软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2023,34(6):2959−2978 [doi: 10.13328/j.cnki.jos.006507]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于无干扰理论的虚拟机可信启动研究*

黄浩翔[1,2], 张建标[1,2], 袁艺林[1,2], 王 晓[3]

[1](北京工业大学 信息学部, 北京 100124)

[2](可信计算北京市重点实验室, 北京 100124)

[3](天津财经大学 理工学院, 天津 300222)

通信作者: 张建标, E-mail: zjb@bjut.edu.cn

摘 要: 云计算作为一种新型高价值计算系统, 目前被广泛应用于各行业领域; 等保 2.0 中也提出了对其应用主动免疫可信计算技术进行动态可信验证的要求. 云计算模式下, 虚拟机作为用户使用云服务的直接载体, 其可信启动是虚拟机运行环境可信的基础. 但由于虚拟机以进程的形式运行在物理节点上, 其启动过程呈现出高动态性, 且多虚拟机域间存在非预期干扰等特点; 而现有的虚拟机可信启动方案存在虚拟机启动过程的动态防护性不足、缺乏多虚拟域间非预期干扰性排除等问题. 针对上述问题, 提出一种基于无干扰理论的虚拟机可信启动研究方案. 首先, 基于无干扰理论, 提出了虚拟机进程的运行时可信定理; 进一步地, 给出了虚拟机可信启动的定义并证明了虚拟机可信启动判定定理. 其次, 依据虚拟机可信启动判定定理, 基于系统调用设计监测控制逻辑, 对虚拟机启动过程进行主动动态度量与主动控制. 实验结果表明所提方案能够有效排除复杂云环境下多虚拟机间非预期干扰, 保证虚拟机启动过程的动态可信性, 且性能开销较小.

关键词: 无干扰理论; 虚拟机进程; 可信启动; 动态可信; 主动度量; 主动控制

中图法分类号: TP316

中文引用格式: 黄浩翔, 张建标, 袁艺林, 王晓. 基于无干扰理论的虚拟机可信启动研究. 软件学报, 2023, 34(6): 2959–2978. http://www.jos.org.cn/1000-9825/6507.htm

英文引用格式: Huang HX, Zhang JB, Yuan YL, Wang X. Research on Trusted Startup of Virtual Machine Based on Non-interference Theory. Ruan Jian Xue Bao/Journal of Software, 2023, 34(6): 2959–2978 (in Chinese). http://www.jos.org.cn/1000-9825/6507.htm

Research on Trusted Startup of Virtual Machine Based on Non-interference Theory

HUANG Hao-Xiang[1,2], ZHANG Jian-Biao[1,2], YUAN Yi-Lin[1,2], WANG Xiao[3]

[1](Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China)

[2](Beijing Key Laboratory of Tusted Computing, Beijing 100124, China)

[3](School of Science and Technology, Tianjin University of Finance and Economics, Tianjin 300222, China)

Abstract: As a new type of high-value computing system, cloud computing has been widely used in various industries fields. Classified protection 2.0 also puts forward the requirement of dynamic trust verification for its application of active immune trusted computing technology. In the cloud computing mode, the virtual machine is the direct carrier for users to use cloud services, and its trusted startup is the basis for the trustworthiness of the virtual machine operating environment. However, since the virtual machine runs on the physical node in the form of process, its characteristics of startup process are high dynamic and unexpected interference between multiple virtual machine domains. But the existing trusted startup schemes of virtual machine have problems such as insufficient dynamic protection during virtual machine startup process and lack of elimination of unexpected interference between multiple virtual domains. To solve the above problems, this study proposes a scheme that research on trusted startup of virtual machine based on non-interference theory. Firstly, based on the non-interference theory, the run-time trusted theorem of virtual machine process is proposed. In addition, the definition of trusted

* 基金项目: 北京市自然科学基金 (M21039); 国防科研试验信息安全实验室基础研究项目 (2017XXAQ09)
收稿时间: 2021-03-18; 修改时间: 2021-06-07, 2021-08-20, 2021-10-09; 采用时间: 2021-10-12; jos 在线出版时间: 2022-11-30
CNKI 网络首发时间: 2022-12-01

launch of virtual machine is given and the judgement theorem of trusted boot of virtual machine is well proved. Then, according to the trusted startup theorem of virtual machine, the monitoring and control logic is designed based on system call, and the virtual machine startup process is actively measured and controlled. Finally, the experimental evaluation shows that the proposed scheme can effectively eliminate the unexpected interference between multiple virtual machines in complex cloud environment, ensure the dynamic credibility of virtual machine startup process, and greatly reduce the performance overhead.

Key words: non-interference theory; virtual machine process; trusted startup; dynamic trusted; active measurement; active control

2020 年“数智时代”的来临使得云计算这种高价值计算系统[1]的应用与需求场景变得更加广泛与深入, 为了应对云计算技术在社会各领域深层次的下沉与应用带来的安全风险, 国家在等保 2.0[2]中提出了对包括云计算在内的重要信息系统都要进行可信验证的要求. 云计算模式下, 由于用户对于托管在云端的私人数据缺乏有效的直接控制能力[3,4], 且云服务提供商拥有云环境中数据优先访问权[5], 导致用户与云服务提供商在云环境是否可信的问题上难以达成共识. 虚拟机作为用户使用云计算服务的直接载体[6], 其运行环境的可信是云平台获取用户信任的关键. 基于可信计算技术, 将可信启动技术应用到云环境中构建虚拟机可信运行环境[7], 从而促使云计算技术向更深层次领域发展与应用, 已经成为社会各界的共识.

针对虚拟机可信启动, 研究人员多从虚拟机安全[8,9]、节点可信执行环境构建[10]、虚拟机完整性[11]等角度提及, 缺少专门针对虚拟机可信启动的研究. 且涉及虚拟机可信启动的相关研究方案多采用基于可信计算组织 TCG (trusted computing group)[12]所定义的装载前度量的方式, 通过将物理计算节点可信启动“逐级度量, 逐级信任”的链式信任传递模式应用到虚拟环境中来解决虚拟机的可信启动问题, 面临着如下挑战.

(1) 虚拟机启动过程的特殊性. 云环境中, qemu-kvm 模式下, KVM (kernel-based virtual machine) 虚拟机在物理节点中是以 Linux 常规进程的形式存在的[13], 其利用软件模拟来实现的 BIOS (baisc input output syatem)、BootLoader 等过程属于 qemu-kvm 进程运行过程的一部分.装载前度量模式“先度量, 后执行”的机制更侧重于通过对系统相关静态组件被加载执行前的完整性度量来保证可信性, 并不能保证 qemu-kvm 进程生命周期内的运行时可信.

(2) 静态对象表征动态可信的局限性. 虚拟机启动是 qemu-kvm 进程运行的动态过程, 其可信体现为 qemu-kvm 进程运行过程中的行为符合预期. qemu-kvm 进程模拟虚拟机启动过程, 加载的主要静态对象包括: SeaBIOS 文件、虚拟机增量镜像文件等, 由于仅针对静态对象 (如虚拟 BIOS、镜像文件等) 进行装载前的可信度量, 未考虑动态行为, 得到的可信结果不能有效表征虚拟机进程的动态可信, 也不符合“实体行为符合预期”的可信本质.

(3) 缺乏对云环境中多虚拟域间非预期干扰排除的考量. 云计算模式下, 同一物理节点存在有多个虚拟节点启动或运行. 虚拟机启动过程容易受到同一物理节点环境下其他虚拟节点的非预期 (恶意) 干扰.

为应对上述挑战, 从保证 KVM 虚拟机进程运行时可信的角度出发, 本文基于无干扰理论, 提出了一种虚拟机可信启动研究方案, 主要特点如下.

(1) 无干扰理论通过清除对安全域非预期的干扰来保证系统行为符合预期, 和“行为符合预期”的实体可信定义在数学上的等价性, 使得保证虚拟机启动过程的动态可信成为可能. 因此, 基于无干扰理论, 提出了虚拟机进程的运行时可信定理, 进一步地, 给出了虚拟机可信的定义并证明了虚拟机可信启动判定定理.

(2) 本文依据虚拟机可信启动判定定理, 从系统调用层面入手, 一方面通过建立符合进程可信运行行为的系统调用行为白名单, 防止异常的系统调用行为执行; 另一方面对虚拟机启动过程加载的关键客体数据进行可信度量, 保证被加载数据的完整性. 综合考虑动态行为和静态对象, 有效阻止了恶意进程的非预期干扰, 共同保证了多虚拟机进程间的隔离性, 能够有效解决虚拟机启动过程的不可信问题.

本文第 1 节对可信的概念及应用、无干扰理论、可信启动的相关技术研究及可信云的相关研究进行了简单介绍. 第 2 节介绍了本文工作的理论基础及相关基本符号及定义. 第 3 节基于无干扰理论和虚拟机进程运行时可信定理, 提出了虚拟机可信启动研究方案, 并对相关判定定理进行了证明. 第 4 节对虚拟机可信启动研究方案原型系统的基本思想做了介绍, 并给出了原型系统的总体框架设计. 第 5 节对原型系统相关功能进行了实验与分析. 第 6 节对本文工作进行简单的总结与展望.

1 相关技术研究

1.1 可信的概念及应用

可信计算将人类社会中较为成熟完善的成功管理经验引入到计算系统平台中, 其以密码学为基础, 以可信芯片为信任源头, 通过在计算系统平台中构建一个作为平台信任起点的信任根, 并借助于信任链将信任关系从底层硬件平台逐步扩展至上层应用, 从而保证整个计算环境从下而上的可信[14−16], 它的提出为解决计算平台的安全问题提供了一种新思路. 目前, 主流的可信定义有以下几种.

(1) TCG 用实体行为的预期性来定义可信, 即如果实体的行为总是以预期的方式, 实现预期的目标, 则该实体是可信的[17].

(2) 沈昌祥院士结合人类免疫系统的理念提出了可信计算新的理论阐述[18], 即计算全程可测可控、不被干扰, 是一种以密码为基因抗体, 具有身份识别、状态度量、保密存储等功能, 运算和防护并存的主动免疫的新计算模式. 其特点是计算运算的同时开启对系统本身的安全防护, 为网络系统培育免疫能力, 使操作和过程行为在任意条件下的计算结果总是与预期一致.

可信密码模块 TCM (trusted cryptography module) 是一个具备密码引擎、随机数发生器和物理安全计算环境的可信平台基础支撑部件[14,19], 是可信思想在工程领域的具体实现. TCM 采用椭圆曲线密码算法和对称密码算法为基础[20], 为计算平台可信环境的建立提供密码技术上的支撑, 其结构如图 1 所示.

