CoVE：迈向 RISC-V 平台上的机密计算

Ravi Sahita Rivos公司 美国俄勒冈州波特兰
ravi@rivosinc.com

Vedvyas Shanbhogue Rivos公司 美国德克萨斯州奥斯汀
ved@rivosinc.com

Andrew Bresticker Rivos公司 美国纽约州纽约市
abrestic@rivosinc.com

Atul Khare Rivos公司 美国华盛顿州西雅图
at
ulkhare@rivosinc.com

Atish Patra Rivos公司 美国加利福尼亚州山景城
atishp@rivosinc.com

Samuel Ortiz Rivos公司 法国蒙彼利埃
sameo@rivosinc.com

Dylan Reid Rivos公司 美国加利福尼亚州山景城
dylan@rivosinc.com

Rajnesh Kanwal Rivos公司 英国英格兰沃特福德
rkanwal@rivosinc.com

摘要

多租户计算平台通常由多个软硬件组件构成，包括平台固件、主机操作系统内核、虚拟化监控器，以及在其上运行的实际租户负载（通常运行于虚拟机、容器或应用程序中）。这种模式已在大规模商业部署中得到广泛应用，但其缺点在于，平台的所有组件和操作人员都属于租户的可信计算基（TCB）。对于旨在尽可能缩小可信计算基规模、注重隐私的工作负载而言，这种情况并不理想。机密计算为实现可量化的计算可信计算基提供了良好的起点。机密计算[1] 需要使用经硬件证明的可信执行环境来保护使用中的数据。RISC-V 架构为以全新方式满足机密计算及其他安全范式的要求奠定了坚实基础。本文介绍了一种参考架构，并探讨了 RISC-V 平台机密计算所需的指令集架构、非指令集架构和片上系统（SoC）要求。本文还讨论了面向 RISC-V 平台的机密虚拟机指令集架构和非指令集架构扩展提案，称为 CoVE。

1. RISC-V 指令集架构及其用法

RISC-V 硬件线程（hart）以某一特权级运行，该特权级在一个或多个控制与状态寄存器（CSRs）中编码为一种模式。表 1 列出了目前定义的三种 RISC-V 特权级 [2] 。特权级用于保护软件栈中的不同组件；尝试执行当前特权模式不允许的操作时，会引发异常。这些异常通常会导致陷入底层（更高特权级）的执行环境。

级别	编码	Name	缩写
0	00	用户/应用程序	U
1	01	监管者	S
2	10	保留	
3	11	机器	M

表 1：RISC-V 特权级别

机器级具有最高特权，是 RISC-V 硬件平台唯一强制要求的特权级。在机器模式（M-mode）下运行的代码通常被视为可信，因为它可以低级访问机器实现。用户模式（U-mode）和监管者模式（S-mode）分别用于常规应用程序和操作系统。每个特权级都有一组核心特权 ISA 扩展，以及可选扩展和变体。例如，监管者模式经过扩展后可支持虚拟机监控程序执行，并用于 CoVE（见第 3 节）。实现可以提供 1 到 3 种特权模式，在降低隔离能力与降低实现成本之间权衡。所有硬件实现都必须提供机器模式（M-mode），因为只有该模式能够不受限制地访问整台机器。

RISC-V 支持用于上下文切换的特权指令，以便向特权环境发出请求，或在陷阱处理后将执行返回到较低特权级别的环境。ECALL 用于向更高特权级别的环境发出请求，而 MRET 和 SRET 分别用于返回到 S 模式和 U 模式特权级别。

2. 对手与基本威胁模型

所考虑的对手类别如下：

- 非特权软件对手- 包括由监管者模式系统软件管理的、在用户模式下执行的软件。此对手可以访问用户模式控制状态寄存器、进程/任务内存以及进程上下文中的 CPU 寄存器。
- 系统软件对手——包括在监管者模式或虚拟监管者模式下执行的系统软件。此类对手可以访问特权控制状态寄存器、分配的系统内存、CPU 寄存器、I/O 设备，并诱发设备访问。
- 启动软件对手——包括在启动时执行的系统软件（机器模式），例如 BIOS、内存配置、能够访问系统内存的设备固件、CPU 寄存器、关键 I/O 设备和 IOMMU。

- 简单硬件对手——包括硬件攻击，例如使用总线中间设备监视内存/设备接口、进行电压/时钟毛刺攻击、观测电磁辐射及其他辐射、通过仪器或探测电源轨分析功耗；这些攻击可能使对手能够篡改内存中的数据。

