虚拟化软件栈安全研究（计算机学报-中科院信工所-朱民）

至彤

2020 年 7 月 11 日

1 摘要

在虚拟化软件栈，虚拟机监控器具有最高权限和较小的可信计算基，故而能为虚拟化系统提供安全监控和保护．但同时也引入了新的软件层，增加了脆弱性，增大了攻击面．另外，多租户模式以及软硬件平台资源共享，更加剧了新软件栈的安全威胁。

* 分析虚拟化软件栈的安全威胁、攻击方式和威胁机理

* 比较了国内外相关安全方案和技术，并指出了当前仍然存在的安全问题．

* 对未来的研究方向进行了探讨和分析，给出了虚拟化软件栈的安全增强方案。

2 引言

* 虚拟化扩增了传统服务器的软件栈．软件栈越大、越复杂，攻击面和脆弱性就越多，安全性则更难以保障。
* 虚拟化技术提供的隔离性并不强
* 软件漏洞，攻击者利用侧信道攻击也可窃取其它虚拟机的敏感数据
* 当前虚拟化的研究主要集中在对Hypervisor的保护、对虚拟机的隔离以及对VM的内部系统、应用的保护，甚至将虚拟化从可信计算基中剔除，以此来增强虚拟化软件栈的安全．

3 云计算和虚拟化

Hypervisor的实现方式：

* 独立模式-Vmware ESXi
* 宿主模式- Vmware Workstation
* 混合模式（与独立模式类似，但IO设备仿真由特权虚拟机处理）-Xen
* Kvm与宿主机为同一层次，需要硬件支持（如Intel VT技术或者AMD V技术)负责cpu虚拟化+内存虚拟化，同时使用QEMU模拟IO设备（网卡，磁盘等）
* 全虚拟化
* 半虚拟化
* 硬件辅助的虚拟化：全虚拟化的硬件实现-IntelVT，AMD-V，ARM VE

4 虚拟化软件栈安全威胁

* 数据泄露和丢失：DMA攻击、侧信道攻击、虚拟机跨域访问
* 控制流截获以及后门，rootkit：代码完整性，控制流完整性，影子备份
* 拒绝服务攻击：资源监控，吞吐量限制
* 虚拟机镜像威胁：完整性验证
* 运行时代码，数据篡改：缓冲区溢出，库函数映射
* 权限提升：虚拟机逃逸
* 不可信的云内部人员
* DMA攻击
* 多重映射和虚拟机跨域访问：DMA攻击、 VLAN跳跃攻击和CACHE变更
* 跨虚拟机的cache攻击：基于cache的侧信道攻击与隐蔽信道攻击
* 快照内存转存威胁：内部攻击
* 物理攻击和线路窃听：内部攻击
* GPT：GVA-¿GPA
* EPT/影子页表:GPA-¿HPA

5 虚拟化软件栈安全防御

5.1 基于Hypervisor的虚拟机安全保护

5.1.1 GOS完整性防护：

* 传统os防护：可信启动（可信计算）与运行时保护（恶意代码嵌入和rootkits攻击-静动态数据完整性或控制流图的完整性）
* 定期监控控制流图完整性
* 透明地将已验证的虚拟机内核和可加载模块的代码拷贝到一个隔离的物理内存中作备份，指令执行时与备份对比发现rootkits。
* 保护内核数据结构和钩子：页保护，钩子函数访问限制，钩子函数写操作验证
* LKM隔离机制：利用轻量级Hypervisor提供的内存虚拟化功能对内核/静态模块以及动态扩展模块进行地址空间隔离,使得不同的模块位于不同的地址空间。不同地址空间互相交互信息时需要经过严格的仲裁机制。

5.1.2 GOS完整性防护

* 传统os防护：可信启动（可信计算）与运行时保护（恶意代码嵌入和rootkits攻击-静动态数据完整性或控制流图的完整性）
* 定期监控控制流图完整性
* 透明地将已验证的虚拟机内核和可加载模块的代码拷贝到一个隔离的物理内存中作备份，指令执行时与备份对比发现rootkits。
* 保护内核数据结构和钩子：页保护，钩子函数访问限制，钩子函数写操作验证
* LKM隔离机制：利用轻量级Hypervisor提供的内存虚拟化功能对内核/静态模块以及动态扩展模块进行地址空间隔离,使得不同的模块位于不同的地址空间。不同地址空间互相交互信息时需要经过严格的仲裁机制。

