第 1 卷 第 4 期
2012 年 11 月

集 成 技 术
JOURNAL OF INTEGRATION TECHNOLOGY

Vol. 1 No. 4
Nov. 2012

基于体系结构扩展的云计算安全增强研究

刘宇涛 夏虞斌 陈海波

（ 上海交通大学并行与分布式系统研究所 上海 200240 ）

摘 要 近年来，云计算的安全问题得到了广泛的关注。由于云计算模式的共享、外包和开发的特性，用户并不拥有对计算和数据的完全控制；相反，云平台内部的恶意系统管理员可在用户不知情的情况下窃取或篡改用户的关键数据。研究者们为保护云计算数据的隐私性与完整性，对体系结构进行了扩展，尝试缩小云安全所依赖的软硬件栈。本文阐述了这些研究中采用的主要技术类型，包括内存隔离增强、安全处理器加密等。

关键词 云计算安全；虚拟化安全；体系结构扩展

Researches on Architecture Extensions for Cloud Security

LIU Yu-tao XIA Yu-bin CHEN Hai-bo

(*Institution of Parallel and Distributed Systems, Shanghai Jiao Tong University, Shanghai* 200240, *China*)

Abstract The security issue of cloud computing has received widely attention in recent years. Due to the sharing, out-sourcing and openness features of cloud computing mode, the end-users don't gain the complete control over their computing and data. Instead, a malicious system operator can tamper with or steal user's critical data without user's awareness. Some researchers tried to protect the privacy and integrity of data in the cloud by extending architecture and reduce the hardware/software stack that cloud security relies on. This paper presents technologies in these researches, including enforcing memory isolation, encryption by secure processor, et al.

Keywords cloud security; virtualization security; architecture extension

1 引 言

安全问题已经成为云计算的关键问题。在云计算模式中，用户把运算与存储外包给云服务商以降低计算成本，然而与此同时却失去了对计算和数据的完全控制。云平台内部的系统管理员对软硬件有最高的访问权限，可在用户不知情的情况下窃取或篡改用户的关键数据；2010年Google的两名员工被曝长时间窥探用户的邮件与聊天记录。目前，如何保证云计算用户数据的隐私性与完整性是一个亟待解决的问题。

由于云计算的软件栈变得越来越庞大，其安全缺陷的数量也随之增加[1]。在这些构成基础架构的软件中，虚拟化软件栈的安全尤其重要。其中，虚拟化监控器（Virtual Machine Monitor,VMM）拥有最高权限，若其安全缺陷被攻击者利用，会影响到其上运行的所有虚拟机，严重威胁到云计算的安全。为此，许多研究者尝试通过对体系结构的扩展，将虚拟化软件栈从TCB中排除，以进一步缩小“计算可信基”（Trusted Computing Base，TCB），减小甚至消除安全性对软件的依赖，从而增强云计算的安全。

本文总结了近年来学术界为增强云计算安全在体系结构扩展方面的尝试，包括增强VMM和VM（Virtual Machine）之间的内存隔离、利用安全处理器对VM内存进行加密，以及对利用x86已有的特性，如SMM（System Management Mode）等，对VMM的操作进行检查和限制。本文列举了这些技术对应的系统，分析了其特点与缺陷，并展望了在体系结构领域增强云计算安全的可能研究方向。

作者简介：刘宇涛，博士研究生，研究方向为虚拟化、系统安全，E-mail: yutaoliu@sjtu.edu.cn；夏虞斌，博士，讲师，研究方向为虚拟化、操作系统、系统安全；陈海波，博士，教授，研究方向为虚拟化、操作系统、分布式系统、系统安全。

2　基于内存管理限制的保护技术

数据在内存中的安全是云计算安全最重要的前提之一。VMM管理物理主机的所有内存，并对其上运行的VM提供内存虚拟服务。VM的应用程序使用客户虚拟地址（Guest Virtual Address，GVA），这通过VM的客户页表翻译成客户物理地址（Guest Physical Address，GPA）。GPA是进一步转化为主机通过保持在VMM的扩展页表（Extended Pages Tables，EPT）的物理地址（Host Physical Address，HPA）。GPA从客户VM的角度来看是连续的内存空间，但可以映射到不连续的物理内存空间（HPA）。VMM通过P2M（Physical to Machine）表来管理从GPA到HPA的映射。在传统的体系结构上，VMM可直接映射并访问其上运行的任意VM的内存，因此恶意管理员可直接窃取或篡改用户数据。

