未来基于 RISC-V 的云基础设施的确认、验证与测试（VVT）：Vitamin-V 欧洲地平线计划项目视角

Mart´ı Alonso[1], David Andreu[1], Ramon Canal[1],Stefano Di Carlo[2],Cristiano Chenet[2],Juanjo Costa[1], Andreu Girones[1], Dimitris Gizopoulos[3], Vasileios Karakostas[3], Beatriz Otero [1], George Papadimitriou[3], Eva Rodr´ıguez[1],Alessandro Savino[2]

[1]加泰罗尼亚理工大学，西班牙巴塞罗那˙都灵理工大学，意大利都灵[2]雅典大学，希腊雅典 联系邮箱：stefano.dicarlo@polito.it

arXiv:2305.01983v1 [cs.DC] 3 May 2023

摘要——**Vitamin-V 是一项由欧洲地平线计划资助、实施期为2023—2025年的项目。该项目旨在为 RISC-V 打造完整的开源软件栈，以支持云服务。该软件栈的性能目标是达到 x86 架构的同等水平；目前，x86 架构在云计算行业占据主导地位。此外，该项目还旨在打造强大的虚拟执行环境，用于软件开发、确认、验证与测试。该虚拟环境将考虑云部署所需的相关 RISC-V 指令集架构扩展。商用云系统采用了一些目前 RISC-V 虚拟环境尚不具备的硬件特性，包括虚拟化、密码学和向量化。为此，Vitamin-V 将在三种虚拟环境中支持这些特性：QEMU、gem5 和云 FPGA 原型平台。该项目将重点支持基于 EPI 的 RISC-V 设计，涵盖主 CPU 和对云计算至关重要的加速器，例如内存压缩。项目还将为这些指令集架构扩展添加基于 LLVM 的编译器和工具链支持。此外，Vitamin-V 将开发新方法，以确认、验证和测试软件的可信度。本文着重介绍 Vitamin-V 项目支持云应用确认、验证与测试的计划和愿景，尤其强调将提供的硬件支持。**

索引词—**RISC-V、确认、验证、测试、云计算、模拟**

I. 引言

RISC-V 是一种革命性的开源指令集架构（ISA），旨在实现简洁、模块化和可扩展性 [1]。这一令人振奋的发展相较于专有处理器架构带来了诸多优势，包括定制化的可能性以及更低的许可成本 [2]。

尽管 RISC-V 应用已开始在嵌入式领域崭露头角，且具备上述优势 [3], ，但在 RISC-V 才能广泛应用于云应用领域。一个关键障碍是 RISC-V 生态系统的成熟度。近年来，该平台势头强劲，但围绕 RISC-V 处理器的生态系统仍在发展之中，涵盖硬件和软件工具，以及可用供应商和支持服务的数量 [4]。随着生态系统不断成熟，预计这一问题将不再那么令人担忧。

另一个潜在挑战是性能。尽管 RISC-V 处理器能够提供良好的性能，但在某些特定应用中，它们可能尚无法达到较成熟架构（如 x86 或 ARM）的性能水平。这可能会限制 RISC-V 在对性能敏感的云应用中的普及。随着技术不断发展，这一障碍或许会逐渐减小。

兼容性是 RISC-V 广泛普及面临的另一项潜在挑战。许多云应用是为在 x86 或 ARM 架构上运行而设计的，可能无法兼容 RISC-V 处理器。这可能会限制 RISC-V 在特定云环境中的应用。目前正在通过开发仿真和虚拟化解决方案来应对这一问题。

安全性也是一个值得关注的问题。随着 RISC-V 处理器得到越来越广泛的应用，遭受安全攻击的风险也在不断增加。确保基于 RISC-V 的云应用的安全性，将是技术发展过程中需要应对的一项重要挑战。

最后，标准化是另一个需要解决的领域。尽管 RISC-V 是一项开放标准，但内存管理和输入/输出接口等领域仍需进一步标准化。这可能导致不同 RISC-V 实现之间出现兼容性问题，并限制基于 RISC-V 的云应用的可移植性。目前正在努力开发标准化解决方案，以促进互操作性。

