软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2011,22(3):572−591 [doi: 10.3724/SP.J.1001.2011.03802]

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

AES 访问驱动 Cache 计时攻击*

赵新杰[1+], 王 韬[1], 郭世泽[2], 郑媛媛[1]

[1](军械工程学院 计算机工程系,河北 石家庄 050003)

[2](北方电子设备研究所,北京 100083)

Access Driven Cache Timing Attack Against AES

ZHAO Xin-Jie[1+], WANG Tao[1], GUO Shi-Ze[2], ZHENG Yuan-Yuan[1]

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

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

+ Corresponding author: E-mail: zhaoxinjieem@163.com

Zhao XJ, Wang T, Guo SZ, Zheng YY. Access driven cache timing attack against AES. *Journal of Software*, 2011,22(3):572−591. http://www.jos.org.cn/1000-9825/3802.htm

Abstract: Firstly, this paper displays an access driven Cache timing attack model, proposes non-elimination and elimination two general methods to analyze Cache information leakage during AES encryption, and builds the Cache information leakage model. Next, it uses quantitative analysis to attack a sample with the above elimination analysis method, and provides some solutions for the potential problems of a real attack. Finally, this paper describes 12 local and remote attacks on AES in OpenSSL v.0.9.8a, v.0.9.8j. Experiment results demonstrate that: the access driven Cache timing attack has strong applicability in both local and remote environments; the AES lookup table and Cache structure decide that AES is vulnerable to this type of attack, the least sample size required to recover a full AES key is about 13; the last round AES implementation in OpenSSL v.0.9.8j, which abandoned the T_4 lookup table, cannot secure itself from the access driven Cache timing attack; the attack results strongly verify the correctness of the quantitative Cache information leakage theory and key analysis methods above.

Key words: AES; access driven; Cache timing attack; remote attack; OpenSSL

摘 要: 首先给出了访问驱动 Cache 计时攻击的模型,提出了该模型下直接分析、排除分析两种通用的 AES 加密泄漏 Cache 信息分析方法;然后建立了 AES 加密 Cache 信息泄露模型,并在此基础上对排除分析攻击所需样本量进行了定量分析,给出了攻击中可能遇到问题的解决方案;最后结合 OpenSSL v.0.9.8a,v.0.9.8j 中两种典型的 AES 实现在 Windows 环境下进行了本地和远程攻击共 12 个实验.实验结果表明,访问驱动 Cache 计时攻击在本地和远程均具有良好的可行性;AES 查找表和 Cache 结构本身决定了 AES 易遭受访问驱动 Cache 计时攻击威胁,攻击最小样本量仅为 13;去除 T_4 表的 OpenSSL v.0.9.8j 中 AES 最后一轮实现并不能防御该攻击;实验结果多次验证了 AES 加密 Cache 信息泄露和密钥分析理论的正确性.

关键词: 高级加密标准;访问驱动;Cache 计时攻击;远程攻击;OpenSSL

中图法分类号: TP309 **文献标识码**: A

* 基金项目: 国家自然科学基金(60772082); 河北省自然科学基金(08M010)
收稿时间: 2009-05-09; 定稿时间: 2009-12-02

AES 是密码界目前公认最安全的分组密码算法,具有很强的抗差分和线性分析能力.然而近年来的研究表明,正是由于 AES 的查找表结构,使其面临着一种基于 Cache 时间特性的旁路攻击威胁.

高速缓冲存储器 Cache 的行为信息可能作为密码破解的旁路思想,最早是由 Kocher[1]和 Kelsey[2]等人提出.2002 年,Page[3]提出了一种针对 DES 的 Cache 模型攻击方法,将 DES 密钥搜索空间由 56 位降低到 32 位,但其攻击方法尚不实用.2003 年,Tsunoo[4]通过分析 DES 查表索引和 Cache 访问特征实现了针对 DES 的首例真实 Cache 攻击,在一台 600-MHz Pentium III 的个人计算机上,用 2^{23} 个明文样本成功获取 DES 全部密钥.2004 年,Bernstein[5]在强制消除网络传输时延条件下实现了一种针对 OpenSSL[6]中 AES 的远程时序驱动计时攻击,其中加密服务端负责采集 AES 加密时间,本质上仍属本地计时攻击.2005 年,Bonneau[7]提出了一种利用密码程序在加密中的内部数据访问冲突导致的 Cache 命中和失效信息来进行密码分析的方法.2005 年,Percival[8]提出多线程间共享 Cache 存储器访问方式不仅提供了线程间数据一个简单、高带宽的泄露隐通道,而且也给恶意线程监视其他线程提供了入口,使得恶意线程能够窃取加密密钥,并设计实现了一种针对 RSA 的计时攻击.2005 年,Osvik[9]借鉴 Percival 信息采集方法实现了多例针对 AES 的 Cache 计时攻击.2007 年,Neve[10]将 Osvik 攻击[9]切入点转移到最后一轮,提出了一种新的 AES 最后一轮访问驱动 Cache 计时分析方法.2007 年,Acıiçmez 提出了一种新的旁路攻击概念:微架构攻击[11](microarchitectural attack,简称 MA),将攻击的微处理器部件单元由数据 Cache 扩展到包括数据 Cache、指令 Cache 和分支预测单元更大的部件范围,并提出了一种改进的 AES 时序驱动远程 Cache 计时攻击[12],其攻击端和加密服务端以 C/S 模式部署在同一 PC 机上,也是在强制消除网络传输时延的前提下成功实现攻击.

国内在密码 Cache 计时攻击方面起步比较晚,主要对 Cache 旁路攻击机理进行研究、对国外攻击进行复制或改进实验.文献[13]使用能量攻击对 MARS 和 Rijndael 进行了深入的分析;文献[14]对包括 Cache 攻击在内近 10 年的旁路攻击进展进行了系统介绍和分析;文献[15]介绍了基于 Cache 的 AES 攻击研究最新进展,分析了攻击的现实可行性,说明了反制攻击的措施建议;文献[16]参考 Bonneau 攻击[7]思想,在 Intel Celeron 1.99GHz 和 Pentium 4 3.6GHz CPU 环境中,分别在 2^{21} 和 2^{25} 个随机明文样本条件下,5 分钟内成功恢复 OpenSSL v.0.9.8a 库中 128 位 AES 密钥;文献[18]对 Bernstein 攻击[5]在 Pentium III,OpenSSL v.0.9.8a 和 Miracl 环境下进行了改进实验.