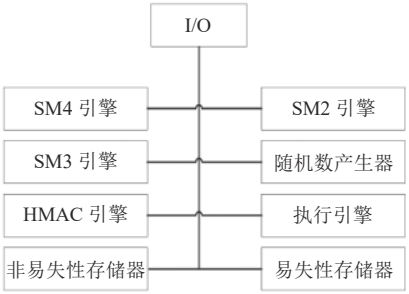

图 1 可信密码模块结构

1.2 无干扰理论研究及应用

1982 年 Goguen 等人[21]最早以状态机的形式提出了基于信息流的无干扰思想, 其基本思想是: 一组用户 A, 使用一组命令集合α操作之后, 如果对另一组用户 B 所能观察到的结果没有影响, 则称用户组 A 对用户组 B 是无干扰的. Rushby[22]为无干扰理论建立了标准的数学模型并给出了策略安全的约束条件, 该模型使得从动作符合预期的角度去研究可信成为了可能; 但是该模型仅从状态变迁会导致状态差异的角度来观察系统, 并未对观察者是否能够同时观察到所执行的不同动作的差异进行研究. 随后 van der Meyden[23]总结 Rushby 模型的不足之处, 并重新定义了安全约束更强的 TA-Security 和 TO-Security. Eggert 等人[24]对传递无干扰、非传递无干扰、TA-Security、TO-Security 这 4 种无干扰属性的时间和空间复杂度进行了推演. 随后, 针对 Eggert 并未给出相关属性验证算法这一问题, 张帆等人[25]基于无干扰理论对云环境中的动作行为进行了可信性分析, 提出了基于状态递归等价的行为可信的充要条件, 解决了目前尚无有效的行为可信性验证方法的问题.

在国内, 无干扰理论被广泛应用到可信计算领域, 用于解决可信测评[26]、信任链可信[27]、进程可信[28]和终端隔离等方面的实际问题[29]. 在可信防护领域, 基于无干扰理论的研究工作多是理论推导及安全性证明, 面临着难以实践验证的难题. 张兴等人[30]基于无干扰理论模型提出了一种分析和判定可信计算平台信任链传递的方法, 即只有组件间信息的流动符合严格的非传递无干扰安全策略的时候, 信任链才是有效的. 但是该方案无条件信任虚拟机监视器, 安全隔离依赖于虚拟机监视器, 具有不确定性; 同时该方案将虚拟机作为一个整体进行校验, 存在着监控较为粗粒度、缺乏动态性等问题. 赵佳等人[31]从动态的角度建立了基于无干扰理论的可信链模型, 并通过内

核执行时前一进程校验后一进程的静态完整性度量方式来实现 Linux 内核的可信引导. 但是, Linux 内核引导过程是一个复杂的过程, 相关进程存在并行执行的情况, 因此该方案存在安全性不足的问题. 张兴等人[28]借鉴信息流的基本无干扰理论对进程运行的可信进行了形式化定义, 给出并证明了系统运行的可信判定定理. 其是国内将无干扰理论应用到可信计算领域的早期代表性研究方案, 但是该方案仅进行了理论推导, 缺乏实验验证. 陈亮等人[32]从进程的角度提出了一种基于无干扰理论的信任传递模型, 该模型将系统状态的变化视为执行动作、发出动作的进程共同作用的结果, 从系统行为的层面对系统运行时的可信进行验证. 但是, 该方案仅进行了形式化定义及安全性证明, 缺乏工程实践验证. 张帆等人[29]通过无干扰模型来判断软件真实行为与预期行为间的一致性, 实现了软件运行过程中的实时可信度量与主动防护, 是无干扰理论应用到可信防护领域的一次工程实践尝试. Zhang 等人[33]通过分析云平台存在的威胁, 基于无干扰理论提出了一种可信域层次模型, 给出了云用户可信域可信运行的约束条件及可信行为的决策定理, 实现了云租户工作负载的完整性和安全性保护, 有效阻止了租户间的非预期干扰.

相比之下, 本文所提出的基于无干扰理论的虚拟机可信启动研究方案, 将虚拟机的启动过程视为 qemu-kvm 进程运行的动态过程. 通过无干扰理论建立数学模型, 提出并证明了虚拟机可信启动的判定定理, 基于系统调用刻画虚拟机进程运行过程的预期行为, 具备动态性、细粒度性, 也更符合可信定义的本质. 同时, 实验证明方案能够有效对虚拟机启动过程进行动态监控, 保证虚拟机启动过程的动态可信性, 且具备较小的性能损耗.

因此, 综合国内外主流的可信定义[17,18]及国内应用无干扰解决可信计算领域问题的实践[28−32], 应用无干扰理论模型去刻画行为符合预期实体可信是极具研究价值的.

1.3 可信启动相关研究

随着云计算技术在社会各行业领域的深层次下沉, 虚拟机计算环境及运行数据的安全性、可用性愈发受到用户关注, 利用可信启动技术为虚拟机构建信任基础[34,35], 成为了研究虚拟机可信的热门领域. 可信启动是可信计算技术中用于保证计算节点可信的技术之一[36], 其借用可信计算组织 TCG“信任链”和“可信度量”的理念, 从一个初始的“信任根”出发, 采用装载前度量的方式将源于物理可信根 TPM (trusted platform module) 的信任关系扩展到整个计算平台 (其过程如图 2 所示), 被广泛应用于多种主流操作系统中, 如 Linux 平台的 IMA (integrity measurement architecture) 架构[32].

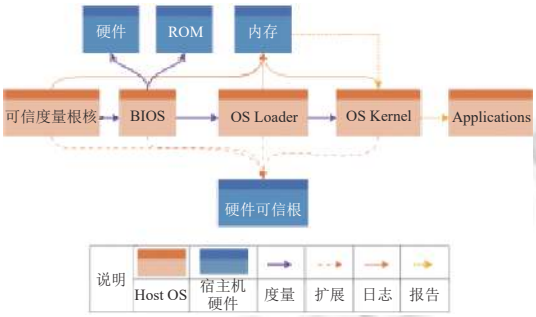

图 2 TCG 信任度量机制

近年来, 针对虚拟机可信启动的相关研究多采用静态的装载前度量方式来将源于信任根的信任关系通过可信度量转移到虚拟机内部, 可信转移过程如图 3 所示. 例如, Srivastava 等人[8]采用基于 TPM 对用户虚拟机运行状态信息进行静态度量方式来保证虚拟机的可信启动; 王庆飞等人[9]从虚拟机镜像的存储加解密及虚拟机镜像关键组件的静态可信度量两个方面来进行研究, 保证 IaaS 云虚拟机启动过程的可信; Jin 等人[37]为了将源于硬件 TPM 的信任链传递到云虚拟机中, 提出了一种利用虚拟机监视器度量虚拟机 BIOS 及 OS Loader 的方案, 保证虚拟机的可信启动, 从而建立一个基于硬件可信根的初始信任. 尽管上述方案保证虚拟机可信启动的方式有所不同, 但本质上都是采用“先度量, 后执行”的装载前度量的方式将源于可信根的信任关系逐级扩展到整个虚拟计算平台. 在早期云平台相对轻量级且攻击手段较为单一时, 装载前的静态度量方式不失为解决虚拟机启动过程可信的一种有效方

案, 但对于云虚拟机启动过程的高动态性而言, 静态度量方式仍然存在易受 TOC-TOU (time of check-time of use) 攻击、缺乏实时性等问题.

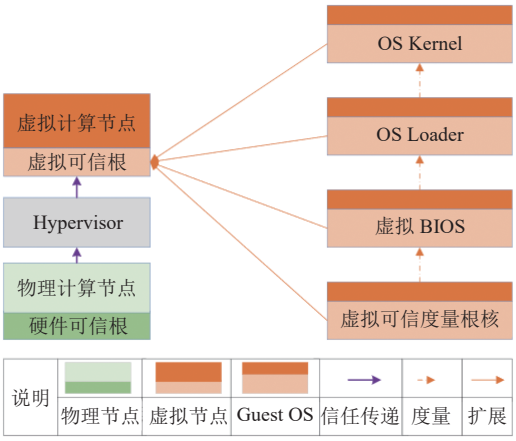

图 3 虚拟节点可信转移过程

随着云计算技术的成熟, 云平台逐渐朝着复杂化、高动态方向发展, 由于虚拟节点是以进程的形式存在于物理节点, 其启动过程的相关阶段是利用软件仿真来实现, 一些学者开始注意到虚拟机启动过程面临的动态可信问题. 例如, 刘川意等人[10]从保证虚拟机执行环境可信的角度提出了一种将可信启动和可信审计技术结合起来的用户可信运行环境构建与实时审计方案, 但针对虚拟机启动相关阶段采用周期性度量的方式进行可信度量, 仍存在动态性不足的问题; 同时所选取的被度量对象为静态代码, 存在有被度量对象特征过于简单, 无法有效表征动态可信性. 林杰等人[38]提出了一种基于虚拟机自省的完整性度量方案, 通过采用组件独立度量的方式进行动态可信度量, 但度量触发机制为周期性触发, 仍然面临着动态性不足的问题. 上述方案均未从虚拟机启动过程实体行为的角度研究虚拟机动态可信度量, 不符合“实体行为符合预期”的可信本质.

云是一种多域虚拟化环境, 各虚拟域间存在着干扰性问题, 且虚拟机域的状态受虚拟机行为的影响. 如周振吉等人[39]针对云计算虚拟环境多域并发的特点, 提出了一种树状可信度量模型. 该方案将管理域和用户域间的信任关系分离, 通过管理域的完整性度量及用户域系统调用行为的可信度量来共同保证用户域启动及运行过程的可信性, 但该方案未考虑多虚拟机域间的相互干扰性给虚拟机的启动带来非预期的干扰性问题.

1.4 可信云相关研究

随着云计算在社会各行业领域的深层次下沉与应用, 云计算平台自身的安全问题愈发突出, 将可信计算技术应用到云计算环境中构建可信云环境[7], 是解决云安全问题的一种新思路.

在多虚拟计算节点的云环境中, 将物理可信根虚拟化从而为云虚拟计算节点提供可信锚点是构建可信云环境的重要基础. 可信根虚拟化最早源于 2006 年 IBM 基于 Xen 架构所提出的 TPM 虚拟化方案[40], 但该方案存在两个重大缺点: 首先, 其充当云平台虚拟计算节点可信支撑的 vTPM 只是一个运行在特权虚拟机 Dom0 中的纯软件应用, 并没有与底层的硬件 TPM 建立起信任关系; 其次, 该方案中特权虚拟机 Dom0 若是被攻击, 则运行在特权虚拟机中的 vTPM 就会失去其可信性. 从而导致虚拟机的不可信. 文献 [41,42] 利用隔离的思想将 vTPM 实例运行在安全的隔离域中, 提高了虚拟根系统的安全性.

在云环境中, 虚拟可信根作为云虚拟计算节点可信源点, 其纯软应用的本质使得自身的安全性无法得到有效保障.因此基于虚拟机自省 VMI (virtual machine introspection)[43]技术使用安全虚拟机来监控云环境中目标虚拟计算节点的方案被提出.文献 [44,45] 等基于安全虚拟机对运行中的系统进行动态防护, 利用物理内存虚拟化方法实时拦截目标虚拟机系统异常行为, 但面临着频繁陷入虚拟机管理层导致性能损耗较大的难题. Du 等人[46]提出了 VMI 和事件捕获技术的动态完整性度量模型 DIMM, 该模型基于虚拟可信根 vTPM 提出了通过自修改动态测量

策略, 降低了可信度量带来的性能开销.

对于宿主机而言, 云虚拟计算节点与 Linux 普通进程没有区别[47], 其启动过程具有高动态性, 仅针对虚拟计算节点启动不同阶段进行完整性校验无法防止基于控制流劫持的代码重用攻击[48]. 文献 [49] 为应用程序提出了一个轻量级的控制流完整性保护方案 TZMCFI, 能够防止应用在运行过程中受到如 ROP[50]等为代表的控制流劫持攻击. 文献 [51] 指出从虚拟节点级别去监视系统调用行为足以检测到针对虚拟节点的大部分攻击, 提出了一种 IaaS 环境下 KVM 虚拟机入侵检测方法, 在基于 Linux 环境下通过实验和统计分析证明了该方案的可行性和有效性.

