云环境中基于cache负载实时定噪的同驻分析方法

何佩聪 黄汝维 陈宁江 赵搏文 刘 洋
(广西大学计算机与电子信息学院 南宁 530004)

摘 要 云计算具有使用便捷、可按需定制服务、优化资源利用等特点,成为提供外包服务的主要计算模式。云环境中的虚拟机侧通道攻击是云计算的主要潜在威胁之一,同驻是云环境中侧通道攻击的前提。针对如何在多租户云环境下进行同驻检测,提出基于链式结构的Prime-Probe测量cache负载方法MCLPPLS和针对云环境噪声复杂多变问题的实时噪声分析机制RTNAM。结合MCLPPLS与RTNAM提出一种新型的同驻检测分析方法。实验表明,该方法能减少突发噪声对同驻检测的干扰,有较高的同驻检测正确率及较低的同驻检测时耗,表现出良好的性能。

关键词 云计算,侧通道攻击,同驻检测
中图法分类号 TP393 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2017.05.019

Co-residency Detection Scheme Based on Cache Load and Real Time Noise Ascertainment in Cloud

HE Pei-cong HUANG Ru-wei CHEN Ning-jiang ZHAO Bo-wen LIU Yang
(School of Computer and Electronic Information, Guangxi University, Nanning 530004, China)

Abstract Cloud computing has the advantages of convenient use, designing customized service on need base, optimizing resource utilization etc. It has become the main computing model for outsourcing services. The side channel attack of virtual machines in the cloud environment is one of the main potential threats of cloud computing, and the co-residency is the premise of the side channel attack in the cloud environment. In view of how to carry out the co-residency detection in multi tenant cloud environment, this paper presented the measurement of cache load by Prime-Probe with linked struct (MCLPPLS) and real time noise ascertainment mechanism(RTNAM). Based on MCLPPLS and RTNAM, we proposed a new method for the analysis of the co-residency detection. The experimental results show that the method can reduce the interference of the burst noise to the co-residency detection, and has higher true detection rate and lower detection time, which shows good performance.

Keywords Cloud computing, Side channel attacks, Co-residency detection

1 引言

云计算[1]作为一种新兴计算模式,由于具有可扩展性和高可用性,被广泛用于外包服务。云计算的可扩展性和高可用性都源于虚拟资源池。虚拟资源池对服务器集群的资源进行统一管理、分配、调度,实现动态扩充、按需定制等服务,并具有对用户透明的优点。各大厂商非常青睐这种灵活方便的模式,并争相推出各自的云平台,如Google的GCE、亚马逊的EC2、微软的Azure、开源的Openstack等。然而,正是由于虚拟资源池共享物理硬件,因此只使用虚拟化隔离的方式会使其存在严重的安全隐患——侧通道攻击。

侧通道攻击是一种恶意用户利用物理资源获取其他用户状态或隐私信息的攻击方法。恶意用户可以通过探测分析本地共享资源信息,推测出与其共享物理资源的虚拟机状态,甚至获取隐私信息。侧通道攻击已经成为云计算的潜在安全威胁之一[2-4]。在云环境中的侧通道攻击与传统的侧通道攻击有所不同,其主要分为两步:1)检测同驻;2)隐私信息提取[2]。在云环境中,如果两台虚拟机运行在同一台物理服务器上,则称为同驻。检测同驻是在不知道云服务商的虚拟机分配机制的情况下,检测目标主机与攻击主机是否处于同一物理云服务器上,从而具备共享物理资源条件;隐私信息提取则是基于共享物理硬件进行状态分析及信息窃取。由此可见,同驻是云环境中发起侧通道攻击的前提。因此,已有学者通过共享资源测试来判断同驻情况,但在云环境中存在以下特点:1)共

到稿日期:2016-11-18 返修日期:2017-01-13 本文受国家自然科学基金项目(61640203,61363003),广西自然科学基金项目(2016GXNSFAA380115),国家科技支撑计划课题(2015BAH55F02),广西大学科研基金项目(XBZ120257,XJZ151321)资助。
何佩聪(1988—),男,硕士生,主要研究方向为云计算、云安全,E-mail:hpc9527@163.com;黄汝维(1978—),女,博士,副教授,主要研究方向为服务计算、云安全、同态加密,E-mail:ruweih@126.com(通信作者);陈宁江(1976—),男,博士,教授,主要研究方向为软件工程、网络分布式计算、中间件技术;赵搏文(1992—),男,硕士生,主要研究方向为信息安全与密码学;刘 洋(1992—),男,硕士生,主要研究方向为人工智能、图像处理与模式识别。

享资源使用情况复杂多变且对用户透明;2)攻击者只是共享资源的使用者而非拥有者,其掌握信息的能力与多少受到限制。这些特点使得同驻检测的难度大幅提升,同时严重影响着同驻检测的正确性。鉴于此,本文提出一种具有结构链表的 Prime-Probe 测量 cache 负载方法 MCLPPLS(Measurement of Cache Load by Prime-Probe with Linked Struct)与实时定噪分析机制 RTNAM(Real Time Noise Ascertainment Mechanism)的同驻检测方法,该方法能更准确地采集 cache 负载信息并抵抗噪声干扰,提高同驻检测的正确率。