Vitamin-V 是一项由欧盟委员会在 2023—2025 年期间资助的研究项目，旨在提出创新解决方案，应对这些挑战。Vitamin-

由欧盟资助。但文中表达的观点和意见仅代表作者本人，并不一定反映欧盟或 HaDEA 的观点。欧盟和资助机构均不对此承担责任。项目编号：101093062

V 将基于面向 RISC-V 核心（尤其是 EPI 核心）的前沿云开源技术，部署完整的云服务 RISC-V 硬件与软件栈。Vitamin-V 旨在提供创新的 RISC-V 虚拟执行环境，支持硬件仿真、模拟和 FPGA 原型验证，从而在实际硬件发布前开展软件开发、验证和确认。此外，Vitamin-V 还致力于为即将推出的 RISC-V EPI 处理器移植完整的交叉编译工具链、软件栈和必需的应用程序库 [5]。这一解决方案有望在推动 RISC-V 于云应用中的采用方面发挥关键作用 [6]。

特别是，在软件开发和部署过程中，确认、验证与测试（VVT）活动是最关键的活动之一，同时云应用的安全性和保密性也面临潜在风险。在概述 Vitamin-V 活动后，本文将重点介绍 Vitamin-V 项目为支持云应用的确认、验证与测试活动而制定的计划和愿景，着重探讨硬件本身能够为这些活动提供的支持。

本文的组织结构如下：第二节概述 Vitamin-V 项目的组织结构，第三节将更深入地介绍该项目针对 RISC-V 云应用的确认、验证与测试技术的实施计划。最后，第四节总结本文的主要贡献。

II. 概念和方法论

Vitamin-V 的目标是创建完整的 RISC-V 云软件栈，在性能方面与占主导地位的 x86 同类系统展开竞争，如图 1 所示。然而，缺乏成熟完整的 RISC-V 系统，给高级云环境配置和软件栈的移植与评估带来了重大挑战。商用云系统使用的部分硬件功能在 RISC-V 虚拟环境和商用硬件核心中已经具备。这些功能包括虚拟化、密码学和向量扩展。

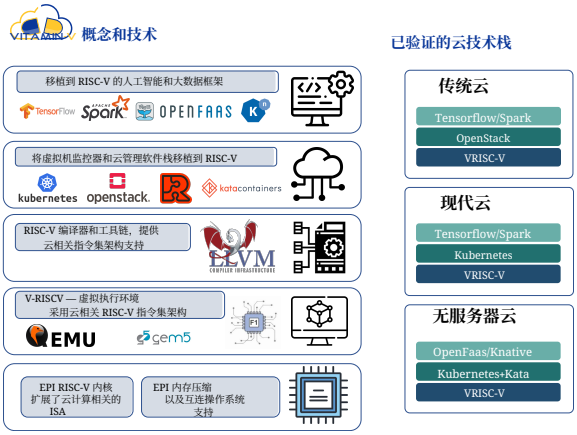

图 1. VITAMIN-V 概念和架构

为应对这一挑战，Vitamin-V 旨在开发*Vitamin RISC-V* 虚拟执行环境 （VRISC-V）。这是一种基于三项前沿技术的多层高性能 RISC-V 虚拟执行环境：功能仿真（QEMU [7]）、周期精确模拟（gem5） [8]„ 以及能够在 AWS EC2 F1 FPGA 上运行的基于 FPGA 的硬件原型系统节点（因而可扩展至数百个节点）（FPGA [9]）。这些技术提供了软件开发、确认、验证和测试所必需的独特功能，使软件开发人员更容易采用 RISC-V。借助 QEMU 和基于 FPGA 的 RISC-V，该项目旨在模拟包含超过 100 个核心的多节点系统。其目标是在 gem5 中以周期精确度模拟多节点系统，并提供微体系结构配置的灵活性 [10]。

Vitamin-V 将在 VRISC-V 虚拟执行环境中支持加速器，例如内存压缩，并提供基于 LLVM 的成熟编译器工具链 [11]，以处理完整的 RISC-V 指令集架构及其扩展。此外，该项目将开发一套确认、验证与测试（VVT）工具集，用于识别软件漏洞和恶意代码序列，确保未来 RISC-V 系统中各软件层的可信度。在成熟硬件问世之前，VVT 工具将利用 VRISC-V 平台，促进软件移植和原型开发。这将推动向云端新型 RISC-V 服务器的迁移。

