TPM 功能与可信平台模块交互，通过所述 Rust-TPM 功能与完整性测量架构交互，将所述 Rust-TPM 功能和所述 Rust-TPM 功能整合到所述完整性测量架构中，并用所述信任域扩展的运行时测量寄存器替换可信平台模块的平台配置寄存器，得到 Asterinas-IMA 内核。

8．一种基于信任域扩展的操作系统内核设计装置，其特征在于，所述装置包括：

第一 Rust 库编写模块，用于使用 Rust 语言实现 Rust-TPM 功能，其中所述 Rust-TPM 功能用于在 Asterinas 内核上实现可信平台模块；

第二 Rust 库编写模块，用于使用 Rust 语言实现 Rust-IMA 功能，其中所述 Rust-IMA 功能用于在 Asterinas 内核上实现完整性测量架构；

第一系统内核获取模块，用于基于所述 Rust-TPM 功能和所述 Rust-IMA 功能，得到 Asterinas-IMA 内核；

第二系统内核获取模块，用于将所述 Asterinas-IMA 内核部署到因特尔信任域扩展上，得到基于信任域扩展的 Asterinas-IMA 内核。

9．一种智能终端，其特征在于，所述智能终端包括存储器、处理器及存储在所述存储器中并可在所述处理器上运行的基于信任域扩展的操作系统内核设计程序，所述处理器执行所述基于信任域扩展的操作系统内核设计程序时，实现如权利要求 1-7 任一项所述的基于信任域扩展的操作系统内核设计方法的步骤。

10．一种计算机可读存储介质，其特征在于，所述计算机可读存储介质上存储有基于信任域扩展的操作系统内核设计程序，所述基于信任域扩展的操作系统内核设计程序被处理器执行时，实现如权利要求 1-7 任一项所述的基于信任域扩展的操作系统内核设计方法的步骤。

说明书

一种操作系统中拓展信任域的隐私保护方法

技术领域

本发明涉及计算机技术领域，具体涉及一种基于信任域扩展的隐私保护方法及其对应操作系统内核设计方法及装置。

背景技术

Linux 是一种开源电脑操作系统内核。它是一个主要用 C 语言编写，符合 POSIX 标准的类 Unix 操作系统，由于其开源和模块化的特点，Linux 在服务器、嵌入式系统、桌面计算等多个领域都有广泛应用。尽管 Linux 在安全性方面做了许多努力，但由于 C 语言本身存在一些固有的安全问题，如：内存管理错误问题、缺乏类型安全问题和手动内存管理问题，如开发者需要手动管理内存分配和释放而导致的内存泄漏和其它内存相关的漏洞等，导致系统崩溃或被恶意攻击者利用。

因此，现有技术还有待于改进和发展。Asterinas 开发团队研发了使用 Rust 语言编写的 Asterinas 操作系统，解决了由 C 语言带来的安全问题。但 Asterinas 系统仍处于上升阶段，尚未实现对隐私数据的保护和对执行完整性的检测。

发明内容

本发明要解决的技术问题在于，针对现有技术的上述缺陷，提供一种基于信任域扩展的操作系统内核设计方法及装置，旨在为 Asterinas 系统添加执行完整性的检测，同时基于信任域拓展，增强对隐私数据的保护。

本发明解决技术问题所采用的技术方案如下：

第一方面，本发明提供一种基于信任域扩展的操作系统内核设计方法，支持隐私保护和隐私计算，其中所述方法包括：

使用 Rust 语言实现 Rust-TPM 功能，其中所述 Rust-TPM 功能用于在 Asterinas 内核上实现可信平台模块；

使用 Rust 语言实现 Rust-IMA 功能，其中所述 Rust-IMA 功能用于在 Asterinas 内核上实现完整性测量架构；

基于所述 Rust-TPM 功能和所述 Rust-IMA 功能，得到 Asterinas-IMA 内核；

将所述 Asterinas-IMA 内核部署到因特尔信任域扩展上，得到基于信任域扩展的 Asterinas-IMA 内核。因特尔信任域扩展创建了一个独立于主机操作系统和虚拟机管理程序的信任域，这些信任域通过硬件隔离保证了代码和数据的私密性，即使主机操作系统或虚拟机管理程序遭到破坏，恶意实体也无法访问信任域内的数据。这种隔离机制为敏感数据的保护提供了强有力的基础，使得数据可以在隔离的环境中进行计算，不会被外部环境窃取或篡改。此外，因特尔信任域扩展提供了可信的执行环境，即使在不可信的系统中，用户也可以确保其代码和数据在信任域内的执行是安全的。这使得数据在计算过程中可以避免未经授权的访问，从而支持隐私计算。隐私计算依赖于这样的可信环境，保证数据在处理过程中不暴露给外部实体。因此，基于信任域扩展的 Asterinas-IMA 内核支持隐私保护和隐私计算功能，能够在不影响数据隐私的前提下完成计算任务。

在一种实现方式中，所述使用 Rust 语言实现 Rust-TPM 功能，包括：

使用 Rust 语言编写 Asterinas 内核的可信平台模块组件和硬件接口驱动程序，其中所述可信平台模块组件用于生成密钥、硬件加密、硬件解密、安全启动和远程认证；

根据所述可信平台模块组件和硬件接口驱动程序，得到所述 Rust-TPM 功能。

在一种实现方式中，所述使用 Rust 语言实现 Rust-IMA 功能，包括：

使用 Rust 语言编写 Asterinas 内核的系统调用拦截器组件、测量单元组件、完整性锚点组件和硬件辅助验证组件，其中所述测量单元组件用于处理测量请求、储存测量结果和验证测量数据；