- 高级硬件对手——包括以下攻击：
  对设备拥有不受限制的物理访问权限，以及利用各种机制篡改或逆向分析硬件可信计算基的攻击，例如利用扫描电子显微镜、纤维注入攻击、侵入式开封等手段提取密钥，以及供应链攻击。
- 侧信道／隐蔽信道对手——包括以下攻击：
  利用任何显式或隐式共享状态（架构层面或微架构层面），通过推断共享资源的特征（例如缓存、分支预测状态、内部微架构缓冲区、队列），跨特权边界泄露信息。此类攻击可能需要使用高精度计时器来泄露信息。

需要考虑的主要威胁类别包括：机密工作负载的使用中数据的机密性、完整性或重放保护遭到破坏。使用中数据可能位于内存中或内部硬件状态中，并可能通过以下方式遭受攻击：

{v1)带内 CPU 访问（通过 CPU 加载/存储 ISA 进行软件访问）。
- 由于软件诱发的侧信道导致机密性丧失而产生的旁路 CPU 访问，可分为以下四个子类别——

  - 可信计算基（TCB）中的瞬态执行侧信道攻击
    组件或工作负载，例如通过共享缓存、分支预测器投毒、逐步触发页面错误。o 因共享缓存及其他共享资源而导致的架构侧信道攻击，例如 prime/probe、flush/reload。o 利用已知明文访问密文，对内存加密密钥发起字典攻击。o 因性能监控和/或调试状态导致的侧信道。

- 旁路软件引起的内存变更，例如 row-hammer 导致完整性丧失。

- 来自设备/加速器的未授权外部 DMA。
- 固件通过以下方式进行旁路 CPU 访问：o 利用物理内存别名。o 篡改关键配置（解码器、I/O 结构的路由表）。

- 由于以下原因导致工作负载执行错误：o 恶意注入故障。o 丢弃中断。o 操纵 M 模式控制状态寄存器中的时间。

- 导致恶意地址的软件或硬件攻击
  通过攻击内存中的 MMU、IOMMU 结构以及硬件中的相关地址转换缓存结构（例如转换后备缓冲器、分页级缓存）进行地址转换

- 由针对 CPU/关键硬件协处理器（例如 RoT）发起的硬件攻击

  - 毛刺攻击、故障注入及其他物理攻击。
    o 暴露给其他 CPU 插槽、内存的接口/链路。o 硬件木马和供应链攻击。

- 通过利用其中的弱点发起密码学攻击
  用于密封机密数据或进行远程证明的密码学方法/协议。

- 将系统的可信计算基（或可信计算基衍生的信息，例如远程证明证据）降级或伪造为不安全的配置、旧版（存在漏洞）版本，或加载/回滚至恶意软件、固件组件。

保护租户工作负载使用中的数据的机密性和完整性至关重要；同时，防范工作负载对主机可用性造成的威胁也属于范围之内——即防止机密工作负载对主机平台造成拒绝服务。另一方面，反过来保护机密工作负载的可用性则不属于此威胁模型的范围。

3. RISC-V 虚拟机监控程序扩展

RISC-V 虚拟机监控器扩展对监管者级架构进行虚拟化，以支持使用一型或二型虚拟机监控程序高效托管客户操作系统。虚拟机监控程序扩展将监管者模式转换为虚拟机监控程序扩展监管者模式（HS 模式——参见表 2），虚拟机监控程序或支持托管的操作系统在此模式下运行。虚拟机监控程序扩展增加了一个新的地址转换阶段（G 阶段），将客户物理地址转换为监管者物理地址，从而为客户操作系统虚拟化内存和内存映射 I/O 子系统。HS 模式的行为与 S 模式相同，但增加了用于控制新地址转换阶段并支持在虚拟 S 模式（VS 模式）下托管客户操作系统的指令和控制与状态寄存器。S 模式操作系统无需修改，即可在 HS 模式下运行或作为 VS 模式客户运行。

Virtualization Mode (V)	Nominal Privilege	Abbreviation	Name	Two-Stage Translation
0	U	U-mode	User mode	Off
0	S	HS-mode	Hypervisor-extended supervisor mode	Off
0	M	M-mode	Machine mode	Off
1	U	VU-mode	Virtual user mode	On
1	S	VS-mode	Virtual supervisor mode	On

表 2：RISC-V 特权级别和虚拟化模式

3.1 RISC-V CoVE

本节介绍所提出的模型，该模型使用 RISC-V 虚拟机监控程序扩展来支持 CoVE。CoVE 提出了非指令集架构和指令集架构扩展，以对一个新的 HS 模式软件模块——可信执行环境安全管理器（TSM）进行空间与时间隔离。该模块负责管理工作负载资产的安全属性，防止主机操作系统和虚拟机监控程序访问这些资产。通过一个经硬件证明的机器模式可信计算基组件使用指令集架构扩展（见第 4 节），可实现 TSM 与主机之间的隔离。