为实现在 VRISC-V 虚拟执行环境中运行完整的云技术栈，Vitamin-V 将为相关开源云软件发行版移植所有必要的机器相关模块。这些模块包括对运行完整虚拟机（VM）、容器和轻量级虚拟机（KVM、QEMU、Docker、RustVMM）、安全可信执行环境（VOSySMonitorV [12]）、云管理软件（OpenStack、Kubernetes、Kata Containers）以及人工智能和大数据库（Tensorflow、Spark）的支持。

该项目将涵盖经典云栈（面向由 OpenStack 管理的完整虚拟机的运行）、现代云环境配置（面向由 Kubernetes 管理的完整虚拟机和容器），以及无服务器云栈（面向由 Kubernetes 搭配 Kata Containers 管理的轻量级虚拟机的运行）。Vitamin-V 将针对相关人工智能应用（如 GoogleNet、ResNet、VGG19）、大数据应用（基于 Apache Spark 的 TPC-DS）和无服务器应用（FunctionBench、ServerlessBench），对这三种可运行的云环境配置进行基准测试。目标是使软件性能达到对应 x86 系统的水平，并采用 CPU 核心 mark 分数进行公平评估。总体而言，这项开发将建立一个 RISC-V 云栈生态系统，推动其在市场上的应用。

III. 面向 RISC-V 云服务的确认、验证与测试

在云服务时代，在不同计算平台之间转移软件工作负载，同时又不牺牲性能和可信度，往往很困难。

[13]. 从一种架构迁移到另一种架构时，多种因素都可能导致性能下降，包括软件实现不佳、数据结构效率低下以及缓存利用不足。此外，软件在其整个生命周期的任何阶段都可能受到错误或恶意代码的影响。因此，为了优化应用程序并确保其可信度，必须监控其执行过程、识别性能瓶颈，并根据底层硬件定制软件。然而，仅使用执行时间或时钟周期等简单性能指标无法完成这项任务。将软件映射到现代计算系统时，需要考虑多种因素，包括顺序执行与乱序执行引擎、流水线阶段、执行端口及其相应的延迟、重排序缓冲区、加载/存储队列以及缓存组织结构。因此，为了实现深入的架构建模和优化流程，捕获并分析详细的性能指标正变得日益必要 [14]。

本节介绍 Vitamin-V 项目计划如何利用 RISC-V 硬件性能监控器（HPM）单元，为 RISC-V 云服务实现先进的确认、验证与测试。

A. 硬件性能监控

与其他现代处理器一样，RISC-V 处理器配备了硬件性能监控单元，用于跟踪处理器性能。近几十年来，处理器复杂性不断增加，因此需要分层缓存子系统、非统一内存、多线程同时执行以及乱序执行，这使得这些单元变得不可或缺。能够理解并适应资源利用情况的软件，有助于提升性能和效率。

RISC-V 指令集架构的 HPM 单元比 x86 更简单，但它定义了一种灵活的开源性能监控解决方案，可通过多种方式实现。Vitamin-V 计划支持最新的 RISC-V HPM 规范，首先在 VRISC-V 中进行硬件模拟，然后利用 Linux perf [15] 和 PAPI [16]等开源库将其集成到操作系统中。实现的 HPM 将包含用于记录微架构事件的硬件寄存器和计数器，软件可以访问这些寄存器和计数器；此外，还将包含 gem5 中有助于调试的硬件断言 [17]。Vitamin-V 旨在于开发早期至少构建基本的计数器事件，使用现有的 RISC-V 核心，例如 SemiDynamics 的 Atrevido Core [18]。

硬件性能监控单元会跟踪与处理器架构和微架构相关的各种事件，例如已退役指令、分支预测、缓存命中与未命中、浮点运算、硬件中断、经过的核心时钟周期以及核心频率。这些事件会生成多个参数，但处理器只有少量寄存器可用于存储这些参数，因此在任何给定时刻可用的硬件性能计数器数量有限。并发监控事件的设计复杂性和成本