2.1　扩展MMU安全

H-SVM[2]扩展了现有的虚拟化体系架构，限制VMM对P2M表的修改必须通过新增的硬件管理接口进行。H-SVM在处理器中添加了4段微代码，来向VMM提供虚拟机的管理接口，包括创建虚拟机、创建内存页映射、取消内存页映射和虚拟机调度。硬件可通过这四个接口对VMM创建的内存映射进行检查，阻止VMM或其他VM对某一个虚拟机的非法内存访问。同时，在现有的物理内存中指定了一块受保护的内存区域，用来存储三部分信息：NPT（Nested Page Table）表，虚拟机控制信息，以及物理页所有者表（Page Ownership Table）；这段受保护的内存区域只有安全硬件才具有访问权限。

VMM需要调用H-SVM提供的4个接口来进行VM的内存管理。当VMM启动一个虚拟机时，向H-SVM显式发送创建请求，H-SVM会初始化该虚拟机的控制信息，建立相应的P2M表。当VMM请求创建一个内存页映射时，H-SVM会根据物理页所有者表来判断该物理页是否属于其它的虚拟机：如果是，则拒绝该请求；否则，将该物理页的拥有者标记为A。同样，当VMM从某个虚拟机中回收一个物理页时，也需要向H-SVM请求，H-SVM会将该物理页内容清零，并在POT表中重置该页的拥有者。当VMM要将某个虚拟机调度到一个处理器核上运行时，需调用H-SVM提供的接口，后者根据该虚拟机的控制信息恢复寄存器状态，并设置好相应NPT的指针信息，然后进行调度。

除此之外，HSVM还允许通过和虚拟机协商来支持虚拟机内存共享和内存置换机制；同时，HSVM将IOMMU架构的I/O页表所在的内存进行保护，从而可防止恶意的DMA请求对虚拟机隐私数据的窃取和篡改。

HSVM通过利用CPU的微代码机制，在一定程度上减少了对现有硬件体系结构的修改；但是，由于内存中数据均以明文状态存在，H-SVM无法保护物理攻击所带来的隐私窃取等安全问题。

2.2　扩展CIP表

HyperWall[3]提出了一种称为虚拟机监控器安全的虚拟化技术（VMM-secure virtualization）。为了在虚拟机监控器被排除在可信基之外的情况下保证用户隐私数据的安全存储，引入了“隐私性和完整性保护表”（Confidentiality and Integrity Protection Table，CIP）的机制，通过修改MMU（Memory Management Unit）、IOMMU、TLB（Translation Look-aside Buffers），以及增加部分CPU指令等硬件支持，使得CIP能够安全地保存和更新虚拟机监控器和DMA（Dynamic Memory Access）对于用户虚拟机内存的访问控制范围和权限，从而保证隐私保护表的实时性。

在系统启动时，内存管理器会将物理内存尾端的一段空间分配给CIP表，同时HyperWall会生成一段用于加密和哈希的秘钥，存放在相应内存区域，这些内存区域都被标记为只能被硬件访问。当用户请求启动虚拟机的时候，需要同时提供一个需要保护的内存列表（pre-CIP）；在虚拟机启动的时候，虚拟机监控器可以自由地映射虚拟机物理地址（Guest Physical Memory）和主机的物理地址（Machine Memory），生成对应的P2M表；HyperWall结合pre-CIP表对虚拟机监控器的P2M表进行遍历，将用户指定的需要保护的内存页在CIP表中标记为相应的状态（如VMM可访问、DMA可访问、VMM不可访问等），同时防止在CIP表中已经被赋值的页被VMM映射到其它的虚拟机；当虚拟机监控器要访问用户虚拟机相应内存页时会发生TLB未命中的情况，此时HyperWall会检查该页在CIP表中是否经过授权，防止虚拟机监控器偷窥用户隐私数据。另外，如果用户虚拟机产生中断，所有的通用寄存器都会被加密并且他们的哈希值会被储存在安全寄存器中，以防止数据的泄露和篡改。

除此之外，HyperWall还为用户提供了远程验证，可信证据收集等功能，并对磁盘IO和网络IO进行了加解密的处理，进一步防止了用户隐私的泄露。

H-SVM与HyperWall都需要大量的修改客户操作系

统以及VMM。例如，H-SVM需要VMM处理复杂的虚拟机交互，并需要集中修改客户操作系统和VMM的数据共享。而且H-SVM不保护外部设备的数据，而是保护客户VM本身，以确保其I/O数据。 HyperWall还要求客户操作系统来决定哪些内存页被保护，这是一个复杂的任务，要求程序员对不同操作系统的内存使用有深刻的理解。此外，HyperWall还忽略了VMM和客户操作系统之间复杂的数据交互。