5.1.3 防护恶意GuestOS对进程的攻击

* 影子页表：对同一进程建立两个不同的地址空间，对应于系统的内核态和用户，而当内核态访问用户态的数据时，通过截获缺页错误，可以对数据的机密性进行保护
* 隔离的执行环境：是将内核和进程的地址空间进行隔
* 访问控制：由程序自身设置进程地址空间的访问权限，然后在Hypervisor的监控下系统内核按照既定的安全策略进行内存访问.
* OverShadow：双影子页表分别展现密文数据和明文数据，shim机制保证进程和GOS之间切换的安全。
* InkTag：利用超级调用在切换的过程中对HAP的上下文进行保护，并对内存页进行完整性检查和机密性保护，但需要对GOS进行更改，同时需要对HAP进行重新编写，使其支持超级系统调用

* TrustVisor: 用硬件虚拟化的特性和TPM为进程和内核提供一个隔离的运行环境。

* AppSec：AppSec能够在运行时通过存储在Hypervisor中的哈希值对共享动态链接程序进行验证，保证共享动态链接库不被篡改。该方案在运行时截获应用程序的系统调用，通过分析系统调用的参数，追踪受保护的信息。而当GOS内核对受保护内存页进行访存时，AppSec查看GOS访存地址是否在应用进程给予的访问范围之内。

5.1.4 虚拟机自省VMI

* 对GOS内部行为进行监控与分析

* 在GOS外部将监控数据进行语义还原

* 在GOS内部根据自身的内部信息提供结构数据

* 根据硬件结构的语义信息从客户虚拟机中获取所需数据

* LibVMI：提供了一套桥接GOS和Hypervisor语义鸿沟的通用库，获取GOS内部信息。（基于LibVMI的入侵检测系统）

* VMI-PL：对虚拟机的内存数据、信息流、寄存器以及事件进行监控和拦截，以系统库的形式为应用提供数据、事件和数据流三类探针。

* RTKDSM：两种模式：通用VMI模式和监控模式。通用VMI模式由一个专用虚拟机利用VMI原理对虚拟机内部的数据进行提取，而监控模式是利用hypervisor和EPT页表

* 项的权限机制保证数据在发生更改后能够及时地被VMI机制的。

5.2 Hypervisor及特权域

虚拟化TCB（减少攻击面和代码）的减小和虚拟化自身完整性保护

5.2.1 减少攻击面

减少Hypervisor代码量和改变虚拟化层的结构以减少GOS与Hypervisor的交互。 Xen虚拟化架构特权域Dom0有两个功能：设备模拟和虚拟机管理，Dom0只能访问到DomU的密文信息，反之，当DomU访问自身信息时，则首先会对密文进行完整性验证，然后才进行数据访问，防止代码篡改和嵌入。对于外映射内存则由Hypervisor强制Dom0释放相应页面。 TrustOSV：虚拟机初始状态远程验证，资源与分配机制（防止跨域访问与虚拟机cache攻击） NOVA：微内核思想，Hypervisor拆分成Micro-Hypervisor(CPU root模式内核层)，根分区管理器（CPU root模式用户层）VMM（独立用户VMM进程）设备驱动（以进程的形式运行在用户空间），每个模块依据权限机制拥有最小特权。

5.3 Hypervisor完整性

* 启动安全性：可信启动

* 运行时安全：保护运行时完整性：HyperCheck

5.4 不安全虚拟化环境下的安全防护

5.4.1 基于隔离机制的防护

威胁：同一物理机的虚拟机隔离性-¿内存去重机制-¿客户虚拟机的数据泄露，跨域访问，控制流截获 CloudVisor:机制与策略分离，嵌套虚拟化（将决策权下降给底层微小的嵌套Hypervisor），页属主机制，overshadow页加密机制，IOMMU页错误处理DMA访问

5.4.2 基于加密机制的防护

对CPU进行加/解密安全扩展，使其能够处理内存中的加密数据。为云虚拟化环境提供由软件到硬件的可信链路。

5.4.3 基于访问控制的防护

利用权限预设定机制来保护物理页不被Hypervisor和DMA恶意访问.

5.4.4 侧信道攻击和隐蔽信道攻击的防护

对于虚拟机共享cache的测信道攻击，破坏攻击条件。