根据所述系统调用拦截器组件、测量单元组件、完整性锚点组件和硬件辅助验证组件，得到所述 Rust-IMA 功能。

在一种实现方式中，所述系统调用拦截器组件用于：

当获取系统调用处理程序的调用指令时，通过系统调用拦截器向所述测量单元组件发送测量请求，其中所述调用指令用于调用系统文件；

根据所述测量请求测量被调用的系统文件，得到系统文件的哈希值；

将所述系统文件的哈希值与测量参考值进行比较，得到比较结果；

若所述比较结果为所述系统文件的哈希值与预设的测量参考值一致，则根据所述调用指令调用所述系统文件；

若所述比较结果为所述系统文件的哈希值与预设的测量参考值不一致，则不执行所述调用指令。

在一种实现方式中，所述完整性锚点组件用于：

获取测量单元的测量记录，并将所述测量记录与测量列表中的历史测量结果进行比较，以验证测量列表中内容的完整性，其中所述测量列表部署在完整性测量架构中用于保存历史测量结果。

在一种实现方式中，所述硬件辅助验证组件用于：

在 Asterinas 内核上运行信任域扩展驱动程序，并通过所述信任域扩展驱动程序触发测量单元向所述信任域扩展发送 TDCALL 请求；

根据所述 TDCALL 请求，记录运行时测量数据；

通过 SHA384 算法计算所述运行时测量数据，得到本地运行时测量寄存器扩展数值，并将所述本地运行时测量寄存器扩展数值存储在信任域扩展的运行时测量寄存器中；

通过信任域扩展认证报告读取所述本地运行时测量寄存器扩展数值，并将所述本地运行时测量寄存器扩展数值与所述测量列表中的历史测量结果进行比较，若所述本地运行时测量寄存器扩展数值与所述测量列表中的历史测量结果一致，则通过验证。

在一种实现方式中，所述基于所述 Rust-TPM 功能和所述 Rust-IMA 功能，得到 Asterinas-IMA 内核，包括：

通过所述 Rust-TPM 功能与可信平台模块交互，通过所述 Rust-TPM 功能与完整性测量架构交互，将所述 Rust-TPM 功能和所述 Rust-TPM 功能整合到所述完整性测量架构中，并用所述信任域扩展的运行时测量寄存器替换可信平台模块的平台配置寄存器，得到 Asterinas-IMA 内核。

第二方面，本发明实施例还提供一种基于信任域扩展的操作系统内核设计装置，其中，所述装置包括：

第一 Rust 库编写模块，用于使用 Rust 语言实现 Rust-TPM 功能，其中所述 Rust-TPM 功能用于在 Asterinas 内核上实现可信平台模块；

第二 Rust 库编写模块，用于使用 Rust 语言实现 Rust-IMA 功能，其中所述 Rust-IMA 功能用于在 Asterinas 内核上实现完整性测量架构；

第一系统内核获取模块，用于基于所述 Rust-TPM 功能和所述 Rust-IMA 功能，得到 Asterinas-IMA 内核；

第二系统内核获取模块，用于将所述 Asterinas-IMA 内核部署到因特尔信任域扩展上，得到基于信任域扩展的 Asterinas-IMA 内核。

第三方面，本发明实施例还提供一种智能终端，其中，所述智能终端包括存储器、处理器及存储在所述存储器中并可在所述处理器上运行的基于信任域扩展的操作系统内核设计程序，所述处理器执行所述基于信任域扩展的操作系统内核设计程序时，实现如以上任一项所述的基于信任域扩展的操作系统内核设计方法的步骤。

第四方面，本发明实施例还提供一种计算机可读存储介质，其中，所述计算机可读存储介质上存储有基于信任域扩展的操作系统内核设计程序，所述基于信任域扩展的操作系统内核设计程序被处理器执行时，实现如以上任一项所述的基于信任域扩展的操作系统内核设计方法的步骤。

有益效果：与现有技术相比，本发明提供了一种基于信任域扩展的操作系统内核设计方法，所述方法包括：使用 Rust 语言实现 Rust-TPM 功能和 Rust-IMA 功能，利用 Rust 自身的安全属性，减少安全漏洞且提高代码的可靠性；基于 Rust-TPM 功能和 Rust-IMA 功能，得到 Asterinas-IMA 内核，通过 Rust 实现的系统内核将能够有效地减少安全漏洞；将 Asterinas-IMA 内核部署到因特尔信任域扩展上，得到基于信任域扩展的 Asterinas-IMA 内核，借助于因特尔信任域扩展的隔离效果，支持隐私保护和隐私计算，增强了系统的安全性。本发明利用 Rust 和信任域扩展的内存安全特性重新实现完整性测量架构功能，确保了信任域中的代码没有被篡改，增强了用户对数据处理过程的信任，从而大幅提升了操作系统的安全性。

附图说明

为了更清楚地说明本发明实施例或现有技术中的技术方案，下面将对实施例或现有技术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中记载的一些实施例，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他的附图。

图 1 是本发明实施例提供的基于信任域扩展的操作系统内核设计方法流程示意图。

图 2 是本发明实施例提供的 Asterinas-IMA 内核架构图。

图 3 是本发明实施例提供的基于信任域扩展的操作系统内核设计装置的原理框图。

图 4 是本发明实施例提供的智能终端的内部结构原理框图。

具体实施方式

为使本发明的目的、技术方案及效果更加清楚、明确，以下参照附图并举实施例对本发明进一步详细说明。应当理解，此处所描述的具体实施例仅用以解释本发明，并不用于限定本发明。