总的来说,国内外现有 Cache 计时攻击大多基于时序驱动和访问驱动方式展开.时序驱动攻击通过采集密码进程加解密整个时间,结合明文或密文进行密钥分析,所需样本量很大,现有远程攻击都是在强制消除网络传输时延前提下实现.在真实远程环境中,由于网络传输时延甚至其抖动时延噪声都远大于加密时间,远程攻击适用性不强.访问驱动攻击则绕开远程网络传输时延噪声问题,采用间谍进程采集密码进程加解密过程中访问 Cache 特征信息,然后结合加密算法开展密钥分析,既可以在本地实现,也可以被远程利用实现,攻击适用性比较强.但国外现有攻击都是针对本地环境开展的,Cache 信息分析方法比较少,且很少有公开实验细节和实验结果,目前也尚未见到国内外有关远程访问驱动攻击的报道.

本文的主要贡献是:

(1) 给出了针对 AES 的访问驱动 Cache 计时分析问题的通用描述,并提出了直接分析、排除分析两种通用的密钥分析方法;

(2) 基于概率统计理论,给出了一种通用的 AES 快速实现加密查表 Cache 信息泄露和排除分析所需样本量的定量分析方法,并对 OpenSSL v.0.9.8a 和 OpenSSL v.0.9.8j 中两种典型的 AES 快速实现算法第 1 轮和最后一轮加密 Cache 信息泄露、本地和远程攻击样本量进行了定量分析;

(3) 对访问驱动 Cache 计时攻击中的难点问题进行了分析,并给出了相应解决方案;

(4) 针对 OpenSSL v.0.9.8a,OpenSSL v.0.9.8j 密码库中两种典型的 AES 算法实现,利用上面的两种通用密钥分析方法,设计实验成功对 AES 进行多次本地和远程访问驱动计时攻击,验证了密钥分析方法、AES 加密 Cache 信息泄露与攻击样本量分析理论正确性.

本文的创新点在于:目前,国内外尚没有关于通用的针对 AES 的访问驱动 Cache 信息分析方法、加密 Cache

信息泄露和攻击所需样本量定量分析方法,也未见到有关利用访问驱动方式进行远程攻击的先例;且现有攻击大都针对 OpenSSL v.0.9.8a 中 AES 实现进行,从 OpenSSL v.0.9.8c~v.0.9.8j 版本均打了相应补丁,最后一轮使用 T_0,T_1,T_2,T_3 表代替 T_4 查找表,可有效防御已公布的 AES 最后一轮 Cache 计时攻击,目前也未见到有关针对 OpenSSL v.0.9.8j 中 AES 的最后一轮 Cache 计时攻击.我们的攻击实验结果表明,OpenSSL v.0.9.8j 依然不能有效防御访问驱动 Cache 计时攻击,并且攻击在本地和远程均可成功实施.

1 相关知识

1.1 Cache

现代微处理器大都使用高速缓存 Cache[19]来解决 CPU 与主存之间速度不匹配的问题.由于 Cache 访问命中和失效会带来时间和能量消耗差异,而大多数分组密码在加密过程中使用了 S 盒进行查表操作访问 Cache,其 Cache 访问特征可以通过时间特征信息泄露出来.图 1 中,密码进程查找 S 盒访问数据 B_1 时,其当前是否在 Cache 中的访问时钟周期有很大区别,因而 Cache 为密码进程加密提供了时间信息泄露源.

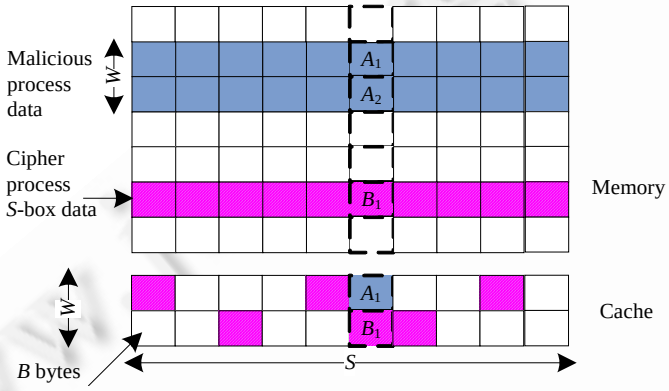


Fig.1 Cache information leakage analysis

图 1 Cache 信息泄漏分析

在组映像 Cache 中,多进程在主存中数据可能被映射到同一 Cache 组中,进而共享 Cache 存储空间.但这样会带来一个严重的问题,恶意进程可以通过对其私有数据 Cache 访问“命中”和“失效”带来的时间差异信息来推测其他进程的 Cache 访问特征.虽然 Cache 中的数据元素是受存储器保护的,攻击者无法直接获取,但是 Cache 访问地址信息却会被泄露出来.而 Cache 访问地址信息和分组密码查找表索引有密切关系,可被攻击者利用来进行密码分析.如图 1 所示,恶意进程数据 A_1 和 A_2 与密码进程的查表数据 B_1 被映射到同一 Cache 组中,其可以通过访问 A_1,A_2 的命中和失效信息来推测密码进程加密中是否访问过 B_1.由此可知,Cache 也为密码进程提供了时间信息泄露隐通道.

1.2 AES算法

AES 作为传统对称加密算法 DES 的替代者,2001 年被美国国家技术标准局(NIST)选用,其前身是 Rijmen 和 Daemen 提出的 Rijndael 算法,算法支持 128 位、192 位和 256 位密钥的不同版本.本文只关注 128 位密钥的 AES,192 位和 256 位密钥版本只是采用了不同的密钥扩展算法和更多的加密轮数.