本文第 2 节介绍了传统侧通道攻击和云环境下的侧通道攻击的相关研究;第 3 节给出了攻击模型以及实现思路;第 4 节详细描述了 MCLPPLS 和 RTNAM 的思想及实现,并给出了基于 MCLPPLS 和 RTNAM 的同驻检测方案;第 5 节通过大量实验验证了本文同驻检测算法的正确性及采用 RTNAM 分析的优势;最后对研究工作进行了总结和展望。

2 相关研究

1996 年,Kocher[5]首先提出基于时间的侧通道攻击,针对 Diffie-Hellman 或 RSA 等特定的基于有限群的离散对数加密算法运算过程,可通过测量加密时间信息破解密钥。文献[6]指出多线程环境下 cache 共享机制的潜在威胁,并通过两个进程构建隐蔽信道,从而窃取 RSA 密钥。文献[7]基于多线程对指令高速缓存的共享,从理论上阐述通过间谍程序监控指令高速缓存,并通过分析指令执行时间,获取加密过程中所用到的加密指令,从而窃取 RSA 密钥信息。文献[8]实现了间谍程序的安装部署,并通过矢量量化和隐马尔可夫模型提高指令高速缓存数据分析技术。文献[9]基于分支预测技术,提出分支预测分析攻击方法,通过不断检测分支目标地址缓存访问时间来判断加密操作,从而实现 RSA 密钥的提取。文献[10]指出共享物理硬件导致的系统威胁,并提出一种字段共享方法攻击(shared FU attack),即针对模乘(Modular multiplication)与平方(squaring)操作运行时间不同且该运算单元被多个线程共享的攻击,该攻击通过窃取 RSA 密钥证明存在威胁。文献[11]通过监控程序利用完全公平调度器(Completely Fair Scheduler)技术观察 cache 的命中情况,并根据 AES 的加密特点分析得出 AES 密钥。文献[12]鉴于 AES 的内存访问模式极易受到密码分析攻击的特点,首次描述 Prime+Probe 与 Evict+Time 探测方法,针对 AES 轮函数特性提出第一轮攻击与第二轮攻击方法,实现 AES 密钥窃取。这些方法都是基于已同驻的情况,利用共享资源的访问时间信息窃取密钥。但是,在云环境中,虚拟机被随机分配到物理服务器集群中,共享物理硬件的条件被破坏,这使得云环境中的虚拟机可以避免遭受侧通道攻击。

然而,随着 Ristenpart[2]在 2006 年首次提出云环境中的虚拟机同驻安全问题,云环境下的虚拟机也将面临侧通道攻击的威胁。因为对于云环境中的攻击者来说,如果能确定目标虚拟机与己方虚拟机同驻,有共享的物理资源,即可根据从 cache 中窃取私密信息的思想进行侧通道攻击。文献[13-14]分别针对云环境中的同驻安全问题提出末级 cache 攻击和基于内存物理地址的 Prime+Probe 攻击,证明云环境下虚拟机已然开始面临侧通道攻击的威胁。

针对云计算环境下的同驻检测,目前已有一些研究成果。文献[2]利用 hping,nmap 以及 wget 网络分析工具探测 Xen 中 Dom0 主机 IP 的一致性,从而判断是否同驻,但是可以通过关闭某些特定端口避免这类同驻检测。文献[3]利用 Prime-Probe 同驻检测方法验证用户是否独享物理主机,由于其目的是确认独享物理主机,因此不需考虑噪声干扰。文献[15]通过事先采集大量负载数据构建正态云模型,来降低噪声干扰。文献[16]为避免硬件与软件特征的干扰,设计了基于三次样条插值的预处理器和基于线性回归模型的 cache 负载预测器,从而可以更准确地进行负载特征的提取与分析。此类方法虽然考虑噪声干扰,但都需事先进行数据采集与预估,不仅十分耗时,而且采集的完整性与预估的准确性将直接影响同驻检测的正确性。文献[17]利用云模型对同驻虚拟机侧通道攻击威胁进行评定,以判断云用户的威胁级别。文献[18]通过介绍云环境中同驻虚拟机所面临的一系列威胁,体现同驻安全问题的重要性。本文针对上述问题,提出一种基于 MCLPPLS 与 RTNAM 的同驻检测方法,通过 MCLPPLS 提高 cache 负载采样的准确性,通过 RTNAM 机制在不需要前期采样的情况下对噪声进行实时分析,消除噪声对同驻判断的影响,从而缩短同驻检测的总体耗时并提升同驻判断的正确率。