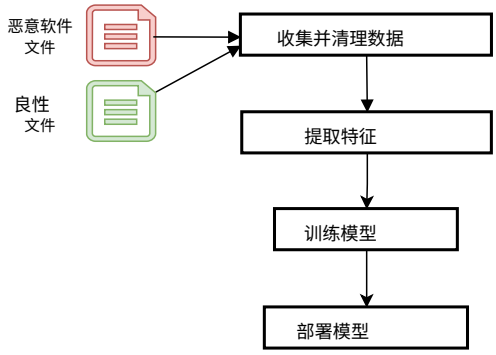

图 2. 用于创建 VITAMIN-V 中所用深度学习模型的方法。

限制了可用硬件性能计数器的数量[19]–[21]。

性能监控指令允许开发人员通过读取或写入计数器的值来访问这些计数器。Vitamin-V 将研究 VRISC-V 的硬件监控扩展，以满足云环境中软件确认、验证与测试的需求。该项目计划利用硬件监控来评估部署在 RISC-V 上的软件的可信度。Vitamin-V 将利用先进的 gem5模拟器 [8] 实现各种监控事件，以支持未来在 RISC-V 核心中实现软件确认、验证与测试。

B. 通过静态分析进行验证、确认与测试（*VVT*）

在软件执行之前的早期阶段评估其可信度至关重要。Vitamin-V 计划开发一款机器学习（ML）工具，用于分析可执行文件的静态内容，以确定其属于良性还是恶意。此前利用可执行文件静态分析来检测软件漏洞和安全威胁 [22], [23] 的研究，促使 Vitamin-V 决定采用深度学习（DL）技术，从大量带标签数据中识别复杂模式。然而，有些情况下，数据集可能不够充分，或者需要检测零日攻击。在这种情况下，可以采用迁移学习（TL）技术来提高检测过程的准确性。迁移学习是指将预训练模型重新用于不同的任务 [24]。

所提出的方法基于与Haddadpajouh等人 [23] 所采用的类似方法，包含四个主要步骤，如图2所示：数据集创建、特征提取、模型训练以及模型部署与调优。

训练任何神经网络模型的第一步，都是准备一份由良性和恶意代码组成的均衡集合。Vitamin-V 专注于检测各种类型的恶意代码，包括 Spectre [25],MeltDown [26],病毒等硬件攻击，以及其他恶意软件。为此，Vitamin-V 将从 VirusTotal [27], VirusShare [28], 或 SourceFinder[29] 等热门公共代码仓库中收集不同的源代码。不过，由于目前缺少恶意 RISC-V 代码的源代码

仓库，计划使用已收集恶意软件的 AMD64 版本，并将其重新编译到目标平台。这种方法适合当前情况；当面向 RISC-V 架构的新恶意代码可用时，也可以将其加入数据集。Vitamin-V 计划从 Debian 仓库下载一组选定的常见 Linux 应用程序包，该仓库已经提供 RISC-V 版本，用于良性代码。最终将构建由两组代码组成的均衡集合。

要对可执行文件进行分类，必须将每个程序表示为特征向量。Vitamin-V 使用 Linux 作为操作系统，因此二进制文件采用可执行与链接格式（ELF）。这些 ELF 二进制文件可以使用 Linux 默认工具反汇编为文本格式，并提取其指令序列（操作码）。通过分析指令序列，可以统计不同操作码序列（n元语法）的频率，进而为每个序列计算特征向量，用于训练神经网络。

将分类后的特征向量输入深度学习网络进行训练，即可生成能够检测良性或恶意代码的模型。然而，由于前文提到的 RISC-V 恶意代码匮乏，生成的模型最初只能检测来自 AMD64 平台的恶意代码。

为克服这一限制，Vitamin-V 计划采用迁移学习技术，将这些知识迁移到一个新的神经网络，使其能够检测 RISC-V 二进制文件中的恶意代码和零日攻击。为微调模型，将收集新的数据，例如硬件性能计数器数据，以调整预训练模型的参数，并提升其检测恶意软件攻击的性能。在经过一段评估期、确认检测机制运行正常后，将部署所提出的模型。为使模型长期保持准确有效，可能还需要进一步收集数据并进行微调。