TSM驱动程序。TSM使用G阶段页表，强制执行以机密虚拟机运行的工作负载之间的隔离属性。机密虚拟机被称为 CoVE TEE 虚拟机或 CoVE TVM。

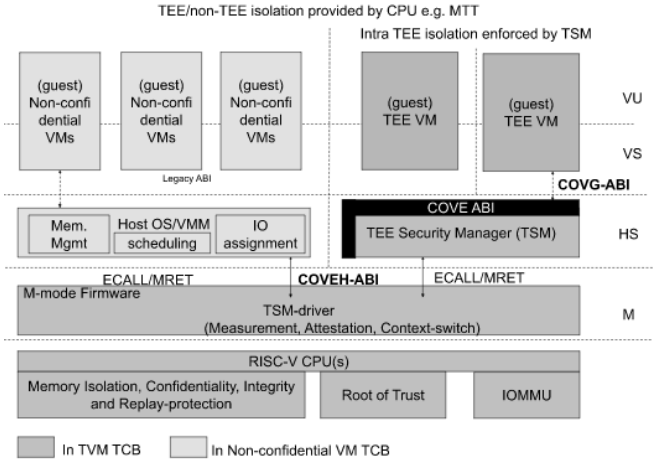


图 1：RISC-V CoVE 参考架构

CoVE TCB 由 TSM 构成，TSM 在可信执行环境与非可信执行环境组件之间充当可信计算基中介。片上系统信任根（RoT）会测量 TSM 驱动程序和 TSM，以支持远程证明。TSM 组件的设计力求精简，仅专注于为 TEE 虚拟机强制实施安全属性。可信计算基还包括负责为使用中的工作负载数据强制实施机密性和完整性属性的硬件组件（处理器和片上系统）。VMM 不受信任，并继续管理所有工作负载（包括机密和非机密工作负载）的资源。

CoVE 使操作系统/虚拟机监控器能够继续担任资源管理器，全面掌控平台资源，即使对于 TVM 也不例外。不受信任的操作系统/虚拟机监控器所管理的资源包括内存、CPU、I/O 资源，以及执行 TVM 工作负载所需的平台功能。将 H 扩展用作机密工作负载的指令集架构契约后，大多数工作负载都可以迁移到 TVM，无需重构应用程序，并提升易用性 [4]。这种方法能够广泛采用指令集架构原语，并最大限度地减少针对机密工作负载的指令集架构变更，从而降低复杂性、可信计算基规模和验证负担。

TSM 对主机的内存隔离通过 CoVE 提出的新型硬件隔离原语——内存跟踪表（MTT）实现。该表允许虚拟机监控程序将内存区域（及页面）捐赠给 CoVE TVM，并由 TSM 驱动程序和 TSM 设置跟踪与分配策略。由 MTT 隔离或实施访问控制的内存无法被主机访问，还可以通过密码机制进一步防止物理访问（以确保机密性、完整性和重放保护）。硬件通过为每个硬件线程维护的机密限定符，对机密内存区域实施访问控制。每个硬件线程可通过 TEECALL（一种通过 ECALL 和 MRET 实现的水平陷阱）启用此模式／限定符，并通过 TEERET 将其禁用——这些上下文切换流程由 TSM 驱动程序支持。只有在设置了机密限定符时，硬件线程才可访问机密内存。第 4 节将详细介绍这些扩展。

TSM 驱动程序将 TSM 隔离功能委托给 TSM，具体包括向 TVM 分配机密内存并对其进行隔离。TSM 驱动程序执行以下 TCB 功能：

- 通过 MMU 扩展（MTT、IOMMU）建立 TSM 的空间与时间隔离。- 为 TSM 执行进行硬件状态上下文切换（通过 ECALL、MRET）。- 提供固件信任根接口，以连接硬件信任根并支持对 TCB 进行远程证明。

TSM 通过向主机公开的 ABI，与平台上的特定于机器的功能交互。TSM 提供的 ABI 包含以下功能：

- 用于管理 TVM 生命周期的 COVH-ABI，例如创建 TVM、向 TVM 添加页面和虚拟硬件线程，以及调度 TVM 虚拟硬件线程执行。- 面向 TVM 上下文的 ABI，称为 COVG-ABI，使 TVM 工作负载能够请求远程证明、内存共享/取消共享或半虚拟化输入输出功能。- 第三个 ABI，即 COVI-SBI，用于管理安全中断设施（使用 RISC-V高级中断架构 [11]）。