3 检测模型

为方便检测模型描述,作以下假设。

假设 1 目标虚拟机在云环境中的位置如图 1 所示。攻击者对此并不知情,但清楚目标虚拟机对外提供何种服务,并能请求该种服务。

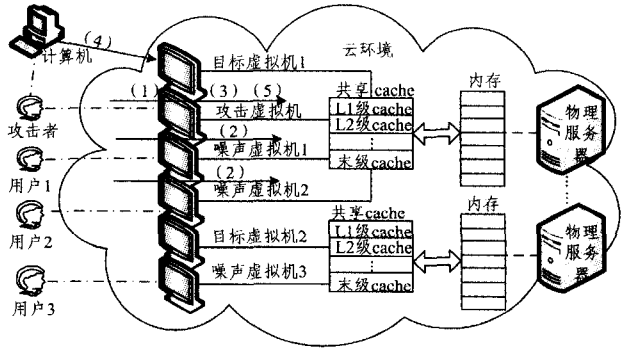


图 1 同驻检测模型

假设 2 目标虚拟机对外公开提供加密服务,所有联网计算机都可向目标虚拟机发起服务请求并获得服务响应。

同驻检测的依据是计算机对不同存储设备的访问时间差异很大,cache 为提高命中率会根据最近访问情况更新内容。因此可进行有针对性的数据访问,并通过访问时耗判断是否对 cache 内容造成修改,从而根据是否共享 cache 来检测同驻。

如图 1 所示,攻击者对云环境中向外提供服务的目标虚

拟机 1 进行同驻检测的具体过程如下。

(1)开辟一块大于末级 cache 大小的数据空间，并将其通过读操作写入到共享 cache。(2)等待在同一物理服务器中的噪声虚拟机 1，2 运行一段时间，以便替换 cache 中的内容。(3)再次访问这组数据，得到时间 $T1$，并再次把数据写入共享 cache。(4)通过另一台计算机对目标虚拟机 1 不断发起服务请求，使其高负载运行。(5)再次访问这组数据，得到时间 $T2$。(6)通过对比 $T1$ 与 $T2$ 判断目标虚拟机 1 与攻击虚拟机是否存在共享的 cache 空间，从而达到同驻检测的目的。

在此过程中，如果攻击虚拟机与目标虚拟机同驻，则在不断发起服务请求的过程中，目标主机将大量替换 cache 中的内容，从而使 $T2$ 大幅增加；如不同驻，则无明显时间变化。但是，如下情况也将产生同驻时耗特点：假设此时攻击者同驻检测对象为目标虚拟机 2，在(4)中，攻击虚拟机对目标虚拟机 2 不断发起服务请求，此时噪声虚拟机 1 或 2 突然高负载运行，导致(5)中时间 $T2$ 大大增加，从而得到同驻结论。然而我们可以清楚地从图 1 中得知，攻击者的虚拟机与目标虚拟机 2 并不同驻。为解决此类问题，本文提出基于 MCLPPLS 与 RTNAM 的同驻检测方法。

4　基于 MCLPPLS 与 RTNAM 的同驻检测方法

4.1　基于结构链表的 Prime-Probe 方法 MCLPPLS

文献[12]提出两种 cache 负载探测方法：Evict＋Time (ET)和 Prime＋Probe(PP)。ET 方法是在同一段明文两次加密的过程中进行内存访问，并比较两次加密的时间差。但是这种方法存在缺点：时间差对加密算法的选取依赖性很强。而 PP 算法则是在两次读取同一段大于 cache 大小的数据之间进行一次明文加密，并比较两次读取的时间差。鉴于 ET 算法对操作的敏感性，本文采用 PP 算法作为负载采集的基准方法，并对其进行改进，通过开辟一组链式结构体数据实现 pointer-chasing[12]技术，以减少因硬件预取技术而导致的时耗，并通过嵌套使用汇编指令 lfence 和 rdtsc 保证严格顺序执行的同时提高采样精度。

开辟一块结构体链表数据空间，如图 2 所示。该空间比末级 cache 缓存稍大，以确保 cache 缓存空间能完全被该块数据重写，并通过链式存储方式避免硬件预取技术的影响。

```
typedef struct Node{
    data;//用来充填 cache 的大量数据
    struct Node * link;//指向下一块数据的指
针
}Node, * linklist;
```

图 2　MCLPPLS 中数据结构体定义图

MCLPPLS 主要包括以下 3 个步骤。

(1)Prime：开辟一块大于 cache 大小的结构体链表数据，并使用带 lfence 的嵌套汇编程序对该数据进行运算，使其写入到 cache 中，为之后的 Probe 初始化环境。

(2)Operate：在此过程中，可空闲等待其他同驻虚拟机共享 cache 访问并替换其中数据，也可不断对目标虚拟机进行服务请求，采取哪种操作取决于步骤(3)需要测量的数据。其中两种操作时间一致。

(3)Probe：再次对这段数据进行操作运算，并记录完成这段操作的耗时 t。每次 Probe 操作亦可被视为下次的 Prime，为再次 Probe 操作做好准备。

MCLPPLS 采集算法如算法 1 所示。

算法 1　MCLPPLS 采集算法

```
Input:T or AT, * linklist Data; //T,AT 为攻击操作指示符
      // * linklist Data 用来重写 cache 数据块首地址
Output:t[]//t[]表示哪类时长由输入的 T 或 AT 决定
long rdtsc( * linklist Data){
    //利用汇编指令 lfence、rdtsc 完成对 * linklist Data 数据块的
      运算操作后设置时间戳
  }
long start,end,t[];
if(T)
  start=rdtsc( * linklist Data);
  for i←0 to n−1 do
    wait(){
        //挂起等待
        }
    end=rdtsc( * linklist Data);
    t[i]=end−start;
    start=end;
  endfor
else
  start=rdtsc( * linklist Data);
for i←0 to n−1 do
    ap_service(){
            //不断向目标主机请求服务;
            }
    end=rdtsc( * linklist Data);
    t[i]=end−start;
    start=end;
  endfor
endif
return t[];
```

