针对 AES 的 Cache 计时模板攻击研究

王 韬[1)] 赵新杰[1)] 郭世泽[2)] 张 帆[3)] 刘会英[1)] 郑天明[1)]

[1)](军械工程学院计算机工程系 石家庄 050003)

[2)](北方电子设备研究所 北京 100083)

[3)](康涅狄格大学计算机科学与工程系 斯托斯 康涅狄格州 06269 美国)

摘 要 受微处理器硬件架构和操作系统的影响,分组密码查找S盒不同索引执行时间存在差异,构成了S盒索引的天然泄漏源.该文采用"面向字节、分而治之"的旁路攻击思想,对AES抗Cache计时模板攻击能力进行了研究.首先分析了分组密码访问Cache时间差异泄漏机理,直观地给出了基于碰撞和模板的两种Cache计时攻击方法;其次给出了Cache计时外部模板攻击模型,提出了基于Pearson相关性的模板匹配算法,对128位AES加密第一轮和最后一轮分别进行了攻击应用;为克服外部模板攻击需要一个模板密码服务器的限制,提出了Cache计时内部模板攻击模型,并对AES进行了攻击应用;最后,在不同环境、操作系统、加密Cache初始状态、密码库中,分别进行攻击实验,同前人工作进行了比较分析,并给出了攻击的有效防御措施.

关键词 高级加密标准;分组密码;Cache计时攻击;模板攻击;查找S盒;防御措施

中图法分类号 TP393 **DOI号**: 10.3724/SP.J.1016.2012.00325

Research of Cache Timing Template Attacks on AES

WANG Tao[1)] ZHAO Xin-Jie[1)] GUO Shi-Ze[2)] ZHANG Fan[3)] LIU Hui-Ying[1)] ZHENG Tian-Ming[1)]

[1)](*Department of Computer Engineering, Ordnance Engineering College, Shijiazhuang* 050003)

[2)](*Institute of North Electronic Equipment, Beijing* 100083)

[3)](*Department of Computer Science and Engineering, University of Connecticut, Storrs, CT* 06269, *USA*)

Abstract Due to the impact of micro-architectures and operating systems, the execution time of different S-Box lookups in block ciphers have some variance, which may cause the leakages of the S-Box lookup indexes. Utilizing the "byte oriented and divide and conquer" strategy in side channel analysis, this paper analyzes the resistance of AES implementations against Cache timing template attacks. First, the mechanism of different Cache access time is analyzed, and two Cache timing attacks are provided (collision-based and template-based). Second, the model of Cache timing external template attack is built, and a new template matching algorithm is proposed which is based on Pearson correlation factor. Two real attacks on the first and the last round of AES are launched successfully. To overcome the requirement of a template platform in external template attacks, an internal template attack is proposed and applied to AES. Finally, several extended attacks on AES are conducted under different settings, operating systems, Cache initial states, and crypto libraries. The experimental results are compared with the previous work, and an effective countermeasure is also suggested.

Keywords AES; block cipher; Cache timing attacks; template attacks; S-Box lookup; countermeasures

收稿日期:2010-11-30;最终修改稿收到日期:2012-01-05.本课题得到国家自然科学基金(60772082,61173191)资助.**王 韬**,男,1964年生,博士,教授,博士生导师,主要研究领域为信息安全和密码学.**赵新杰**(通信作者),男,1986年生,博士研究生,主要研究方向为分组密码旁路分析和故障分析.E-mail:zhaoxinjieem@163.com.**郭世泽**,男,1969年生,博士,研究员,博士生导师,主要研究领域为信息安全和密码学.**张 帆**,男,1978年生,博士,主要研究方向为密码旁路分析和故障分析、计算机体系结构、无线传感器网络安全.**刘会英**,男,1984年生,博士研究生,主要研究方向为密码旁路分析.**郑天明**,男,1985年生,硕士研究生,主要研究方向为卫星网络安全与密码学.

1 引 言

1.1 相关工作

Cache 访问过程中,可通过执行时间、能量消耗(电磁辐射)等旁路产生一定的信息泄露. Kocher[1] 和 Kelsey[2] 在 20 世纪 90 年代中期提出将这些旁路信息用于密码破解的思想,随后密码学家在此基础上实现了针对分组密码的各种 Cache 攻击[3-33]. 根据所采集物理效应的分类不同,可将 Cache 攻击分为 Cache 计时攻击和 Cache 功耗分析(电磁)攻击两种;根据信息采集模型不同,可将 Cache 攻击分为时序驱动(time driven)、访问驱动(access driven)、踪迹驱动(trace driven) 3 种.

时序驱动攻击[4-11] 主要通过计时手段采集整个密码加/解密时间,利用统计方法分析密钥. 文献[4-7]利用 Cache 命中和失效次数同密码整体的执行时间关系对 DES、AES 进行了攻击,这种攻击可简称为"Cache 碰撞计时攻击". 文献[8]利用查找 S 盒不同索引的执行时间同密码整体执行时间的关系,通过模板分析的方法对 AES 进行了远程计时攻击,该攻击可简称为"Cache 计时模板攻击". 后续研究者对文献[8]中的攻击进行了再现[9]、分析[10] 以及扩展[11]. 时序驱动攻击所需样本量大,分析方法相对复杂,但采集方法简单,跨平台适用能力强.

访问驱动攻击[12-25] 主要利用多进程共享 Cache 资源特性,使用一个恶意进程 S 通过读取私有数据在密码进程 V 执行(某次查找表或整个加密)前清空 Cache,然后在 V 执行后再次启动 S 观察私有数据被替换情况,根据二次访问执行时间推断 S 和 V 的外部 Cache 访问碰撞,获取 V 访问 Cache 行地址,并结合算法进行密钥分析. 文献[12]首次指出多线程共享 Cache 访问方式可为恶意线程监视采集密码线程 Cache 访问地址提供入口,使得恶意线程能够窃取加密密钥,并实现了针对 RSA 的访问驱动 Cache 计时攻击. 之后,基于该思想,研究者对 AES[13-19]、Camellia[20]、ARIA[21]、SMS4[22] 等分组密码,HC-256[23]、SNOW 3G[24]、RC4[25] 等流密码进行了攻击应用. 同时序驱动攻击相比,访问驱动攻击采集方法比时序驱动稍显复杂,但分析方法简单,一般通过计时手段实现,主要适用于支持多进程或线程的微处理器平台.

踪迹驱动攻击[26-33] 需精确采集一次密码加密多次查表导致的内部 Cache 访问碰撞信息,得到每次加密所有查表的 Cache 命中和失效序列,在此基础上结合算法进行密钥分析. 文献[26]首先提出踪迹驱动攻击思想,文献[27]首次通过功耗仿真对 AES 进行了攻击尝试,后续研究者通过仿真手段对 OPENSSL 中的快速 AES 实现的第一轮[28]、第二轮[29]、最后一轮[30] 进行了分析,通过功耗[32]、电磁[33] 手段对适用紧凑型 S 盒的 AES 实现进行了物理攻击实验. 一般来说,Cache 访问内部碰撞采集通过计时手段很难实现,常通过功耗或电磁采集手段进行,攻击主要适用于嵌入式处理器平台,需物理接触密码设备,远程攻击可行性不强.

上述 3 种攻击中,通过分区 Cache、将 S 盒预先加载到 Cache 中等实现[8,14,34-36],可消除 Cache 碰撞信息对整体执行时间的影响、阻止攻击者采集 Cache 碰撞信息,进而可有效地防御时序驱动攻击中的 Cache 碰撞计时攻击、访问驱动攻击和踪迹驱动攻击. 但由于查找 S 盒不同索引的时间差异受 CPU 硬件架构、操作系统等多种因素影响,很难从根本上消除掉,现有措施不能有效防御 Cache 计时模板攻击. 有研究者指出[11],即使在加密前将 S 盒提前加载到 Cache 中,使得加密每次查表均发生 Cache 命中,仍不能完全消除掉查表执行时间差异,存在遭受 Cache 计时模板攻击的风险. 因此,本文主要对 Cache 计时模板攻击进行研究.

在 Cache 计时模板攻击方面,Bernstein[8] 在强制消除网络传输时延条件下,实现了一种针对 OpenSSL-0.9.7a 中 AES 加密第一轮的远程 Cache 计时攻击,使用 $2^{27.5}$ 个样本恢复出 128 位 AES 密钥. 攻击中,密码服务端负责采集 AES 加密时间并发送给攻击端,本质仍属本地攻击. Bernstein 首先搭建了一台同目标密码服务器相同配置的模板密码服务器,在已知密钥情况下采集到对大量样本的加密时间,并计算第一轮每个查表索引对应 256 个聚类的平均时间标准差(每个聚类平均时间同所有聚类的平均时间之差),得到一条模板曲线;然后对目标密码服务器在未知密钥情况下采集大量样本的加密时间,通过预测每个密钥字节候选值,计算第一轮每个查表索引对应的 256 个聚类的平均时间标准差,得到一条预测曲线;再通过计算这两条曲线 256 个点的平均时间标准差乘积并求和,得出两条曲线匹配度. 正确的密钥字节候选值对应的匹配度较高,否则较低. Bernstein 攻击存在以下潜在问题:一是计算的曲线匹配度未进行归一化处理,匹配算法准确度有待提高;二是攻击条件过于严格,需获取同目标密码服务器一样配置的模板密码服务器,并能控制其在密钥已知时进行密码运算,搭建时间模板. 本

文工作主要是围绕上述两个问题开展的.

1.2　本文的贡献

本文对 Cache 计时模板攻击进行了一定的研究，并以 AES 为例进行了攻击应用，主要研究贡献如下：

(1) 分析了分组密码查找 S 盒 Cache 访问时间泄漏的机理，直观给出了如何利用时间差异进行密钥分析的方法.

分组密码时序驱动 Cache 攻击主要是利用 S 盒操作时间差异对整体加密时间的影响进行密钥分析. 我们认为 S 盒查表操作时间差异可划分为两类：一是加密两次，查找同一个 S 盒 Cache 访问时命中和失效的时间差异，二是利用加密一次查找同一个 S 盒不同索引的执行时间差异. 现有时序驱动 Cache 攻击[4-11]本质上都属于这两类中的某一类，大都仅给出原理说明，细节阐述很少. 本文首次直观地给出了这两类时间差异的特征以及如何利用其进行密钥分析的方法，同时对时间差异泄漏机理和攻击防御难度也进行了比较分析.

(2) 提出了一种新的基于计算 Pearson 相关性的 Cache 计时模板匹配算法.

Bernstein[8]给出了一种 Cache 计时外部模板攻击，主要针对 AES 加密第一轮进行，其模板匹配方法是将已知密钥时查表索引对应计时标准差模板曲线、未知密钥字节预测查表索引对应计时标准差曲线上 256 个点简单乘积加和，最大匹配度对应的预测值即为正确密钥字节值，匹配度未进行归一化处理. 本文首先给出 Cache 计时外部模板攻击模型，并提出一种新的基于计算 Pearson 相关性的 Cache 计时模板匹配算法，然后对 AES 加密第一轮和最后一轮均进行攻击和分析. 新模板匹配算法通过计算模板曲线、预测曲线间的 Pearson 相关性系数，对模板匹配度进行归一化处理，提高了正确密钥字节可识别度，降低了攻击所需的样本量.

(3) 提出了一种新的 Cache 计时内部模板攻击模型.

Bernstein 攻击[8]属于外部模板攻击，攻击假定可预先获取一个同目标密码服务器一样配置的模板密码服务器，并能够使用已知密钥执行大量的加密操作，采集加密时间构建模板. 实际情况下，这种条件过于严格，而且在外部模板攻击中，用于搭建模板的密码服务器上的运行环境很难保证同目标密码服务器完全一致，而这种环境的差异信息对攻击成功率存在很大的影响.