C. 使用硬件性能计数器进行动态分析

尽管某个应用可能通过静态验证而被认为是可信的，但在运行时仍可能发生损坏，因此有必要进行动态监控。这种损坏可能由环境条件、电磁场、空间辐射、老化、设计缺陷、制造瑕疵以及蓄意攻击引起，进而导致处理器出现异常行为，危及现代计算系统的安全性、可靠性和信息安全。异常检测涉及识别数据中偏离预期行为的模式 [30]。Vitamin-V 计划利用 RISC-V 硬件性能监控单元来实现这一目的，正如图3所报告的概念验证所展示的那样，其中硬件计数器的使用能够区分良性应用程序和恶意应用程序。该图展示了特定比率（L3 缓存未命中次数与 L1 缓存未命中次数之比）的频率箱线图，清晰地识别出三种恶意代码（MeltDown、Spectre 和 ZombieLoad），并将其与三个良性应用程序（其中两个应用程序来自）区分开来。MiBench 基准测试和 Firefox 浏览器中的 YouTube 视频可视化）。

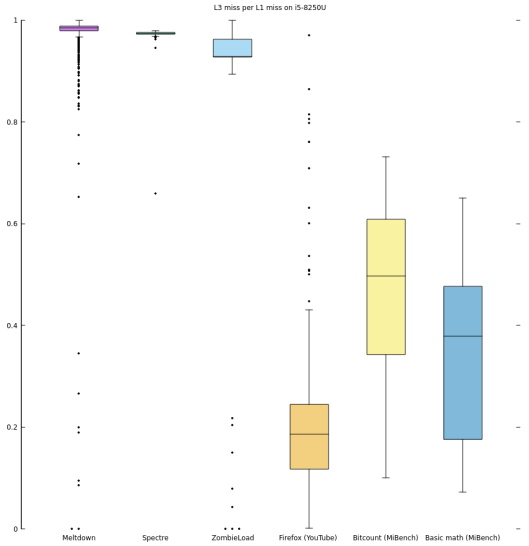

图 3. 使用硬件计数器检测良性和恶意代码。

通过硬件和人工智能（AI）进行异常检测，涉及利用机器学习（ML）算法对处理器中的微架构事件进行动态分析，从而区分正常行为与异常行为。这种方法最早于2013年由 Demme 等人在其提出的基于硬件的恶意软件检测器中引入 [31]，随后被应用于 [32], [33]中的软错误检测。

基于硬件性能计数器（HPC）检测异常的原理源于这样一个事实：程序会呈现阶段性行为 [34], [35]。程序在不同阶段执行不同活动，这些活动可能对应于架构和微架构事件的模式，从而能够根据硬件性能计数器的时间行为模式检测异常。

Vitamin-V 硬件与人工智能异常检测器框架如图 4 所示，其主要目标是区分正常执行与异常执行。该框架由三个基本模块组成：(i) 配备硬件性能计数器（HPC）的 RISC-V 处理器，(ii) 数据收集过程，以及 (iii) 由机器学习分类器执行的异常检测。通过评估检测器的性能和效率，确定其有效性。

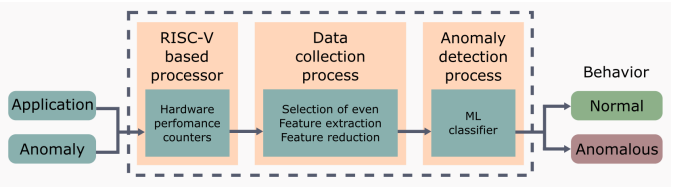

图 4. 基于硬件和人工智能的异常检测器框架。作者绘制。

由于在推测执行条件下监控事件的设计复杂度和成本较高，添加硬件性能计数器面临诸多挑战。移动设备和物联网设备的资源限制更加严格，因此添加更多硬件性能计数器也更为困难。需要权衡硬件事件数量与检测准确率。为达到合理的准确率，一些研究会分析多于可用硬件性能计数器数量的事件，因此需要多次运行应用程序 [31], [36], [37]。这限制了基于硬件的异常检测在运行时的适用性。为弥补因分析事件较少而导致的特征表征不足，需要谨慎设计机器学习分类器。

数据收集过程包括事件选择、特征提取和特征降维 [38]。在硬件和人工智能驱动的异常检测场景中，事件选择是指从处理器提供的多个硬件事件中选取事件，以供异常检测器使用。特征提取是指捕获并存储已记录的硬件性能计数器（HPC），将其置于向量空间中，以供机器学习（ML）模型分析。特征降维（也称为降维）是一种数据处理方式，用于处理高维空间中的冗余维度。冗余维度会增加训练数据集中的噪声测量，从而降低测试的检测率。