AES 是一个基于有限域运算的迭代密码,首先输入 16 字节明文 P,输入的 16 字节密钥被扩展成 44 个 32 位字所组成的数组 $w[r]$,P 同初始密钥 K 进行异或作为初始输入状态变量;然后,第 r 轮迭代根据 16 字节的输入状态 x^{r-1} 和一个 16 字节的子密钥 K^r 产生一个 16 字节的输出状态 x^r.除最后一轮之外,每一轮迭代运算都包括对 x^{r-1} 的以下 4 种代数运算:字节代换、行移位、列混淆和轮子密钥 K^r 异或.

为提高加密速度,AES 加密算法快速实现[20]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

作预先进行计算,计算结果存储在多个查找表(OpenSSL v.0.9.8a 中,AES 快速实现使用了 T_0,T_1,T_2,T_3,T_4 共 5 个查找表;OpenSSL v.0.9.8j 中,AES 快速实现使用了 T_0,T_1,T_2,T_3 共 4 个查找表)中.整个 AES 加密过程就由 160 次查表和 176 次异或操作组成,执行效率非常高.但是,频繁的查表操作访问数据 Cache 也为其带来了严重的 Cache 计时攻击威胁.攻击者如果采集到足够多的 AES 加密 Cache 计时信息,将其转换为查表索引值,结合明文或密文信息就可以推断出某一轮扩展密钥信息.由于 AES 密钥扩展结构具有可逆性,攻击者只要恢复了中间任意轮扩展密钥就相当于恢复了初始密钥.这些都为针对 AES 的 Cache 计时攻击提供了很好的思路.

1.3 Cache计时攻击

Cache 计时攻击[11]主要利用密码算法在加解密过程中通过微部件单元 Cache 泄漏出来的时间信息,结合明文或密文信息来进行的分析技术.其原理是:相同的数据或指令在访问 Cache 时,由于涉及到的目标数据当前是否处于 Cache 中,同 Cache 的历史状态有关,因此,此时是否发生访问命中是不确定的.这将导致数据访问的不确定,而这种不确定性可以通过测量时间差异信息来获得.同时,由于这类时间差异信息和密钥往往是紧密相关的,所以只要采集到足够多的时间旁路信息,结合算法设计就有可能推测出密钥.

根据计时攻击的 Cache 部件不同[11],可将计时攻击分为针对数据 Cache、指令 Cache 两种.由于现代分组密码大多使用 S 盒查找表实现非线性混淆,而查找表需要对数据 Cache 进行访问,所以目前利用数据 Cache 进行的攻击对象主要为分组密码,如 DES,AES;同样,现代公钥加密系统大都利用经典数学问题的单向陷门特性,使用了大量的指令访问操作,其加解密过程中由于密钥位值不同所要进行的指令访问操作数目有很大的区别,此时会导致对指令 Cache 访问次数及时间存在很大区别.所以,目前利用指令 Cache 的攻击对象主要是公钥加密系统,如 RSA,ECC.因此,本文中的计时攻击主要针对数据 Cache.

根据所采集的时间信息不同[9],又可将计时攻击分为时序驱动、访问驱动、踪迹驱动 3 种方式.时序驱动攻击[5,7,9,12]采集的是密码进程整个加解密时间,采集方法简单,平台适用性强,但所需样本量大,一般都要百万计,离线分析方法比较复杂.更重要的是,在远程环境中,网络传输时延甚至是抖动时延都要远大于加密时间,采集到精确的加密时间显得极为困难,远程攻击适用性不强.访问驱动攻击[8,10]主要利用间谍进程采集密码进程加解密中访问的 Cache 组集合信息,采集方法比时序驱动稍显复杂,但分析方法比较简单,在木马植入技术日趋成熟的今天,攻击本地和远程实现可行性比较强.踪迹驱动攻击[21]比访问驱动攻击信息采集精度更高,攻击方需精确采集密码进程一次加解密过程中每次查表 Cache 访问的命中和失效信息,仅通过计时手段很难得以实现,一般都通过功耗检测手段进行,需物理接触密码设备,因此,攻击不论在本地还是远程通过计时实现可行性都不强.

根据对 Cache、AES 算法、计时攻击的研究可知,针对 AES 的访问驱动数据 Cache 计时攻击可行性比较强,可在本地乃至远程加以实现,这也是本文攻击方法选取的重要依据.

1.4 预先定义

为了更好地描述攻击,现将攻击中的一些参量给出以下定义和说明.

(1) OpenSSL v.0.9.8a 与 OpenSSL v.0.9.8j

本文主要描述针对 OpenSSL 密码库中 AES 快速实现算法的 Cache 计时攻击,典型的 AES 快速实现有两大版本,从 OpenSSL v.0.9.7~v.0.9.8b 中 AES 算法基本相同,为防御现有 Cache 计时攻击;OpenSSL v.0.9.8c~v.0.9.8j 将查找表 T_4 去除,通过查 T_0,T_1,T_2,T_3 表取而代之.本文主要介绍针对这两大版本 AES 快速实现的 Cache 计时攻击.为描述方便,下面章节中分别以 OpenSSL v.0.9.8a,v.0.9.8j 为例加以说明.

(2) Cache 参数

定义 Cache 行大小为 B 字节,每个 Cache 行由 δ 个元素组成,Cache 组个数为 S,每组由 W 个 Cache 行组成,即相联度为 W,整个 Cache 大小为 $S\times W\times B$ 字节.

(3) $M(G,S_l^{CS},S_l^y)$

$M(G,S_l^{CS},S_l^y)$ 代表所采集的访问驱动 Cache 计时信息,其中,G 为已知明文或密文,S_l^{CS} 为查 T_l 表过程中所访问的 Cache 组集合,S_l^y 为加密查 T_l 表的索引值集合.

(4) O_l

O_l为查找表 T_l起始元素在 Cache 行中的起始偏移量,根据说明(2),在确定查找表 T_l起始元素所在 Cache 行的前提下,O_l对应δ种可能性.

(5) ATP,AP,SP

AP 表示 AES 加密进程,SP 表示间谍进程,ATP 表示攻击端进程.

另外,若无特别说明,本文的 Cache 计时攻击特指针对 128 位 AES 算法的访问驱动数据 Cache 计时攻击.