TSM 的功能在设计上仅限于提供必要的安全原语，以确保操

TVM 执行与非机密虚拟机的运行类似，不同之处在于，

在第 4节至第6节中，我们将介绍构成 RISC-VCoVE基础构件的指令

4. ISA 能力

以下 ISA 扩展可保障 CoVE 工作负载的安全属性，并使其能够高性能地扩展至大型服务器平台。

4.1 RISC-V 硬件线程的机密限定符：将 TVM 与主机软件隔离

需要额外的空间与时间隔离原语。其目标是在允许主机软件保留特权运行模式的同时，实现非可信计算基（TCB）元素与可信计算基元素之间的分离，具体包括主机虚拟机监控程序、TSM 和 TVM 工作负载。为弥补这一缺口，CoVE 提出一种按处理单元（例如硬件线程）维护的“机密模式限定符”，并将其传递至物理内存保护（PMP）、内存管理单元（MMU）查找以及片上系统（SoC）互连结构。该限定符在硬件线程中至少占用 1 位状态；互连结构上还可以携带额外位，以向平台传达其他访问类别或模式。表 3 展示了纳入机密限定符后的不同运行模式。第 4.2 节将介绍基于此模型、用于空间隔离的 PMP 和 MMU 扩展。

4.2. 机密限定符之外的内存和设备域分配（处理单元上的机密限定符）

为了使 TCB 固件能够声明处理单元处于机密模式，我们还需要具备隔离物理地址的能力，从而对内存和设备实施强制所有权，使其归属于安全域（例如机密域），并映射到机密性/完整性等属性。即使主机软件完全控制现有的访问控制机制（例如页表（内存管理单元）和/或物理内存保护寄存器（物理内存保护）），也必须如此。本节介绍了域分配物理内存属性（PMA）机制，以满足这一要求——该属性是与内存关联的动态属性，可分配给不同的隔离域，也可从中撤销。此处所说的域，仅指相互隔离的物理地址区域。

完整系统的物理内存映射包含各种地址范围，其中一些范围可能支持特定属性，而另一些则可能不支持。在 RISC-V 系统中，机器物理地址空间各区域的这些属性称为物理内存属性（PMA）。硬件 PMA 检查器会针对任何物理内存访问检查 PMA，包括经过虚拟地址到物理地址转换的访问。通常，同一特权级的软件对所有内存的访问权限都相同——ISA 没有提供支持在特权级内实现隔离的机制——这一缺口可通过增加一层动态 PMA 强制执行来弥补。为此，本文提出域分配 PMA：其思路是为每个物理内存区域（和页面）维护元数据，用于指定域所有权。元数据可以是最精简的 1 位（本文将其指定为 C 位），也可以包含其他元数据；在仅支持 M 或 M/U 模式的系统中，它可以与 PMP 关联，而在支持 M、S/H 和 U 级别的系统中，则可通过 MMU 扩展强制执行。在此方案中，使用一种 MMU 扩展来查询存储在内存中的内存跟踪表（MTT，图 2），以强制执行域分配。

表 3：硬件线程机密限定符与特权模式

Vir虚拟化模式 (V)	名义上的特权	机密限定符 (C)	缩写和名称	地址转换
0	U	0	U 模式（用户模式）	[单级 /裸模式] + [物理内存保护] + **MTT**
0	S	0	HS 模式（虚拟机监控程序-扩展监管者模式）	[单阶段/ /裸模式] + [物理内存保护] + **MTT**
0	M	0	机器模式（机器模式）	裸模式（TSM-可信计算基中的驱动程序）*
1	U	0	VU 模式（虚拟用户）	[两阶段翻译] + [物理内存保护] + MTT
1	S	0	VS 模式 (虚拟监管者)	
1	U	1	机密 VU 模式（虚拟用户）	[两阶段地址转换] + [物理内存保护] + MTT
1	S	1	机密 VS-模式（虚拟监管者）	

*关于在 RISC-V 安全社区中隔离机器模式的讨论仍在进行中（目前，机器模式的可信计算基可能会通过将组件降级到用户模式或监管者模式来加以限制）；另一种方案是将 TSM 驱动程序的功能并入 CPU ISA。

启用 MTT 的硬件线程必须强制执行以下属性：- 处于非机密模式的硬件线程不得访问 MTT 标记为机密的内存（即为其他隔离域隔离的内存）。

- 与机密工作负载关联的硬件线程 必须是根据 MTT 唯一获准访问该工作负载所拥有内存的硬件线程；此类硬件线程可以选择访问全局共享内存，以支持输入输出用例。还可以强制实施机密工作负载所需的其他属性，例如，代码提取和页表遍历应始终通过机密内存进行。