事件选择和特征降维旨在选取与异常检测最相关、最具预测性的数据。由于处理器中可用的硬件事件只有几百种，因此可以凭借对各事件在体系结构和微体系结构中的含义的经验知识，或依据其他研究，手动进行事件选择。然而，在某些情况下，研究人员可能需要这些信息，于是开始收集处理器中所有可用的硬件事件，从而产生大数据。在这种情况下，手动分析是不可能的，必须使用某些特征选择技术。常用方法包括主成分分析（PCA）、Fisher评分[39],基于皮尔逊相关系数 [40] 的技术，以及信息增益（也称为互信息） [41]。

特征提取的一个关键特征是硬件性能计数器（HPC）的采样周期。这个值没有固定规则。不过，在基于硬件的检测实验中，采样周期通常以毫秒或秒计，有时也会是处理器周期或指令周期的整数倍。

该框架中的第三个基础模块是异常检测本身，由机器学习分类器执行。这些分类器经过输入数据训练后，能够自动将数据归入一个或多个类别（因为训练数据与测试数据具有相似的统计分布）。分类器在多个方面可能存在差异，包括可用标签（多分类和单分类）、学习方式（监督学习、无监督学习和半监督学习）、所实现的数学公式/算法（基于神经网络、基于线性模型、基于决策树、基于规则等），以及其排列方式。

分类器（多阶段）。由于每种分类器配置在不同指标（包括性能、效率和硬件设计开销）上会产生不同结果，因此设计机器学习分类器至关重要。

基于多类分类的异常检测假定，训练期间的数据包含标记为属于多个正常类别的实例。测试新实例时，如果训练得到的分类器都未将其分类为正常，则该实例会被视为异常。另一方面，基于单类分类的异常检测技术假定所有训练实例只有一个类别标签，任何落在学习边界之外的新实例都会被判定为异常 [30]。

监督学习指使用带标签样本训练的分类器，而无监督学习使用无标签样本。半监督学习在数据集 [42] 中结合带标签样本和无标签样本。然而，获取带标签数据可能成本高昂，尤其是在考虑异常行为时，这类行为往往具有动态性，难以准确标注。因此，无监督学习分类器更常用于异常检测 [30]。

最后，为了提高分类器的准确率，会使用更复杂的算法，例如集成学习技术 [43]。集成学习是机器学习的一个分支，它结合多个基学习器的结果来提高决策准确率。集成学习算法包括提升法（AdaBoost 实现 [44]）和装袋法（自助聚合 [45]）。多阶段分类器也可用于提高异常检测的准确率 [46]。

IV. 结论

本文概述了 Vitamin-V 项目。该项目由欧洲地平线计划资助，实施期为2023年至2025年。项目旨在为 RISC-V 打造一套完整的开源虚拟化软件栈，用于云服务。项目开展的各项活动中，主要目标之一是构建稳健的虚拟执行环境，可用于软件开发、确认、验证与测试，从而提高 RISC-V 云应用的可信度。感兴趣的读者可通过 https://www.vitamin-v.eu 了解项目的最新进展。

参考文献

[1] A. Waterman、Y. Lee 等，《RISC-V 指令集手册第二卷：特权架构 1.9 版》，加州大学伯克利分校 EECS 系，技术报告，2016 年。[2] S. Greengard，《RISC-V 会革新计算吗？》，《计算机协会通讯》，第 63 卷第 5 期，第 30–32 页，2020 年。[3] A. D¨orflinger、M. Albers 等，《开源应用级 RISC-V 处理器实现的比较性综述》，载于第 *18* 届 *ACM* 国际计算前沿会议论文集，CF ’21 系列。美国纽约州纽约市：计算机协会，2021 年，第 12–20 页。 [在线]。可访问：https: //doi.org/ 10 1145 3457388 3458657。 / 。B. W. Mezger、D. A. Santos 等，[4]《RISC-V 架构软件支持综述》，《IEEE Access》，第 10 卷，第 51 394–51 411 页，2022 年。[5] 欧盟委员会。欧洲处理器倡议。https://www. european-processor-initiative.eu/。访问日期：2023-03-17。