2 访问驱动攻击模型

访问驱动 Cache 计时攻击主要利用 Cache 访问时间不确定性和 Cache 存储空间资源共享机制,利用间谍进程监视密码进程 Cache 访问操作,通过计时方法采集密码进程一次或多次加密后对间谍进程数据二次访问的 Cache“命中”和“失效”特征信息,间接得到加密进程的 Cache 访问信息并对其进行分析,预测加密查表索引信息,根据查表索引、明文密文对以及密钥之间的关系缩小密钥搜索空间.具体攻击模型如下:

(1) Cache 信息采集:令间谍进程 SP 和密码进程 AP 在同一 PC 机上并发运行,SP 和 AP 在主存中所占内存块分别为 SCB 和 ACB,SCB 大小和 Cache 存储空间大小相同(如图 2(a)、图 2(b)所示).在 AP 加密之前,SP 将私有内存数据全部加载到 Cache 中,清空 Cache(如图 2(c)所示);然后触发 AP 执行加密操作,AP 在加密过程中,会对 Cache 进行多次查表访问,将 SP 部分预先加载数据从 Cache 中驱逐出去(如图 2(d)所示);SP 在对所有私有数据进行再次访问时,对那些从 Cache 中驱除出来的数据进行访问会发生“Cache 失效”、需访问二级缓存或主存进而消耗较多的时钟周期,通过时间差异对比 SP 可得到 AP 在每次加密中查 T_i表所访问的 Cache 行或 Cache 组集合 S_i^{CS} 以及没有访问过的 Cache 行或 Cache 组集合 $\tilde{S}_i^{CS}$ (如图 2(e)所示).

(2) Cache 信息分析:根据所采集 Cache 旁路信息,利用查找表 Cache 组和查表索引之间的映射关系,得到可能的查表索引值集合 S_i^y 或不可能值集合 $\tilde{S}_i^y$,然后结合明文或密文通过分析预测密钥.

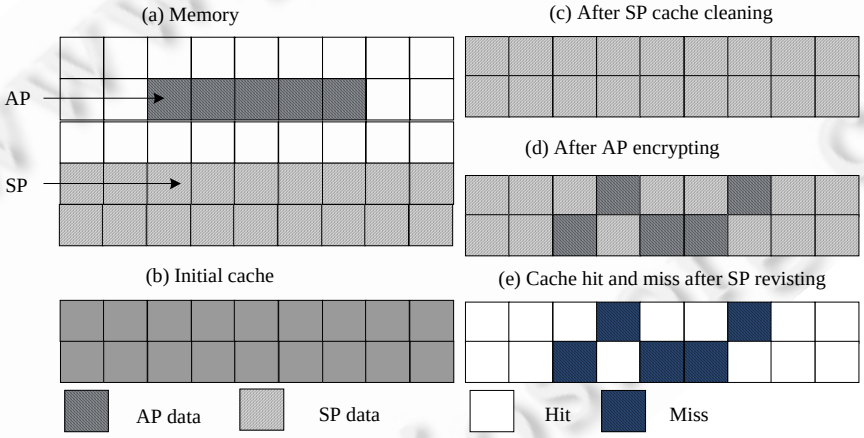


Fig.2 Access driven Cache timing attack model

图 2 访问驱动 Cache 计时攻击模型

3 信息分析方法

AES 访问驱动 Cache 信息分析主要利用所采集 AES 加密查表访问 Cache 信息,将其转换为查表索引或查表结果值,然后结合已知明文或密文信息进行.由于中间轮状态变量不像明文和密文一样,一般很难直接获取到,所以目前针对 AES 的 Cache 计时分析主要针对加密起始轮和结束轮,如第 1 轮和最后一轮.

3.1 问题描述

由第 1.2 节 AES 算法原理可知,128 位 AES 加密由多次查表、异或操作组成,第 1 轮输入状态变量为

$$x_i^0 = P_i \oplus K_i^0 \tag{1}$$

其中, x_i^0 为查 T_l 表(l=i mod 4)的输入索引;P_i, K_i^0 分别为明文和初始密钥第 i 个字节, K_i^0 也常表示为 K_i.

同样,对于 AES 加密最后一轮,输出状态变量即密文为

$$C_i = T_l[x_j^9] \oplus K_i^{10} \tag{2}$$

x_j^9 为最后一轮查 T_l 表索引(j=(5i) mod 16), $T_l[x_j^9]$ 为查表结果,C_i, K_i^{10} 分别为密文和最后一轮扩展密钥的第 i 个字节.

为了更好地描述攻击,给出公式(1)、公式(2)的通用描述:

$$G_i = U_j^{r-1} \oplus K_i^r \tag{3}$$

其中,r=(1,…,10);G_i 为攻击者可以直接获取的加密输入或输出状态值:在第 1 轮时表示明文字节 P_i,在最后一轮时表示密文字节 C_i;U_j^{r-1} 为加密过程中的查找表相关信息:在第 1 轮时表示查表索引值,在最后一轮时表示查表结果值; K_i^r 表示轮密钥第 i 个字节,在第 1 轮表示初始密钥 K_i,最后一轮表示扩展密钥 K_i^{10}.

公式(3)又等价于

$$K_i^r = G_i \oplus U_j^{r-1} \tag{4}$$

由公式(4)可知,如果我们采集到 AES 加密过程中查表访问 Cache 信息 U,再根据已知的明文或密文 G 信息,结合一定的分析方法,就有可能对 K 进行直接或间接推断,获取部分甚至全部密钥.

3.2 直接分析

直接分析方法主要根据公式(4)对扩展密钥 K^r 进行直接推断.首先对 K^r 的第 i 个字节 K_i^r 进行推断,具体步骤如下:

步骤 1. 筛选大聚类.

根据 G 字节数(128 位对应 16 个字节)将采集信息 M 分为 16 个大聚类,每个大聚类对应一个 G 字节值,然后将 M 中第 i 个字节 G_i 相同的样本分为第 i 个大聚类中的多个小聚类,便于步骤 2 进行分析.由于密钥 K 恒定,则扩展密钥 K^r 不变.如果 G_i 相同,由公式(4)可知,M 中必然包含一个共同的查表索引,即 M 中查找 T_l 表访问的 Cache 组集合中也必然包含一个相同 Cache 组.