本技术领域技术人员可以理解，除非特意声明，这里使用的单数形式“一”、“一个”、“所述”和“该”也可包括复数形式。应该进一步理解的是，本发明的说明书中使用的措辞“包括”是指存在所述特征、整数、步骤、操作、元件和/或组件，但是并不排除存在或添加一个或多个其他特征、整数、步骤、操作、元件、组件和/或它们的组。 应该理解，当我们称元件被“连接”或“耦接”到另一元件时，它可以直接连接或耦接到其他元件，或者也可以存在中间元件。此外，这里使用的“连接”或“耦接”可以包括无线连接或无线耦接。这里使用的措辞“和/或”包括一个或更多个相关联的列出项的全部或任一单元和全部组合。

本技术领域技术人员可以理解，除非另外定义，这里使用的所有术语(包括技术术语和科学术语)，具有与本发明所属领域中的普通技术人员的一般理解相同的意义。还应该理解的是，诸如通用字典中定义的那些术语，应该被理解为具有与现有技术的上下文中的意义一致的意义，并且除非像这里一样被特定定义，否则不会用理想化或过于正式的含义来解释。

Linux 是一个基于 C 语言的操作系统，自 1991 年发布以来，已经成为全球范围内使用最广泛的操作系统之一。由于其开源和模块化的特点，Linux 在服务器、嵌入式系统、桌面计算等多个领域都有广泛应用。在安全性方面，Linux 实现了 TPM（Trusted Platform Module，可信平台模块）和 IMA（Integrity Measurement Architecture，完整性测量架构）等功能，这些功能极大地增强了系统的安全性。但由于 C 语言本身存在一些固有的安全问题，导致 Linux 存在内存管理错误、缺乏类型安全等安全缺陷。为

了克服 C 语言的安全缺陷，开发者选择用更加注重安全的 Rust 语言实现新的操作系统。

Asterinas（星绽操作系统）使用 Rust 作为唯一的编程语言，是一个安全、快速、通用的操作系统内核。它提供与 Linux 类似的接口，可无缝运行部分 Linux 应用，但比 Linux 更加内存安全和开发者友好。本发明采用 Asterinas 作为系统的内核，为了增强 Asterinas 的安全性，本发明为 Asterinas 开发适配 TDX（Trust Domain Extensions，信任域扩展）的版本，同时引入 Rust 语言实现的 IMA（Integrity Measurement Architecture，完整性测量架构）和 TPM（Trusted Platform Module，可信平台模块）功能。

示例性方法

本实施例提供一种基于信任域扩展的操作系统内核设计方法，本实施例可应用于操作系统内核。如图 1 所示，所述方法包括如下步骤：

步骤 S100、使用 Rust 语言实现 Rust-TPM 功能，其中所述 Rust-TPM 功能用于在 Asterinas 内核上实现可信平台模块；

具体地，在 Linux 系统中，TPM（Trusted Platform Module，可信平台模块）提供了一系列硬件级别的安全功能，包括生成和管理加密密钥。利用基于硬件的加密技术，TPM 为存储信息提供了更高的保护，防止外部软件攻击。TPM 支持的各种应用提高了安全措施，特别是在需要防止未经授权访问敏感数据的场景，例如设备被盗。通过确保平台配置未被篡改 TPM 使得应用程序可以拒绝访问数据和机密信息，从而加强了安全协议。

星绽（Asterinas）是一个安全、快速、通用的操作系统内核。它提供与 Linux 类似的接口，可无缝运行部分 Linux 应用。星绽在内存安全性方面，使用 Rust 作为唯一的编程语言，从而提高了其安全性。通过将不安全的部分 Rust 代码使用限制在一个明确定义且最小的可信计算基础上，使得星绽成为一个更安全、更可靠的内核选择。而 Rust 中的所有内存访问都经

过了编译器的严格检查,且在运行时不会出现空指针异常或数据竞争等问题。这意味着 Rust 代码具有更高的可靠性和安全性,可以避免常见的安全漏洞。

在本实施例中，使用 Rust 语言实现 Rust-TPM 功能，从而在 Asterinas 内核上实现 TPM，使得内核运行更安全可靠。

在一种实现方式中，本实施例所述步骤 S100 包括如下步骤：

步骤 S101、使用 Rust 语言编写 Asterinas 内核的可信平台模块组件和硬件接口驱动程序，其中所述可信平台模块组件用于生成密钥、硬件加密、硬件解密和安全启动；

步骤 S102、根据所述可信平台模块组件和硬件接口驱动程序，得到所述 Rust-TPM 功能。

在本实施例中，通过将 TPM 用 Rust 语言实现，通过开发 Rust 库，用于与 TPM 进行交互，包括密钥生成、硬件加密/解密、安全启动和远程认证等功能，再实现 TPM 的硬件接口和驱动程序，以确保 TPM 可以与 Asterinas 操作系统无缝集成。这样就在 Asterinas 内核上实现可信平台模块，增强了系统的安全性。

步骤 S200、使用 Rust 语言实现 Rust-IMA 功能，其中所述 Rust-IMA 功能用于在 Asterinas 内核上实现完整性测量架构；

具体地，IMA（Integrity Measurement Architecture，完整性测量架构）是一个操作系统的安全框架，通过计算和比较文件的哈希值来确保系统在运行期间没有被篡改。IMA 的测量结果可存储在 TPM 中，以获得更强的安全性，并可用于远程认证。但仅使用 IMA 的系统依赖于内核的安全性和配置。若内核或配置存在漏洞或不当配置，可能会影响 IMA 的保护效果。例如，某些关键文件可能未被监控，或 IMA 配置错误可能导致保护不足。