2 工作基础

2.1 理论场景

本文以一个小型的 OpenStack[52]私有云环境作为场景示例, 包括 1 个控制节点、1 个计算节点, 且控制节点和计算节点都是具备物理可信根的可信节点, 架构如图 4 所示.

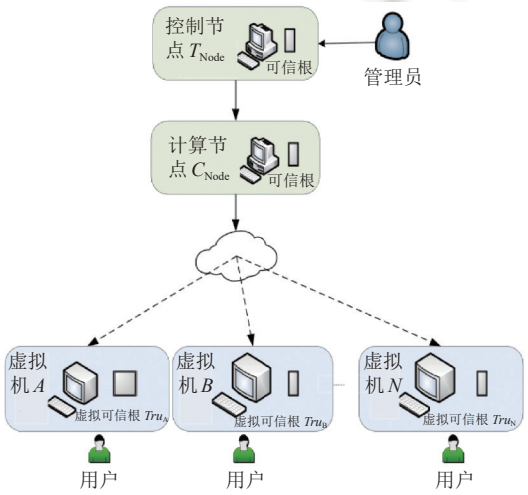

图 4 私有云场景

从 Linux 2.6.20 开始, KVM 被集成到标准 Linux 内核中, 负责 CPU 和内存的虚拟化, 使得 Linux 内核成为一个轻量级的虚拟机监视器. 在用户空间, KVM 与修改后的 QEMU 协同工作, 用于虚拟机的创建、管理和必要设备的模拟[47]. 在宿主机操作系统看来, 虚拟机与其使用的虚拟 CPU 都以常规 Linux 进程的形式存在, 且由 Linux 内核的调度程序进行管理[13]. 值得指出的是, 若无特殊说明, 文中所述虚拟机均指 qemu-kvm 模式下创建的虚拟机, 在文中统一简称为 KVM 虚拟机.

因此, 在图 4 所示场景中, 可信的计算节点 C_{Node} 在可信控制节点 T_{Node} 的控制下创建并运行多个 KVM 虚拟机, 且每个 KVM 虚拟机在可信计算节点 C_{Node} 中都是以 qemu-kvm 进程的形式存在. 其中 KVM 虚拟机 A 运行在可信计算节点中, 且该虚拟机拥有与之唯一对应且绑定的虚拟可信根 Tru_A.

基于上述场景, 虚拟机 A 在可信计算节点 C_{Node} 中是以 qemu-kvm[52]进程的形式存在的, qemu-kvm 进程能够完成虚拟计算节点 A 启动阶段的硬件仿真模拟. 与普通 Linux 进程不同的是, qemu-kvm 进程利用 ioctl 系统调用通过字符设备/dev/kvm 调用底层 KVM 模块完成 CPU 和内存的虚拟化, 支持虚拟机运行. 映射给虚拟机的内存实际上是映射到 qemu-kvm 进程的虚拟内存, 通过对 qemu-kvm 进程进行监控, 能够有效发现虚拟机启动过程的恶意攻击. 对于 qemu-kvm 进程而言, 其运行时可信应从两方面考虑, 即动态行为的可信和静态客体文件的可信. 其中, 动态行为表现为系统调用的执行, 对虚拟机进程运行过程中系统调用行为的监控能够有效排除其他进程对其的恶意干扰. 因此, 从虚拟机进程的角度去保证虚拟机的启动过程满足“实体行为”符合“预期”的可信定义是极具可行性的.

云计算使用虚拟化技术实现对底层计算、网络和存储资源的封装, 并以虚拟机的形式提供给远程, 是一种多域虚拟化环境. 多虚拟机进程同时启动或运行的情况下, 如何保障目标虚拟机启动过程不受其他虚拟机进程的非预期干扰是本文关注的重点之一. 基于信息流的无干扰理论通过清除动作序列中来自其他进程的非预期或恶意干扰因素, 从保证系统运行结果符合预期的角度建立安全策略模型, 和"实体行为符合预期"的可信定义在数学上是等价的. 因此, 结合无干扰理论对虚拟机进程运行过程行为是否符合预期进行研究是具备理论优越性及可行性的.

2.2 基本符号定义

本文基于系统"行为"在运行过程中总是符合"预期"且提供可信赖的服务的可信定义, 沿用文献 [22] 的部分符号定义将虚拟机启动系统形式化描述为进程、动作及状态输出的有限集合.

定义 1. 系统 $M=(S,P,A,Q,O,F)$, 包含如下元素.

(1) 系统状态集 $S=(s_0,s_1,\ldots,s_n)$, 其中 s_0 为初始状态.

(2) 系统进程集 $P=(p_0,p_1,\ldots,p_m)$, 其中单个进程使用 p 表示.

(3) 进程客体对象集 $Sub=(sub_0,sub_1,\ldots,sub_s)$.

(4) 动作行为集 $A=(a_0,a_1,\ \ldots,\ a_k)$.

(5) 动作行为序列集 $Q=(\alpha_0,\alpha_1,\ldots,\alpha_l)$, 其中 α_i $(1\leqslant i\leqslant l)$ 为动作行为集 A 中多个元素的集合, 可以表示为: $\alpha_{i(1\leqslant i\leqslant l)}=a_0\circ a_1\circ\ldots\circ a_i$; 其中, 动作或动作序列为空时, 使用 $\wedge$ 表示.

(6) 进程运行输出结果集 O.

函数集合 F:

(7) 动作-进程映射函数 $dom:A\to P$, 返回一个原子动作 $a\in A$ 所属的进程 $dom(a)$;

(8) 单步函数 $step:S\times A\to S$, 系统 M 的某一状态 S_i 经过执行单个原子动作 $a_i\in A$ 后所达到的新状态 S_{i+1}.

(9) 多步运行函数 $process:S\times Q\to S$, 表示系统 M 从某一状态经过一个动作序列 α 后所达到的新的状态.

(10) 观测函数 $observe:A\to Sub$ 表示在系统状态 $s_i\in S$ 下, 所观察到的当前进程执行动作 $a_i\in A$ 时所加载的客体对象 $sub_i\in Sub$.

(11) 执行结果函数 $execute:S\times Sub\to O$; 表示系统在某状态 $s_i\in S$ 下, 进程执行动作 $a_i\in A$ 所调用的客体对象 $sub_i\in Sub$ 后产生的结果 $o\in O$, 也是发出动作 $a_i\in A$ 的进程 p 对此时系统状态的观测结果.

关系集合 R:

(12) 干扰关系 ~>: 进程 p 在运行的过程中, 如果受到进程 q 发出的动作的影响, 则认为进程 q 对进程 p 具有干扰性, 记为 $q\rightsquigarrow p$.

(13) 无干扰关系 $\not\rightsquigarrow$: 和干扰关系互为补集, $q\not\rightsquigarrow p$ 表示进程 q 对进程 p 无干扰性.

(14) 进程观察等价关系: $s\overset{p}{=}t$ 表示从进程 p 的角度观察系统状态 s 和状态 t 是等价的.

定义 2. 清除函数 $pure:Q\times P\to P$: 对于 $\forall p\in P$ 和 $\alpha\in Q$, $pure(\alpha,p)$ 表示将所有对进程 p 有正常干扰关系的进程发出的动作保留, 从动作序列 α 中将对进程 p 无干扰关系的进程产生的动作删除, 表示如下:

$$pure(\wedge,p)=\wedge \tag{1}$$

$$pure(a\circ\alpha,p)=\begin{cases}a\circ pure(\alpha,p), & \text{if } dom(a)\rightsquigarrow p\\ pure(\alpha,p), & \text{otherwise}\end{cases} \tag{2}$$

定义 2 的清除函数目的是对一个进程运行过程中的原始动作序列进行简化, 将原始动作序列中对该进程没有干扰的动作删除, 只保留能够使该进程状态发生预期跃迁的干扰动作.

3 虚拟机可信启动研究方案

3.1 基于无干扰理论的虚拟机进程可信判定

定义 3. 根据第 1.1 节中操作可预测、结果符合预期的可信定义, 满足下式的进程 $p=dom(a)$ 是运行可信的:

$$execute(process(s_0,\alpha),observe(a)) = execute(process(s_0,pure(\alpha,dom(a))),observe(a)) \tag{3}$$

在进程观察等价的条件下, 把公式 (3) 的左右两边分别视为等价自动机 EMA (equivalent machine's automaton) 和预期等价自动机 EEMA (expect equivalent machine's automaton). EMA 表示进程 p 从初始可信状态 s_0 经过动作序列 α 达到实际状态 $process(s_0,\alpha)$ 后, 利用测试动作 $a \in A$ 从进程 P 的角度观察到的系统真实输出结果. EEMA 则代表一个相对严格控制信息流动的安全策略的控制下, 排除掉动作序列中对进程没有干扰的动作而得到的预期执行结果 (该结果在策略控制下的理论预期执行结果, 是可信的). 若等式成立, 表明从进程 p 的角度对系统进行观察, 无论针对进程的攻击方式如何多变 ($\forall a$), 进程的实际执行状态总是与预期状态保持一致, 说明进程 $p = dom(a)$ 没有受到潜在的、非预期的干扰, 即进程 p 是可信的.

定理 1 (进程可信运行定理). 如果一个进程 $dom(a)$ 满足以下性质:

(1) 输出隔离性, $s \overset{dom(a)}{=} t \Rightarrow execute(s,observe(a)) = execute(t,observe(a))$;

(2) 单步隔离性, $s \overset{dom(a)}{=} t \Rightarrow step(s,a) \overset{dom(a)}{=} step(t,a)$;

(3) 局部无干扰性, $dom(a) \nrightarrow p \Rightarrow s \overset{p}{=} step(s,a)$,

则进程 $dom(a)$ 是运行时可信的.

证明: 首先假设进程 p 满足上述 3 个性质, 要证明进程 p 可信, 则进程 p 必须满足定义 3 中公式 (3). 因为进程 p 满足输出隔离性, 所以只要证明进程 p 在具有观察等价性质的系统内满足公式 (4) 即可.

$$s \overset{p}{=} t \Rightarrow process(s,\alpha) \overset{p}{=} process(t,pure(\alpha,p)) \tag{4}$$

(1) 当 α 为 $\wedge$ 时, 式 (4) 成立.

(2) 当 α 不为 $\wedge$ 时, 采用假设归纳法进行证明.

首先, 假设动作序列 α 长度为 n 时, 公式 (4) 成立. 那么当动作序列长度为 (n+1) 时, 设动作序列 $\alpha' = a \circ \alpha$.

对于公式 (4) 左边:

$$process(s,\alpha') = process(s,a \circ \alpha) = process(step(s,a),\alpha) \tag{5}$$

对于公式 (4) 右边:

$$process(t,pure(\alpha',p)) = process(t,pure(a \circ \alpha,p)) \tag{6}$$

对于公式 (6) 分两种情况进行讨论.