为解决上述问题，本文提出了一种新的 Cache 计时内部模板攻击模型，并对 AES 第一轮和最后一轮进行了攻击分析. 攻击者可直接采集目标服务器加密时间，然后利用不同次查找同一个 S 盒的时间搭建内部计时模板，通过计算 Pearson 相关性系数的模板匹配方法恢复相关密钥. 由于内部模板搭建可在目标密码服务器上直接进行，模板搭建和匹配准确度相对较高，攻击样本量比外部模板攻击相对要小.

(4) 分析和评估了不同环境下 AES 实现抗 Cache 计时模板攻击的能力，并给出攻击的一种有效防御措施.

现有的时序驱动 Cache 计时攻击[4-11]大都在单一环境下针对单一 AES 实现进行，我们则利用 Cache 计时模板攻击对本地远程环境、不同操作系统、不同加密前 Cache 初始状态、不同密码库下的 AES 实现安全性进行了分析和评估，试图研究 AES 实现在不同环境下的抗 Cache 计时模板攻击的能力，并给出有效防御措施. 结果表明：由于网络传输时延甚至其抖动远大于不同输入的加密时间差异，远程环境下的分组密码 Cache 计时模板攻击可行性不高；加密前将 S 盒预先加载到 Cache 中可有效防御 Cache 碰撞计时攻击、访问驱动和踪迹驱动 Cache 计时攻击，但由于其不能消除查找不同 S 盒索引的访问时间差异，因而不能有效防御 Cache 计时模板攻击；最新的各类密码库，如 OpenSSL-0.9.8j、Miracl 5.0、LibTomCrypt 1.17、Crypto++ 5.6.1 中 AES 实现仍然是不安全的；防御可通过在 AES 加密首轮、末轮将查找表预先加载到 Cache 中，并增加一定的随机时延来实现.

1.3　结构组织

本文第 2 节给出 Cache 访问机制、查找 S 盒访问 Cache 时间泄漏机理和密钥分析方法；第 3 节给出 Cache 计时外部模板攻击模型和对 AES 攻击应用，并和前人工作进行了比较；第 4 节给出 Cache 计时内部模板攻击模型和对 AES 攻击应用，并对两种模板攻击进行比较分析；第 5 节对不同环境、不同操作系统、不同加密 Cache 初始状态、不同密码库下的 AES 实现安全性进行分析和评估；第 6 节给出一种攻击防御措施；第 7 节总结全文.

2　Cache 结构与信息泄露分析

2.1　Cache 访问机制

现代微处理器大都使用高速缓存 Cache 来解决

CPU 与主存之间速度不匹配的问题. 假设整个 Cache 包括 S 个 Cache 组,每组有 W 个 Cache 行,每行有 δ 个元素(B 字节),则整个 Cache 大小为 $S\times W\times\delta\times B$ 字节. CPU 读取主存中的一个字 a 时,首先将 a 地址放入地址寄存器,Cache 控制逻辑依据地址判断 a 当前是否在 Cache 中,如果是则地址变换成功,发生"Cache 命中",从 Cache 中直接读取 a;否则发生"Cache 失效",根据程序局部性原理,把包括 a 在内的一整块数据(δ 个 Cache 元素)从主存中读出来,装载到 Cache 中去,然后从 Cache 中读取 a. 对于典型处理器来说,Cache 命中所需时间较少,一般为 2～3 个时钟周期,而失效则一般需要 10～14 个. 需要指出的是,如无特殊说明,文中的加密时间都是以时钟周期为基本单位.

现代分组密码大都使用 S 盒查表操作,访问 Cache,提高算法的执行效率,但由于其整体加密时间受 Cache 结构和访问机制影响,可在某种程度上泄露出 S 盒查表索引信息,给密钥安全带来严重威胁. 影响密码整体加密时间的因素有很多,本文主要从 S 盒查表操作时间对整个加密时间的影响进行分析. 总的来说,S 盒查表操作时间对整体加密时间的影响分为两类:一是加密两次查找 S 盒 Cache 访问命中和失效的时间差异,常用于"Cache 碰撞计时攻击";二是利用加密查找 S 盒不同索引的执行时间差异,常用于"Cache 计时模板攻击". 下面分别给出如何利用这两类信息进行密钥分析的方法.

2.2 查找 S 盒 Cache 访问命中和失效时间信息泄露分析

如图 1 所示,p_i 和 p_j 是一次加密过程中两个明文字节,k_i 和 k_j 是对应的密钥字节. 以 AES[37] 加密第一轮查找表① T_0 为例,对于两次查 T_0 表操作,首先按照明文字节差分 $p_i\oplus p_j$ 将大量样本的整个加密时间划分为 256 个聚类,并计算每个聚类的平均加密时间. 如果两次查表索引相同,则第二次查表会发生 Cache 命中,满足

$$p_i\oplus k_i=p_j\oplus k_j\Rightarrow k_i\oplus k_j=p_i\oplus p_j \tag{1}$$

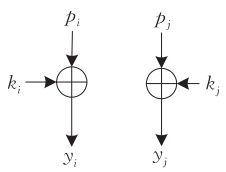


图 1 AES 加密第一轮两次查表

此时 $p_i\oplus p_j$ 值对应的聚类样本加密时间较短,否则较长. 也就是说,加密时间较短聚类对应的即为 $p_i\oplus p_j$ 的值,事实上就是 $k_i\oplus k_j$ 值. 当考虑到 Cache 调度策略和程度的局部性原理,发生 Cache 命中时,AES 加密两次查找 T_0 表的索引值 y 的高 $8-\log_2(\delta)$ 位(表示为$\langle y\rangle$)相同,即

$$\langle p_i\oplus k_i\rangle=\langle p_j\oplus k_j\rangle\Rightarrow\langle k_i\oplus k_j\rangle=\langle p_i\oplus p_j\rangle \tag{2}$$

此时会有连续的 δ 个 $p_i\oplus p_j$ 值对应聚类的平均加密时间较短. 图 2 为 Athlon 64 3000+1.81 GHz 处理器下($\delta=16$)2^{21} 样本时,OpenSSL-0.9.8a[38] 中 AES 第一轮前两次查找 T_0 表根据 $p_0\oplus p_4$ 值划分的 256 个聚类的平均访问时间,可以看出,当 $p_0\oplus p_4$ 的高 4 位值为 7 时,其对应平均加密时间较短,而真实的 $p_0\oplus p_4$ 值为 0x78.

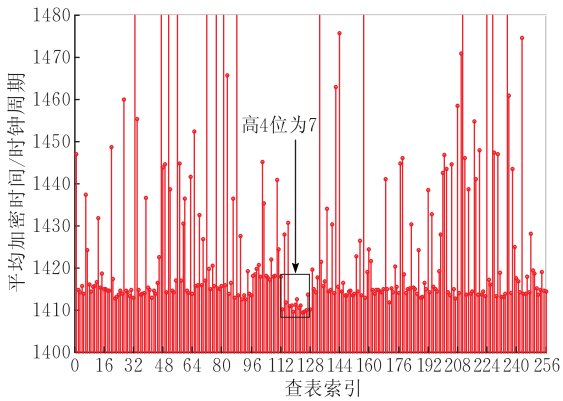


图 2 $p_0\oplus p_4$ 的 256 候选值对应的 AES 平均加密时间

2.3 查找 S 盒不同索引时间信息泄露分析

2.2 节考虑的是同一次加密两次查找 S 盒时,第二次查找 S 盒发生的 Cache 命中和失效对整体加密时间的影响,本节则主要考虑单次查找 S 盒不同索引访问时间对整体加密时间影响.

受处理器硬件架构、操作系统、Cache 调度策略等因素影响,分组密码查找 S 盒不同索引的访问时间也不尽相同. Bernstein[8] 指出:大量样本的查找分组密码 S 盒不同索引的平均访问时间可以用来进行精确建模,作为计时模板信息. 图 3 为 Windows XP SP2 环境、Athlon 64 3000+1.81 GHz 处理器下($\delta=16$),使用两个已知的不同随机密钥对 2^{21} 样本加密时,OpenSSL-0.9.8a 中 AES 第一轮查找 T_0 表(图 3(a))、T_3 表(图 3(b))的 256 个索引值对应的平均加密时间标准差. 平均加密时间标准差是指某个索引值聚类对应的平均加密时间减去所有加密样本平均加密时间差值. 可以看出,同一查找表的 256 个索引值对应平均加密时间标准差确实不尽相同,而且不同查找表的相同查表索引对应的平均加密时间

① 这里的查找表实际上就是 S 盒.

标准差也不相同.

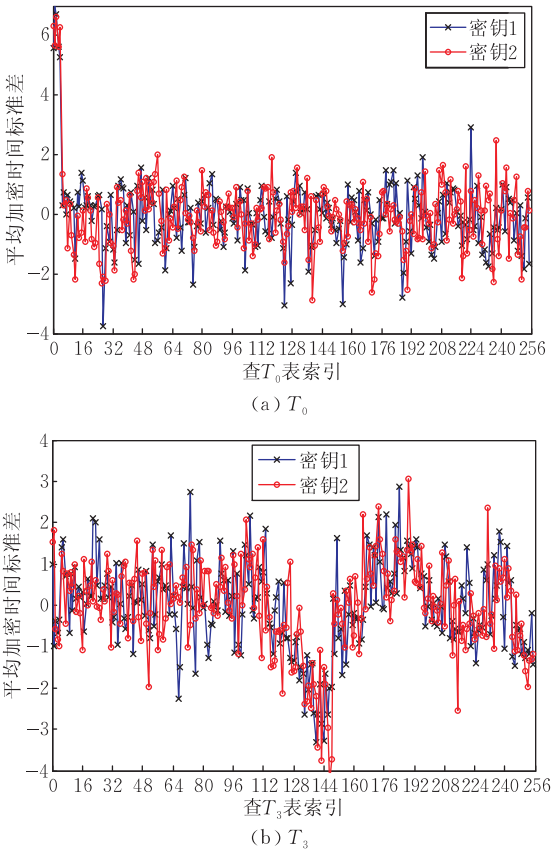


图 3　不同密钥对应的 AES 第一轮查表 256 个索引和平均加密时间标准差

如果攻击者能够掌控一台同目标密码服务器相同的模板密码服务器，并能使用已知密钥进行加密，采集到 AES 第一轮 16 次查找 S 盒的所有索引值的平均加密时间，搭建类似图 3 的 16 条 Cache 计时模板曲线，然后对未知密钥的目标密码服务器采集大量的加密时间信息，分别预测每个密钥字节的值，将根据每个密钥字节候选值将样本划分为 256 个索引值聚类，得到对应的一条平均加密时间标准差，计算其同前面模板曲线的匹配度，正确密钥字节候选值对应的匹配度往往较高.

在加密前进行 Cache 预热，将 S 盒提前加载到 Cache 中，使得加密每次查表均发生 Cache 命中，可消除 2.2 节提到的时间差异，进而有效防御 Cache 碰撞计时攻击[4-7]、访问驱动攻击[12-19]和踪迹驱动攻击[26-33]. 然而本节攻击利用的单次查找 S 盒不同索引的时间不受 Cache 访问命中和失效时间的影响，时间差异根源来自于 CPU 硬件架构、操作系统等因素，很难消除. 即使在加密前将整个 S 盒提前加载到 Cache 中，单次查找 S 盒不同索引值的时间仍然可以用于构建模板，用于进一步的密钥分析.

考虑到利用这类时间差异信息进行攻击的较强适用性，本文主要研究利用单次查找 S 盒不同索引值的访问时间差异进行的密钥分析，即 Cache 计时模板攻击技术.