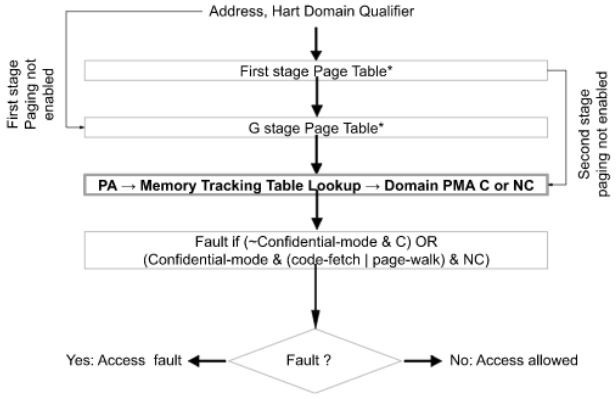


图 2：内存跟踪表强制执行

MTT 能力仍由可信计算基（TCB）控制——通过将其编程接口与硬件线程的机密限定符相关联——因此仅允许 TSM 驱动程序以及由 TSM 驱动程序明确委派的 TSM 动态管理该能力。C 位（以及可能的其他元数据位）可以

选择加密密钥，以针对机密内存与非机密内存实施额外的物理内存保护。实现可以选择使用与 TVM 工作负载相关联的独立密钥。

另一个关键要求是确保 TSM 的硬件线程上下文状态具有时间隔离性——上述内存隔离模型可以支持由域上下文切换触发的状态隔离。在 CoVE 中，这些域上下文切换流程可以通过非指令集架构 ABI 实现——CoVE 参考架构（图 1）描述了新的 ECALL 流程（TEECALL、TEERET），用于在不可信的主机虚拟机监控程序与 TSM 之间实现隔离；这些流程由机器模式下的 TSM 驱动程序协调。此流程也可以由硬件线程承载并作为指令集架构的一部分实现，从而利用域隔离内存完成上下文切换。

目标是创建一种通用流程，用于调用可信计算基组件；该流程可由不可信的机器模式调用，以调用可信的机器模式组件。机器模式可信计算基组件可以将此流程扩展到更低的特权级别。

4.3. 机密中断 RISC-V 高级中

中断架构 [11] 定义了 VS 模式客户机的中断文件虚拟化。机密计算引入的要求是隔离（并保留）与 TVM 相关联的客户机中断文件（VS 模式级寄存器文件）——这通过中断文件的机密模式限定符以及用于内存驻留中断文件的 MTT 来支持。当硬件线程不处于机密模式时，访问 TEE 分配的 VS 级寄存器文件状态必须导致非法指令异常（当 V==0 时）或虚拟指令异常（当 V==1 时）。中断传递模型保持不变。

5. 非指令集架构 [SW ABI]

CoVE TVM 的生命周期经历以下阶段：TVM的构建与初始化、执行以及与主机操作系统/虚拟机监控器的相关运行时交互，最后是拆除TVM 并回收分配给 TVM的资源。当主机操作系统/虚拟机监控器通过 COVH-ABI 调用 TSM，或 TVM 通过 COVG-ABI 调用 TSM 时，这些操作由 TSM 执行。

在 CoVE TVM 中执行机密工作负载，需要按顺序执行以下一个或多个步骤。这些步骤由不受信任的操作系统/虚拟机监控器（主机）或 TVM 执行，并调用 TSM 来强制实施安全不变量。

- 平台 TSM 检测与能力枚举——检测 TSM 是否存在、其版本及其能力。
- 将非机密内存转换为机密内存：一种允许 VMM 将内存捐赠给 TVM 的内在操作。
- 创建可信虚拟机（TVM）：一种为主机 TVM 状态分配机密内存的内在操作。
- 通过 G 阶段分页结构将机密内存页捐赠给 TSM，以进行 TVM 页映射。
- 定义 TVM 内存区域：将 TVM 地址空间分配为机密或非机密，并对后续向这些地址空间进行内存分配的操作添加谓词条件。
- 将 TVM 代码和数据载荷初始化到机密内存区域：作为 TVM 构建过程的一部分，这种内在操作会对内存进行度量，并将未度量的内存页清零。
- 创建 TVM 虚拟 CPU：为主机 TVM 虚拟 hart 上下文分配机密内存页。TSM 会在 TVM 执行前保存上下文结构，并在执行后恢复这些结构。
- 完成TVM 创建：这些内建函数用于完成远程证明的 TVM 度量过程。
- TVM 执行：用于调度、注入中断以及处理 TVM 故障和退出的内在函数。
- 映射TVM 按需清零机密内存区域：支持将内存延迟添加到 TVM 中。TSM 验证机密内存的分配和映射属性未被违反。
- 映射 TVM 非机密共享页——支持按需将共享内存映射到用于输入输出或受控 CPU 状态共享的 TVM 共享内存区域中。TSM 会验证机密内存的分配和映射属性未被违反。
- 拆除 TVM——用于停止 TVM 执行并释放机密内存资源（归还给 TSM）的内置机制。
- 机密内存的重新分配——通过 TSM 强制机密内存由各个 TVM 互斥所有权，实现对未分配机密内存的重复使用。
- 回收机密内存 – 用于将内存从机密状态转换回非机密状态，以便用于非机密虚拟机/主机工作负载。