1) $dom(a) \rightsquigarrow p$

公式 (6) 转换为公式 (7) 结果:

$$process(t,a \circ pure(\alpha,p)) = process(step(t,a),pure(\alpha,p)) \tag{7}$$

又因为进程 p 满足单步隔离性, 所以 $step(s,a) \overset{p}{=} step(t,a)$. 因此, 在关于进程 p 的观察等价条件下, 根据归纳假设总有公式 (8) 成立:

$$process(step(s,a),\alpha) \overset{p}{=} process(step(t,a),pure(\alpha,p)) \tag{8}$$

综合公式 (5), 公式 (7) 和公式 (8), 由假设条件可得, $process(s,a \circ \alpha) \overset{p}{=} process(t,pure(a \circ \alpha,p))$ 成立.

2) $dom(a) \nrightarrow p$

公式 (6) 转换为公式 (9) 结果:

$$process(t,pure(a \circ \alpha,p)) = process(t,pure(\alpha,p)) \tag{9}$$

又因为进程 p 满足局部无干扰性, 所以 $s \overset{p}{=} step(s,a)$, 又因为 $s \overset{p}{=} t$, 所以 $step(s,a) \overset{p}{=} t$ 成立. 因此, 在关于进程 p 的观察等价条件下, 根据归纳假设总有公式 (10) 成立:

$$process(step(s,a),\alpha) \overset{p}{=} process(t,pure(\alpha,p)) \tag{10}$$

综合公式 (5), 公式 (9) 和公式 (10), 由假设条件可得, $process(s,a \circ \alpha) \overset{p}{=} process(t,pure(a \circ \alpha,p))$ 成立.

综合 (1) 和 (2), 对任意序列长度, 公式 (6) 均成立. 证毕.

定理 1 的本质是: 对于满足输出隔离性、单步隔离性、局部无干扰性的进程 p 而言, 其进程运行过程中的实际行为和预期行为在同步执行任意动作或动作序列后总是能够保证状态等价性, 体现了进程运行过程中的动态可

信性. 其中, 隔离性表示进程的状态只与预期的动作相关.

3.2 虚拟机可信启动判定定理

定义 4. 当虚拟机启动过程满足以下条件时:

(1) 系统的初始状态 s_0 是可信的;

(2) 虚拟机进程运行时可信,

则称虚拟机启动过程是可信的.

定义 5 (可预期状态). KVM 虚拟机中, 虚拟机进程从初始可信状态 s_0 执行预期的系统调用行为序列后所达到的状态为虚拟机启动过程的可预期状态.

定理 2 (状态可预期判定定理). 从虚拟机进程 p 的角度观察, 对任意虚拟机状态 $s_i=(p,\alpha_i)$, 其中 $s_i\in S$, $\alpha_i\in Q$; 若 α_i 对公式 (11) 恒成立:

$$process(s_0,\alpha_i)\overset{p}{=}process(s_0,pure(\alpha_i,p)) \tag{11}$$

则认为状态 s_i 是可预期的.

证明: (反证法) 假设 $s_i=(p,\alpha_i)$ 是不可预期的, 则对于进程 p 而言, 其从 s_0 到 $s_i\in S$ 的状态跃迁过程中必定受到了非预期的干扰. 即 $\exists a\in A$ 是动作序列 α_i 的单个动作, 使得 $dom(a)$ 对 p 产生非预期的干扰. 那么对于 $\alpha_i=\alpha_1'\circ a$ 而言, 显然 $pure(\alpha_i,p)=a\circ pure(\alpha_i',p)$, 所以 $process(s_0,pure(\alpha_i,p))=process(step(s_0,a),pure(\alpha_i',p))$ 一定成立. 而在 $s_i=(p,\alpha_i)$ 是符合预期的情况下 $process(s_0,pure(\alpha_i,p))=process(pure(\alpha_i',p))$. 显然, $process(pure(\alpha_i',p))\neq process(step(s_0,a),pure(\alpha_i',p))$.

定义 6. 进程状态跃迁是通过系统调用行为来体现的, 因此对于任意可预期状态 $s_i\in S$ 执行符合预期的单个系统调用行为 $a_i\in A$ 会到达下一个可预期状态 $s_{i+1}\in S$.

定义 7. 验证函数: $verify: S\times S\to\{\text{true},\text{false}\}$. 若 $s_i\in S$ 满足 $process(s_0,\alpha_i)\overset{p}{=}process(s_0,pure(\alpha_i,p))$, 且对于 $s_{i+1}\in S=step(s_i,a_i)$ 而言, a_i 符合预期且 $observe(a_i)$ 完整性未受到破坏, 则认为状态 s_i 和 s_{i+1} 间的转移是可信的, 即 $verify(s_i,s_{i+1})=\text{true}$.

定理 3 (虚拟机可信启动判定定理). 当虚拟机启动过程满足如下条件时:

(1) 虚拟机启动过程基于可信根开始运行.

(2) 对 $\forall s_i\in S$ 为可预期状态, $\exists a_i\in A$ 使得 $verify(s_i,s_{i+1})=\text{true}$.

则虚拟机启动过程是可信的.

证明: 要证明定理 3 成立, 只需要证明虚拟机的启动过程满足定义 4 中的两个条件即可.

(1) 利用虚拟化技术为虚拟机配置唯一绑定且对应的虚拟可信根 vtpcm (virtual trusted platform control module), 虚拟可信根先于虚拟机的启动而初始化, 为虚拟机的启动、运行提供可信支撑. 虚拟可信根的可信性由位于宿主机底层的物理可信芯片保证, 是一种无条件可信, 无法从进程可信和动作序列推出, 满足定义 4 中的条件 (1). 接下来只需证明定理 3 的条件 (2) 满足定义 4 中的条件 (2) 即可.

(2) 对于任意可预期状态 $s_i\in S$, 由定理 2 可知, 总有公式 (12) 成立:

$$process(s_0,\alpha_i)\overset{p}{=}proccess(s_0,pure(\alpha_i,p)) \tag{12}$$

由 $verify(s_i,s_{i+1})=\text{true}$ 及定义 7 可知, $\exists a_i\in A$ 为符合预期的系统调用行为, 则对于 $s_{i+1}=step(s_i,a_i)$:

1) $a_i\in A$ 为对虚拟机启动过程产生干扰的预期行为, 则 $pure(\alpha_{i+1},p)=pure(\alpha_i\circ a_i,p)=a_i\circ pure(\alpha_i,p)$, 所以, $proccess(s_0,pure(\alpha_{i+1},p))=proccess(s_0,pure(\alpha_i,p)\circ a_i)=step(process(s_0,pure(\alpha_i,p)),a_i)=process(s_0,\alpha_{i+1})$.

2) $a_i\in A$ 为对虚拟机启动过程无干扰的预期行为, 则 $pure(\alpha_{i+1},p)=pure(\alpha_i\circ a_i,p)=pure(\alpha_i,p)$, 所以,

$$proccess(s_0,pure(\alpha_{i+1},p))=proccess(s_0,pure(\alpha_i,p))=process(s_0,\alpha_i).$$

又因为 $a_i\in A$ 对虚拟机进程无干扰, 所以由局部无干扰性可知 $s_i\overset{p}{=}step(s_i,a_i)=s_{i+1}$, 所以 $proccess(s_0,pure(\alpha_{i+1},p))=process(s_0,\alpha_i)=process(s_0,\alpha_{i+1})$ 成立.

综上, 对 $\forall s_i\in S$ 为可预期状态, 且 $verify(s_i,s_{i+1})=\text{true}$ 时, 状态 s 满足 $process(s_0,\alpha)\overset{p}{=}proccess(s_0,pure(\alpha,p))$.

令 $s=t=t_0$, 总有公式 (13) 成立:

$$process(s,\alpha) \overset{p}{=} proccess(t,pure(\alpha,p)) \tag{13}$$

又因为 $observe(a_i)$ 完整性未受到破坏, 则在 $s \overset{p}{=} t$ 等价观察条件下, 必有公式 (14) 成立:

$$execute(s,observe(a_i)) \overset{p}{=} execute(t,observe(a_i)) \tag{14}$$

因此, 虚拟机进程满足输出隔离性. 由定理 1 可得, 满足输出隔离性且满足公式 (12) 的进程是运行时可信的.

综上所述, 得证.

定理 3 在定理 1 的理论基础之上给出了虚拟机启动过程的可信判定定理, 该定理给出了虚拟机进程运行过程中系统的实际状态与预期状态总是在同步执行的情况下保持等价的条件, 即一个从可信初始状态 s_0 开始运行的虚拟机进程, 保证任意可预期状态到下一个状态间的转移过程是可信的, 且在状态转移过程中系统调用的客体对象是可信的, 就能够有效保证虚拟机进程运行过程中的输出隔离、单步隔离、局部无干扰, 是定理 1 的实际应用.

4 总体架构

4.1 基本思想

基于第 2.1 节构造场景: 云环境中, 多虚拟机进程共同运行在同一物理节点上, 虚拟机进程系统调用的执行会导致虚拟机进程状态发生跃迁. 目标虚拟机启动时, 该虚拟机进程运行时可能会受到不可信虚拟机进程的非预期干扰. 通过系统调用攻击正常虚拟机进程势必会导致虚拟机启动过程的不可信, 从而导致目标虚拟机状态的不可控. 通过系统调用攻击虚拟机进程, 主要有两种方式: 一是修改正常系统调用序列实现攻击目的; 二是系统调用重定向.

本文基于无干扰理论, 从保证虚拟机进程运行时可信的角度出发, 选择虚拟机进程系统调用作为原子动作, 在计算节点 C_{Node} 中利用 ptrace 命令实时获取虚拟机进程相关的系统调用, 通过分析系统调用, 排除对虚拟机启动相关无干扰的系统调用行为, 建立虚拟机进程可信运行产生预期干扰的系统调用行为白名单. 同时以 Linux 操作系统作为原型环境, 基于 Linux 安全模块 LSM (Linux security module)[53,54]机制, 利用 SELinux[55]在宿主机操作系统内核层通过重写钩子函数的方式来对虚拟机的启动过程的重点系统调用做主动控制, 并通过系统调用的相应接口函数对虚拟机启动过程加载的目标客体文件做主动度量. 根据第 3.2 节中的定理 3, 对相应度量监测结果进行实时判定, 综合考量虚拟机进程状态的跃迁是否符合预期, 从而对虚拟机启动过程可信状态进行评估.

值得指出的是, 虽然本文捕获系统调用的方式是通过修改 Linux 内核实现, 但是基于 Linux 内核本身集成的安全框架, 不会对内核的运行造成过大负担.

4.2 总体框架设计

基于无干扰理论的虚拟机可信启动研究方案总体架构如图 5 所示, 主要包括: 根据定理 3 对虚拟机状态进行评估的可信管理模块, 对虚拟机启动命令进行监控的监测机制, 结合可信管理模块的判定结果对虚拟机进程相关系统调用执行进行管控的控制机制.

(1) 监测机制: 对用户层的虚拟机启动命令进行监测, 并在监测到虚拟机启动命令后调用控制机制对虚拟机进程在内核层的系统调用行为进行拦截控制.

(2) 控制机制: 是监测机制发挥作用的入口, 接受监测机制的调用, 基于 Linux 内核的 LSM 通用框架实现对虚拟机进程运行过程中的系统调用行为进行控制, 主要控制过程包括捕获系统调用行为, 获取行为相关的客体、操作等信息, 将其发送至可信管理模块进行处理, 并根据判定机制对虚拟机进程运行状态的可信判定结果管控虚拟机相关系统调用的执行.

(3) 可信管理模块: 对虚拟机进程运行过程中的动态行为及加载的静态客体进行可信判定, 并将判定结果发送至控制机制.