3　AES Cache 计时外部模板攻击

3.1　Cache 计时外部模板攻击模型

需要说明的是，本文中的 Cache 计时模板攻击主要针对 S 盒查表操作，特别适用于使用了 S 盒的分组密码甚至流密码. 攻击一般由模板搭建、目标设备 Cache 计时信息采集、模板匹配 3 个步骤组成(图 4). 不失一般性，下面以对分组密码加密第一轮使用明文 P 和密钥 K 异或值查表分析为例进行

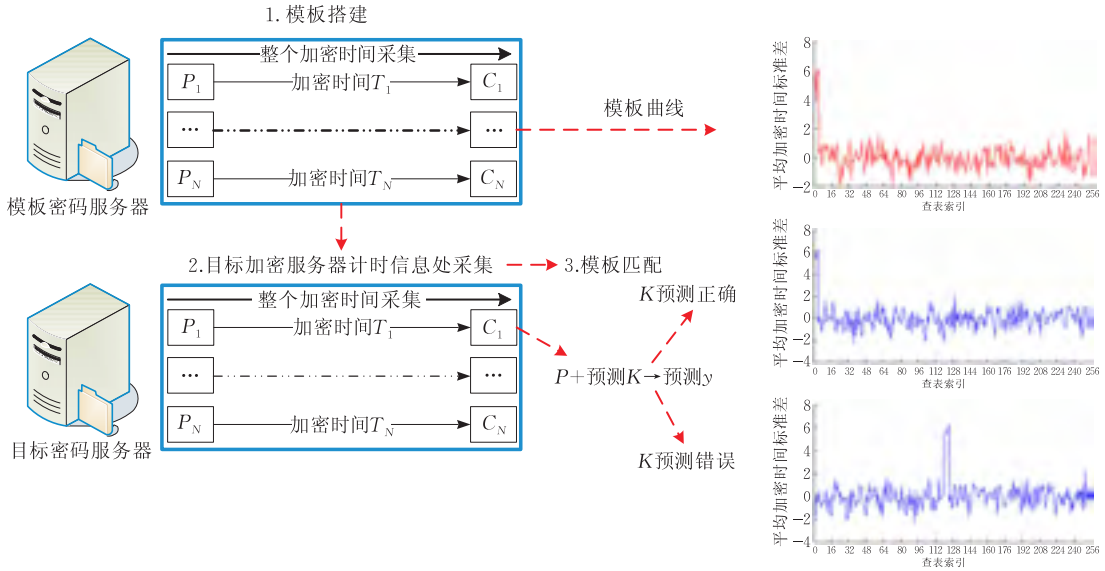


图 4　Cache 外部计时模板攻击模型

介绍.

(1)模板搭建

攻击者能够掌控一台同目标密码服务器一样的模板密码服务器,并能对其使用已知密钥进行加密操作.

①对掌控模板密码服务器,使用已知密钥 K^T 为 N 个随机明文样本执行加密操作,将采集加密时间存储在数组 $TM[i](1\leq i\leq N)$.

②对于第一轮加密过程中的 m 次(假如 $m=16$)查表操作对应的明文、密钥块(假如块大小为一个字节),根据 16 个索引值 $p_i\oplus k_i(1\leq i\leq N)$ 的候选值,分别将 $TM[i]$ 划分为 256 个聚类,并计算每个聚类的平均加密时间标准差 $TMC[i][j](0\leq i\leq 15,0\leq j\leq 255)$ 中.

(2)目标设备 Cache 计时信息采集

①对未知密钥目标密码服务器,使用未知密钥 K 采集对 N 个随机明文加密时间 $T[i](1\leq i\leq N)$.

②对于第一轮加密中 m 次查表操作对应的明文、密钥块,以第 i 次查表操作为例,首先预测 k_i 的 256 个候选值,对于每个 k_i 的第 j 个候选值,计算 N 个明文样本对应 $p_i\oplus k_i$ 值 q,将 $T[i]$ 划分为 256 个聚类,并计算平均加密时间标准差 $TC[i][j][q](0\leq i\leq 15,0\leq j\leq 255,0\leq q\leq 255)$.

(3)模板匹配

①以分析密钥 k_i 为例. 假设 $i=0,k_i=0$,利用式(3)计算匹配向量 $\boldsymbol{X}=TC[0][0][q]$ 同计时模板向量 $\boldsymbol{Y}=TMC[0][q]$ 的 Pearson 相关性系数,得到 k_i 的 256 个候选值对应的相关性系数向量 $\boldsymbol{R}[0][j](0\leq j\leq 255)$.

$$r\langle \boldsymbol{X},\boldsymbol{Y}\rangle=\frac{\sum_{q=0}^{255}(X_q-\overline{\boldsymbol{X}})(Y_q-\overline{\boldsymbol{Y}})}{\sqrt{\sum_{q=0}^{255}(X_q-\overline{\boldsymbol{X}})^2}\sqrt{\sum_{q=0}^{255}(Y_q-\boldsymbol{Y})^2}} \tag{3}$$

②将向量 $\boldsymbol{R}[0][j](0\leq j\leq 255)$ 按大小进行排序,最大值对应的 j 即为正确 k_i.

③参考上面步骤通过模板匹配获取所有 k_i 值,通过进一步分析获取初始密钥 K.

需要说明的是,上面模板攻击过程同样适用于利用密文和加密时间对最后一轮扩展密钥的分析.

3.2 第一轮攻击

为提高 AES 软件加密速度,OpenSSL-0.9.8a 中 AES 在每一轮中,将除与轮密钥异或以外的操作合并为 16 次查表操作. 整个加密过程由 160 次查表和 176 次异或操作组成,执行效率非常高,前 9 轮分别对 $T_0\sim T_3$ 表执行 4 次查表操作,最后一轮仅对 T_3 表执行了 16 次查表操作. AES 第一轮加密 16 次查表操作索引为

$$p_i\oplus k_i=y_i(0\leq i<16) \tag{4}$$

应用 3.1 节攻击模型,通过采集已知密钥的 AES 密码服务器加密时间构建模板,并采集未知密钥的密码服务器加密时间,然后预测每个密钥字节 k_i,并根据其利用式(3)~(4)计算每个索引字节对应的平均加密时间标准差,同模板进行匹配,匹配度最大的即为正确密钥字节值.

在 Windows 环境下对 OpenSSL-0.9.8a 中的 AES 进行了 10 次 Cache 计时外部模板攻击实验,样本量为 2^{21},16 个正确密钥字节在 256 个候选值模板匹配度集合中排序如表 1 所示,其中匹配度越高,序号越小.

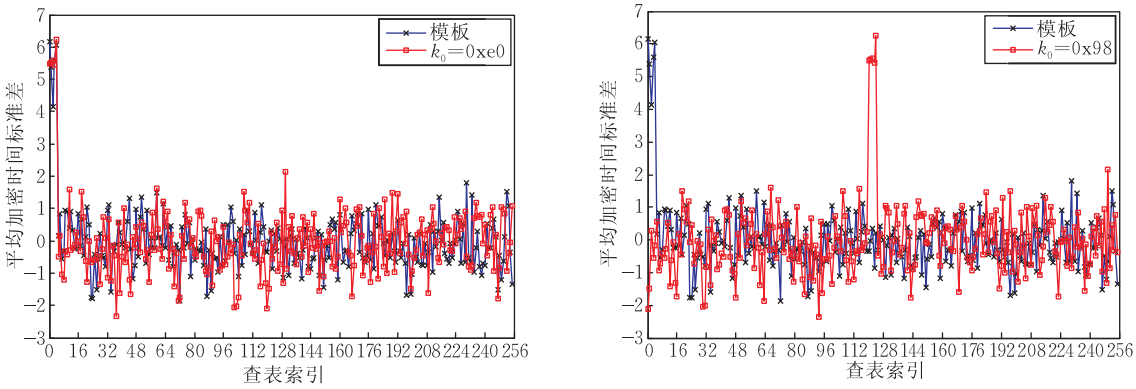
表 1 10 次 OpenSSL-0.9.8a 中 AES 第一轮 Cache 计时外部模板攻击结果

序号	k_0	k_1	k_2	k_3	k_4	k_5	k_6	k_7	k_8	k_9	k_{10}	k_{11}	k_{12}	k_{13}	k_{14}	k_{15}
1	1	52	155	1	1	10	188	1	1	6	51	1	2	170	2	1
2	1	14	82	1	1	35	230	1	1	19	40	1	3	187	31	1
3	1	201	91	1	1	14	26	1	1	151	33	1	1	11	66	1
4	1	3	142	1	1	13	150	1	2	88	139	1	1	89	13	1
5	1	229	65	1	1	6	138	1	2	224	16	1	1	116	44	1
6	1	135	100	1	1	12	9	1	1	129	81	1	1	133	16	1
7	1	188	33	1	1	2	27	1	1	7	21	1	1	205	175	1
8	1	176	195	1	1	32	127	1	1	250	78	1	1	127	4	1
9	1	39	68	1	1	17	136	1	1	99	23	1	1	57	50	2
10	1	11	50	1	1	11	113	1	1	4	52	1	1	43	13	1

从表 1 可以看出,T_0 表和 T_3 表对应的 8 个密钥字节恢复效果较好,而 T_1 表和 T_2 表对应字节恢复效果较差. 图 5(a)、(b)分别为猜测 $k_0=0$xe0 和 $k_0=0$x98 时预测查 T_0 表索引聚类和模板中索引聚类对应平均加密时间标准差曲线. 正确的密钥值是 0xe0. 当 $k_0=0$xe0 时,两条曲线十分接近(图 5(a)),而 $k_0=0$x98 时,两条曲线差异较大(图 5(b)).

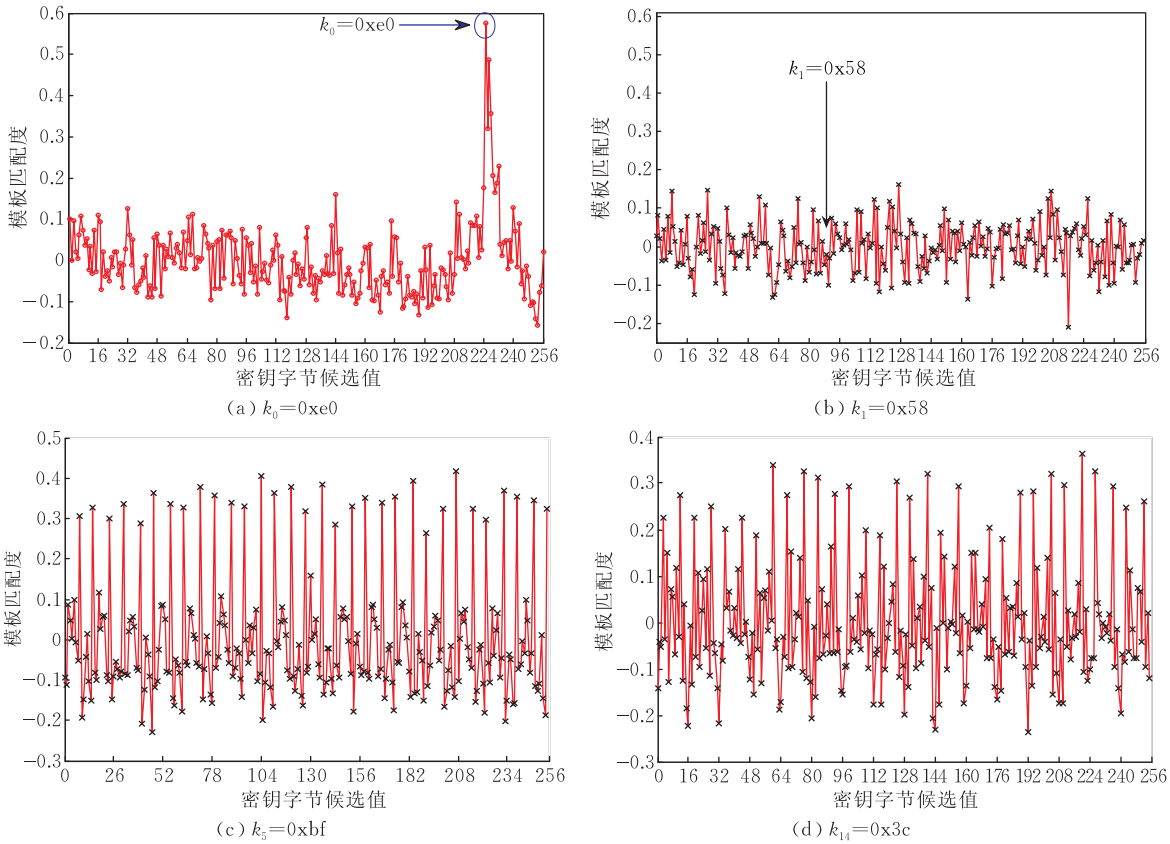
图 6(a)为 k_0 的 256 个候选值的匹配曲线,可以

看出，$k_0=0$xe0 对应匹配度比较大，匹配曲线出现明显的峰值. 需要说明的是，当考虑到程序的局部性原理时，每次查找表访问 1 个索引值就会将同一个块的 16 个 Cache 元素都加载到 Cache 中，这样相邻的 16 个索引查表时间十分接近，这样索引值的低 4 位值有时候很难精确预测. 在图 6(a)中，当 k_0 的高 4 位为 e 时，其匹配度都相对比较高.