ABI 函数的详细信息可参见 AP-TEE 任务组制定的详细 RVI CoVE 规范 [3] 。

6. 非指令集架构 [SoC 能力]

本节介绍为机密计算所需的硬件线程功能提供补充的平台功能。

6.1 SoC 信任根

RoT 硬件引擎是平台可信计算基（TCB）的度量、报告和更新根。它负责管理片上系统的安全生命周期和密钥材料，因此应由芯片制造商认证。RoT 会验证片上系统固件和硬件模块的配置，并实现设备身份组合引擎（TCG DICE[5]），其中每个 TCB 层都会度量下一层并配置凭据。随后，依赖方可使用远程证明协议验证以 RoT 为根的工作负载度量值。远程证明协议用于验证从硬件到 TCB 固件的信任链，其中包括 TSM 驱动程序和 TSM。每个 TVM 都会从 TSM 派生自己的远程证明证书，以便使用 IETF RATS [6] 等框架支持标准远程证明协议。

6.2. 机密内存的片上系统考量

存储在片上系统封装之外（例如内存中）的使用中的工作负载数据，可能需要明确的密码学保护。平台上的内存最初属于非机密世界中的不可信资源，之后通过 TSM 转换为对应的机密资源；TSM 使用片上系统专用机制，启用对内存的保护，以防止主机访问。转换过程完成后，VMM 可通过 TSM 将机密内存分配给 TVM。不再使用的已转换机密资源可以自由分配给另一个 TVM。不过，VMM 必须回收未使用的机密资源，以便在非机密世界中使用（此过程由 TSM 跟踪并强制执行）。

为满足这些要求，片上系统必须将域分配PMA映射到发往片上系统内存控制器和互连结构的事务中，以区分机密内存流量和非机密内存流量。必须在片上系统缓存中实施隔离限定符（包括机密模式位），并根据所应对的威胁模型采用适当的访问控制、加密、完整性和重放保护。图 3 展示了一个映射示例。

6.3. 片上系统平台数据保护

大规模服务器平台需要可靠性、可用性和可服务性（RAS）。此外，平台调试和性能监控是调整和调试工作负载的关键功能。然而，这些设施可能被用作攻击向量，因此片上系统必须提供适当的控制措施，确保机密数据不会通过这些机制遭到恶意访问。

为缓解来自可靠性、可用性和可服务性、服务质量、调试和性能监控平台机制的攻击，必须使用适当的授权技术 [12] 来激活这些机制，并向 TVM 提供主动选择控制（由 RoT 记录，并由 TSM 和 TSM 驱动程序强制执行），从而对向主机暴露的 TVM 运行时状态提供明确控制。最后，特定 TVM 的调试和性能监控激活状态会在其证明状态中明确报告。

6.4. 片上系统输入输出和设备

将设备绑定到 TVM 会扩大 TVM 的可信计算基，并引入新的威胁，例如互连结构上的设备标识符欺骗、设备链路上的数据泄露、设备固件篡改，以及对 IOMMU 和输入输出桥的恶意编程。

将设备连接到 TVM需要片上系统在输入输出桥和根端口中提供支持，以便保护从输入输出桥到设备的链路。要为发送到设备的数据设置端到端保护，前提条件之一是验证设备的真实性。目前正在积极开展标准化工作，通过 DMTFSPDM [7]，验证设备证书的协议，以及使用IDE [8] 保护 PCIe等互连结构上数据的完整性和机密性。TSM 可以控制设备接口的信任状态