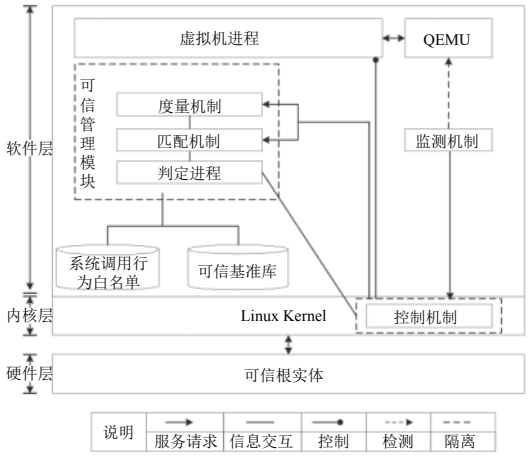

图 5 总体架构图

● 度量机制: 响应控制机制的服务请求, 在虚拟可信根的支持下, 对虚拟机进程运行过程中加载的重要客体文件进行可信度量, 并将度量结果发送至判定机制. 例如: 当用户层的虚拟机启动命令发出后, 虚拟机进程会通过 ioctl 系统调用进入 KVM 内核层, 完成虚拟机的相关初始化工作后, 跳转到虚拟 BIOS 的代码入口处去执行, 然后通过系统调用加载镜像数据. 此时需要对虚拟 BIOS、增量镜像数据等文件进行可信度量, 防止被修改过的数据载入内存.

● 匹配机制: 响应控制机制的服务请求, 对虚拟机进程运行过程中的重要系统调用行为与系统调用行为白名单进行匹配, 并将结果发送至判定机制.

● 判定机制: 接收度量机制及匹配机制发送的结果, 根据定理 3 实时判定虚拟机启动过程是否可信, 并将判定结果发送至控制机制.

虚拟机可信判定算法伪代码如算法 1 所示.

算法 1. 虚拟机可信判定算法.

输入: 系统调用行为匹配结果 (Result_Syscall), 客体文件度量结果 (Ob_measurment), 判定策略 (定理 3: Theorem 3), 虚拟机初始状态 (s_0);
输出: 虚拟机状态 Result_state (false 代表不可信, true 代表可信).

```
Begin
Result_state ← false;
//把虚拟机状态假定为不可信的;
If (s0 ← true){
//判断虚拟机初始状态是否可信
While vm_ImageFile not be load
//判断镜像文件是否完成加载
{
  if(Result_Syscall ⊆ Theorem 3&&Ob_measurment⊆ Theorem3)
  Result_state ← True;
  //基于相关结果, 根据判定定理 3, 实时判定虚拟机当前状态可信;
  else {
        Result_state ← False;
```

```
            //把虚拟机状态置为不可信;
            break;
    }
}
    return Result_state;
}
else
    return False;
End
```

(4) 系统调用行为白名单: 首先在可信的情况下, 利用 ptrace 命令对虚拟机进程的运行过程进行监测, 收集虚拟机进程运行过程中正常的系统调用行为. 通过实验分析选取对虚拟机启动过程影响较大的重要系统调用作为系统调用行为白名单的主要来源, 建立对虚拟机启动过程产生预期干扰的系统调用行为白名单. 其中, 白名单中的系统调用应包括相应的参数内容.

(5) 可信基准库: 主要为度量机制提供可信基准值校验数据, 包括但不仅限于虚拟机的虚拟 BIOS、增量镜像等基准值数据. 本文可信基准库的获取基于这样一个安全前提假设, 即在确保可信的宿主机平台上进行虚拟机完整启动过程监测, 并对虚拟机启动过程加载的静态数据进行完整性度量, 并获取相关度量值, 然后将度量值存储到可信基准库中.

需要指出的是本文基于国产可信根 TPCM (trusted platform control module), 利用虚拟化技术为云虚拟机提供虚拟可信根 vtpcm, 将信任关系从物理平台传递到云环境中, 保证虚拟机启动的初始状态 s_0 是可信的.

5 实验分析

本文通过在局域网内搭建并运行 OpenSatck 平台[52]来模拟云环境下虚拟机的运行环境, 其中包括一个控制节点及两个计算节点, 控制节点用于对 OpenStack 平台下的虚拟机进行管理, 计算节点用于给虚拟机提供运行环境. 本文基于可信计算北京市重点实验室开发的虚拟可信根对虚拟机的启动过程做可信支撑. 节点软硬件配置如表 1 所示.

表 1 节点软硬件配置表

名称	配置
操作系统	CentOS 7.0
虚拟化方式	KVM+QEMU
可信软件基框架	Cube 1.3[56]
密码算法	国密算法包GM-SM系列
vTPCM测试工具	vTPCM_utils
内存、硬盘、网卡	8 GB, 1 TB, 1 000 Mb/s以太网有线网卡

5.1 安全功能分析

为了测试本文研究方案是否能够动态防护针对虚拟机进程的攻击, 我们基于两种内核级的 rootkit[57]建立两种威胁场景, 并模拟它们的攻击来对虚拟机可信启动研究方案的安全性能进行测试, 表 2 展示了我们在实验中选择的 rootkit 及其恶意攻击方式.

威胁场景假设: (1) 本文假设所构造的 OpenStack 云场景中使用的物理计算节点是基于可信引导完成启动过程, 且具备相应的安全机制能够保证物理计算节点可信; (2) qemu-kvm 进程在模拟虚拟机启动的过程中受到修改

其系统调用行为的恶意攻击, 值得指出的是, 攻击能否在拥有可信根的计算节点上实施并不属于本文关注的内容; (3) 每个 KVM 虚拟机都视为一个 Linux 普通进程, 监视 qemu-kvm 进程的系统调用行为足以保证虚拟机应对修改其系统调用行为的攻击. 考虑两种攻击场景.

表 2　节点 IP 对应表

rootkit	攻击级别	攻击方式
Enyelkm	内核级	替换system_call和sysenter_entry的若干指令, 使任何系统调用请求跳转到攻击者的恶意模块
Knark	内核级	修改系统调用表, 将某些系统调用编号对应的服务函数替换为恶意函数, 使程序执行重定向

场景 1: Enyelkm[58]是一款基于可加载内核模块 LKM (Linux kernel module) 的内核级 rootkit, 被实现为一个可加载的隐藏内核模块.其在不修改系统调用表的基础之上, 通过替换中断描述符表的 system_call 函数中检测系统调用号是否越界的指令无条件跳转指令的方式, 使进程运行过程所请求的系统调用都会被转向攻击者预先设定的恶意处理函数执行逻辑, 从而实现攻击.

本文使用基于 Enyelkm, 通过将其函数执行逻辑替换为自定义的打印“hello world”的程序文件, 然后编译成为一个简单的内核模块 lkm_example, 并在虚拟机运行时挂载该模块. 当模块 lkm_example 被挂载后, 能够自动隐藏不被 lsmod 搜索到, 此时挂载后的模块会扩充到虚拟机镜像文件的增量镜像中, 完成了对虚拟机镜像文件的恶意修改攻击行为的模拟. 本文研究方案的监测机制通过对虚拟机进程运行过程系统调用加载的重要客体文件进行可信度量, 能够及时发现恶意程序对正常系统调用的 hook, 防止恶意隐藏模块被挂载, 保证虚拟机可信启动.

场景 2: Knark 是基于 LKM 的内核级 rootkit, 其通过修改系统调用表将某些系统调用编号对应的服务函数替换为攻击者的恶意处理函数, 从而实现 root 提权、非法加载进程等非法行为. 在计算节点中, 虚拟机进程通过 read 系统调用加载虚拟机配置文件时, Knark 通过替换该系统调用为自定义系统调用, 实现对虚拟机配置文件的修改或其他非法恶意操作行为.

本文通过对 Linux 系统调用函数进行劫持替换的目的来模拟 Knar 攻击 (如图 6 所示), 以此来测试本方案的检测能力. 以 sys_read 为例, 在 x86_64 架构机器中展示劫持系统调用的过程: 首先, 通过 kallsyms_lookup_name (“sys_call_table”) 获取系统调用表地址 (其中 system call table 是一个数组, 可以根据下标为“ __NR_read”来找到内核具体的系统调用函数 sys_read() 地址), 然后将该地址替换为自定义的“恶意函数”即可. 由于系统调用表 sys_call_table 地址符号所对应的区域是只读的, 因此在进行函数地址替换前需要对“只读”权限进行修改. 具体的: 如果控制寄存器 CR0 的第 16 位置位, 则表示禁止系统进程写那些只有只读权限的文件, 所以在修改系统调用表 sys_call_table 之前先将 CR0 的第 16 位清零 (具体代码见附录), 当修改完后重新恢复置位即可.

通过实时监控虚拟机进程的系统调用, 并与系统调用行为白名单进行匹配, 能够及时发现虚拟机进程执行过程中系统调用行为的异常调用, 保证虚拟机启动过程的可信.

5.2　性能开销分析

本节采用时间负载 ϕ 作为本文研究方案的性能评价指标, 其中引起时间负载的因素包括 3 个部分: (1) 捕获系统调用所用的时间 t_1; (2) 匹配系统调用行为白名单的时间 t_2; (3) 合法系统调用加载客体数据的可信度量时间 t_3.

在实际运行及测试中, 虚拟机启动过程中各阶段的执行流程及配置文件的读取顺序是固定不变的, 因此在 qemu-kvm 进程模拟虚拟机启动过程的时间段内, 启动过程的系统调用流序列呈现出局部稳定的状态. 可信启动相较于普通启动过程而言, 其差别在于系统调用执行过程的监控及可信度量行为, 因此使用系统调用执行时间和能够有效表征启动时间. 能够利用 strace 命令得到了系统调用的执行时间, 设虚拟机可信启动过程的所有系统调用执行时间和为 $t_{\text{trust\_s}}$, 虚拟机正常启动过程系统调用执行时间和为 $t_{\text{normal\_s}}$.

因为本文原型实验对系统调用的捕获是基于 SELinux[55]机制完成, 通过在 SELinux 实现的 HOOK 函数中添加处理逻辑, 完成对系统调用的监控. 而 SELinux 作为一种基于强制访问控制的 LSM 通用安全框架的具体实现, 属于 Linux 内核的一部分, 且在 centOS 中是默认开启的, 即正常启动时, SELinux 也会对进程系统调用进行捕获, 从而通过策略规则来控制进程对内核对象进行访问. 因此认为, $t_{\text{trust\_s}} = t_{\text{normal\_s}} + t_2 + t_3$. 其中, $t_1 \in t_{\text{normal\_s}}$.