4.2　实时定噪分析机制 RTNAM

本文交替采集两类 cache 时间负载情况，一类为执行 wait 操作的计时数组 $t[]$，记为正常时耗 t；另一类为攻击时耗，记为 $at(attack\_time)$。为更直观体现当前噪声的波动情况，可连接 t 和 at 数组中的时耗点，得到时间段时耗曲线，并记为 T，同理得到 AT，并依次交替采集 T，AT 时耗 5 次，如图 3 所示。

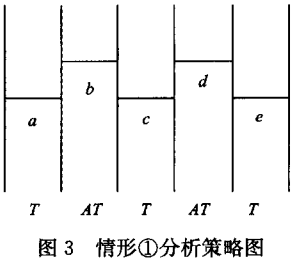


图 3　情形①分析策略图

为了方便描述同驻策略，引入以下定义说明。

定义 1 在一个时间段内相邻前后时间点差的绝对值的平均值和方差称为波动均值 $\overline{X}$ 和波动方差 σ_X。即第一个时间点与第二个时间点差的绝对值记为 X_1，第二个时间点与第三个时间点差的绝对值记为 X_2，以此类推，然后根据式(1)可计算波动均值 $\overline{X}$，再根据式(2)计算波动方差 σ_X。

$$\overline{X}=\frac{1}{n}\sum_{i=1}^{n}X_i \tag{1}$$

$$\sigma_X=\frac{1}{n}\sum_{i=1}^{n}(X_i-\overline{X})^2 \tag{2}$$

定义 2 一个时间段内的平均时耗称为时间段均值，即 n 个时耗点的平均值。

定义 3 对两个时间段样本进行比较，若两个时间段样本的时间段均值差与波动方差比均满足多维阈值 $\theta(x,y)$，则称这两个时间段样本相似，记作$\backsim$；否则不相似，记作$\neq$。阈值 θ 是一个多维数据，其中 x 为时间段均值差的上限，y 为波动方差比的上限，在波动方差比值中，数值大的为分子。

定义 4 对两个时间段样本 $T_1(t_{11},t_{12},\cdots,t_{1n})$和 $T_2(t_{21},t_{22},\cdots,t_{2n})$，如果 T_1 满足 $t_{11}-t_{21},t_{12}-t_{22},\cdots,t_{1n}-t_{2n}$均大于 3 倍的 T_2 的 $\overline{X}$，则称 T_1 远大于 T_2，记为 $T_1\gg T_2$。

如图 3 所示，对 5 段时间分别进行标号，其中 a,c,e 为正常时耗 T；b,d 为攻击时耗 AT，竖直高度为 cache 负载时间，水平宽度为采样时间点，且为方便分析描述，用直线代表波动曲线。选取 5 段时间分析方法的原因如下。情况 1：相邻两个 T 时段相似，这能反映这两段时耗采集过程中背景噪声一致；情况 2：3 个 T 时段相似，这能反映这 3 段时耗采集过程中背景噪声一致。在情况 1 中，并不能判断两段相邻 T 时段中的 AT 时段是否混杂噪声。而在情况 2 中，可对比两次 AT 时耗是否都远大于 T 时段，从而排除由于噪声干扰引起的单独一段 AT 时段远大于 T 时段的同驻错判。虽然也可能在两段 AT 中都出现噪声干扰，但要出现图 3 的情形则需满足：噪声在 T 结束时立刻产生，并在 AT 结束时立刻消失，然后在下次 T 结束时再次立刻产生，并于下次 AT 结束时立刻消失。在随机噪声中，这种噪声的出现是小概率的，故不影响判断分析，且可通过多次同驻判断解决这一问题。具体同驻检测策略如下。

比较 a,c,e 3 段时间的值，分以下 3 种情况：①$a\backsim c\backsim e$；②$a\backsim c\neq e$或者 $a\neq c\backsim e$；③其他。在①中，认为在 $a-e$ 这段时间中的噪声背景大致是相同的，因为正常时耗能反映当前攻击虚拟机所在服务器的 cache 替换活跃程度。如果噪声背景不同，则会导致 a,c,e 成为 3 段随机波动曲线而毫不相似。然后比较 b,d 与 c 的时耗情况。如果有 $b\gg c$ 且 $d\gg c$，则认为同驻，否则不同驻。因为在 $a-c$ 与 $c-d$ 两段相同方法采样区间中得到不同结果，说明 $b\gg c$ 或 $d\gg c$ 的情况并不由不断请求所导致，应为某些虚拟机突然负载增加所致。在②中，同理，认为相似时段内噪声背景相同，不相似时段内噪声背景发生了变化。故分别讨论。首先分析在相似区间内的攻击时耗是否远大于 c，然后判断非相似时段的时耗与另一段攻击时耗是否有相同的广义单调性。即剩下的一段 T 与 AT 有相同大小的时耗增幅或降幅，如图 4 所示，d,e 都上升了大约 Δ 大小的负载。如果有相同广义单调性，则认为同驻，否则不同驻。而③中，由于噪声背景不稳定，不具备分析条件，应该丢弃数据，重新检测。