步骤 2. 查找每个小聚类中查 T_l 表访问的共同 Cache 组.

对每个小聚类中查 T_l 表所访问的 Cache 组信息进行分析,查找每个小聚类中查 T_l 表共同访问的 Cache 组;

步骤 3. 利用不同小聚类 G_i 字节差异信息推断 K_i^r .

假设根据步骤 1 获得两个 G_i 小聚类, $G_i^a = 0\text{x}00, G_i^b = 0\text{x}01$,分别为这两个聚类第 i 个字节相同的密文字节,如由步骤 2 可知, G_i^a 和 G_i^b 所对应的共同访问 Cache 组分别为 CS_2,CS_5,则满足:

$$\begin{cases} K_i^r = G_i^a \oplus U_j^{a,r-1} \\ K_i^r = G_i^b \oplus U_j^{b,r-1} \end{cases} \Rightarrow G_i^a \oplus G_i^b = U_j^{a,r-1} \oplus U_j^{b,r-1} \tag{5}$$

G_i^a 对应的 Cache 组 CS_2 和 G_i^b 所对应的 Cache 组 CS_5 查 T_l 表索引或结果分别对应 16 个可能值,这两组可能值的 256 个组合中满足公式(5)的组合值可以通过有限次代入得到.

假如得到满足公式(5)的组合值 $U_j^{a,r-1} = 0\text{xfd}, U_j^{b,r-1} = 0\text{xfc}$,并将 G_i^a 和 $U_j^{a,r-1}$ 值代入公式(4),可得到 K_i^r 候选值为 0xfd.对步骤 2、步骤 3 进行多次迭代获取 K_i^r ,同样方法可以应用到密钥 K^r 其他字节候选值.

步骤 4. 由 K^r 候选值逆推初始密钥并进行密钥验证.

由步骤 3 可得 K^r 候选值集合 S,对 S 中每个 K^r 候选值通过密钥逆推得到初始密钥候选值 K,对相同明文使用真实密钥和预测密钥进行加密,如果密文相等则密钥验证成功.

3.3 排除分析

第 3.2 节公式(4)经进一步转换,有

$$K_i^r \neq G_i \oplus \tilde{U}_j^{r-1} \tag{6}$$

仍以 K_i^r 为例.令 V_i 代表 K_i^r 所有可能的候选值集合,V_i 初始有 256 个可能候选值,可表示为 V_i={q:0≤q≤255|q}.我们的目的是通过排除方法将 V_i 中错误候选值剔除出去,最终得到唯一候选值 $V_i=\{K_i^r\}$.假设 G_i=0x4c,一次加密中 16 次查 T_l 表访问 Cache 组集合信息见表 1:第 2 行表示间谍进程对 T_l 表对应 Cache 组进行二次访问所需时钟周期;H 表示间谍进程二次访问该 Cache 组发生“Cache 命中”,说明该 Cache 组在加密过程中查 T_l 表没有被访问过;M 则表示该 Cache 组在加密过程中查 T_l 表被访问过.

Table 1 T_l table related Cache sets data access patterns of spy process after an AES encryption

表 1 间谍进程采集一次 AES 加密后 T_l 表对应 Cache 组数据访问特征信息

Cache set	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Access clock cycles	13	14	13	14	3	14	14	13	3	15	15	14	14	14	4	15
Access pattern	*M*	*M*	*M*	*M*	*H*	*M*	*M*	*M*	*H*	*M*	*M*	*M*	*M*	*M*	*H*	*M*

由表 1 可知,一次加密中查 T_l 表访问到的 Cache 组集合 A={0,1,2,3,5,6,7,9,10,11,12,13,15},没有被访问到的 Cache 组集合 $\tilde{A}$={4,8,14}.排除分析法主要关注集合 $\tilde{A}$,令 N 表示 $\tilde{A}$ 中 Cache 组总数,显然,此时 N=3.

根据公式(6),每一个没有被访问到的 Cache 组对应的 16 个 $\tilde{U}_j^{r-1}$ 值同 G_i 异或结果将会从 V_i 中排除出去,则对于集合 $\tilde{A}$ 来说,理想状态下会得到 48 个不同 K_i^r 错误候选值,V_i 总数被减小为 256−48=208 个,然后通过对其他样本进行排除分析可将 V_i 中元素缩小为 1 个.同样方法可以应用到 K^r 其他字节.这样,对一定的样本 M 进行分析可推算出 K^r,然后根据密钥扩展算法逆推得到初始密钥候选值,经密钥验证得到正确初始密钥 K.

4 加密信息泄漏与攻击分析

4.1 Cache信息泄露模型

假定 AES 加密需要查 T_l 表 b 次,令 m 为 T_l 表在理想情况下(查表行和 Cache 组对齐时:即 Cache 行大小为 64 字节时,一个查表行往往对应一个 Cache 行所在的 Cache 组)对应 Cache 组个数,$p(b)$为查 T_l 表 b 次访问某一 Cache 组概率,$P(b)$为在 $p(b)$概率下对应 Cache 元素个数.同样有 $p_n(b)$为查找 T_l 表 b 次没有访问某一 Cache 组概率,$P_n(b)$为在 $p_n(b)$概率下对应 Cache 元素个数.对于一次查 T_l 表操作访问的某个 Cache 组来说

$$p(1)=1/m,\ p_n(1)=1-1/m \tag{7}$$

则一次 AES 加密查找 T_l 表 b 次均未访问某一 Cache 组概率为

$$p_n(b)=(1-1/m)^b \tag{8}$$

则 b 次查找 T_l 表没有访问的平均 Cache 组个数为

$$E(P_n(b))=m(1-1/m)^b \tag{9}$$

b 次查找 T_l 表访问过的平均 Cache 组个数为

$$E(P(b))=m(1-(1-1/m)^b) \tag{10}$$

在实际环境中采集 AES 加密查表访问 Cache 组信息时,由于系统进程和其他用户进程采集中会访问 Cache 给攻击带来一定干扰,同时在远程环境下网络发包、拆包也需多次访问 Cache 加剧这种干扰,因此所能采集到的查表访问过的 Cache 组就会多于理论值;同样,采集的加密查表没有访问的 Cache 组集合就会相应减少.