(a) 正确预测k_0=0xe0　　(b) 错误预测k_0=0x98

图 5　OpenSSL-0.9.8a 中 AES 第一轮查 T_0 表索引值 256 聚类对应平均加密时间标准差



(a) k_0=0xe0　　(b) k_1=0x58

(c) k_5=0xbf　　(d) k_{14}=0x3c

图 6　OpenSSL-0.9.8a 中 AES 加密第一轮密钥字节候选值模板匹配度

实验中我们发现，经模板分析有些密钥字节的恢复效果比较好，如第 k_0，k_3，k_4，k_7，k_8，k_{11}，k_{12}，k_{15} 字节，正确的密钥字节值对应的匹配度最高，峰值比较明显(图 6(a))，有些密钥字节(如 k_1)的恢复效果则较差(图 6(b))，正确的密钥字节对应匹配度不一定是最高的. 有趣的是，正确 k_5，k_{14} 字节的低 2～4 位也可以恢复出来，如图 6(c)、(d)所示，会出现 32 个峰值，其对应密钥字节的低 2～4 位相同. 这样，在 Windows 环境下，2^{21} 样本下，应用本节攻击，可恢复 OpenSSL-0.9.8a 中 AES 的 70 位密钥. 进一步降低密钥搜索空间可通过下面两个途径获得：一是加大攻击样本量，二是对 AES 加密第二轮进行

进一步模板分析.

3.3 最后一轮攻击

Bernstein 攻击[8]主要针对 AES 加密第一轮，使用了 $2^{27.5}$ 个样本，经几天时间才恢复 128 位 AES 密钥. 我们在第一轮攻击的基础上，尝试将攻击转向最后一轮，检验 AES 最后一轮抗 Cache 计时模板攻击能力. OpenSSL-0.9.8a 中 AES 最后一轮仅对 T_4 表进行了 16 次查表操作，16 次查表索引 y_i 为

$$T_4[y_i]\oplus k_i^9=c_i^9\ (0\leq i<16)\Rightarrow y_i=T_4^{-1}[c_i^9\oplus k_i^9] \tag{5}$$

应用 3.1 节攻击模型，通过采集已知密钥的密码服务器上加密时间构建模板，并采集未知密钥的密码服务器加密时间，然后预测每个密钥字节 k_i^9，并根据其利用式(3)和式(5)计算最后一轮每次查表索引字节对应的平均加密时间标准差，同模板进行匹配，匹配度最大的即为正确密钥字节值.

在 Windows 环境下对 OpenSSL-0.9.8a 中的 AES 进行了本节所述外部模板攻击实验，样本量大小为 2^{20}，攻击均能在有限复杂度内恢复完整密钥，10 次攻击最后一轮密钥字节的模板匹配度排序如表 2 所示. 从中可以看出，最后一轮攻击(表 2)大部分正确密钥字节对应的排序要高于第一轮攻击(表 1)，攻击效果要好.

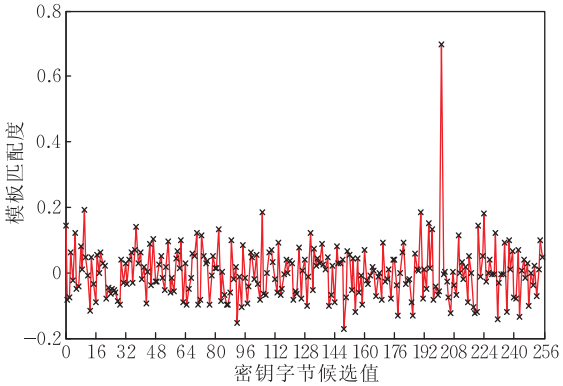
表 2 10 次 OpenSSL-0.9.8a 中 AES 最后一轮 Cache 计时外部模板攻击结果

序号	k_0^9	k_1^9	k_2^9	k_3^9	k_4^9	k_5^9	k_6^9	k_7^9	k_8^9	k_9^9	k_{10}^9	k_{11}^9	k_{12}^9	k_{13}^9	k_{14}^9	k_{15}^9
1	1	12	2	1	1	1	1	1	1	1	1	1	12	1	1	9
2	1	1	2	1	1	1	1	1	1	1	1	1	1	1	1	1
3	1	30	1	2	1	1	1	1	1	1	1	1	1	2	1	1
4	2	149	1	1	1	1	1	1	1	1	1	1	1	2	1	2
5	1	4	2	94	1	1	1	1	1	1	1	1	2	1	1	1
6	1	65	2	1	1	1	1	1	1	1	1	1	1	1	1	1
7	1	3	1	2	1	1	1	1	1	1	1	1	1	1	1	1
8	1	2	1	1	1	1	1	1	1	1	1	1	1	1	2	1
9	1	124	46	1	1	1	1	1	1	1	1	1	5	22	1	1
10	2	60	16	1	1	2	1	1	1	1	1	1	1	1	1	1

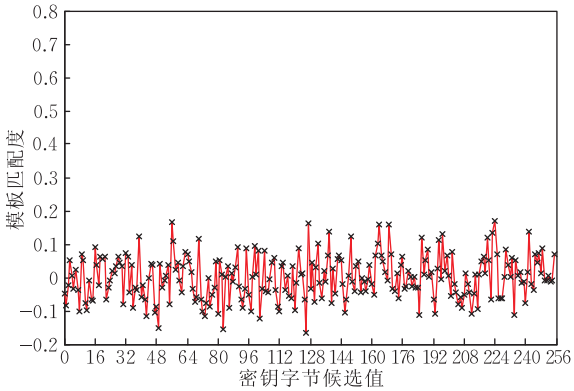
图 7(a)为 AES 最后一轮 k_0^9 的 256 个候选值匹配曲线，同图 6 比较可以看出，正确密钥候选值 $k_0^9=$0xc9 对应匹配度比较大，匹配曲线会出现唯一的峰值，效果要好于第一轮攻击. 原因有两个方面：

(1)只有最后一轮对 T_4 表执行了 16 次查表操作，整体加密时间受每次查 T_4 表操作时间影响比较明显，而第一轮攻击关注的是对加密整体时间受第一轮查 $T_0\sim T_3$ 4 个表的影响，由于加密其它轮也要查找这 4 个表，进而对整体加密时间也有较大影响，因而第一轮查表时间差异对整体时间影响较小，噪声较大.

(2)文中每个 Cache 行包含 $\delta=16$ 个 AES 查表元素. 考虑到程序的局部性原理，根据第一轮攻击中利用的式(4)可知，与正确密钥高 4 位相同的 16 个密钥字节的查表时间对整体时间影响大致相同，故第一轮理论上一般可获取每个密钥字节的高 4 位，而根据最后一轮攻击利用的式(5)可知，由于 T_4 表的雪崩扩散作用，同正确密钥高 4 位相同的 16 个密钥字节的查表时间对整体时间的影响也有很大区别，每个预测密钥字节对整体加密时间的影响差异都大，故正确密钥字节对应的匹配曲线峰值最为明显.



(a) $k_0^9=$0xc9



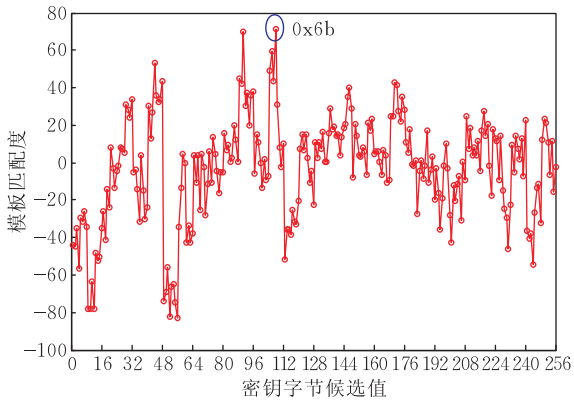
(b) $k_1^9=$b6

图 7 OpenSSL-0.9.8a 中 AES 加密最后一轮密钥字节候选值模板匹配度

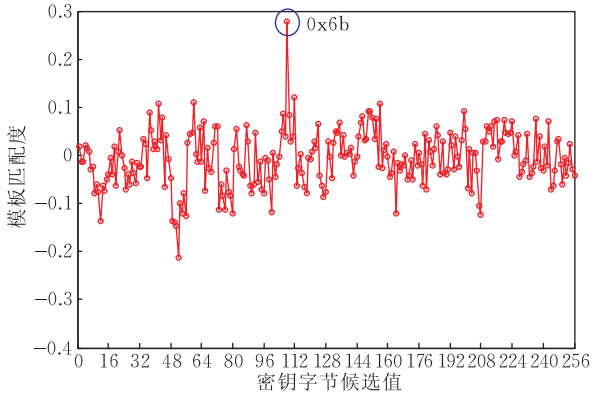
但是最后一轮 k_1^9 密钥字节的猜测结果不是很理想(图 7(b)),匹配曲线比较平缓,没有出现明显的峰值,匹配度普遍比较低,我们可以分析得出结论 k_1^9 猜测错误,最后通过暴力破解获取 k_1^9.

3.4　同前人工作比较

Bernstein[8] 攻击是典型的 Cache 计时外部模板攻击,主要针对 AES 加密第一轮进行,使用 $2^{27.5}$ 个样本恢复出 128 位 AES 密钥,攻击得到的模板匹配度未进行归一化处理. 我们在对 Bernstein 攻击基础上,提出了一种新的基于Pearson相关性系数的模板匹配算法,对 AES 第一轮和最后一轮均进行了攻击分析. 新匹配算法对匹配度进行了归一化处理,提高了 Bernstein 攻击的效率,可降低攻击样本量和提高攻击准确度. 图 8(a)、(b)分别为外部模板攻击中,2^{20} 样本下,应用 Bernstein 攻击方法和本文模板匹配方法对 OpenSSL-0.9.8a AES 加密最后一轮进行攻击(被攻击密钥字节相同,0x6b)的结果. 可以看出,本文方法对模板匹配度进行了归一化处理,正确密钥字节峰值更为明显.



(a) Bernstein攻击模板匹配方法



(b) 本文模板匹配方法

图 8　本文模板匹配方法同 Bernstein 攻击比较

4　AES Cache 计时内部模板攻击

4.1　Cache 计时内部模板攻击模型

第 3 节模板攻击属于“两步攻击”,攻击者需掌控一个与目标密码服务器一样配置的密码设备,并能够对已知密钥执行加密操作. 本节则尝试去除该条件,构造一种“一步攻击”. 攻击者直接采集目标服务器的加密时间,然后利用不同次查找同一个 S 盒时间搭建内部计时模板,通过模板匹配恢复相关密钥. 下面将详细描述 Cache 内部模板攻击模型.