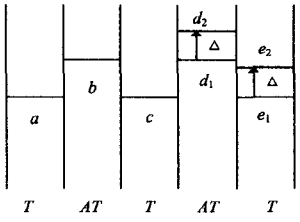


图 4 情形②分析策略图

RTNAM 算法如算法 2 所示。

算法 2 RTNAM 算法(使用图 3 标记)

```
Input:a;b;c;d;e//分别对应 5 个时间段的时耗点数组
Output:co-residency ;not ;renew //分别为同驻,不同驻
                                //重新采集
similar(t1[],t2[]){
  //判断两个数组是否相似。相似则返回 1,否则返回 0
}
overmore(t1[],t2[]){
  //判断 t1[]是否远大于 t2[]。t1[]远大于 t2[]则返回 1,否则返回 0
}
timesub(t1[],t2[]){
  //返回数组 t3[],t3 中的元素为 t1 与 t2 两数组中对应元素的差的
    绝对值
}
monotone(t1[],t2[]){
  //判断 t1[],t2[]是否具有广义单调性,若有广义单调则返回 1,否
    则返回 0
}
if (similar(a,c) && similar(c,e))
  if (overmore(b,c) && overmore(d,c))
    co-residency
  else
    not
  endif
elseif ((similar(a,c) && (!similar(c,e))) || (similar(c,e) &&
(!similar(a,c))))
  if(((similar(a,c))&&(overmore(b,c))&&
    (monotone(timesub(d,b),timesub(e,c))))
      co-residency
  elseif(((similar(c,e))&&(overmore(d,c))&&
    (monotone(timesub(a,c),timesub(b,d))))
      co-residency
  else
      not
  endif
else
  renew
endif
```

4.3 RTNAM 方法采集时段分析

在 RTNAM 方法中，选取相邻的奇数时间段中的 cache 负载进行同驻分析判断，即 3 段、5 段、7 段等。该方法的主要思想为：对比两段相邻时段负载，判断负载变化情况，并分析该种变化由何引起。需要通过两个相邻的 T 时段来判断背景噪声，即发起 AT 采样前的背景噪声与 AT 采样后的背景噪声推测出在 AT 采样过程中的噪声情况，所以必定以 T 开始且以 T 结束。

为分析时段数对 RTNAM 的影响，分别选取 3 段、5 段、7 段、9 段时间段样本进行同驻检测，每组实验进行 50 次同驻检测，分别进行 10 组实验，得到检测正确率及检测平均耗时，如图 5 和图 6 所示。

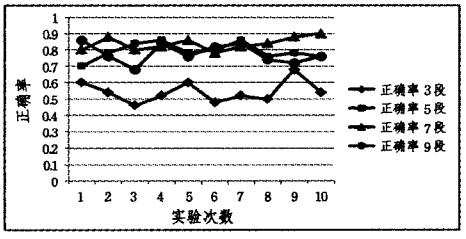


图 5　不同时段数在 RTNAM 中的正确率

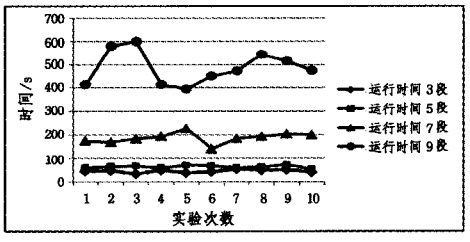


图 6　不同时段数在 RTNAM 中的时耗

在实验过程中，我们发现选取 3 段进行同驻检测的正确率最低，这是因为其并不能判断出在 AT 时段中的负载增加是否由噪声引起；其他 3 种判断方法都取得较高的正确率，但 9 段方法由于时间跨度长，采样中混入噪声情况复杂，正确率反而有所下降。在时耗方面，9 段方法由于其采集时间长，且噪声复杂导致不能满足相似条件，经常要丢弃重新采集数据，因此时耗很大，且有几次超过了 10 分钟。通过实验发现，大于 5 段选取方法具有较高正确率，但时段选取越大，时耗增加越大。若想提高正确率，应采取多次同驻判断方法，而非增加时段选取。观察可知，选取 5 段分析在正确率与时耗方面均有不错表现，故本文选取 5 段时耗数据进行同驻检测分析。

4.4 基于 MCLPPLS 与 RTNAM 的同驻检测方法的实现

同驻检测方法如下：每次先通过 MCLPPLS 采集 cache 负载数据，然后传递给 RTNAM 进行同驻判断，如未得出同驻判断结果，则重新进行采集分析判断，直至得出同驻结果。

算法 3　同驻检测算法

```
While(未得出判断){
    MCLPPLS 模块数据采集
    RTNAM 模块分析
}
```

5 实验与分析

由 10 台浪潮英信服务器 NF5280M3 组成基于 KVM 的虚拟化运行环境，其主要配置信息为：Xeon E5-2620 CPU，32GB 内存，1TB 硬盘，centos6.4 系统。虚拟机配置信息为：2VCPU，2GB 内存，15G 硬盘，Windows7 操作系统。虚拟机中均带有 eclipse，dev-c++等编程环境，攻击目标虚拟机提供 AES 文件加密服务。