在本实施例中，通过Rust语言实现IMA，基于Rust语言的安全性，开发Rust-IMA功能用于计算和验证系统组件的哈希值，确保系统完整性，实现运行时完整性检查，持续监测关键文件和进程的完整性，集成审计和日志记录功能，提供全面的安全审计能力，支持基于策略的完整性测量和验证，允许管理员定义和管理安全策略。即通过在Asterinas中，使用Rust实现的IMA，有效地检测系统文件是否被恶意软件篡改，并减少开发人员在编写和维护安全代码时的负担。

在一种实现方式中，本实施例所述步骤S200包括如下步骤：

步骤S201、使用Rust语言编写Asterinas内核的系统调用拦截器组件、测量单元组件、完整性锚点组件和硬件辅助验证组件，其中所述测量单元组件用于处理测量请求、储存测量结果和验证测量数据；

步骤S202、根据所述系统调用拦截器组件、测量单元组件、完整性锚点组件和硬件辅助验证组件，得到所述Rust-IMA功能。

具体地，如图2所示，Asterinas-IMA的详细架构中，IMA的实现分为几个主要组件，包括系统调用拦截器组件、测量单元组件、完整性锚点组件和硬件辅助验证组件，这些组件共同工作以确保文件和系统的完整性。

在一种实现方式中，本实施例所述步骤S201包括如下步骤：

步骤S2011、当获取系统调用处理程序的调用指令时，通过系统调用拦截器向所述测量单元组件发送测量请求，其中所述调用指令用于调用系统文件；

步骤S2012、根据所述测量请求测量被调用的系统文件，得到系统文件的哈希值；

步骤S2013、将所述系统文件的哈希值与测量参考值进行比较，得到比较结果；

步骤 S2014、若所述比较结果为所述系统文件的哈希值与预设的测量参考值一致，则说明所述系统文件未被篡改，可以根据所述调用指令调用所述系统文件；

步骤 S2015、若所述比较结果为所述系统文件的哈希值与预设的测量参考值不一致，则说明所述系统文件被篡改，不执行所述调用指令。

具体地，在 Asterinas 中，定义了各种系统调用接口及其处理程序，这些程序解析系统调用请求及其参数，并将系统调用请求及其参数传递给具体的处理函数。其中，系统调用拦截器负责拦截操作系统的系统调用请求。每当操作系统处理系统调用请求时，CPU 会触发中断，操作系统进入陷阱帧处理程序，拦截器路由到具体的处理函数前进行拦截。

在本实施例中，为了实现运行时测量，在 Asterinas 系统调用处理程序中添加了一个系统调用拦截器，以绑定注册的系统调用处理程序，并允许在系统调用处理任务前后插入测量代码。具体地，当执行诸如 mmap() 这样的文件操作敏感系统调用时，系统调用拦截器会向测量单元发送测量请求。测量单元会测量被访问的文件，并将当前的系统文件的哈希值与预设的测量参考值进行比较。这个过程确保了文件在运行时的完整性，并对文件操作进行监控。

在一种实现方式中，本实施例所述步骤 S201 包括如下步骤：

步骤 S2016、获取测量单元的测量记录，并将所述测量记录与测量列表中的历史测量结果进行比较，以验证测量列表中内容的完整性，其中所述测量列表部署在完整性测量架构中用于保存历史测量结果。

具体地，测量单元负责处理测量请求、结果存储和验证测量数据。测量列表（Measurement List，ML）是实现 IMA 功能的关键数据结构，它保存当前的完整测量历史。ML 由内核安全模块维护，当生成新的测量记录时，一条新的测量条目会被添加到列表中。由于测量列表是维护系统的测量日志，历史条目不能被删除，因此它对外部世界是只读的。

在本实施例中，为了保存测量参考值，Asterinas 文件系统设计了一个扩展文件属性（Extended File Attribute，Xattr）机制。扩展属性是一种与特定文件永久关联的键值对，通常用于提供文件系统的附加功能，例如权限控制和日志记录，集成在访问控制列表（ACL）中。在 IMA 实现中，使用 security.ima 属性来指定当前文件的参考测量值。内核可以使用操作 Xattr 的 API 来设置、获取和列出文件的扩展属性。文件系统在系统初始化期间遍历文件系统，以初始化测量参考值，并将扩展属性写入作为原始参考值。后续系统进行完整性测量时便可以与参考值进行比较判断文件是否完整，同时文件修改后需要同步更新对应参考值。

在一种实现方式中，本实施例所述步骤 S201 包括如下步骤：

步骤 S2017、在 Asterinas 内核上运行信任域扩展驱动程序，并由测量单元向所述信任域扩展驱动程序发送 TDCALL 请求；

步骤 S2018、根据所述 TDCALL 请求，记录运行时测量数据；

具体地，完整性锚点（Measurement Anchor，MA）是 IMA 信任链中的一个重要节点，用于确保测量列表中内容的完整性。每当将新的测量记录提交到测量列表时，测量单元会将更改同步到测量锚点，以保持记录的一致性。为了使完整性锚点可信，它依赖于安全硬件，例如 TPM 芯片或 TDX 环境中的测量锚点。在本实施例中，选择基于 Intel TDX 的测量锚点作为安全固件平台，具体地，TDX 固件的驱动模块，即 tdx_guest，在内核启动时被挂载到/dev/tdx_guest 目录下。在内核可调用该模块函数操作 TDX 相关寄存器。作为 TDX 中的信任域（TD），Asterinas 可以使用 TDCALL 来提交测量数据并获取认证报告，从而读取数据。