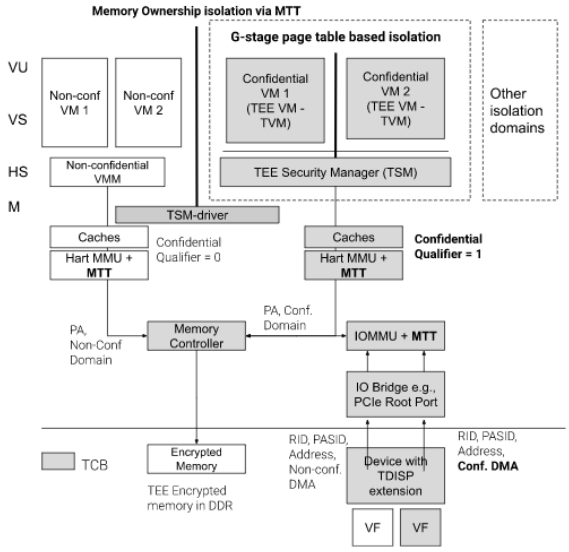


图 3：片上系统视图：保密限定符与域分配

通过PCIeTEE 设备接口安全协议（TDISP [9]）进行细粒度分配（虚拟功能）。在片上系统一侧，RISC-V 正在推进 IO-PMP，以便能够将输入输出控制器编程接口隔离至TCB元素。类似地，所提出的 RISC-V输入输出内存管理单元 [10] 可以提供专用的安全编程接口，这些接口可由处于机密模式的硬件线程强制执行，从而使 TSM 能够对输入输出内存管理单元进行编程，并确保满足 TVM 安全属性。

7. 相关工作

本节简要综述相关的商业或学术可信执行环境方案。

Keystone [13] 使用RISC-V M模式固件实现安全监控器，并使用物理内存保护来隔离机密区。如定义所述，Keystone 不支持虚拟机机密区。Keystone 机密区由连续内存构成，这限制了启动后操作系统对机密区进行内存管理时的可扩展性。每个机密区都需要一个物理内存保护条目；对于具有N 个硬件物理内存保护寄存器的架构，最多可支持N-2 个机密区。相比之下，CoVE使用MTT提供机密内存隔离，并使用G阶段页表进行机密内存分配，从而简化操作系统的内存管理，因为其以架构页大小进行操作。

Intel TDX [14] 在一种称为安全仲裁的模式下运行 TVM，其功能与 CoVE TSM 类似。TDX 模块是一个在 CPU VMX-root 模式下运行的软件组件。Intel TDX 将 TVM 地址空间划分为私有地址空间和共享地址空间，并使用 Secure-EPT（相当于 CoVE 中的 G 阶段页表）管理机密内存分配。每个信任域都会分配一个唯一的临时密钥，用于内存加密。TVM 私有内存和 Secure-EPT 页通过可选的加密完整性机制防止篡改，该机制使用截断的 MAC 存储，并结合

内存 ECC（这会限制可靠性机制）。Intel TDX 支持基于硬件根密钥的远程证明。Intel TDX 将内存加密密钥标识符嵌入物理地址中，这限制了其可扩展性。

AMDSEV-ES-SNP[15] 使用平台安全处理器（PSP）作为固件可信计算基组件来加密TVM，并为每个TVM分配一个临时内存加密密钥。SNP使用反向映射表，通过将主机地址映射到与页面关联的客户机地址来跟踪内存所有权。然而，反向查找仅对写操作强制执行，从而可能暴露密文。对于大型核心系统和频繁的可信操作（例如页面迁移），PSP可能会限制可扩展性。AMD SEV支持基于硬件根的远程证明。

ARM CCA [16] 引入了 Realm 虚拟机，这些虚拟机在 RealmManagementMonitor（RMM）的控制下运行——RMM 作为不受信任的虚拟机监控程序的对等方运行——它是 CoVE TSM 的对应机制。CCA 扩展了ARMISA，以便从主机调用 RMM。Realm 与其他上下文之间的隔离通过 Granule Protection Table（GPT）提供，该表用于跟踪所有访问中的内存所有权。Realm 支持硬件远程证明。

上述所有方法的一个常见缺点是缺乏对机密输入输出的支持——这仍是一个持续推进的工作领域。

8. 结论与未来工作

本文介绍了 CoVE，这是首个面向应用级处理器的框架，旨在为运行丰富操作系统和虚拟机监控程序的 RISC-V 平台提供可扩展的机密虚拟机支持。本文还介绍了 RISC-V 平台支持机密虚拟机所需的基础扩展。这些扩展正在 RISC-V国际组织的技术组中进行开发，以推动其在 RISC-V 平台上获得批准。CoVE 的一个活跃的开源概念验证实现托管于 GitHub [17]。

致谢

作者谨此感谢 RISC-V国际组织论坛中各位贡献者所开展的技术讨论并提供的反馈。这些论坛正在讨论 ISA、非指令集架构和平台规范，以推动其获得批准，具体包括：安全横向委员会（HC）、可信计算特别兴趣小组（SIG）、应用处理器可信执行环境任务组（TG）、特权架构任务组（TG）、平台运行时服务任务组（TG）、虚拟机监控程序特别兴趣小组（SIG）和片上系统基础设施特别兴趣小组（SIG）。