5.1 同驻检测抗噪声实验

在 10 台服务器中设置若干台虚拟机，并在所有虚拟机中运行随机噪声函数，该函数对随机数进行模运算，根据余数情况随机触发噪声与噪声规模，通过设置充填 cache 数据，可分为 3 类负载噪声规模：弱噪声、中噪声、强噪声，其数据重复率分别为 75%，50%，25%。与本文进行对比实验的是文献[15]的同驻检测算法（简称文献方法）以及直接比较负载时间方法（简称直接对比法）。文献方法需要大量数据采集准备，通过聚类算法提取特征值，然后基于正态云模型进行判定。直接对比法则直接对比时耗判定。分别进行 10 组实验，10 组实验中的噪声虚拟机分别为 1，2，3，…，10 台。每组实验进行 50 次同驻判断，然后进行正确率对比。

实验结果如图 7 所示，本文方法能保证近 70%的正确率；文献方法在噪声虚拟机较少的情况下，有近 80%的正确率，但随着虚拟机台数增多，正确率明显下降；而直接对比法只在一台噪声虚拟机情况下体现 60%的正确率，随后大幅下降，甚至不足 20%。文献方法基于数字特征采集，当噪声增加且处于变噪环境中时，数字特征被大量随机噪声数据掩盖，且预估变得十分困难，数据采集是否完整将直接影响正确率，故当噪声规模增大时，正确率下降明显。直接对比法受噪声干扰十分严重，当噪声规模增大时，几乎不能进行同驻判断。本文方法基于对当前噪声的分析，当噪声数量规模增长至较稳定时，噪声突变被隐藏在噪声波动中，从而降低了噪声干扰，体现了良好的抗噪声性，总体效果稳定平稳。

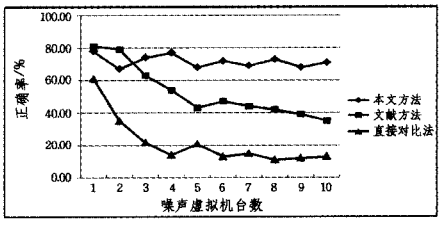


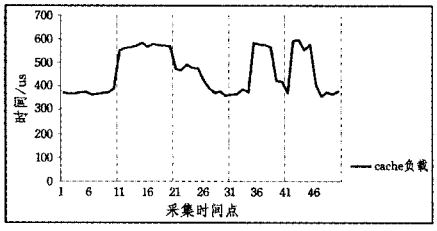
图 7　噪声规模与正确率对比

5.2 RTNAM 方法的优势分析

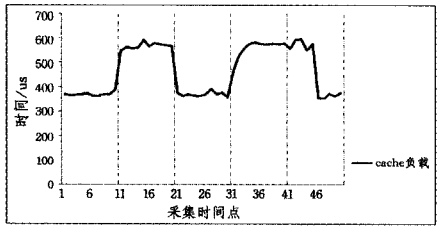
在上述实验中，RTNAM 方法有较高正确率。从实验数据中选取以下 3 种典型易误判情形（如图 8 所示）来分析说明 RTNAM 方法如何确保正确性，其中只有情形 3 为真实同驻。

图 8 中每一情形的 5 段时间区间与图 3 中的 A，AT 时段一一对应。在情形 1—情形 3 中，如果只观察前两段时耗，均能得出同驻的结论。如果观察前 3 段时耗，由于情形 1 中的 a，c 时段攻击都在进行 wait 操作，因此 a，c 反映的是攻击虚

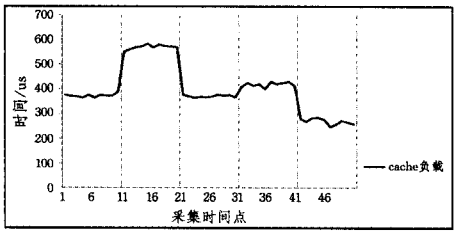
拟机所在服务器上的噪声背景情况，故可以得出情形 1 中的噪声背景产生变化，b 段的时耗增加可能是噪声所致，所以情形 1 并不能得出同驻结；情形 2，3 认为同驻。如果观察前 4 段时耗，情形 2 中 a，c 相似，b，d 时耗增加，被认为同驻；而情形 3 中，b，d 同为不断请求访问时耗，然而 b，d 表现出不同时耗情况，被认为非同驻。如果观察 5 段时耗，情形 2 中，c，e 时段不相似，可推出 $c-e$ 时段内噪声背景发生变化，比较 c，e 可知，e 相对于 c 有所增加，即在 $c-e$ 时段内产生正噪声，这个正噪声必然增大 d 的时耗负载，然而从图 8 可知，d 负载并无增加，故发现之前判断错误，应为不同驻；在情形 3 中，c，e 时段不相似，且发现在 $c-e$ 内产生负噪声，d 相对于 b 有一个相似的下降，故之前判断错误，应为同驻。对引起误判的数据集合的分析表明，采取 5 段时间分析能有效提高同驻判断的正确率，减少漏判、错判。