Cache 内部模板攻击一般由目标设备加密时间信息采集、内部模板构建、模板匹配 3 个步骤组成. 不失一般性,下面仍以对分组密码加密第一轮使用明文 P 和密钥 K 的异或值查表的分析为例进行介绍.

目标设备加密时间采集过程不再赘述. 分组密码两次查找同一个 S 盒时,根据其索引 y_i,y_j(如$i=0$,$j=4$)划分的 256 个聚类平均时间标准差分布特征应该基本相似,如图 9(a)、(b)所示. 攻击者采集到目标密码服务器使用未知密钥的加密时间后,由于密钥未知,故无法根据查找表索引 y 将所有样本加密时间划分为 256 个聚类,而只可以根据每次查表对应的明文字节进行聚类划分. 具体内部模板构建、模板匹配方法如下:

(1) 首先按照 p_i 值进行聚类划分,得到 256 个聚类平均时间标准差,绘制一条模板曲线,如图 9(c)所示.

(2) 然后攻击者可通过预测 $k_i \oplus k_j$ 值,针对每一个 $k_i \oplus k_j$ 值,得到 $k_i \oplus k_j \oplus p_j$ 值对应的 256 个聚类(根据 p_j 划分)的平均时间标准差.

(3) 因为 $k_i \oplus k_j = p_i \oplus p_j$,如果 $k_i \oplus k_j$ 预测正确,$k_i \oplus k_j \oplus p_j$ 所划分的 256 个聚类平均时间标准差曲线和 p_i 值划分的模板曲线应该最匹配(如图 9(d)),计算两条曲线间的 Pearson 相关性系数值最大;反之,如果 $k_i \oplus k_j$ 预测错误,得到的预测曲线同模板曲线差别较大(如图 9(e)),得到的 Pearson 相关性系数值较小.

需要说明的是,基于内部模板攻击可得到某轮查同一类型 S 盒对应的不同密钥字节异或值,并不能直接得到密钥字节值. 同样,上面攻击模型适用于利用密文和加密时间对最后一轮扩展密钥异或值进行的分析.

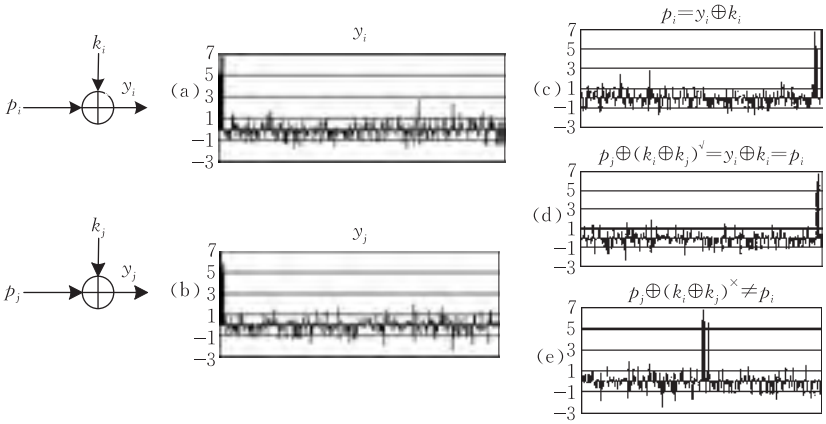


图 9 Cache 计时内部模板攻击模型

4.2 第一轮攻击

OpenSSL-0.9.8a 中 AES 第一轮对 $T_0\sim T_3$，4 个表分别查找 4 次，共查找 16 次表，第 j 次查找的是 $T_{j\%4}$ 表. 根据 4.1 节，第一轮攻击可获取 24 ($4\times(4\times3)/2$)个 $k_i\oplus k_j$ ($0\leq i, j\leq 15$, $i\%4=j\%4$)密钥字节，如图 10 所示.

这样，第一轮攻击理想情况下可获取到 96($4\times3\times8$)位 AES 密钥，但当考虑到 Cache 访问的局部性原理时，对于 64 字节 Cache 行大小，每个 Cache 行 16 个 S 盒元素来说，理论上可获取 48($4\times3\times4$)位密钥.

对 OpenSSL-0.9.8a 中的 AES 进行了第一轮内部模板攻击实验，样本量大小为 2^{20}，10 次攻击各个相关密钥字节的模板匹配度排序如表 3 所示.

表 3 OpenSSL-0.9.8a 中 AES 第一轮 10 次 Cache 计时内部模板攻击结果

序号	$k_4\oplus k_0$	$k_5\oplus k_1$	$k_6\oplus k_2$	$k_7\oplus k_3$	$k_8\oplus k_0$	$k_8\oplus k_4$	$k_9\oplus k_1$	$k_9\oplus k_5$	$k_{10}\oplus k_2$	$k_{10}\oplus k_6$	$k_{11}\oplus k_3$	$k_{11}\oplus k_7$
1	1	97	25	146	1	2	220	79	124	69	254	1
2	1	70	38	157	1	1	98	81	35	28	250	1
3	1	139	6	79	1	1	227	74	135	15	255	1
4	1	79	65	84	1	1	194	106	16	58	252	1
5	1	121	30	67	1	1	205	25	197	49	250	1
6	1	78	79	139	1	1	101	27	12	9	248	1
7	1	208	15	95	1	1	170	67	109	93	253	1
8	1	22	40	126	1	1	22	9	49	11	248	1
9	1	144	60	104	1	1	200	67	113	9	250	1
10	1	67	45	92	1	1	58	101	17	10	246	1
序号	$k_{12}\oplus k_0$	$k_{12}\oplus k_4$	$k_{12}\oplus k_8$	$k_{13}\oplus k_1$	$k_{13}\oplus k_5$	$k_{13}\oplus k_9$	$k_{14}\oplus k_2$	$k_{14}\oplus k_6$	$k_{14}\oplus k_{10}$	$k_{15}\oplus k_3$	$k_{15}\oplus k_7$	$k_{15}\oplus k_{11}$
1	2	3	2	19	197	218	61	72	31	4	230	256
2	1	2	1	151	95	57	108	152	50	5	256	255
3	1	1	1	42	206	140	36	20	54	3	246	254
4	1	3	2	59	69	99	147	111	27	3	255	254
5	2	2	1	30	56	169	28	48	46	3	230	255
6	1	1	1	106	105	101	69	60	26	4	255	254
7	1	3	1	97	134	131	22	133	6	4	237	256
8	2	1	3	140	95	73	87	75	51	1	256	255
9	1	2	1	37	164	234	35	91	16	6	251	256
10	1	2	1	32	85	104	79	67	46	2	255	255

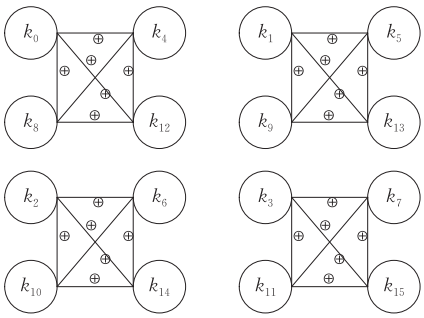
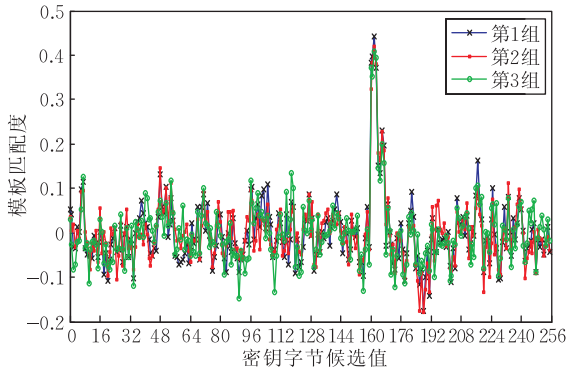


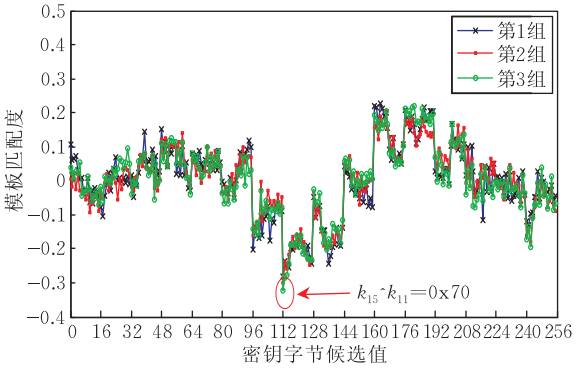
图 10 AES 第一轮 Cache 计时内部模板攻击恢复密钥

实验中发现：正确的 $k_4\oplus k_0$，$k_8\oplus k_0$，$k_8\oplus k_4$，$k_{11}\oplus k_7$，$k_{12}\oplus k_0$，$k_{12}\oplus k_4$，$k_{12}\oplus k_8$，$k_{15}\oplus k_3$ 字节对应匹配度一般是最高的. 如图 11(a)所示，3 组攻击中的 256 个 $k_4\oplus k_0$ 候选值匹配度(正确值为 0xa2)最高. 这同 3.2 节 OpenSSL-0.9.8a 中的 AES 第一轮外部模板攻击可精确恢复 k_0，k_3，k_4，k_7，k_8，k_{11}，k_{12}，k_{15} 字节的结果也基本对应. 实验中，我们还发现正确的 $k_{11}\oplus k_3$，$k_{15}\oplus k_{11}$ 字节对应匹配度常常是最小的，3 组攻击中的 256 个 $k_{15}\oplus k_{11}$ 候选值匹配度

(正确值为 0xb6)如图 11(b)所示，利用该特性，也可以恢复 $k_{11}\oplus k_3$，$k_{15}\oplus k_{11}$ 字节.



(a) 正确$k_4\oplus k_0$=0xa2



(b) 正确$k_{15}\oplus k_{11}$=b6

图 11　OpenSSL-0.9.8a 中 AES 加密第一轮密钥字节候选值模板匹配度

这样，在 Windows 环境下，使用 2^{21} 样本，应用本节攻击，通过对恢复的 $k_4\oplus k_0$，$k_8\oplus k_0$，$k_8\oplus k_4$，$k_{11}\oplus k_7$，$k_{12}\oplus k_0$，$k_{12}\oplus k_4$，$k_{12}\oplus k_8$，$k_{15}\oplus k_3$，$k_{11}\oplus k_3$，$k_{15}\oplus k_{11}$ 字节进行分析，可恢复 OpenSSL-0.9.8a 中 AES 的 48 位密钥. 进一步降低密钥搜索空间可通过下面加大攻击样本量或开展第二轮分析进行.

4.3　最后一轮攻击

OpenSSL-9.8.a 中 AES 最后一轮仅对 T_4 表进行了 16 次查表. 根据 4.1 节原理，最后一轮攻击最多可获取 120 个 $k_i^9\oplus k_j^9$ 密钥字节异或值(图 12)，分析获取最后一轮扩展密钥 K^9，经密钥逆推恢复初始主密钥 K.

应用 4.1 节分析模型，对 OpenSSL-0.9.8a 中 AES 最后一轮进行攻击，2^{19} 样本下，120 个正确 $k_i^9\oplus k_j^9$ 的密钥字节匹配度排名如图 13 所示. 可以看出，OpenSSL-9.8.a 中 AES 最后一轮攻击后，大约 110 个左右的 $k_i^9\oplus k_j^9$ 对应排名均为第 1，经分析可恢复 AES 第 10 轮的 120 位扩展密钥，经逆推和暴力破解恢复初始密钥.