步骤 S2019、通过 SHA384 算法计算所述运行时测量数据，得到本地运行时测量寄存器扩展数值，并将所述本地运行时测量寄存器扩展数值存储在信任域扩展的运行时测量寄存器中；

步骤 S2020、通过信任域扩展认证报告读取所述本地运行时测量寄存器扩展数值，并将所述本地运行时测量寄存器扩展数值与所述测量列表中的历史测量结果进行比较，若所述本地运行时测量寄存器扩展数值与所述测量列表中的历史测量结果一致，则通过验证。

具体地，硬件辅助验证机制基于因特尔信任域扩展的运行时测量寄存器（RTMR）扩展。在因特尔信任域扩展标准中，运行时测量寄存器用于记录运行时数据，其中 RTMR[2] 和 RTMR[3] 允许操作系统和用户程序记录运行时测量数据。首先，我们在内核中实现了一个简单的 TDX 驱动程序，该驱动程序基于 Intel TDX Guest API，允许测量单元发送 TDCALL 以扩展运行时测量寄存器。TDX 在接收到请求后，会将提交的值与旧值一起进行扩展操作，通过 SHA384 (new value + old value) 算法计算并覆盖旧值。

需要注意的是，信任域扩展标准不提供直接读取运行时测量寄存器的方式，而是通过信任域扩展认证报告解码来获取。信任域扩展 Guest 可以使用 TDCALL[TDG.MR.REPORT] 来获取来自信任域扩展模块的 TDREPORT (TDREPORT_STRUCT)。在信任域扩展认证报告中，运行时测量寄存器值被编码在特定的字段内。通过读取 TDREPORT_BUFFER 中的相应位置来获取 RTMR 值，即测量列表中的历史测量结果，并与本地运行时测量寄存器扩展数值进行比较。如果一致，这证明了测量列表中存储的测量记录的完整性，则通过验证。

步骤 S300、基于所述 Rust-TPM 功能和所述 Rust-IMA 功能，得到 Asterinas-IMA 内核；

具体地，仅使用 IMA 的系统依赖于内核的安全性和配置。若内核或配置存在漏洞或不当配置，可能会影响 IMA 的保护效果。例如，某些关键文件可能未被监控，或 IMA 配置错误可能导致保护不足。而 TDX 提供了硬件级别的隔离，将虚拟机和宿主机系统的操作环境分开，可以防止

宿主机和虚拟机内部的潜在威胁影响数据的完整性。在 Asterinas 中，IMA 提供了完整性保护，同时受益于 TDX 的隔离效果，增强了系统的安全性。

在一种实现方式中，本实施例所述步骤 S300 包括如下步骤：

步骤 S301、通过所述 Rust-TPM 功能与可信平台模块交互，通过所述 Rust-TPM 功能与完整性测量架构交互，将所述 Rust-TPM 功能和所述 Rust-TPM 功能整合到所述完整性测量架构中，并用所述信任域扩展的运行时测量寄存器替换可信平台模块的平台配置寄存器，得到 Asterinas-IMA 内核。

具体地，Asterinas 通过实现一个基于信任域扩展的运行时测量寄存器扩展的验证机制，在测量数据记录和验证中引入了新的方法。本实施例中，使用运行时测量寄存器的扩展机制代替传统可信平台模块的平台配置寄存器记录运行时数据，这种机制增强了测量数据的可信性和安全性。同时系统使用了 TDX 标准认证报告获取扩展值的方式，相较于传统 TPM 相比，实现 TPM 和 IMA 的协同工作，增强系统的整体安全性。

步骤 S400、将所述 Asterinas-IMA 内核部署到因特尔信任域扩展上，得到基于信任域扩展的 Asterinas-IMA 内核。

具体地，仅使用 IMA 的系统依赖于内核的安全性和配置。若内核或配置存在漏洞或不当配置，可能会影响 IMA 的保护效果。例如，某些关键文件可能未被监控，或 IMA 配置错误可能导致保护不足。而因特尔信任域扩展提供了一种硬件级别的隔离机制，将虚拟机的执行环境与宿主操作系统隔离开来，可以防止宿主机和虚拟机内部的潜在威胁影响数据的完整性。

在本实施例中，通过将所述 Asterinas-IMA 内核部署到因特尔信任域扩展上，隔离保护了虚拟机中的数据和代码免受宿主系统或其他虚拟机的影响，从而降低了攻击面。例如，即使宿主操作系统被攻陷，攻击者也无

法直接访问或篡改在信任域内部运行的虚拟机数据。RTMRs（运行时测量寄存器）用于记录虚拟机在运行时的测量数据，并使用安全硬件对这些数据进行保护和验证。这种机制确保了测量数据的完整性，使得系统能够实时检测和响应潜在的安全问题，例如文件篡改或系统配置的异常。在系统内部维护运行时测量寄存器，可以很好地利用信任域扩展硬件特性提高IMA 系统的测量可信和完整性。

示例性装置

如图 3 中所示，本实施例还提供一种基于信任域扩展的操作系统内核设计装置，所述装置包括：

第一 Rust 库编写模块，用于使用 Rust 语言实现 Rust-TPM 功能，其中所述 Rust-TPM 功能用于在 Asterinas 内核上实现可信平台模块；

第二 Rust 库编写模块，用于使用 Rust 语言实现 Rust-IMA 功能，其中所述 Rust-IMA 功能用于在 Asterinas 内核上实现完整性测量架构；