(a)情形 1



(b)情形 2



(c)情形 3

图 8 典型 cache 负载情况

这套策略的思想基于两次小概率事件同时发生的概率必然小于一次小概率事件发生的概率。在 $a-c$ 及 $c-e$ 中同时因噪声导致的误判断概率要远小于在 $a-c$ 中因噪声导致的误判断概率，因此只接受 $a-c$ 及 $c-e$ 同时得出的同驻判断能提高同驻检测正确性；并且通过 a，c，e 的相似比较确认两次同驻判断的噪声背景情况，通过 b，d 的相似比较判断在同驻检测时间段中的噪声突发情况，从而尽可能减少噪声干扰，并根据背景噪声情况采用不同的同驻检测策略，提高同驻检测的正确性与可靠性。

另外，可通过设置多维阈值 $\theta(x,y)$ 放宽相似判断标准来调节同驻分析时耗。x，y 的取值越小，相似标准越严格，但是过严的相似标准会增加数据采集时耗，并有同驻漏判的可能。

5.3 同驻检测用时分析

本实验主要记录从开始检测到得到结果所需的时间，如图 9 所示。文献方法由于需要事先采集大量数据，从而进行特征值提取，因此需要大量的前期准备时间，平均大概需要 2 分半到 3 分钟。直接对比法采集测量时间少，不需要进行数据分析判断，故耗时最少；而本文方法，只考虑当前噪声稳定情况，只有在不稳定时才会对数据进行丢弃后重新检测，故时间开销并不大。

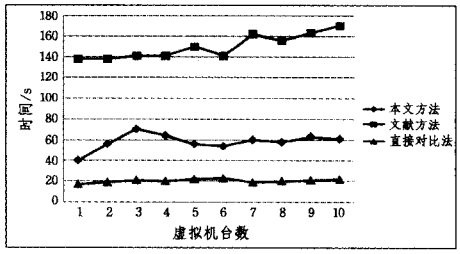


图 9 时耗对比

从 5.1 节的实验中发现，文献方法的正确率下降明显，疑为数据采集不完整所致，提升采样时间后，正确率有所回升。确保正确率在 55%以上的时耗对比如图 10 所示，这表明在复杂的噪声情况下，文献方法为保证正确率需要大量时间进行数据采集，且当环境发生变化时需重新采集数据，这对未知云中的同驻检测造成不便；而本文方法只基于当前噪声分析，在时耗上具备一定优势。

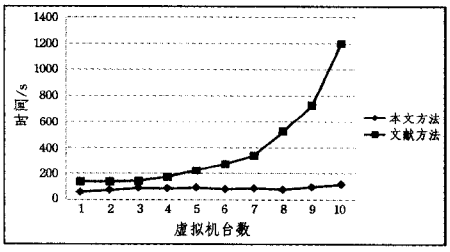


图 10 正确率在 55%以上的时耗对比

结束语 本文提出的 5 段时耗数据分析策略的灵感来自于差分曼彻斯特编码，在负载探测的同驻检测方法中，由于我们只关心 cache 负载的提升是否由不断发起服务请求导致，因此可通过多次对比 cache 负载变化分析原因，确定是否有共享 cache，从而判断同驻。通过 MCLPPLS 算法，提升对 cache 负载采集的精确度；RTNAM 算法通过分析多段 cache 时间负载判断当前噪声情况，降低噪声干扰。本文结合 MCLPPLS 与 RTNAM 的同驻检测方法已具备的良好特性，在接下来的工作中，可对 RTNAM 算法中的多维阈值 $\theta(x,y)$ 的取值进行进一步的研究，根据不同云环境情况进行特定的 x，y 的设置，使其能针对不同云环境进行更高效、更准确的云环境中虚拟机的同驻检测。

参 考 文 献

[1] CHEN K, ZHENG W M. Cloud Computing: System Instances and Current Research[J]. Journal of Software, 2009, 20(5): 1337-1348. (in Chinese)
陈康，郑纬民. 云计算：系统实例与研究现状[J]. 软件学报，2009，20(5)：1337-1348.

(下转第 115 页)

陈炜，路世昌，崔铁军. 基于 AHP 可拓综合方法的公路隧道安全等级判定研究[J]. 中国安全生产科学技术，2014，10(7)：158-163.

[12] CUI T J，MA Y D，BAI R C. Selection of Blast Scheme Based on Coupling of Genetic Algorithm and Artificial Neural Network [J]. China Safety Science Journal，2013，23(2)：64-68. (in Chinese)
崔铁军，马云东，白润才. 基于 ANN 耦合遗传算法的爆破方案选择方法[J]. 中国安全科学学报，2013，23(2)：64-68.

[13] HE F，LIU J，CUI T J. Selection of Blast Schemes Based on Both Structure Element Intuitive Fuzzy Set and Genetic Algorithm[J]. China Safety Science Journal，2013，23(12)：60-65. (in Chinese)
赫飞，刘剑，崔铁军. 基于结构元直接模糊集和 GA 算法的爆破方案选择[J]. 中国安全科学学报，2013，23(12)：60-65.