3 基于安全处理器的保护技术

3.1 安全处理器

在过去的十年，安全处理器已经被广泛研究[5-10]。在这些研究中，主要有以下两个关键技术：“基于计数器的地址独立seed加密”技术（Address Independent Seed Encryption,AISE）和“Bonsai Merkle哈希树”技术（Bonsai Merkle Tree,BMT）。

3.1.1 AISE技术：保护数据隐私性

AISE基于数据加密的：Rogers等提出的“基于计数器的地址独立seed加密（AISE）”[9]内存加密。处理器并不直接加密/解密数据块（即高速缓存块），而是加密/解密数据块的seed，然后将其与明文数据块进行异或生成密文。同样，明文通过异或产生密文异或具有相同垫。seed是由三部分组成：LPID，计数器，地址偏移量。LPID（Logical Page ID）对每一个物理内存页是独一无二的，其值在初始化时分配。每个缓存块计数器和页偏移、LPID和偏移，确保seed空间的唯一性。一个数据块的计数器递增每一个数据块写回内存的时间，以确保seed的时间唯一性。一旦计数器溢出，相应的页面被分配一个新的LPID和重新加密。因此，攻击者不能重用seed进行重放攻击。所有LPID和计数器保存在主内存中，可以对于一个给定的物理地址使用简单的索引。同时引入了一个独立的缓存，名为计数器缓存，用于优化LPID和计数器。AISE加密开销非常低SPEC2K的额外负载仅为1.6%[9]。原因有以下三点：第一，加密只需要在数据高速缓存未命中的情况下进行；第二，对seed加密解密可与内存负载/存储并行；第三，计数器缓存的命中率是非常高。

3.1.2 BMT技术：保护数据完整性

Bonsai Merkle Tree的BMT（BMT）[9]被用来保护数据的完整性。当从内存中加载数据至高速缓存时，CPU获取并比较相应的哈希值。BMT的根节点和中间节点都保存在芯片内部，进一步加速比较过程。在BMT中，只有计数器和LPID的内存区域是由哈希树保护，其基本思路是使用计数器作为数据的版本号，因此只要保护计数器即可确保数据的完整性。当一个内存页被换出到磁盘，此页面的根哈希值被保存在一个受保护的内存区域，同样也通过BMT保护。BMT可以比Merkle哈希树（MHT）性能更好：AISE+BMT的开销仅为1.8[9]。

3.2 HyperCoffer

HyperCoffer[11]系统利用安全处理器的特性，为VM提供数据隐私性和完整性的保护。由于VM在内存中的数据均为密文，因此可以保留VMM原有的内存管理机制，尽可能减少对VMM的修改。

HyperCoffer的设计包括三个部分：首先，对内存虚拟化进行解耦，使得内存管理不依赖于对客户虚拟机内存的访问；其次，对CPU虚拟化进行修改，对不同的陷入情况予以不同的处理，由硬件直接提供必要的信息；再次，对I/O虚拟化进行控制，由客户虚拟机对I/O控制信息与数据进行去保护，从而在不修改硬件设备的前提下，实现与硬件的交互，同时保持尽可能多的数据处于保护状态。

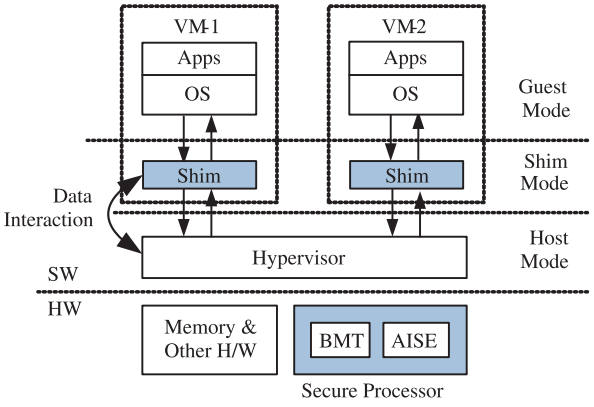

图1 HyperCoffer架构图

为了保持对客户OS的兼容性，HyperCoffer引入了VM-Shim机制。如图1所示，处理器在传统的host模式和guest模式之外，新增了一个shim模式；当状态在host模式与guest模式之间进行切换时，都先要经过shim模式，执行一段由处理器生产商提供的代码。这段shim代码用于在VMM与VM之间传递必要的交互数据，包括下陷指令、页表项值、I/O数据等，从而使得VM不需要经过修改即可在HyperCoffer上运行。通过一层实现简单的软件层，HyperCoffer修复了安全处理器与虚拟机监控器之间的语义鸿沟，很好的解决了传统安全处理器需要大幅修改上层软件的问题。