4.4　两种模板攻击比较分析

在 Athlon 64 3000+1.81GHz 处理器、Windows XP SP2、Microsoft Visual C++6.0 环境下，我们对 OpenSSL-0.9.8a 密码库中 AES 实现进行了 Cache 计时外部和内部模板攻击实验. 结果表明，OpenSSL-0.9.8a 库中 AES 最后一轮外部模板攻击和内部模板攻击中可分别使用 2^{20} 和 2^{19} 样本恢复 128 位 AES 密钥.

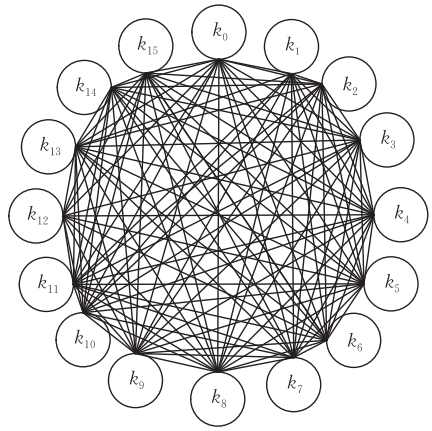


图 12　AES 最后一轮 Cache 计时内部模板攻击恢复密钥

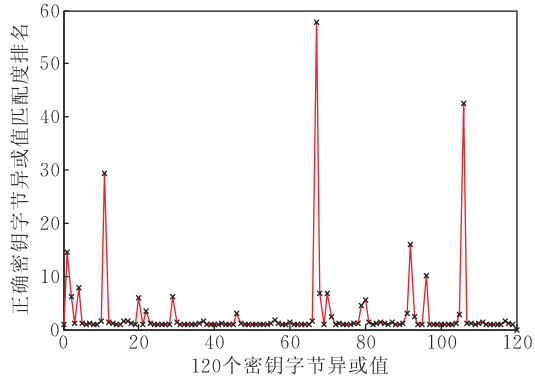


图 13　OpenSSL-0.9.8a 库中 AES 的正确 $k_i^9\oplus k_j^9$ 匹配度排名

同外部模板攻击相比，内部模板攻击具有以下特点：

(1) 降低了攻击代价

外部模板攻击假定攻击者可预先获取一个与目标密码服务器一样配置的模板密码服务器，并能够使用已知密钥执行大量的加密操作，采集加密时间搭建模板. 实际情况下，这种条件过于严格，内部模板攻击则不需要该条件.

(2) 提高了攻击准确度

外部模板攻击中，用于搭建模板密码服务器上的运行环境很难保证同目标密码服务器完全一致，这种环境的差异信息对攻击成功率存在很大的影响. 内部模板攻击可在目标服务器上直接搭建时间模板，准确度较高，攻击样本相对要小.

(3) 可获取密钥字节异或结果

外部模板攻击可直接获取 AES 轮密钥每个字

节.内部模板攻击中,由于攻击主要利用加密不同次查找同一类型 S 盒访问时间差异,只能直接获取对应密钥字节异或值.在 OpenSSL-0.9.8a 中 AES 攻击时,由于第一轮分别查找 4 种查找表 4 次,第一轮内部模板攻击理论上可恢复 96 位密钥,比外部模板攻击 128 位的理论值小;由于最后一轮仅对 T_4 表查找 16 次,内部模板攻击可获取 120 位 AES 密钥,结果比 128 位的理论值小.

5 攻击实验扩展与比较分析

在实验过程中,课题组还在本地远程环境、不同操作系统、不同加密前 Cache 初始状态、不同密码库下的 AES 实现进行了多例攻击实验,以评估其抗 Cache 计时模板攻击安全性.

5.1 本地、远程环境攻击

前面第 3、4 节攻击主要在本地环境下进行.在此基础上,我们在两种远程环境下(Windows 环境)对 OpenSSL-0.9.8a 库中 AES 最后一轮进行了外部模板攻击实验,攻击端、模板密码服务器和目标密码服务器分别被部署在不同的电脑上.

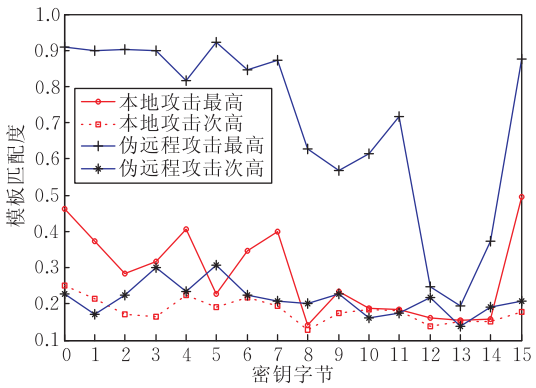
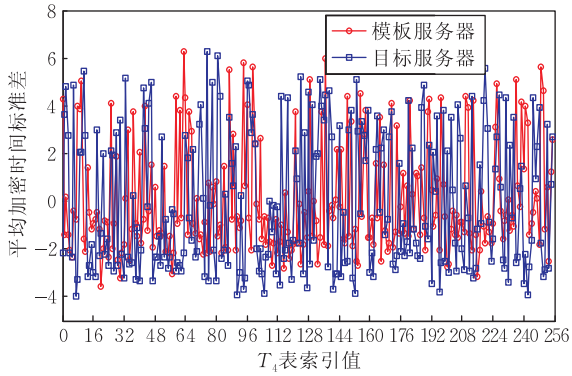


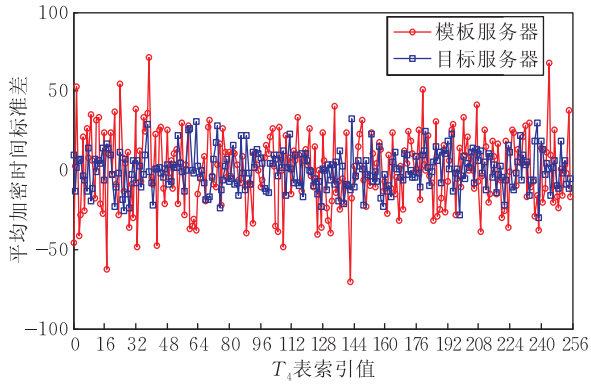
图 14 AES 最后一轮本地攻击、伪远程攻击的最高和次高模板匹配度

(1)第 1 种攻击环境同 Bernstein[8] 攻击类似,加密服务器在收到攻击端发送的明文后,将准确的加密时间同密文一起反馈给攻击端,攻击端再利用 3.1 节方法分析密钥.与 Bernstein 攻击不同的是,攻击端向加密服务端仅发送 16 个字节的明文,而 Bernstein 攻击每次需发送 400(800)字节的明文.由于计时采集不是在攻击端进行,攻击可称之为伪远程攻击.在这种环境下,由于服务端接收明文和发送密文、加密时间需要大量地访问 Cache,能够起到间接的清空 Cache 作用,其攻击的最高模板匹配度同次高模板匹配度的差距甚至要远大于本地攻击时(加密前清空 Cache)最高模板匹配度同次高模板匹配度的差距,攻击效果十分明显,如图 3 所示.实验结果表明,此类伪远程计时攻击,无需像 Bernstein 攻击每次需发送 400(800)字节的明文,正常的 16 个字节明文发送加上上次信息采集中密文和加密时间反馈就可以基本达到清空 Cache,将 AES 查找表从 Cache 中驱逐出去的作用.

(2)第 2 种攻击环境中,攻击端负责采集从发送明文到接收到密文之间的时间,这是实际攻击中常见的场景.对 AES 最后一轮进行外部模板攻击时,采集的模板服务器最后一轮第一次查 T_4 表的 256 个索引值的加密平均时间(90 000 个时钟周期)及标准差同目标服务器(1400 个时钟周期)具有很大不同,时间抖动范围很大(−60 到 80 个时钟周期),差异很大,故即使是正确的密钥字节值,其模板匹配度也很低,整个模板匹配度十分不准确(如图 15(b)).而在伪远程攻击时,模板服务器和目标服务器的查 T_4 表的 256 个索引值加密平均时间标准差基本接近,时间抖动范围非常小(−4 到 6 个时钟周期),十分接近,对于正确的密钥字节值,模板匹



(a)伪远程攻击(密码服务端计时)



(b)真实远程攻击(攻击端计时)

图 15 AES 最后一轮查 T_4 表索引对应平均加密时间标准差

配度很高(如图 15(a)). 可以说明,Cache 计时模板攻击在真实远程环境下的可行性不强,网络传输时延甚至其抖动即可掩盖了加密查同一 S 盒不同索引操作的微弱时间差异.

5.2　不同操作系统下攻击

同一处理器硬件架构在不同操作系统环境下,每个查找表建立的时间模板信息不同,噪声大小也不尽相同. 我们在 Windows 环境实验的基础上,还在 Fedora 8 Linux 系统、Gcc 4.1.8 编译器环境下进行了攻击实验. Linux 环境下,其它进程给攻击带来的噪声较小,S 盒在 Cache 中往往是对齐的(对于 64B 的 Cache 行大小,1KB 大小 S 盒恰好对应 16 个 Cache 行),高 4 位相同的密钥值对应模板匹配度基本相同. 这样,第一轮攻击一般可获取每个密钥字节的高 4 位(图 16);而在 Windows 环境下,由于操作系统进程和其它进程的影响,即使是高 4 位相同的密钥字节,对应的匹配度也有很大差异,大部分情况下可直接获取正确密钥字节候选值.

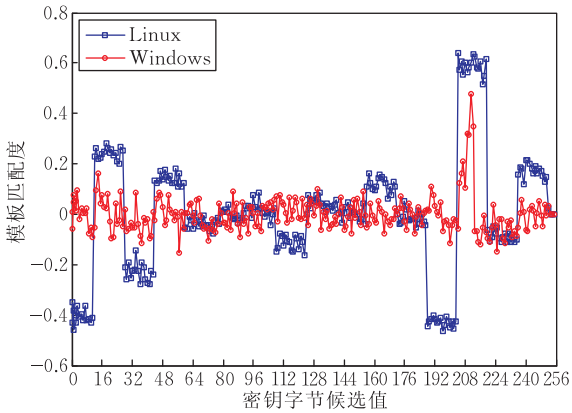


图 16　Linux 和 Windows 环境下 AES 第一轮密钥 k_0 的 256 个候选值匹配曲线

5.3　不同加密初始状态攻击

在前面本地攻击实验中,在加密前大都需进行 Cache 清空操作,使得加密前查找表都没有预先加载到 Cache 中,保证 Cache 的初始状态一致,实验效果非常理想. 当前的部分 AES 软件实现将查找表提前加载到 Cache 中,这样每次 Cache 访问都会发生 Cache 命中,可在一定程度上防御时序驱动 Cache 攻击.

我们尝试在加密前不清空 Cache 条件下,在 Linux 环境下对 OpenSSL-0.9.8a 进行了模板攻击,进行了两类实验. 一类是通过修改算法,将 AES 查找表预先加载到 Cache 中;此外,我们还在加密前不执行任何操作情况下进行了攻击实验,实验结果同加密前将 AES 所有查找表预先加载到 Cache 中攻击效果基本接近.

第一轮攻击中,2^{15} 个样本采集分析后可获取每个密钥字节的低 2～4 位,最终获取 48 位密钥(图 17).

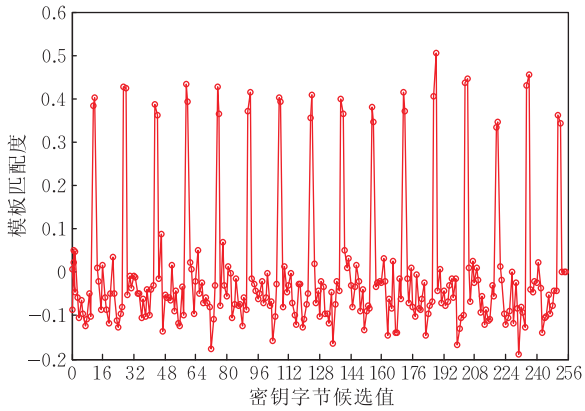


图 17　加密前预先加载 S 盒条件下 AES 第一轮 k_0 256 个候选值匹配曲线

最后一轮攻击中,2^{17} 个样本下可直接获取 k_2^9, k_3^9, k_6^9, k_{10}^9, k_{14}^9 5 个字节密钥(图 18).