[14] CUI T J，MA Y D，BAI R C. Optimization of blast parameters using genetic algorithm optimized by artificial neural network [J]. Journal of Earthquake Engineering and Engineering Vibration，2014，34(2)：258-262. (in Chinese)
崔铁军，马云东，白润才. 基于神经网络优化遗传算法的爆破参数优化[J]. 地震工程与工程振动，2014，34(2)：258-262.

[15] LUO Y Q，XIA J B，CHEN T P. Network performance comprehensive evaluation model based on cloud model and entropy weight[J]. Journal of Chongqing University of Posts and Telecommunications(Natural Science Edition)，2009，21(6)：771-775. (in Chinese)
罗赟骞，夏靖波，陈天平. 基于云模型和熵权的网络性能综合评估模型[J]. 重庆邮电大学学报(自然科学版)，2009，21(6)：771-775.

[16] WANG P Z. Factor spaces and Factor Data-bases[J]. Journal of Liaoning Technical University(Natural Science)，2013，32(10)：1-8. (in Chinese)
汪培庄. 因素空间与因素库[J]. 辽宁工程技术大学学报(自然科学版)，2013，32(10)：1-8.

(上接第 110 页)

[2] RISTENPART T，TROMER E，SHACHAM H，et al. Hey，you，get off of my cloud：exploring information leakage in third-party compute clouds[C]// ACM Conference on Computer and Communications Securtty(CCS 2009). 2009：199-212.

[3] ZHANG Y，JUELS A，OPREA A，et al. HomeAlone：Co-residency Detection in the Cloud via Side-Channel Analysis[C]// Security and Privacy. IEEE，2011：313-328.

[4] ZHANG Y，JUELS A，REITER M K，et al. Cross-VM side channels and their use to extract private keys[C]// ACM Conference on Computer and Communications Security. 2012：305-316.

[5] KOCHER P C. Timing Attacks on Implementations of Diffie-Hellman，RSA，DSS，and Other Systems [C] // International Cryptology Conference on Advances in Cryptology. 1996：104-113.

[6] PERCIVAL C. Cache missing for fun and profit[J]. Proc of Bsdcan，2005.

[7] ACIIÇMEZ O. Yet another microarchitectural attack：exploiting I-cache[C]// Proceedings of the 2007 ACM Workshop on Computer Security Architecture. ACM，2007：11-18.

[8] ACIIÇMEZ O，BRUMLEY B B，GRABHER P. New results on instruction cache attacks [C] // International Conference on Cryptographic Hardware & Embedded Systems. 2010：110-124.

[9] ACIIÇMEZ O，ÇETINKAYA K，SEIFERT J P. On the Power of Simple Branch Prediction Analysis[C]// 2007 ACM Symposium on Information，Computer and Communications Security (ASIACCS'07). 2006：312-320.

[10] ACIIÇMEZ O，SEIFERT J P. Cheap Hardware Parallelism Implies Cheap Security[C]// Workshop on Fault Diagnosis and Tolerance in Cryptography，2007 (FDTC 2007). IEEE，2007：80-91.

[11] GULLASCH D，BANGERTER E，KRENN S. Cache games-bringing access-based cache attacks on AES to practice[C]// In 32nd IEEE Symposium on Security and Privacy. 2011：490-505.

[12] TROMER E，OSVIK D A，SHAMIR A. Efficient Cache Attacks on AES，and Countermeasures[J]. Journal of Cryptology，2010，23(1)：37-71.

[13] YOUNIS Y A，KIFAYAT K，SHI Q，et al. A New Prime and Probe Cache Side-Channel Attack for Cloud Computing[C]// IEEE International Conference on Dependable，Autonomic and Secure Computing. 2015：1718-1724.

[14] LIU F，YAROM Y，GE Q，et al. Last-Level Cache Side-Channel Attacks are Practical[C]// IEEE Symposium on Security & Privacy. 2015：605-622.

[15] YU S，GUI X L，ZHANG X J，et al. Co-residency Detection Scheme based on Shared Cache in the Cloud[J]. Journal of Computer Research and Development，2013，50(12)：2651-2660. (in Chinese)
余思，桂小林，张学军，等. 云环境中基于 cache 共享的虚拟机同驻检测方法[J]. 计算机研究与发展，2013，50(12)：2651-2660.

[16] SI Y，GUI X，LIN J，et al. Detecting VMs Co-residency in Cloud：Using Cache-based Side Channel Attacks [J]. Electronics & Electrical Engineering，2013，19(5)：73-78.

[17] BIAN G Q，ZHAI H，SHAO B L. A Measurement Method of Side-Channel-Attacks Threat for Co-Residency Virtual Machines Based on Cloud Model[J]. Journal of Xi'an Jiaotong University，2016，50(4)：21-27. (in Chinese)
边根庆，翟红，邵必林. 一种采用云模型的同驻虚拟机侧通道攻击威胁度量方法[J]. 西安交通大学学报，2016，50(4)：21-27.

[18] SHEN Q N，LI Q. Review on Co-residency Security Issues of Virtual Machines in Cloud Computing[J]. Journal of Integration Technology，2015(5)：5-17. (in Chinese)
沈晴霓，李卿. 云计算环境中的虚拟机同驻安全问题综述[J]. 集成技术，2015(5)：5-17.