为了更好地描述分析过程,引入噪声变量 u,u 指在信息采集过程中由于其他进程、网络发包、拆包等噪声所访问的 Cache 组数目平均值和数据 Cache 组总数之比.此时,实际攻击所能采集到的 AES 加密查找 T_l 表未访问的平均 Cache 组数目有效值将降低到原来的 1−u 倍.将噪声变量 u 引入公式(9),进一步得到:

$$E(P_n(b))=m(1-u)(1-1/m)^b \tag{11}$$

同样,采集到的一次加密查找 T_l 表访问的平均 Cache 组数目为

$$E(P(b))=m(1-(1-u)(1-1/m)^b) \tag{12}$$

4.2 攻击样本量分析

4.2.1 通用分析模型

在第 4.1 节模型基础上,应用第 3.3 节排除分析方法,由公式(11)可知平均每个样本可以排除掉的 K_i^r 数目:

$$S=E(P_n(b))\times\delta \tag{13}$$

由第 3.2 节可知,V_i 代表 K_i^r 字节候选值集合,经过对 N 个随机样本进行排除分析,V_i 元素数量平均值为

$$N_{V_i}=256(1-S/256)^N=256(1-m\delta(1-u)(1-1/m)^b/256)^N \tag{14}$$

由 AES 加密原理及第 3.2 节分析方法可知,正确的 K_i^r 字节值是不可能被排除掉的,因而 $N_{V_i}\geqslant 1$.我们只需计算 N 多大时 N_{V_i} 值趋近于 1,也就得到排除分析攻击大致所需样本量.

4.2.2 OpenSSL v.0.9.8a 中 AES 攻击样本量分析

OpenSSL v.0.9.8a 中,AES 前 9 轮加密每轮分别对 T_0~T_3 这 4 个表进行 4 次共 144 次查表操作访问 Cache,最后一轮仅使用 T_4 表进行 16 次查表.每个查找表对应 256 个数据元素,共 16 行、16 列,每个元素 4 字节.对于 Cache 行大小为 64 字节(δ=16)来说,每个查找表对应 16 个 Cache 组,即 m=16.

对于第 1 轮加密进行信息分析时,由于采集的数据为一次加密 36 次查找 T_l(l=0,1,2,3)表后 Cache 信息,即 b=36,同时每个查找表 T_l 对应 4 个密钥字节值 K_i(i=4l,l=0,1,2,3);对最后一轮加密进行分析时,其共查找 T_4 表 16 次,即 b=16;同时,在本地攻击环境下 u 值接近于 0,可忽略不计.那么分别将参数 m,δ,u 和第 1 轮、最后一轮对应 b 值代入公式(7)~公式(14),得到表 2.

Table 2 Analysis of AES attack in OpenSSL v.0.9.8a

表 2 OpenSSL v.0.9.8a 中 AES 攻击分析

Parameter	First round	Last round
b	36	16
$p(1)$	1/16	1/16
$p_n(1)$	1–1/16	1–1/16
$p_n(b)$	0.098	0.356
$E(P_n(b))$	1.567	5.697
S	25.1	91.2
Local attack N_{V_i}	$256(1-25.1/256)^N$	$256(1-91.2/256)^N$
Remote attack N_{V_i}	$256(1-25.1(1-u)/256)^N$	$256(1-91.2(1-u)/256)^N$

由 AES 加密原理及第 3.2 节可知,正确的 K_i^{10} 值是不可能被排除的,因而 $N_{V_i}\geqslant 1$.我们只需计算 N 多大时 N_{V_i} 趋近于 1,也就得到排除分析攻击所需样本量.不同 N 值代入本地攻击 N_{V_i},得到 AES 加密第 1 轮和最后一轮排除分析本地攻击中,理论上所需攻击样本量和平均密钥字节候选值数量关系分别如图 3、图 4 所示.

易知,理论上 OpenSSL v.0.9.8a 中 AES 第 1 轮、最后一轮本地攻击可分别在 54~75,13~15 个样本将 K_i^{10} 错误候选值排除尽,获取 K^{10} 正确值,经逆推获取正确密钥 K.

在远程环境中对 AES 实施访问驱动 Cache 计时攻击,由于噪声变量 u 的存在,使得攻击所需样本量比本地攻击要大些.将不同的 u 值代入表 2 远程攻击 N_{V_i} 表达式,得到使 N_{V_i} 接近于 1 的 N 值,即为该 u 值下进行排除分析攻击所需最小样本量.AES 加密第 1 轮和最后一轮排除分析远程攻击理论上 u 值和攻击成功样本量 N 关系分别如图 5、图 6 所示.