HyperCoffer中，安全可信基（Trusted Computing Base,TCB）仅包含安全处理器与VM-Shim软件层，其他的软件与硬件组件均为不可信，从而进一步缩小了TCB。其中，Shim的数据和VM的数据均由安全处理器保护。

HyperCoffer系统结合了安全处理器与硬件虚拟化，通过软硬件协同的方法，在不改变现有客户操作系统的情况下，对客户虚拟机进行系统级保护，从而保证用户数据的隐私性和完整性，并可抵御来自云计算平台内部的物理攻击。同时，HyperCoffer保持了云计算平台对操作系统的兼容性，同时只需要对现有的VMM进行很少的修改，从而提高了可部署性。

4　结论与展望

在云计算环境中，通过面向安全增强技术的体系结构支撑来提高云平台的整体安全性与租户数据的隐私性，并减低相应性能开销，可从如下三个方面开展研究：

（1）面向云可信执行环境构建的硬件支撑。研究从硬件层面提供对包括对虚拟机、云服务以及云用户的多层次与可配置安全隔离与信任链构建的支持，探索多种云服务模式下如何从体系结构层面防止包括时间、能耗、缓存等多种隐形流的信息泄露，并研究相应的软硬件协同，在保证云平台的安全性的同时减少性能开销。

（2）面向云计算的故障与安全攻击的监控、诊断与恢复的硬件支持。研究面向大规模云平台的片上、内存、网络与外存等故障与安全攻击进行动态监控的硬件扩展，提供对故障与安全攻击的跨结点与跨应用的故障与攻击现场回溯的硬件支撑机制，以支持对故障与安全攻击的快速定位与诊断与在线恢复。

（3）面向租客数据安全与隐私性保障的硬件支持。研究相关硬件支持以提供租户对数据的生命周期进行安全与可信管理与确信，提供对包括平台管理员在内的攻击者对租户数据的滥用与泄露进行检测、举证与防止的硬件机制，实现平台管理硬件层面的强制性数据安全。

参 考 文 献

[1] Common vulnerabilities and exposures [EB/OL]. [2012-10]. http://cve.mitre.org/.

[2] Jin S, Ahn J, Cha S, et al. Architectural support for secure virtualization under a vulnerable VMM [C] // Proceedings in the 44th Annual IEEE/ACM International Symposium on Microarchitecture, 2011.

[3] Szefer J, Lee R. Architectural support for VMM-secure virtualization [C] // Proceedings in the Conference of Architectural Support for Programming Languages and Operating Systems, 2012.

[4] Chhabra S, Rogers B, Solihin Y, et al. SecureME : a hardware-software approach to full system security [C] // Proceedings of the international conference on Supercomputing , 2011.

[5] Lie D, Thekkath C, Mitchell M, et al. Architectural support for copy and tamper resistant software [C] // Proceedings of the International Conference on Architectural Support for Programming Languages and Operating Systems, 2000: 168-177.

[6] Lie D, Thekkath C, Horowitz M. Implementing an untrusted operating system on trusted hardware [C] // Proceedings of International Conference on the ACM Symposium on Operating Systems Principles, 2003: 178-192.

[7] Suh G, Clarke D, Gassend B, et al. AEGIS: architecture for tamper-evident and tamper-resistant processing [C] // Proceedings of Supercomputing, 2003.

[8] Champagne D, Lee R. Scalable architectural support for trusted software [C] // Proceedings of the IEEE International Symposium on High Performance Computer Architecture, 2010: 1-12.

[9] Rogers B, Chhabra S, Prvulovic M, et al. Using address independent seed encryption and bonsai merkle trees to make secure processors os- and performance-friendly [C] // Proceedings in the Annual IEEE/ACM International Symposium on Microarchitecture, 2007: 183-196.

[10] Shi W, Lee H-H S. Authentication control point and its implications for secure processor design [C] // Proceedings in the Annual IEEE/ACM International Symposium on Microarchitecture, 2006: 103-112.

[11] Xia Y, Liu Y, Chen H. Architecture support for guest-transparent VM protection from untrusted hypervisor and physical attacks [C] // Proceedings of the IEEE International Symposium on High Performance Computer Architecture, 2013.

[12] Zhang F, Chen J, Chen H, et al. CloudVisor : retrofitting protection of virtual machines in multi-tenant cloud with nested virtualization [C] // Proceedings of International Conference on the ACM Symposium on Operating Systems Principles, 2011: 203-216.