参考文献

[1] 机密计算联盟（2022）《机密计算通用术语》，网址：https://confidentialcomputing.io/wp-content/uploads/sites/85/2023/01/Common-Terminology-for-Confidential-Computing.pdf（访问日期：2023年3月2日）。[2] Waterman, A., Asanović, K., Hauser, J. 等（2021）《RISC-V 指令集手册第二卷：特权架构》，GitHub。RISC-V国际组织。网址：https://github.com/riscv/riscv-isa-manual（访问日期：2023年2月26日）。[3] Sahita, R. 等（2023）《RISC-V 上的机密计算：应用处理器可信执行环境（AP-TEE）》。GitHub。RISC-V国际组织。网址：https://github.com/riscv-non-isa/riscv-ap-tee（访问日期：2023年2月26日）[4] Mark Russinovich、Manuel Costa、Cédric Fournet、David Chisnall、Antoine Delignat-Lavaud、Sylvan Clebsch、Kapil Vaswani 和 Vikas Bhatia。（2021）《迈向机密云计算：将硬件强制加密保护扩展到使用中的数据》。《Queue》19卷1期，第20页（2021年1月至2月），共28页。https://doi.org/10.1145/3454122.3456125[5] Dominik Lorych 和 Lukas Jäger。（2022）《DICE设计空间探索》。载于《第17届可用性、可靠性与安全国际会议》[6] H. Birkholz、D. Thaler、M. Richardson、N. Smith 和 W. Pan。（2023）《远程证明程序（RATS）架构》。RFC编辑部。网址：https://www.rfc-editor.org/info/rfc9334（访问日期：2023年3月2日）[7] Brett Henning 等（2022）《安全协议与数据模型（SPDM）架构白皮书》。DMTF。DSP2058。1.2.0版。网址：https://www.dmtf.org/sites/default/files/standards/documents/DSP2058_1.2.0.pdf（访问日期：2023年3月2日）[8] Claire Ying。（2021）《为什么要采用 PCIe 和 CXL 的 IDE安全技术？》（2021）。Cadence。网址：https://www.chipestimate.com/Why-IDE-Security-Technology-for-PCIe-and-CXL/Cadence/blogs/3512（访问日期：2023年3月2日）[9] Richard Solomon。（2023）《全新的 PCIe TDISP架构通过虚拟服务器保护设备接口》。Synopsys。网址：https://blogs.synopsys.com/from-silicon-to-software/2023/02/08/what-is-tdisp-pcie-io-virtualization/（访问日期：2023年3月2日）[10] Vedvyas Shanbhogue 等（2023）《RISC-V IOMMU架构规范》。GitHub。RISC-V国际组织。网址：https://github.com/riscv-non- isa/riscv-iommu/blob/main/riscv-iommu.pdf[11] John Hauser 等（2023）《RISC-V高级中断架构》。1.0-RC2版。GitHub。RISC-V国际组织。网址：https://github.com/riscv/riscv-aia/releases/download/1.0-RC2/riscv-interrupts-1.0-RC2.pdf[12] Tim Newsome 等（2019）《RISC-V外部调试支持》0.13.2版。RISC-V国际组织。网址：https://riscv.org/wp- content/uploads/2019/03/riscv-debug-release.pdf（访问日期：2023年3月2日）[13] Dayeol Lee、David Kohlbrenner、Shweta Shinde、Krste Asanović 和 Dawn Song。2020。《Keystone：用于构建可信执行环境的开放框架》。载于《第十五届欧洲计算机系统会议论文集》（EuroSys '20）。计算机协会，纽约，纽约州，美国，第38号文章，1–16页。https://doi.org/10.1145/3342195.3387532[14] R. Sahita 等，2021。《虚拟化工作负载的机密计算指令集架构安全分析》，载于《2021年安全与隐私执行环境设计国际研讨会》（SEED），美国华盛顿特区，2021年，第121–131页，doi：10.1109/SEED51797.2021.00024。[15] 超威半导体公司（2020），《AMD SEV-SNP：通过完整性保护等功能增强虚拟机隔离》。AMD白皮书。网址：https://www.amd.com/system/files/TechDocs/SEV-SNP-strengthening-vm-isolation-with-integrity-protection-and-more.pdf（访问日期：2023年3月2日）[16] Mark Knight、Gareth Stockwell。2021年10月。《Arm机密计算架构》。载于《安全与隐私的硬件及架构支持》（HASP）2021论文集。网址：https://haspworkshop.org/2021/slides/HASP-2021-Session2-Arm-CCA.pdf（访问日期：2023年3月2日）[17] Salus：用于TEE开发的RISC-V虚拟机监控器（2023）。GitHub。Rivos公司。网址：https://github.com/rivosinc/salus（访问日期：2023年4月10日）