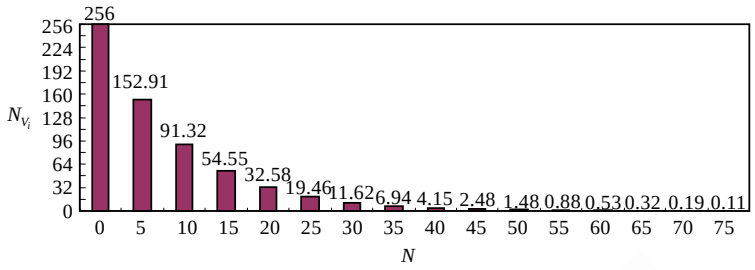


Fig.3 N and N_{V_i} in first round local attack

图 3 第 1 轮本地攻击 N 和 N_{V_i} 关系

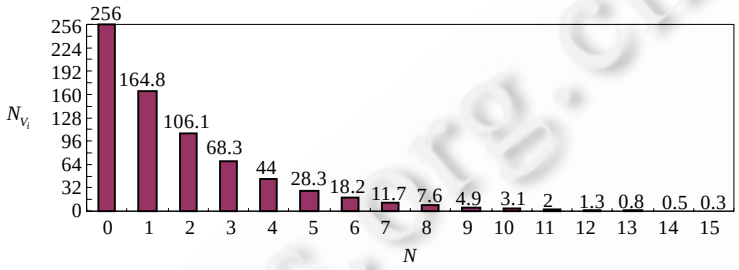


Fig.4 N and N_{V_i} in last round local attack

图 4 最后一轮本地攻击 N 和 N_{V_i} 关系

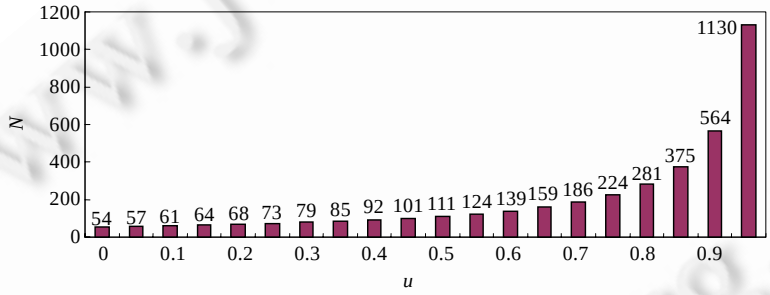


Fig.5 u and N in first round remote attack

图 5 第 1 轮远程攻击 u 和 N 关系

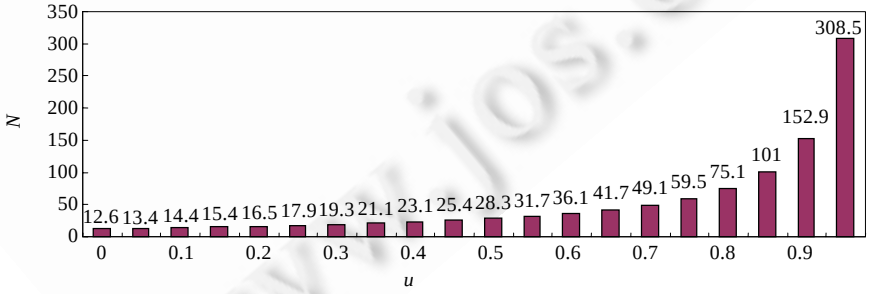


Fig.6 u and N in last round remote attack

图 6 最后一轮远程攻击 u 和 N 关系

4.2.3 OpenSSL v.0.9.8j 中 AES 攻击样本量分析

OpenSSL v.0.9.8j 中,AES 最后一轮加密中将查找表 T_4 去除,分别对 T_0,T_1,T_2,T_3 这 4 个表分别进行 4 次共 16

次查表操作访问 Cache.这样,128 位 AES 一次加密 10 轮分别查找 T_0,T_1,T_2,T_3 这 4 个表各 40 次共 160 次查表操作,即 b=40.那么分别将参数 m,δ,u 和第 1 轮、最后一轮对应 b 值代入公式(7)~公式(14),得到表 3.

Table 3 Analysis of AES attack in OpenSSL v.0.9.8j

表 3 OpenSSL v.0.9.8j 中 AES 攻击分析

Parameter	First round (last round)
B	40
$p(1)$	1/16
$p_n(1)$	1–1/16
$p_n(b)$	0.076
$E(P_n(b))$	1.211
S	19.4
Local attack N_{V_i}	$256(1-19.4/256)^N$
Remote attack N_{V_i}	$256(1-19.4(1-u)/256)^N$

不同 N 值代入表 3 本地攻击 N_{V_i} 表达式,得到 OpenSSL v.0.9.8j 中 AES 排除分析理论上所需攻击样本量和密钥搜索空间关系如图 7 所示.

由图 8 易知,OpenSSL v.0.9.8j 中 AES 最后一轮本地攻击中大约 80~100 个样本可将 K_i^{10} 错误候选值排除尽,获取 K^{10} 正确值,经逆推获取正确密钥 K.将不同的 u 值代入表 3 远程攻击 N_{V_i} 表达式,得到 OpenSSL v.0.9.8j 中 AES 加密第 1 轮和最后一轮排除分析远程攻击理论上 u 值和攻击成功样本量 N 关系如图 8 所示.

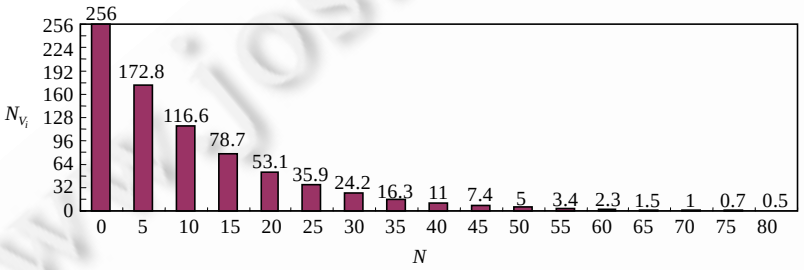


Fig.7 N and N_{V_i} in AES local attack

图 7 AES 本地攻击 N 和 N_{V_i} 关系

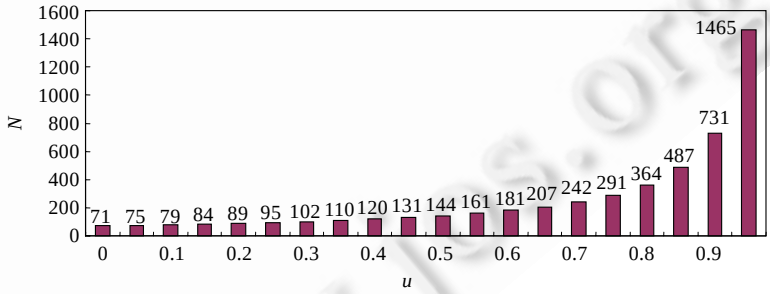


Fig.8 u and N in AES remote attack

图 8 AES 远程攻击 u 和 N 关系

5 攻击难点分析及解决方案

5.1 查找表定位

为采集 AES 进程加密所访问的 Cache 组信息,攻击者需定位 4 个(OpenSSL v.0.9.8j)或 5 个(OpenSSL

v.0.9.8a)查找表映射 Cache 组的起始位置.以 OpenSSL v.0.9.8a 中 5 个查找表为例,具体定位方法如下:

令 AES 加密进程 AP 和间谍进程 SP 运行在同一台 PC 机上,假设 SP 所占内存空间为与 $L1$ 数据 Cache 等大小的字节数组 $A[0,\ldots,S\times W\times B-1]$.