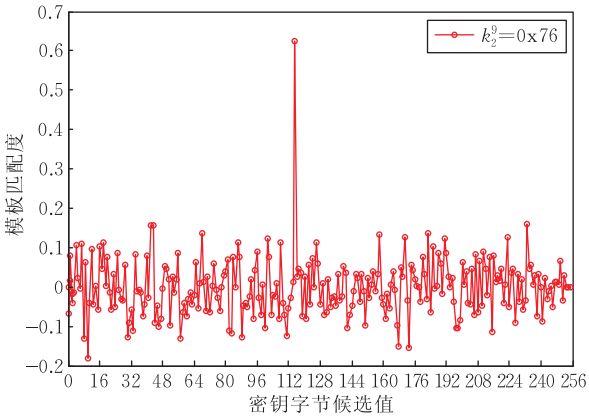


图 18　加密前预先加载 S 盒条件下 AES 最后一轮密钥字节 k_2^9 256 个候选值匹配曲线

5.4　不同密码库下攻击

为检验不同密码库中 AES 实现抗 Cache 计时模板攻击能力,我们以 Cache 计时外部模板攻击为例,针对 OpenSSL-0.9.8a、Miracl 5.0[39]、LibTomCrypt 1.17[40]、Crypto++ 5.6.1[41] 4 类典型密码库中的 AES 实现,在 Windows 和 Linux 环境下,在加密前 Cache 清空和 Cache 预热(提前加载 S 盒到 Cache 中)两种情况下,开展了大量的攻击实验,实验结果如表 4 所示.

表 4 不同密码库中 AES 实现 Cache 计时模板攻击结果比较

密码库	操作系统	攻击轮	加密初始状态	样本量	恢复密钥位
OpenSSL-0.9.8a	Windows	第一轮	Cache 清空	2^{21}	70 位
			Cache 预热	2^{20}	48 位
		最后一轮	Cache 清空	2^{20}	120 位
			Cache 预热	2^{19}	40 位
Miracl 5.0	Linux	第一轮	Cache 清空	2^{20}	128 位
			Cache 预热	2^{19}	96 位
		最后一轮	Cache 清空	2^{19}	128 位
			Cache 预热	2^{18}	120 位
LibTomCrypt 1.17	Windows	第一轮	Cache 清空	2^{21}	96 位
			Cache 预热	2^{23}	48 位
		最后一轮	Cache 清空	2^{22}	128 位
			Cache 预热	2^{22}	112 位
Crypto++5.6.1	Windows	第一轮	Cache 清空	2^{23}	21 位
			Cache 预热	2^{23}	21 位
		最后一轮	Cache 清空	2^{23}	40 位
			Cache 预热	2^{23}	72 位

由表 4 可知，这 4 种典型密码库中的最新 AES 实现仍易遭受 Cache 计时模板攻击的威胁，安全隐患较大. 此外，加密前进行 Cache 预热并不能很好地防御 Cache 计时模板攻击.

6 攻击防御措施

在加密前将查找表预先加载到 Cache 中，可有效防御 Cache 碰撞计时攻击、访问驱动 Cache 攻击、踪迹驱动 Cache 攻击，本文大量的实验证明了该防御措施并不能有效防御 Cache 计时模板攻击. 为了更好地防御此类攻击，可通过在算法实现中加入一定的随机时延[8,14,34-35]（如插入空指令、垃圾指令）来实现.

下面我们以 Linux 系统上 Miracl 5.0 密码库中 AES 实现为例，通过插入随机个（范围为 1～10）空指令，研究随机时延抗 Cache 计时模板攻击有效性. 表 5、图 19、图 20 为未加防御、预先加载 S 盒、预先加载 S 盒并加入随机时延 3 种情况下，对 AES 进行第一轮攻击和最后一轮攻击的结果.

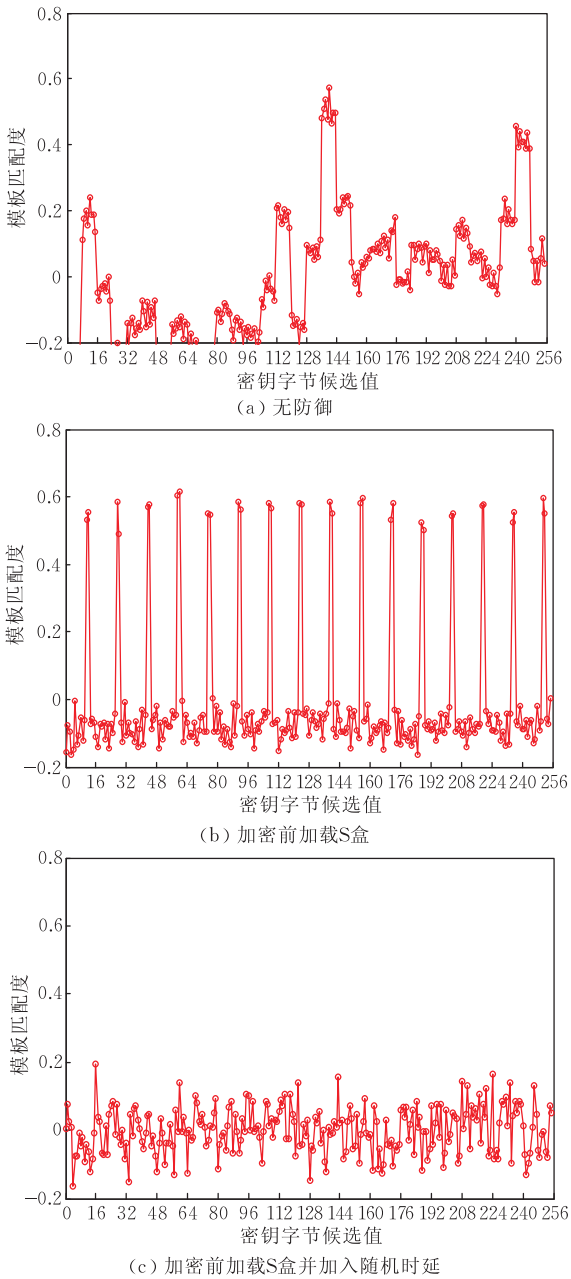
表 5 Miracl 5.0 密码库中几种 AES 实现攻击结果（2^{20} 样本）

AES 实现	第一轮某正确密钥字节(0x8c)模板匹配度排序	最后一轮某正确密钥字节(0xa1)模板匹配度排序	平均加密时钟周期
未加防御	1	1	889
预先加载 S 盒	24	1	887
预先加载 S 盒并加入随机时延	144	149	910

从图中可以看出：

(1) 未加防御措施的 Miracl 5.0 密码库中 AES 实现是不安全的（图 19(a)、图 20(a)）；

(2) 预先加载 S 盒到 Cache 后，Miracl 5.0 中 AES 执行效率接近于未加防御实现，但第一轮攻击可获取到密钥字节低 2～4 位（图 19(b)）；最后一轮

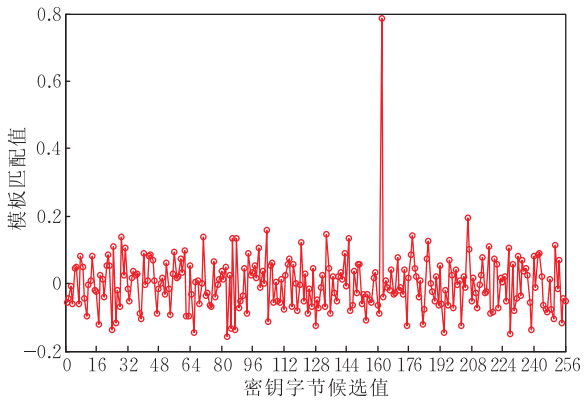


(a) 无防御

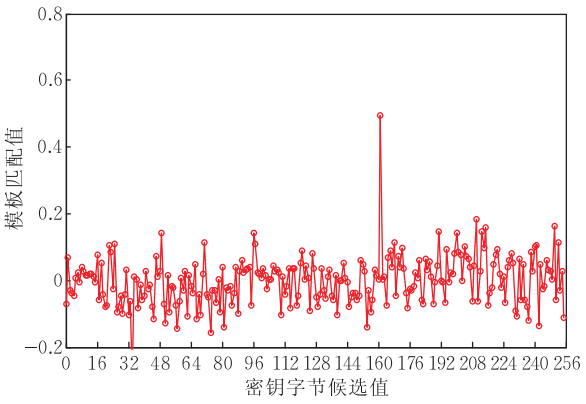
(b) 加密前加载S盒

(c) 加密前加载S盒并加入随机时延

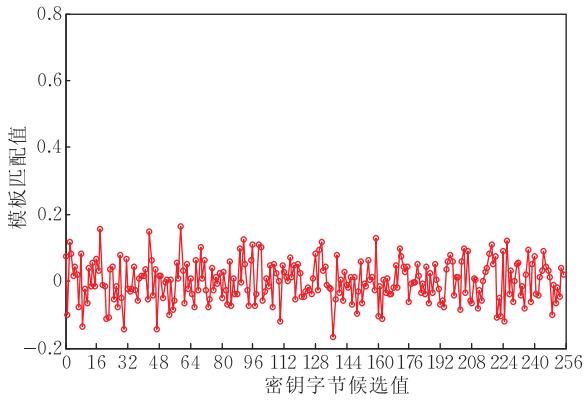
图 19 Miracl 5.0 密码库中 AES 第一轮攻击



(a) 无防御



(b) 加密前加载S盒



(c) 加密前加载S盒并加入随机时延

图 20　Miracl 5.0 密码库中 AES 最后一轮攻击

攻击正确密钥字节模板匹配度有所降低，但仍可攻击成功(图 20(b))；

(3) 加密前预先加载 S 盒到 Cache 中并加入随机时延的 Miracl 5.0 密码库中 AES 实现，其执行效率要高出未加防御的 AES 实现约 20 个时钟周期，但均可有效防御 AES 第一轮(图 19(c))和最后一轮攻击(图 20(c)).

需要注意的是如果算法防御仅加入时延而不进行加密前 Cache 预热处理，并不能有效防御访问驱动 Cache 攻击和踪迹驱动 Cache 攻击. 因此在 AES 实现过程中，将 Cache 预热和随机时延结合起来，可有效防御目前所有的 Cache 计时攻击.

7　总　结

本文对 AES Cache 计时模板攻击进行了研究，在对 Bernstein 的 AES 第一轮 Cache 计时模板攻击基础上，提出了一种新的基于 Pearson 相关性系数的 Cache 计时外部模板匹配算法，对 AES 第一轮和最后一轮进行了分析应用，然后利用一次加密不同次查找同一 S 盒时间搭建内部模板，提出了一种新的 Cache 计时内部模板分析模型，并对 AES 第一轮和最后一轮进行了分析应用；在两种模板分析的模型基础上，对不同操作系统、不同密码库、不同加密 Cache 初始状态，对 AES 实现抗 Cache 计时模板攻击能力进行了研究. 结果表明，加密前将 S 盒预先加载到 Cache 中并不能消除查找不同 S 盒索引的访问时间差异，进而不能有效防御 Cache 计时模板攻击；最新的各类密码库中 AES 实现仍然存在安全隐患，仍易遭受 Cache 计时模板攻击威胁；防御此类攻击可通过在 AES 加密首轮和末轮提前将查找表加载到 Cache 中，并增加一定的随机时延来实现.