第一系统内核获取模块，用于基于所述 Rust-TPM 功能和所述 Rust-IMA 功能，得到 Asterinas-IMA 内核；

第二系统内核获取模块，用于将所述 Asterinas-IMA 内核部署到因特尔信任域扩展上，得到基于信任域扩展的 Asterinas-IMA 内核。

在一种实现方式中，所述第一 Rust 库编写模块包括：

第一组件生成单元，用于使用 Rust 语言编写 Asterinas 内核的可信平台模块组件和硬件接口驱动程序，其中所述可信平台模块组件用于生成密钥、硬件加密、硬件解密、安全启动和远程认证；

Rust-TPM 功能生成单元，用于根据所述可信平台模块组件和硬件接口驱动程序，得到所述 Rust-TPM 功能。

在一种实现方式中，所述第二 Rust 库编写模块包括：

第二组件生成单元，用于使用 Rust 语言编写 Asterinas 内核的系统调用拦截器组件、测量单元组件、完整性锚点组件和硬件辅助验证组件，其中所述测量单元组件用于处理测量请求、储存测量结果和验证测量数据；

Rust-IMA 功能生成单元，用于根据所述系统调用拦截器组件、测量单元组件、完整性锚点组件和硬件辅助验证组件，得到所述 Rust-IMA 功能。

在一种实现方式中，本实施例所述第二组件生成单元包括：

测量请求发送单元，用于当获取系统调用处理程序的调用指令时，通过系统调用拦截器向所述测量单元组件发送测量请求，其中所述调用指令用于调用系统文件；

哈希值获取单元，用于根据所述测量请求测量被调用的系统文件，得到系统文件的哈希值；

第一比较单元，用于将所述系统文件的哈希值与测量参考值进行比较，得到比较结果；

系统文件调用单元，用于若所述比较结果为所述系统文件的哈希值与预设的测量参考值一致，则根据所述调用指令调用所述系统文件；

拦截单元，用于若所述比较结果为所述系统文件的哈希值与预设的测量参考值不一致，则不执行所述调用指令。

第二比较单元，用于获取测量单元的测量记录，并将所述测量记录与测量列表中的历史测量结果进行比较，以验证测量列表中内容的完整性，其中所述测量列表部署在完整性测量架构中用于保存历史测量结果。

请求发送单元，用于在 Asterinas 内核上运行信任域扩展驱动程序，并通过所述信任域扩展驱动程序触发测量单元向所述信任域扩展发送 TDCALL 请求；

测量数据记录单元，用于根据所述 TDCALL 请求，记录运行时测量数据；

运行时测量寄存器扩展数值获取单元，用于通过SHA384算法计算所述运行时测量数据，得到本地运行时测量寄存器扩展数值，并将所述本地运行时测量寄存器扩展数值存储在信任域扩展的运行时测量寄存器中；

第三比较单元，用于通过信任域扩展认证报告读取所述本地运行时测量寄存器扩展数值，并将所述本地运行时测量寄存器扩展数值与所述测量列表中的历史测量结果进行比较，若所述本地运行时测量寄存器扩展数值与所述测量列表中的历史测量结果一致，则通过验证。

在一种实现方式中，所述第一系统内核获取模块包括：

Asterinas-IMA内核获取单元，用于通过所述Rust-TPM功能与可信平台模块交互，通过所述Rust-TPM功能与完整性测量架构交互，将所述Rust-TPM功能和所述Rust-TPM功能整合到所述完整性测量架构中，并用所述信任域扩展的运行时测量寄存器替换可信平台模块的平台配置寄存器，得到Asterinas-IMA内核。

基于上述实施例，本发明还提供了一种智能终端，其原理框图可以如图4所示。该智能终端包括通过系统总线连接的处理器、存储器、网络接口、显示屏、温度传感器。其中，该智能终端的处理器用于提供计算和控制能力。该智能终端的存储器包括非易失性存储介质、内存储器。该非易失性存储介质存储有操作系统和计算机程序。该内存储器为非易失性存储介质中的操作系统和计算机程序的运行提供环境。该智能终端的网络接口用于与外部的终端通过网络连接通信。该计算机程序被处理器执行时以实现一种基于信任域扩展的操作系统内核设计方法。该智能终端的显示屏可以是液晶显示屏或者电子墨水显示屏，该智能终端的温度传感器是预先在智能终端内部设置，用于检测内部设备的运行温度。

本领域技术人员可以理解，图4中示出的原理框图，仅仅是与本发明方案相关的部分结构的框图，并不构成对本发明方案所应用于其上的智能

终端的限定，具体的智能终端以包括比图中所示更多或更少的部件，或者组合某些部件，或者具有不同的部件布置。

在一个实施例中，提供了一种智能终端，智能终端包括存储器、处理器及存储在存储器中并可在处理器上运行的基于信任域扩展的操作系统内核设计程序，处理器执行基于信任域扩展的操作系统内核设计程序时，实现如下操作指令：