[6] M. Kovaˇc, 《欧洲处理器倡议：面向百亿亿次时代的 EuroHPC 工业基石》，载于第 16 届 ACM 国际计算前沿会议论文集，2019 年，第 319 页。[7] 软件自由保护协会。RISCV-QEMU。 [在线] https://www.qemu.org/。访问日期：2023-03-17。[8] 《gem5》，https://github.com/gem5/gem5，访问日期：2023-03-17。[9] 《Amazon F1》，https://aws.amazon.com/ec2/instance-types/f1/，访问日期：2023-03-17。[10] A. Brokalakis、N. Tampouratzis 等，《Cossim：解决系统之系统模拟器鸿沟的开源集成解决方案》，载于 2018 年第 21 届 Euromicro 数字系统设计会议（DSD），2018 年，第 115–120 页。[11] C. Lattner 和 V. Adve，《LLVM：用于终身程序分析与转换的编译框架》，美国加利福尼亚州圣何塞，2004 年 3 月，第 75–88 页。[12] F. Caforio、P. Iannicelli 等，《Vosysmonitorv：面向支持 Linux 的 RISC-V平台的混合关键性解决方案》，载于 2021 年第 10 届地中海嵌入式计算会议（MECO）。IEEE，2021 年，第 1–4 页。[13] N. Tampouratzis 和 I. Papaefstathiou，《一种用于包含定制硬件加速器的异构云系统的新型模拟器》，IEEE 多尺度计算系统汇刊，第 4 卷第 4 期，第 565–576 页，2018 年。[14] D. Marques、A. Ilic 等，《应用驱动的缓存感知屋顶线模型》，未来一代计算机系统，第 107 卷，273，2020 年 257 页。– 。[15]J. M. Domingos、P. Tomas 和 L. Sousa，《通过 Linux（perf）的性能分析工具支持 RISC-V性能计数器》，载于第 5 届 RISC-V计算机体系结构研究研讨会（CARRV ‘21），2021 年，第 935–948 页。 [在线]。可访问：https://arxiv.org/pdf/2112.11767.pdf[16] S. Browne、J. Dongarra 等，《使用硬件计数器进行应用性能调优的可扩展跨平台基础设施》，载于 SC ’00：2000 年 ACM/IEEE超级计算会议论文集，2000 年，第 42–42 页。[17] J. Lowe-Power、A. M. Ahmad 等，《gem5模拟器：20.0+版》，2020 年。[18] 《SemiDynamics 高带宽、支持向量的 RISC-V 核心》，https://semidynamics.com/uploads/semidynamics/pdf/2021.03.31-RISCV-WEEK.pdf，访问日期：2023-04-06。[19] B. Sprunt，《性能监控硬件基础》，IEEE Micro，第 22 卷第 4 期，第 64–71 页，2002 年。[20] C. Malone、M. Zahran 和 R. Karri，《硬件性能计数器是否是进行程序完整性检查的经济有效方式》，载于第六届 ACM 可扩展可信计算研讨会论文集，STC ’11 系列。美国纽约州纽约市：计算机协会，2011 年，第 71–76 页。 [在线]。可访问：https://doi.org/10 1145 2046582 2046596 。 / 。N. C. Doyle、E. Matthews 等，《硬件内存访问跟踪采集的性能影响与局限性》，载于欧洲设计、自动化与测试会议暨展览会（DATE），2017 年，2017 年，第 506–511 页。[22] Y. Ding、W. Dai 等，《基于控制流的操作码行为分析用于恶意软件检测》，计算机与安全，第 44 卷，第 65–74 页，2014 年。[23] H. HaddadPajouh、A. Dehghantanha 等，《基于深度循环神经网络的物联网恶意软件威胁搜寻方法》，未来一代计算机系统，第 85 卷，第 88–96 页，2018 年。[24] E. Rodr´ıguez、P. Valls 等，《基于迁移学习的入侵检测框架用于物联网网络》，第 22 卷第 15 期。MDPI，2022 年，第 5621 页。[25] P. Kocher、J. Horn 等，《Spectre攻击：利用推测执行》，载于第 40 届 IEEE安全与隐私研讨会（S&P’19），2019 年。[26] M. Lipp、M. Schwarz 等，《Meltdown：从用户空间读取内核内存》，载于第 27 届 USENIX安全研讨会（USENIX Security 18），2018 年。[27] 《VirusTotal》，https://www.virustotal.com/。