参　考　文　献

[1] Kocher P C. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems//Proceedings of the Advances in Cryptography (CRYPTO 96). LNCS 1109. Santa Barbara, California, USA, 1996: 104-113

[2] Kelsey J, Schneier B, Wagner D et al. Side channel cryptanalysis of product ciphers. Journal of Computer Security, 2000, 8(2-3): 141-158

[3] Tsunoo Y, Tsujihara E, Minematsu K et al. Cryptanalysis of block ciphers implemented on computers with Cache//Proceedings of the International Symposium on Information Theory and Its Applications (ISITA 2002). Xi'an, China, 2002: 803-806

[4] Tsunoo Y, Saito T, Suzaki T et al. Cryptanalysis of DES implemented on computers with Cache//Proceedings of the Cryptographic Hardware and Embedded Systems (CHES 2003). LNCS 2779. Cologne, Germany, 2003: 62-76

[5] Tsunoo Y, Kubo H, Shigeri M et al. Timing attack on AES using Cache delay in S-boxes//Proceedings of the Symposium on Cryptography and Information Security (SCIS 2003). Hamamatsu, Japan, 2003: 179-184

[6] Bonneau J, Mironov I. Cache-collision timing attacks against AES//Proceedings of the Cryptographic Hardware and Embedded Systems (CHES 2006). Yokohama, Japan, 2006: 201-215

[7] Acıiçmez O, Schindler W, Koç Ç K. Cache-based remote Timing Attack on the AES//Proceedings of the Topics in Cryptology (CT-RSA 2007). San Fancisco, CA, USA, 2007: 271-286

[8] Bernstein D J. Cache-timing attacks on AES, 2005. Available online at http://cr. yp. to/papers. html\#cachetiming

[9] O'Hanlon M, Tonge A. Investigation of Cache-timing attacks on AES, 2005. Available online at http://www. computing. dcu. ie/research/papers/2005/0105. pdf

[10] Neve M, Seifert J, Wang Z. A refined look at Bernstein's AES side-channel analysis//Proceedings of ACM Symposium on Information, Computer and Communications Security (ASIACCS'06). Taipei, Taiwan, China, 2006: 369

[11] Canteaut A, Lauradoux C, Seznec A. Understanding Cache attacks. INRIA, Rocquencourt, France, Research Report RR-5881, 2006

[12] Percival C. Cache missing for fun and profit//Proceedings of the Technical BSD Conference 2005 (BSD2005). Ottawa, 2005: 1-13

[13] Osvik D A, Shamir A, Tromer E. Cache attacks and countermeasures: The case of AES//Proceedings of the Topics in Cryptology (CT-RSA 2006). San Jose, CA, USA, 2006: 1-20

[14] Tromer E, Osvik D A, Shamir A. Efficient Cache attacks on AES, and countermeasures. Journal of Cryptology, 2010, 23(1): 37-71

[15] Neve M, Seifert J P. Advances on access-driven Cache attacks on AES//Proceedings of the Selected Areas in Cryptography (SAC2006). Montreal, Quebec, Canada, 2007: 147-162

[16] Zhao X J, Wang T, Mi D. Robust first two rounds access driven Cache timing attack on AES//Proceedings of the International Conference on Computer Science and Software Engineering (CSSE 2008). Wuhan, China, 2008, 3: 785-788

[17] Zhao Xin-Jie, Wang Tao, Guo Shi-Ze et al. Research on access driven Cache timing attack against AES. Journal of Software, 2011, 22(3): 572-591(in Chinese)
(赵新杰，王韬，郭世泽等. AES 访问驱动 Cache 计时攻击研究. 软件学报，2011，22(3)：572-591)

[18] Bangerter E, Gullasch D, Krenn S. Cache games-bringing access-based Cache attacks on AES to practice//Proceedings of the IEEE Symposium on Security and Privacy (S&P 2011). Berleley, California, USA, 2011: 490-505

[19] Bangerter E, Gullasch D, Krenn S. Cache games-bringing access-based Cache attacks on AES to practice//Proceedings of the Constructive Side-Channel Analysis and Secure Design (COSADE 2011). Darmstadt, Germany, 2011: 215-221

[20] Zhao Xin-Jie, Wang Tao, Zheng Yuan-Yuan. Research on access driven Cache timing attacks against Camellia. Chinese Journal of Computers, 2010, 33(7): 1153-1164(in Chinese)
(赵新杰，王韬，郑媛媛. Camellia 访问驱动 Cache 计时攻击研究. 计算机学报，2010，33(7)：1153-1164)

[21] Zhao Xin-Jie, Guo Shi-Ze, Wang Tao et al. Access driven Cache timing template attack on ARIA. Journal of Huazhong University of Science and Technology (Nature Sciences Edition), 2011, 39(6): 62-65(in Chinese)
(赵新杰，郭世泽，王韬等. ARIA 访问驱动 Cache 计时模板攻击. 华中科技大学学报(自然科学版)，2011，39(6)：62-65)

[22] Zhao Xin-Jie, Wang Tao, Zheng Yuan-Yuan. Cache timing attack on SMS4. Journal on Communications, 2010, 31(6): 89-98(in Chinese)
(赵新杰，王韬，郑媛媛. 针对 SMS4 密码算法的 Cache 计时攻击. 通信学报，2010，31(6)：89-98)

[23] Zenner E. A Cache timing analysis of HC-256//Proceedings of the Selected Areas in Cryptography (SAC 2008). LNCS 5381. Sackville, New Brunswick, Canada, 2009: 199-213

[24] Brumley B B, Hakala R M, Nyberg K, Sovio S. Consecutive S-box lookups: A timing attack on SNOW 3G//Proceedings of the International Conference on Information and Communications Security (ICICS 2010). Barcelona, Spain, 2010: 171-185

[25] Leresteux D, Fouque P A, Chardin T. Cache timing analysis of RC4//Proceedings of the International Conference on Applied Cryptography and Network Security (ACNS 2011). LNCS 6715. Nerja, Spain, 2011: 110-129

[26] Page D. Theoretical use of Cache memory as a cryptanalytic side-channel. Department of Computer Science, University of Bristol: Technical Report CSTR-02-003, 2002: 1-47

[27] Bertoni G, Zaccaria V, Breveglieri L, Monchiero M, Palermo G. AES power attack based on induced Cache miss and countermeasure//Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC 2005). Washington, DC, USA, 2005: 586-591

[28] Lauradoux C. Collision attacks on processors with Cache and countermeasures//Proceedings of the Western European Workshop on Research in Cryptology (WEWoRC 2005). Leuven, Belgium, 2005: 76-85

[29] Acıiçmez O, Koç Ç K. Trace-driven Cache attacks on AES//Proceedings of the International Conference of Information and Communications Security (ICICS 2007). Raleigh, NC, USA, 2006: 112-121

[30] Bonneau J. Robust final-round Cache-trace attacks against AES. Cryptology ePrint Archive, 2006. Available at http://eprint. iacr. org/2006/374. pdf

[31] Zhao Xin-Jie, Guo Shi-Ze, Wang Tao, Liu Hui-Ying. Improved Cache trace attack on AES and CLEFIA by considering Cache miss and S-box misalignment. Journal on Communications, 2011, 32(8): 101-110(in Chinese)
(赵新杰，郭世泽，王韬，刘会英. 针对 AES 和 CLEFIA 的改进 Cache 踪迹驱动攻击. 通信学报，2011，32(8)：101-110)

[32] Gallais J, Kizhvatov I, Tunstall M. Improved trace-driven Cache-collision attacks against embedded AES//Proceedings of the Workshop on Information Security Applications (WISA 2011). Jeju Island, 2011: 243-257

[33] Gallais J, Kizhvatov I. Error-tolerance in trace-driven Cache collision attacks//Proceedings of the Constructive Side-Channel Analysis and Secure Design (COSADE 2011). Darmstadt, Germany, 2011: 222-232

[34] Brickell E, Graunke G, Neve M, Seifert S. Software mitigations to hedge AES against Cache-based software side-chan-

nel vulnerabilities. Cryptology ePrint Archive, 2006. Available online at http://eprint.iacr.org/2006/052.pdf

[35] Blömer J, Krummel V. Analysis of countermeasures against access driven Cache attacks on AES//Proceedings of the Selected Areas in Cryptography (SAC 2007). Ottawa, Canada, 2007: 96-109

[36] Wang Z, Lee R. New Cache designs for thwarting software Cache-based side channel attacks//Proceedings of the International Symposium on Computer Architecture (ISCA 2007). San Diego, California, USA, 2007: 494-505

[37] Daemen J, Rijmen V. The Design of Rijndael: AES — The Advanced Encryption Standard. Berlin Heidelberg: Springer-Verlag, 2002

[38] OpenSSL the open-source toolkit for SSL/TLS, 2010. Available online at http://www.openssl.org/

[39] Miracl — Multiprecision Integer and Rational Arithmetic C/C++ Library. Available on line at http://www.shamus.ie/

[40] LibTomCrypt. Available on line at http://libtom.org/

[41] Crypto++. Available on line at http://www.cryptopp.com/

WANG Tao, born in 1964, Ph. D., professor, Ph. D. supervisor. His main research interests include information security and cryptography.

ZHAO Xin-Jie, born in 1986, Ph. D. candidate. His main research interests include side channel analysis, fault analysis and combined analysis of block ciphers.

GUO Shi-Ze, born in 1969, Ph. D., researcher, Ph. D. supervisor. His main research interests include information security and cryptography.

ZHANG Fan, born in 1978, Ph. D.. His main research interests include side channel analysis and fault analysis in cryptography, computer architecture, security in wireless sensor network.

LIU Hui-Ying, born in 1984, Ph. D. candidate. His main research interests include algebraic side channel analysis of block cipher.

ZHENG Tian-Ming, born in 1985, M. S. candidate. His main research interests include satellite network security and cryptography.

Background

In 1996, Kocher found that the differences of the executing time can be used as a new approach to compromise the cryptosystems. Since then, there have been much active research in the area. Many block ciphers use large tables to improve the nonlinearity and the effciency of the software implementations. However, the table lookup indexes might be leaked through the execution time, which brings the threats of Cache timing attacks to many block ciphers. This paper is supported by the National Natural Science Foundation of China (Grant Nos. 60772082, 61173191), aiming to analyze the security of different cryptographic primitives when the leakages of Cache access time are available.

Due to the impact of micro-architectures and operating systems, the execution time of different S-Box lookups in block ciphers have some variance, which may cause the leakages of the S-Box lookup indexes. Utilizing the "*byte oriented and divide and conquer*" strategy in side channel analysis, this paper analyzes the resistance of AES implementations against Cache timing template attacks. First, the mechanism of different Cache access time is analyzed, and two Cache timing attacks are provided (collision-based and template-based). Second, the model of Cache timing external template attack is built, and a new template matching algorithm is proposed which is based on Pearson correlation factor. Two real attacks on the first and the last round of AES are launched successfully. To overcome the requirement of a template platform in external template attacks, an internal template attack is proposed and applied to AES. Finally, several extended attacks on AES are conducted under different settings, operating systems, Cache initial states, and crypto libraries. The experimental results are compared with the previous work, and an effective countermeasure is also suggested.

Experiment results show that: compared with external template attacks, internal template attacks can remove the requirement of recreating a target platform, improve the accuracy of the template to be built and enhance the efficiency of the whole attack. Since the transmission delay, or even the traffic jitter in a network, is bigger than the difference of different encryption time, it is difficult to launch the remote Cache timing template attacks. Preloading S-box into Cache before encryption can prevent Cache collision timing attacks, access driven Cache timing attacks and trace driven Cache timing attacks. However it cannot eliminate the timing differences of different S-box lookups. As a result, it cannot prevent Cache timing template attack efficiently. A possible countermeasure against Cache timing template attacks is to use the techniques of inserting delays and preloading S-Box simultaneously.