使用 Rust 语言实现 Rust-TPM 功能，其中所述 Rust-TPM 功能用于在 Asterinas 内核上实现可信平台模块；

使用 Rust 语言实现 Rust-IMA 功能，其中所述 Rust-IMA 功能用于在 Asterinas 内核上实现完整性测量架构；

基于所述 Rust-TPM 功能和所述 Rust-IMA 功能，得到 Asterinas-IMA 内核；

将所述 Asterinas-IMA 内核部署到因特尔信任域扩展上，得到基于信任域扩展的 Asterinas-IMA 内核。

本领域普通技术人员可以理解实现上述实施例方法中的全部或部分流程，是可以通过计算机程序来指令相关的硬件来完成，所述的计算机程序可存储于一非易失性计算机可读取存储介质中，该计算机程序在执行时，可包括如上述各方法的实施例的流程。其中，本发明所提供的各实施例中所使用的对存储器、存储、运营数据库或其它介质的任何引用，均可包括非易失性和/或易失性存储器。非易失性存储器可包括只读存储器（ROM）、可编程 ROM（PROM）、电可编程 ROM（EPROM）、电可擦除可编程 ROM（EEPROM）或闪存。易失性存储器可包括随机存取存储器（RAM）或者外部高速缓冲存储器。作为说明而非局限，RAM 以多种形式可得，诸如静态 RAM（SRAM）、动态 RAM（DRAM）、同步 DRAM（SDRAM）、双运营数据率 SDRAM（DDRSDRAM）、增强型 SDRAM（ESDRAM）、同步链路（Synchlink）DRAM（SLDRAM）、存

储器总线（Rambus）直接 RAM（RDRAM）、直接存储器总线动态 RAM（DRDRAM）、以及存储器总线动态 RAM（RDRAM）等。

综上，本发明公开了一种基于信任域扩展的操作系统内核设计方法，所述方法包括：使用 Rust 语言实现 Rust-TPM 功能，其中 Rust-TPM 功能用于在 Asterinas 内核上实现可信平台模块；使用 Rust 语言实现 Rust-IMA 功能，其中 Rust-IMA 功能用于在 Asterinas 内核上实现完整性测量架构；基于 Rust-TPM 功能和 Rust-IMA 功能，得到 Asterinas-IMA 内核；将 Asterinas-IMA 内核部署到因特尔信任域扩展上，得到基于信任域扩展的 Asterinas-IMA 内核。本发明利用 Rust 和信任域扩展的内存安全特性重新实现完整性测量架构功能，从而大幅提升了操作系统的安全性。

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。

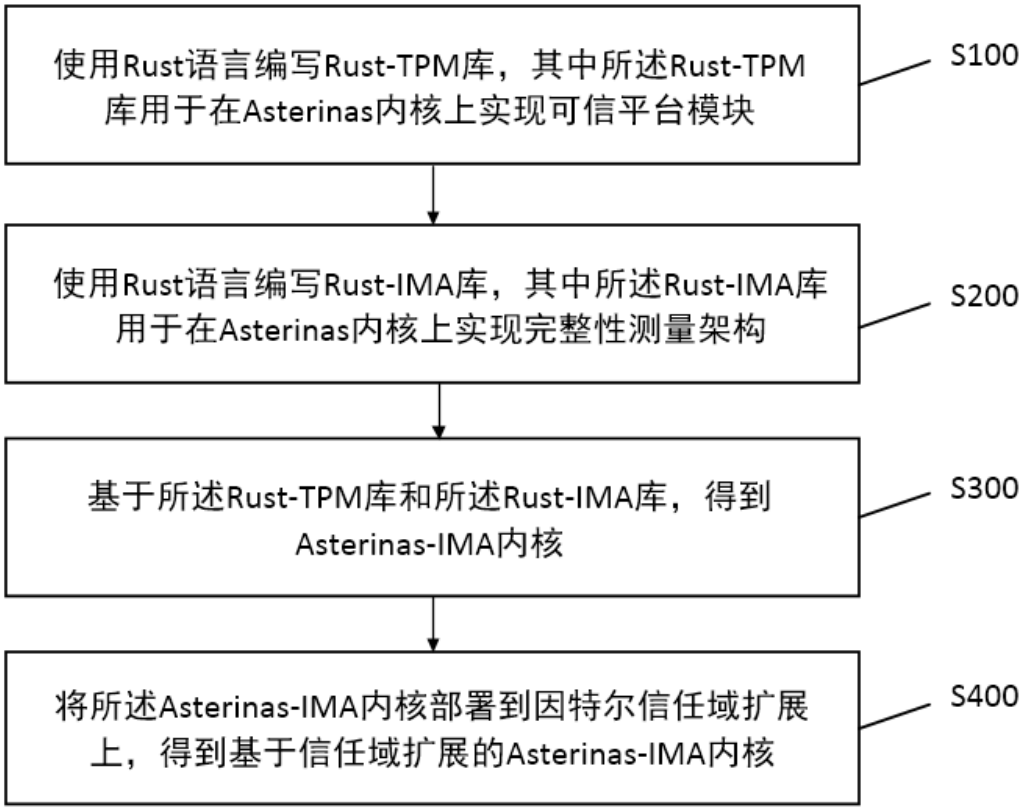

使用Rust语言编写Rust-TPM库，其中所述Rust-TPM库用于在Asterinas内核上实现可信平台模块
S100
使用Rust语言编写Rust-IMA库，其中所述Rust-IMA库用于在Asterinas内核上实现完整性测量架构
S200
基于所述Rust-TPM库和所述Rust-IMA库，得到Asterinas-IMA内核
S300
将所述Asterinas-IMA内核部署到因特尔信任域扩展上，得到基于信任域扩展的Asterinas-IMA内核
S400