[28] 《VirusShare》，https://www.virusshare.com/。[29] M. O. F. Rokon、R. Islam 等，《{SourceFinder}：从 {GitHub} 中公开可用的代码仓库查找恶意软件{源代码} 》，载于第 23 届攻击、入侵与防御研究国际研讨会（RAID 2020），2020 年，第 149–163 页。[30] V. Chandola、A. Banerjee 和 V. Kumar，《异常检测：综述》，ACM 计算评论，第 41 卷第 3 期，2009 年 7 月。 [在线]。可访问：https://doi.org/10.1145/1541880.1541882[31] J. Demme、M. Maycock 等，《基于性能计数器进行在线恶意软件检测的可行性》，SIGARCH Comput. Archit. News，第 41 卷第 3 期，第 559–570 页，2013年 6 月。 [在线]。可访问：https://doi.org/10 1145 2508148 2485970 。/。S. Dutto、A. Savino 和 S. Di Carlo，《探索深度学习在微处理器现场故障检测中的应用》，载于 2021年欧洲设计、自动化与测试会议暨展览会（DATE），2021年，第 1456–1459 页。[33] D. Kasap、A. Carpegna 等，《微架构特征作为嵌入式安全关键系统中软错误诱发故障执行的标记：初步研究》，2023年。[34] T. Sherwood、E. Perelman 等，《发现并利用程序阶段》，IEEE Micro，第 23 卷第 6 期，第 84–93 页，2003年。[35] C. Isci、G. Contreras 和 M. Martonosi，《真实系统上的实时运行时阶段监控与预测及其在动态电源管理中的应用》，载于 2006年 IEEE/ACM 第 39 届年度微架构国际研讨会（MICRO’06），2006年，第 359–370 页。[36] B. Singh、D. Evtyushkin 等，《使用硬件性能计数器检测内核级 rootkit》，载于 2017年 ACM 亚洲计算机与通信安全会议论文集，系列：ASIA CCS ’17。纽约，纽约州，美国：计算机协会，2017年，第 483–493 页。 [在线]。可访问：https://doi.org/10 1145 3052973 3052999 。/。H. Sayadi、N. Patel 等，《基于机器学习的复合核心架构能效预测与调度方法》，载于 2017年 IEEE 国际计算机设计会议（ICCD），2017年，第 129–136 页。[38] C. P. Chenet、A. Savino 和 S. D. Carlo，《基于硬件的恶意软件检测方法综述》，2023年。[39] R. O. Duda、P. E. Hart 和 D. G. Stork，《模式分类》（第 2 版）。美国：Wiley-Interscience，2000年。[40] K. Pearson，《关于双亲情况下回归与遗传的注释》，Proc. R. Soc. Lond.，第 58 卷，第 240–242 页，1895年。[41] H. Peng、F. Long 和 C. Ding，《基于最大依赖、最大相关和最小冗余互信息准则的特征选择》，IEEE Transactions on Pattern Analysis and Machine Intelligence，第 27 卷第 8 期，第 1226–1238 页，2005年。[42] A. Burkov，《百页机器学习书》。Andriy Burkov，2019年。 [在线]。可访问：https://books.google.it/books?id=0jbxwQEACAAJ[43] H. Sayadi、N. Patel 等，《用于有效运行时基于硬件的恶意软件检测的集成学习：综合分析与分类》，载于 2018年 ACM/ESDA/IEEE 设计自动化会议（DAC），2018年，第 1–6 页。[44] Y. Freund 和 R. E. Schapire，《在线学习的决策理论推广及其在提升法中的应用》，Journal of Computer and System Sciences，第 55 卷第 1 期，第 119–139 页，1997年。 [在线]。可访问：
htt
ps://www.sciencedirect.com/science/article/pii/S002200009791504X[45] L. Breiman，《装袋法预测器》，Machine Learning，第 24 卷第 - 期，第 123–140 页，1996年。[46] H. Sayadi、H. M. Makrani 等，《2smart：一种用于运行时专用硬件辅助恶意软件检测的两阶段机器学习方法》，载于 2019年欧洲设计、自动化与测试会议暨展览会（DATE），2019年，第 728–733 页。