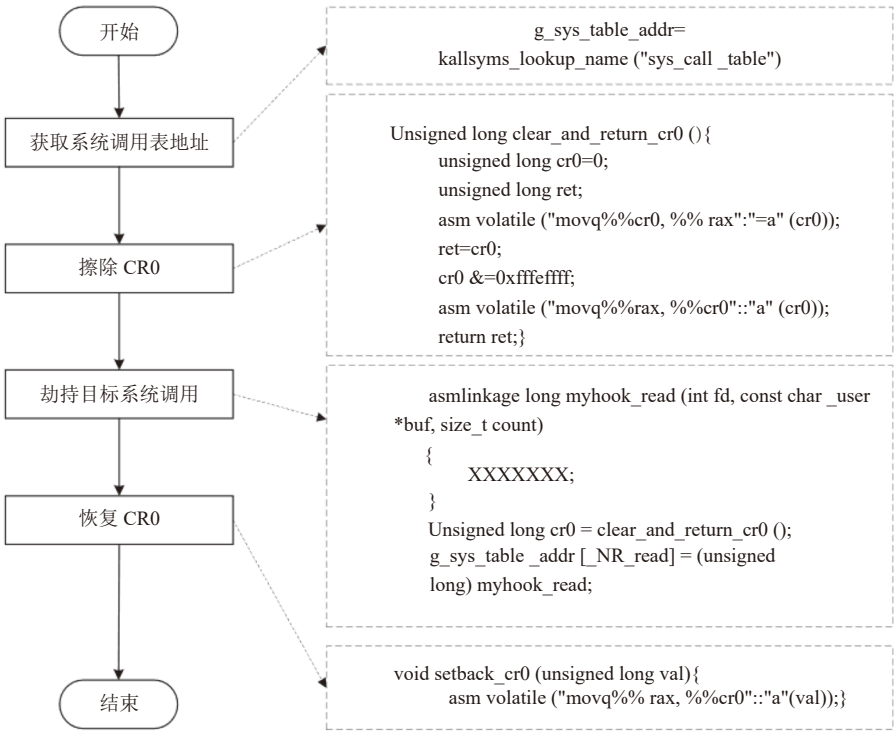

图 6 系统调用劫持示例

因此,

$$\phi=\frac{t_{\text{trust\_s}}-t_{\text{normal\_s}}}{t_{\text{normal\_s}}}=\frac{t_2+t_3}{t_{\text{normal\_s}}} \tag{15}$$

利用 strace 命令, 结合 KVM 虚拟机的启动流程对虚拟机启动过程的相关系统调用进行统计, 两次统计结果表明虚拟机不同启动过程所调用的同一类系统调用次数大致稳定, 如图 7 所示. 分析发现虚拟机启动各阶段主要受配置文件的读写等系统调用影响较大, 因此本文选取 write、read 这两类系统调用作为系统调用白名单规则的主要来源. 当上述两类系统调用被捕获并进行系统调用白名单匹配后, 根据其系统调用参数对其加载的文件进行可信度量.

(1) t_2 主要是字符串比较等指针操作, 进行一次匹配所带来的性能损耗极小. 但是在虚拟机进程模拟虚拟机启动过程的这一阶段内系统调用的次数是相对较大 (如表 3 所示), 如果每一次系统调用都进行白名单匹配, 那么会给内核带来极大的负担. 值得指出的是表 3 中的数据并非是稳定的数值, 表 3 的数据是在实验过程中多次实验得到的平均值, 每次实验的真实值围绕平均值上下浮动.

通过查阅相关文献发现, rootkit 活跃期和周期时间一般都在 10 s 数量级以上, 只要随机时间间隔设置合理能够有效发现 rootkit 攻击. 因此在实际实验过程中可以采取间隔随机时间段的方式来进行白名单规则匹配, 这样虽然在一定程度上降低了安全性, 但是极大地提升了性能. 实验发现, 当随机时间间隔在 [1.5, 3] ms 之间时, t_2 在 [0.7, 1.43] s 之间来回浮动, 如图 8 所示.

(2) t_3 是对重要客体文件进行完整性度量的耗时, 为了提高性能, 只对与虚拟机启动相关度较大的客体文件进行全面可信度量, 即虚拟机增量镜像文件、BIOS 的模拟文件 bios-256k.bin. 本实验发现调用 SM3[59]进行完整性度量所耗费的时间与被度量文件的大小呈线性正比关系, 如图 9 所示.

由于虚拟机进程运行时加载的 BIOS 模拟文件的大小是恒定不变的且数量级较小, 因此 t_3 主要体现在虚拟机增量镜像文件的完整性度量耗时上, 即随着虚拟机增量镜像文件不断变大, 虚拟机启动的时间开销也会逐渐变大.

表 4 给出了客体文件完整性度量的时间消耗, 其中虚拟机增量镜像文件完整性度量耗时最长, 总的完整性度量耗时 t_3 约为 0.64 s. 值得指出的是, 每次重启虚拟机时, 其增量镜像大小会发生变化, 从而导致完整性度量时间发生改变.

从上面测试分别可以获取 t_2 和 t_3 的值, 为了提高测试的精确性和科学性, 进行了 5 次实验, 图 10 给出了不同启动条件下的启动过程时间开销. 根据公式 (15) 计算得出对虚拟机进程的可信监控给虚拟机的启动带来的性能影响在 [6.8%, 9.3%] 区间内徘徊, 在虚拟机实际使用过程中并不会造成过大的影响.

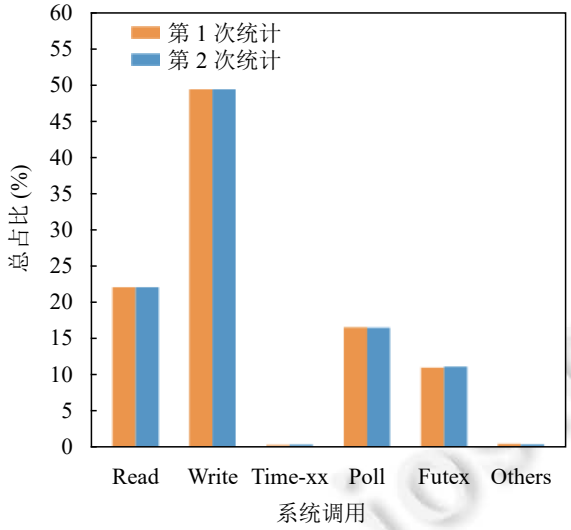

图 7 虚拟机进程系统调用统计

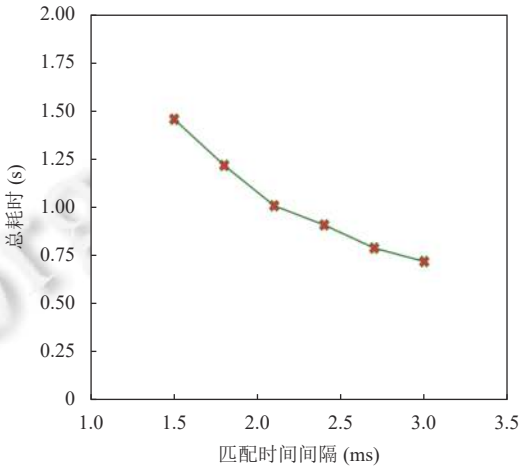

图 8 不同时间间隔匹配耗时

表 3 系统调用统计

系统调用名	调用次数	每条系统调用耗时 (μs)	总耗时 (s)
write	749901	3	2.22
read	334795	3	1.07

表 4 完整性度量时间消耗

客体文件	文件大小 (KB)	完整性度量耗时 (ms)	总耗时 t_3 (s)
bios-256k.bin	256	2.07	0.642
虚拟机增量镜像文	65536	640	

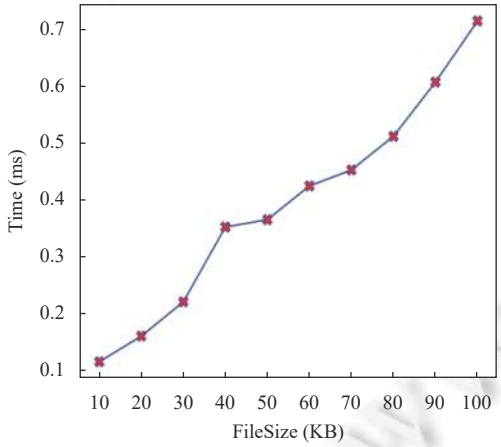

图 9 完整性度量时间开销

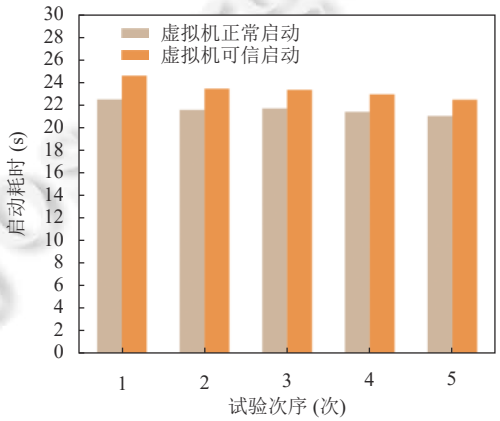

图 10 虚拟机启动时间负载分析

5.3 对比分析

通过对比分析来评估本文提出的基于非传递无干扰理论模型的虚拟机启动框架, 主要同已有的研究进行对

比 (表 5), 对本文所提出方案的优缺点进行分析总结.

结合表 5, 可以看出本文所述基于非传递无干扰理论的虚拟机可信启动方案具有以下几个方面的优势.

(1) 可信支持: 本文方案采用我国自主可信平台控制模块 TPCM 作为底层物理可信根, 并结合虚拟化技术基于 Cube 架构[56]实现了能够对云虚拟机进行可信支持的虚拟可信根 vTPCM.

(2) 细粒度: 本文方案从 qemu-kvm 进程的角度对虚拟机启动过程中系统调用行为进行监测, 能够较为细粒度地对虚拟机启动过程进行可信管控.

(3) 安全性: 本方案基于 SELinux 设置安全监控钩子函数, 能够从系统调用的层次对虚拟机启动过程的相关行为进行管控, 同时结合对虚拟机启动过程加载的静态文件数据进行完整性校验, 能够有效防止针对系统调用的恶意攻击.

(4) 动态性: 云环境中虚拟机启动过程具有高动态性, 装载时度量不能有效保证 qemu-kvm 进程运行时的可信性. 本文方案采用动态行为触发结合随机时间间隔的方式对虚拟机启动过程进行可信监测, 能够极大的提高可信度量的动态性, 有效防止 TOC-TOU 攻击.

表 5 本方案和已有方案对比分析

方案	可信支持		细粒度性	完整性校验	系统调用行为评估	安全性	动态性
	vTPM	vTPCM					
Alarifi[51]	×	×	具备	不具备	具备	系统调用级别安全监测	滑动窗口检测
刘川意等人[10]	√	×	不具备	具备	不具备	完整性度量	不具备
本文方案	×	√	具备	具备	具备	完整性度量动态行为监测	动态行为触发随机时间间隔

6 结束语

将可信计算应用到云环境中保证虚拟机可信启动是解决虚拟机可信的基础, 本文从保证虚拟机进程运行时可信的角度出发, 基于无干扰理论提出了一种虚拟机可信启动研究方案. 针对云计算模式下, 虚拟机启动过程的动态性及运行在同一物理节点上的多虚拟机进程间的非预期干扰问题, 首先基于无干扰理论, 提出并证明了虚拟机可信启动判定定理; 在此基础之上, 以虚拟机进程的系统调用为原子动作, 基于 Linux 内核的安全框架 SELinux, 设计并实现了能够监测、控制虚拟机进程运行过程的原型系统, 兼顾虚拟机进程的动态行为验证及静态组件的可信度量, 实现了对虚拟机启动过程的动态度量及控制, 避免了目标虚拟机启动时多虚拟机进程间的非预期干扰, 保证了虚拟机启动过程的动态可信. 通过实验分析, 验证了本文研究方案的有效性, 且原型系统带来的额外性能开销处于可接受范围内.