图 1

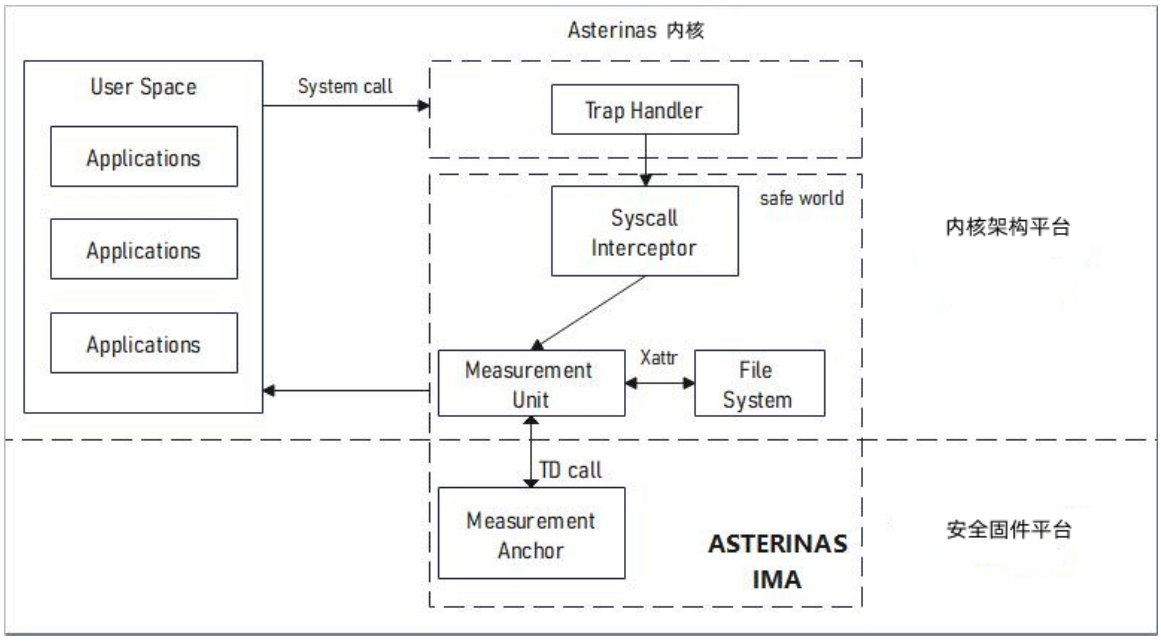

Asterinas 内核
User Space
Applications
Applications
Applications
System call
Trap Handler
safe world
Syscall Interceptor
内核架构平台
Measurement Unit
Xattr
File System
TD call
Measurement Anchor
ASTERINAS IMA
安全固件平台

图 2

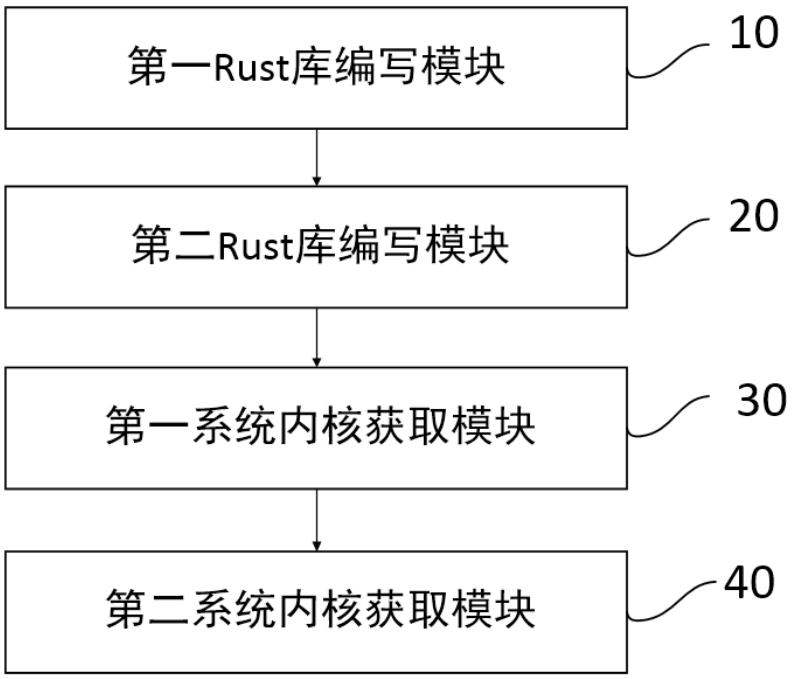

第一Rust库编写模块
10
第二Rust库编写模块
20
第一系统内核获取模块
30
第二系统内核获取模块
40

图 3

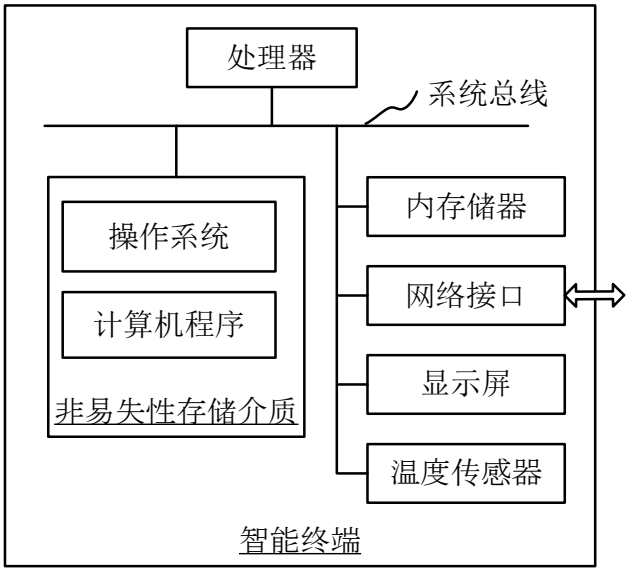

处理器
系统总线
操作系统
计算机程序
非易失性存储介质
内存储器
网络接口
显示屏
温度传感器
智能终端

图 4