在接下来的研究中, 仍有一些工作需要进行: 针对虚拟机增量镜像数据部分的可信度量, 如何兼顾细粒度与低时间性能开销; 增加对系统调用行为不同序列及参数组合的差异性研究, 利用机器学习相关算法[60]构造能够精准反映虚拟机进程运行符合预期的系统调用序列行为模型.

References:

[1] Zhao XK, Yin JW, Chen ZN, Sheng HE. vSpec: Workload-adaptive operating system specialization for virtual machines in cloud computing. Science China Information Sciences, 2016, 59(9): 92105. [doi: 10.1007/s11432-015-5387-6]

[2] State Administration for Market Regulation, Standardization Administration. GB/T 22239-2019 Information Security Technology-baseline for Classified Protection of Cybersecurity. Beijing: China Standards Press, 2019. 92 (in Chinese).

[3] Zhang YQ, Wang XF, Liu XF, Liu L. Survey on cloud computing security. Ruan Jian Xue Bao/Journal of Software, 2016, 27(6): 1328−1348 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/5004.html [doi: 10.13328/j.cnki.jos.005004]

[4] Liu CY, Lin J, Tang B. Dynamic trustworthiness verification mechanism for trusted cloud execution environment. Ruan Jian Xue Bao/Journal of Software, 2014, 25(3): 662−674 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4447.html [doi: 10.13328/j.cnki.jos.004447]

[5] Feng DG, Zhang M, Zhang Y, Xu Z. Study on cloud computing security. Ruan Jian Xue Bao/Journal of Software, 2011, 22(1): 71−83 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/3958.htm [doi: 10.3724/SP.J.1001.2011.03958]

[6] Xiang Z, Gabriel F, Urbano E, Nguyen GT, Reisslein M, Fitzek FHP. Reducing latency in virtual machines: Enabling tactile internet for human-machine Co-working. IEEE Journal on Selected Areas in Communications, 2019, 37(5): 1098–1116. [doi: 10.1109/JSAC.2019.2906788]

[7] Feng DG, Liu JB, Qin Y, Feng W. Trusted computing theory and technology in innovation-driven development. Scientia Sinica Informationis, 2020, 50(8): 1127–1147 (in Chinese with English abstract). [doi: 10.1360/SSI-2020-0096]

[8] Srivastava A, Raj H, Giffin J, England P. Trusted VM snapshots in untrusted cloud infrastructures. In: Proc. of the 15th Int'l Symp. on Research in Attacks, Intrusions, and Defenses. Amsterdam: Springer, 2012. 1−21. [doi: 10.1007/978-3-642-33338-5_1]

[9] Wang QF, Yan F, Wang J, Wang T, Shi X. Secure storage and trusted launch of virtual machine in IaaS. Journal of Wuhan University (Natural Science Edition), 2014, 60(3): 231–236 (in Chinese with English abstract). [doi: 10.14188/j.1671-8836.2014.03.010]

[10] Liu CY, Wang GF, Lin J, Fang BX. Practical construction and audit for trusted cloud execution environment. Chinese Journal of Computers, 2016, 39(2): 339–350 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2016.00339]

[11] Azab AM, Ning P, Sezer EC, Zhang Xl. HIMA: A hypervisor-based integrity measurement agent. In: Proc. of the 2009 Annual Computer Security Applications Conf. Honolulu: IEEE, 2009. 461−470. [doi: 10.1109/ACSAC.2009.50]

[12] Berger B. Trusted computing group history. Information Security Technical Report, 2005, 10(2): 59–62. [doi: 10.1016/j.istr.2005.05.007]

[13] Anand A, Dhingra M, Lakshmi J, Nandy SK. Resource usage monitoring for KVM based virtual machines. In: Proc. of the 18th Int'l Conf. on Advanced Computing and Communications (ADCOM). Bangalore: IEEE, 2012. 66−70. [doi: 10.1109/ADCOM.2012.6563586]

[14] Shen CX, Gong B. The innovation of trusted computing based on the domestic cryptography. Journal of Cryptologic Research, 2015, 2(5): 381–389 (in Chinese with English abstract). [doi: 10.13868/j.cnki.jcr.000087]

[15] Shen CX, Zhang HG, Feng DG, Cao ZF, Huang JW. Survey of information security. Science in China Series F: Information Sciences, 2007, 50(3): 273–298. [doi: 10.1007/s11432-007-0037-2]

[16] Shen CX, Zhang HG, Wang HM, Wang J, Zhao B, Yan F, Yu FJ, Zhang LQ, Xu MD. Research on trusted computing and its development. Science China Information Sciences, 2010, 53(3): 405–433 (in Chinese with English abstract). [doi: 10.1007/s11432-010-0069-x]

[17] TCG Group. TCG specification architecture overview. 2007. https://trustedcomputinggroup.org/resource/tcg-architecture-overview-version-1-4/

[18] Shen CX. To create a positive cyberspace by safeguarding network security with active immune trusted computing 3.0. Journal of Information Security Research, 2018, 4(4): 282–302 (in Chinese with English abstract).

[19] State Cryptography Administration. GM/T 0012-2020 Trusted Computing-trusted Computing Interface Specification of Trusted Cryptography Module. Beijing: China Standards Press, 2020. 136 (in Chinese).

[20] Zhang QY, Feng DG, Zhao SJ. Design and formal analysis of TCM key migration protocols. Ruan Jian Xue Bao/Journal of Software, 2015, 26(9): 2396−2417 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4719.html [doi: 10.13328/j.cnki.jos.004719]

[21] Goguen JA, Meseguer J. Security policies and security models. In: Proc. of the 1982 IEEE Symp. on Security and Privacy. Oakland: IEEE, 1982. 11−11. [doi: 10.1109/SP.1982.10014]

[22] Rushby J, Computer Science Laboratory, SRI International. Noninterference, transitivity, and channel-control security policies. San Mateo: SRI International, 2005.

[23] van der Meyden R. What, indeed, is intransitive noninterference? In: Proc. of the 12th European Symp. on Research in Computer Security. Dresden: Springer, 2007. 235−250.

[24] Eggert S, van der Meyden R, Schnoor H, Wilke T. The complexity of intransitive noninterference. In: Proc. of the 2011 IEEE Symp. on Security and Privacy. Oakland: IEEE, 2011. 196−211. [doi: 10.1109/SP.2011.30]

[25] Zhang F, Zhang C, Chen W, Hu FN, Xu MD. Noninterference analysis of trust of behavior in cloud computing system. Chinese Journal of Computers, 2019, 42(4): 736–755 (in Chinese with English abstract). [doi: 10.11897/SP.J.1016.2019.00736]

[26] Zhang HG, Yan F, Fu JM, Xu MD, Yang Y, He F, Zhan J. Research on theory and key technology of trusted computing platform security testing and evaluation. Science China Information Sciences, 2010, 53(3): 434–453. [doi: 10.1007/s11432-010-0062-4]

[27] Xu MD, Zhang HG, Zhang F, Yang LJ. Survey on chain of trust of trusted system. Acta Electronica Sinica, 2014, 42(10): 2024–2031 (in Chinese with English abstract). [doi: 10.3969/j.issn.0372-2112.2014.10.024]

[28] Zhang X, Chen YL, Shen CX. Non-interference trusted model based on processes. Journal on Communications, 2009, 30(3): 6–11 (in Chinese with English abstract). [doi: 10.3321/j.issn:1000-436X.2009.03.002]

[29] Zhang F, Xu MD, Chao HC, Zhang C, Liu XL, Hu FN. Real time trust measurement of software: Behavior trust analysis approach based on noninterference. Ruan Jian Xue Bao/Journal of Software, 2019, 30(8): 2268−2286 (in Chinese with English abstract). http://www.jos.

org.cn/1000-9825/5768.htm [doi: 10.13328/j.cnki.jos.005768]

[30] Zhang X, Huang Q, Shen CX. A formal method based on noninterference for analyzing trust chain of trusted computing platform. Chinese Journal of Computers, 2010, 33(1): 74–81 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2010.00074]

[31] Zhao J, Shen CX, Liu JQ, Han Z. A noninterference-based trusted chain model. Journal of Computer Research and Development, 2008, 45(6): 974–980 (in Chinese with English abstract).

[32] Chen L, Zeng RR, Li F, Yang WM. Trust chain transfer model based on non-interference theory. Computer Science, 2016, 43(10): 141–144, 181 (in Chinese with English abstract). [doi: 10.11896/j.issn.1002-137X.2016.10.026]

[33] Zhang L, Chen XS, Liu L, Jin X. Trusted domain hierarchical model based on noninterference theory. The Journal of China Universities of Posts and Telecommunications, 2015, 22(4): 7–16. [doi: 10.1016/S1005-8885(15)60662-8]

[34] Sailer R, Zhang XL, Jaeger T, Van Doorn L. Design and implementation of a TCG-based integrity measurement architecture. In: Proc. of the 13th USENIX Security Symp. San Diego: USENIX Association, 2004. 223−238.

[35] Santos N, Gummadi KP, Rodrigues R. Towards trusted cloud computing. In: Proc. of the 2009 USENIX Association Workshop on Hot Topics in Cloud Computing. San Diego: USENIX Association, 2009. 14−19.

[36] He XF, Tian JF, Liu FM. Survey on trusted cloud platform technology. Journal on Communications, 2019, 40(2): 154–163 (in Chinese with English abstract). [doi: 10.11959/j.issn.1000?436x.2019035]

[37] Jin X, Wang QX, Li X, Chen XS, Wang W. Cloud virtual machine lifecycle security framework based on trusted computing. Tsinghua Science and Technology, 2019, 24(5): 520–534. [doi: 10.26599/TST.2018.9010129]

[38] Lin J, Liu CY, Fang BX. IVirt: Runtime environment integrity measurement mechanism based on virtual machine introspection. Chinese Journal of Computers, 2015, 38(1): 191–203 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2015.00191]

[39] Zhou ZJ, Wu LF, Hong Z, Xu MF. Trustworthiness measurement model of virtual machine for cloud computing. Journal of Southeast University (Natural Science Edition), 2014, 44(1): 45–50 (in Chinese with English abstract). [doi: 10.3969/j.issn.1001-0505.2014.01.009]

[40] Berger S, Cáceres R, Goldman KA, Perez R, Sailer R, van Doorn L. vTPM: Virtualizing the trusted platform module. In: Proc. of the 15th USENIX Security Symp. Vancouver: USENIX Association, 2006. 305–320.

[41] Anderson MJ, Moffie M, Dalton CI. Towards trustworthy virtualisation environments: Xen library os security service infrastructure. 2007. https://www.hpl.hp.com/techreports/2007/HPL-2007-69.pdf

[42] Murray DG, Milos G, Hand S. Improving xen security through disaggregation. In: Proc. of the 4th ACM SIGPLAN/SIGOPS Int'l Conf. on Virtual Execution Environments. Seattle: ACM, 2008. 151–160. [doi: 10.1145/1346256.1346278]

[43] Taubmann B, Reiser HP. Towards hypervisor support for enhancing the performance of virtual machine introspection. In: Proc. of the 20th IFIP WG 6.1 Int'l Conf. on Distributed Applications and Interoperable Systems. Valletta: Springer, 2020. 41–54. [doi: 10.1007/978-3-030-50323-9_3]

[44] Seshadri A, Luk M, Qu N, Perrig A. SecVisor: A tiny hypervisor to provide lifetime kernel code integrity for commodity OSes. In: Proc. of the 21st ACM SIGOPS Symp. on Operating Systems Principles. Stevenson: ACM, 2007. 335–350. [doi: 10.1145/1294261.1294294]

[45] Shi GY, Shen CX, Liu Y. Dynamical attestation for trust based on secure virtual machine introspection. Journal on Communications, 2011, 32(11A): 24–38 (in Chinese with English abstract).

[46] Du RZ, Pan WY, Tian JF. Dynamic integrity measurement model based on vTPM. China Communications, 2018, 15(2): 88–99. [doi: 10.1109/CC.2018.8300275]

[47] Wei Z, Gui XL, Wei HR, Si Y. TCP DDOS attack detection on the host in the KVM virtual machine environment. In: Proc. of the 11th IEEE/ACIS Int'l Conf. on Computer and Information Science. Shanghai: IEEE, 2012. 62–67. [doi: 10.1109/ICIS.2012.105]

[48] Dileesh ED, Shanthi AP. An application specific dynamic behaviour model using function-call sequence and memory access-graph for execution integrity verification. Computers & Security, 2021, 107: 102299. [doi: 10.1016/j.cose.2021.102299]

[49] Kawada T, Honda S, Matsubara Y, Takada H. TZmCFI: RTOS-aware control-flow integrity using trustzone for armv8-M. Int'l Journal of Parallel Programming, 2021, 49(2): 216–236. [doi: 10.1007/s10766-020-00673-z]

[50] Lee JH, Jang JS, Jang YJ, Kwak N, Choi Y, Choi C, Kim T, Peinado M, Kang BB. Hacking in darkness: Return-oriented programming against secure enclaves. In: Proc. of the 26th USENIX Security Symp. Vancouver: USENIX Association, 2017. 523–539.

[51] Alarifi SS, Wolthusen SD. Detecting anomalies in IaaS environments through virtual machine host system call analysis. In: Proc. of the 2012 Int'l Conf. for Internet Technology and Secured Trans. London: IEEE, 2012. 211–218.

[52] Rosado T, Bernardino J. Implementation of a low cost IaaS using openstack. In: Proc. of the 11th Int'l Joint Conf. on Software Technologies (ICSOFT 2016). Lisbon: Science and Technology Publications, 2016. 298–303.

[53] Isohara T, Takemori K, Miyake Y, Qu N, Perrig A. LSM-based secure system monitoring using kernel protection schemes. In: Proc. of the 2010 Int'l Conf. on Availability, Reliability and Security. Krakow: IEEE, 2010. 591–596. [doi: 10.1109/ARES.2010.48]

[54] Watson RNM. A decade of OS access-control extensibility: Open source security foundations for mobile and embedded devices. Queue,

2013, 11(1): 20–41. [doi: 10.1145/2428616.2430732]
[55] Smalley S, Vance C, Salamon W. Implementing SELinux as a Linux security module. NAI Labs Report #01-043, 2001.
[56] Hu J, Shen CX, Gong B. Trusted Computing 3.0 Engineering Fundamentals. Beijing: Posts & Telecom Press, 2017. 239 (in Chinese).
[57] Li YG, Chung YC, Hwang K, Li YJ. Virtual wall: Filtering rootkit attacks to protect Linux kernel functions. IEEE Trans. on Computers, 2021, 70(10): 1640–1653. [doi: 10.1109/TC.2020.3022023]
[58] Hofmann OS, Dunn AM, Kim S, Roy I, Witchel E. Ensuring operating system kernel integrity with OSck. ACM SIGARCH Computer Architecture News, 2011, 39(1): 279–290. [doi: 10.1145/1961295.1950398]
[59] Liu LM, Wang XY, Zhang ZF, Wang P. Research on the international standard proposals of commercial cryptographic algorithms. Information Technology & Standardization, 2018, (5): 17–20 (in Chinese with English abstract).
[60] Melvin AR, Kathrine GJW, Johnraja JI. The practicality of using virtual machine introspection technique with machine learning algorithms for the detection of intrusions in cloud. In: Proc. of the 1st Int'l Conf. on Advanced Scientific Innovation in Science, Engineering and Technology. Chennai: ICASISET, 2021. 20–35. [doi: 10.4108/eai.16-5-2020.2303939]

附中文参考文献:

[2] 国家市场监督管理总局, 国家标准化管理委员会. GB/T 22239-2019 信息安全技术网络安全等级保护基本要求. 北京: 中国标准出版社, 2019. 92.
[3] 张玉清, 王晓菲, 刘雪峰, 刘玲. 云计算环境安全综述. 软件学报, 2016, 27(6): 1328−1348. http://www.jos.org.cn/1000-9825/5004.html [doi: 10.13328/j.cnki.jos.005004]
[4] 刘川意, 林杰, 唐博. 面向云计算模式运行环境可信性动态验证机制. 软件学报, 2014, 25(3): 662–674. http://www.jos.org.cn/1000-9825/4447.html [doi: 10.13328/j.cnki.jos.004447]
[5] 冯登国, 张敏, 张妍, 徐震. 云计算安全研究. 软件学报, 2011, 22(1): 71–83. http://www.jos.org.cn/1000-9825/3958.htm [doi: 10.3724/SP.J.1001.2011.03958]
[7] 冯登国, 刘敬彬, 秦宇, 冯伟. 创新发展中的可信计算理论与技术. 中国科学: 信息科学, 2020, 50(8): 1127–1147. [doi: 10.1360/SSI-2020-0096]
[9] 王庆飞,严飞,王鹃,王涛,石翔. IaaS下虚拟机的安全存储和可信启动. 武汉大学学报(理学版), 2014, 60(3): 231–236. [doi: 10.14188/j.1671-8836.2014.03.010]
[10] 刘川意, 王国峰, 林杰, 方滨兴. 可信的云计算运行环境构建和审计. 计算机学报, 2016, 39(2): 339–350. [doi: 10.11897/SP.J.1016.2016.00339]
[14] 沈昌祥, 公备. 基于国产密码体系的可信计算体系框架. 密码学报, 2015, 2(5): 381–389. [doi: 10.13868/j.cnki.jcr.000087]
[16] 沈昌祥, 张焕国, 王怀民, 王戟, 赵波, 严飞, 余发江, 张立强, 徐明迪. 可信计算的研究与发展. 中国科学: 信息科学, 2010, 53(3): 405–433. [doi: 10.1007/s11432-010-0069-x]
[18] 沈昌祥. 用主动免疫可信计算3.0筑牢网络安全防线营造清朗的网络空间. 信息安全研究, 2018, 4(4): 282–302.
[19] 国家密码管理局. GM/T 0012-2020 可信计算可信密码模块接口规范. 北京: 中国标准出版社, 2020. 136.
[20] 张倩颖, 冯登国, 赵世军. TCM密钥迁移协议设计及形式化分析. 软件学报, 2015, 26(9): 2396−2417. http://www.jos.org.cn/1000-9825/4719.html [doi: 10.13328/j.cnki.jos.004719]
[25] 张帆, 张聪, 陈伟, 胡方宁, 徐明迪. 基于无干扰的云计算环境行为可信性分析. 计算机学报, 2019, 42(4): 736–755. [doi: 10.11897/SP.J.1016.2019.00736]
[27] 徐明迪, 张焕国, 张帆, 杨连嘉. 可信系统信任链研究综述. 电子学报, 2014, 42(10): 2024–2031. [doi: 10.3969/j.issn.0372-2112.2014.10.024]
[28] 张兴, 陈幼雷, 沈昌祥. 基于进程的无干扰可信模型. 通信学报, 2009, 30(3): 6–11. [doi: 10.3321/j.issn:1000-436X.2009.03.002]
[29] 张帆, 徐明迪, 赵涵捷, 张聪, 刘小丽, 胡方宁. 软件实时可信度量: 一种无干扰行为可信性分析方法. 软件学报, 2019, 30(8): 2268–2286. http://www.jos.org.cn/1000-9825/5768.htm [doi: 10.13328/j.cnki.jos.005768]
[30] 张兴, 黄强, 沈昌祥. 一种基于无干扰模型的信任链传递分析方法. 计算机学报, 2010, 33(1): 74–81. [doi: 10.3724/SP.J.1016.2010.00074]
[31] 赵佳, 沈昌祥, 刘吉强, 韩臻. 基于无干扰理论的可信链模型. 计算机研究与发展, 2008, 45(6): 974–980.
[32] 陈亮, 曾荣仁, 李峰, 杨伟铭. 基于无干扰理论的信任链传递模型. 计算机科学, 2016, 43(10): 141–144, 181. [doi: 10.11896/j.issn.1002-137X.2016.10.026]
[36] 何欣枫, 田俊峰, 刘凡鸣. 可信云平台技术综述. 通信学报, 2019, 40(2): 154–163. [doi: 10.11959/j.issn.1000?436x.2019035]
[38] 林杰, 刘川意, 方滨兴. IVirt: 基于虚拟机自省的运行环境完整性度量机制. 计算机学报, 2015, 38(1): 191–203. [doi: 10.3724/SP.J.1016.2015.00191]
[39] 周振吉, 吴礼发, 洪征, 徐明飞. 云计算环境下的虚拟机可信度量模型. 东南大学学报(自然科学版), 2014, 44(1): 45–50. [doi: 10.3969/

j.issn.1001-0505.2014.01.009]
[45] 施光源, 沈昌祥, 刘毅. 基于安全虚拟机内省的动态可信证明方法. 通信学报, 2011, 32(11A): 24–38.
[56] 胡俊, 沈昌祥, 公备. 可信计算3.0工程初步. 北京: 人民邮电出版社, 2017. 239.
[59] 刘丽敏, 王小云, 张振峰, 王鹏. 商用密码算法国际标准提案研究. 信息技术与标准化, 2018, (5): 17–20.

附录 A. 部分代码示例

```
Unsigned long clear_and_return_cr0(){
        unsigned long cr0=0;
        unsigned long ret;
        asm volatile("movq %%cr0, %%rax":"=a"(cr0));
        ret=cr0;
        cr0 &= 0xfffeffff;
        asm volatile("movq %%rax, %%cr0"::"a"(cr0));
        return ret;
}
void setback_cr0(unsigned long val){
        asm volatile("movq %%rax, %%cr0"::"a"(val));
}
```

值得指出的是在 32 位和 64 位的系统下汇编指令是有区别的, 上述示例适用环境为 64 位机器; 下面是 32 位机的写法:

```
        unsigned long clear_and_return_cr0(){
        unsigned long cr0=0;
        unsigned long ret;
        asm volatile("movl %%cr0, %%eax":"=a"(cr0));
        ret=cr0;
        cr0 &= 0xfffeffff;
        asm volatile("movl %%eax, %%cr0"::"a"(cr0));
        return ret;}
void setback_cr0(unsigned long val){
        asm volatile("movl %%eax, %%cr0"::"a"(val));}
```

黄浩翔(1992－), 男, 博士生, 主要研究领域为可信计算, 云计算, 访问控制.

袁艺林(1991－), 女, 博士生, 主要研究领域为信息安全, 云计算, 云存储.

张建标(1969－), 男, 博士, 教授, 博士生导师, 主要研究领域为可信计算, 网络安全, 区块链技术.

王晓(1983－), 女, 博士, 讲师, 主要研究领域为信息安全, 可信计算, 云安全.