(a) 通知 SP 启动,从 A 中每隔 B 字节大小读取数组数据清空 Cache,初始化 Cache 为已知状态;

(b) 触发 AP 执行基于随机明文 P 加密操作;

(c) SP 检测到 AP 执行完毕后,立即对 A 中所有数据进行再次访问,采集每个 Cache 组访问时钟周期;

(d) 执行步骤(a)~步骤(c)对 10~12 个样本进行旁路信息采集,多次采集可得到对所有 Cache 组访问平均时钟周期.由于一次 AES 加密要对 5 个查找表进行 160 次访问,多样本采集时可确保 5 个查找表对应所有 Cache 组在加密过程中均被访问过,其所占 Cache 组区域平均访问时钟周期必然远远大于其他区域.因此,在采集到对多样本所有 Cache 组平均访问时钟周期后,可看到连续的很多个平均访问时钟周期较大的 Cache 组,其起始位置显然就是 AES 的 5 个查找表对应 Cache 组起始位置.

对于 Athlon 64 3000+ 1.81 GHz 典型处理器来说,$L1$ Cache 大小为 64KB,Cache 行大小为 64B,两路组相联,共 512 个 Cache 组,AES 的 5 个 1KB 大小查找表对应 80~81 个连续 Cache 组区域.如图 9 所示,其中横坐标表示 Cache 组序号,纵坐标表示 Cache 组访问时钟周期.其中,388~468 之间连续 81 个 Cache 组访问时钟周期比较长,易知 388 即为 T_0 查找表所映射起始 Cache 组,452 为 T_4 查找表所映射起始 Cache 组.

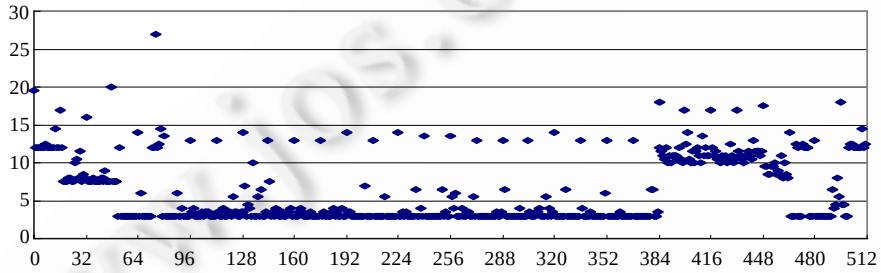


Fig.9 Average access cycles of all Cache sets

图 9 Cache 中所有组平均访问时钟周期采样图

5.2 查表索引同Cache组映射

AES 每个查找表为 1KB,每个元素 4 字节,由 16 个行、列 256 个元素组成.对于 64B 的 Cache 行大小,由于查找表元素在 Cache 中存储是连续的,理论上来说,一个 Cache 组可映射一个查表行中 16 个元素.同样,一个查表行可映射到一个 Cache 组.而实际情况往往不是这样,实验结果表明,查表行和 Cache 组并不是一一对应的,而是不对齐的,一个 Cache 组中的 Cache 行往往对应两个不同的连续查表行所对应的 16 个查找表元素.同样,一个查找表行常常对应两个不同的连续 Cache 组的 Cache 行中的 16 个 Cache 元素,如图 10 所示.

实验中,对于 64B Cache 行大小,T_1 查找表第 1 个元素在对应 Cache 组的 Cache 行中起始位置可能性有 16 种,攻击需对每种可能性执行离线分析操作.对于排除分析法来说,正确的起始位置进行分析后最终会得到唯一的密钥候选值;反之,错误起始位置对应分析结果往往将所有每个密钥字节 256 个候选值全部排除掉.

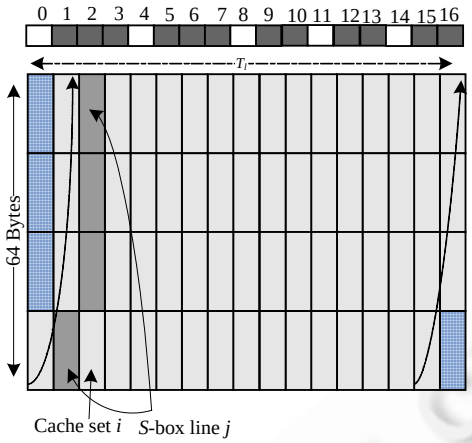


Fig.10 T_l related Cache set and lookup index mapping

图 10 T_l 表对应 Cache 组和查表索引映射关系

6 攻击应用

应用第 3 节信息分析方法,本节主要攻击对象为 OpenSSL 密码库中 v.0.9.8a,v.0.9.8j 两个典型版本中 AES 快速实现,攻击轮为 AES 加密第 1 轮或最后一轮,攻击方式为访问驱动 Cache 计时攻击.

6.1 算法分析

OpenSSL v.0.9.a 中,AES 快速实现将除与轮密钥异或以外的加密操作合并为查表操作,预先进行计算,计算结果存储在 T_0,T_1,T_2,T_3,T_4 这 5 个查找表中.前 9 轮中每轮分别查找 T_0,T_1,T_2,T_3 表各 4 次,最后一轮仅对 T_4 表执行 16 次查表操作,其前 9 轮和最后一轮加密分别如公式(15)、公式(16)所示:

$$\begin{cases}(x_0^r,x_1^r,x_2^r,x_3^r)=T_0[x_0^{r-1}]\oplus T_1[x_5^{r-1}]\oplus T_2[x_{10}^{r-1}]\oplus T_3[x_{15}^{r-1}]\oplus(K_0^r,K_1^r,K_2^r,K_3^r)\\(x_4^r,x_5^r,x_6^r,x_7^r)=T_0[x_4^{r-1}]\oplus T_1[x_9^{r-1}]\oplus T_2[x_{14}^{r-1}]\oplus T_3[x_3^{r-1}]\oplus(K_4^r,K_5^r,K_6^r,K_7^r)\\(x_8^r,x_9^r,x_{10}^r,x_{11}^r)=T_0[x_8^{r-1}]\oplus T_1[x_{13}^{r-1}]\oplus T_2[x_2^{r-1}]\oplus T_3[x_7^{r-1}]\oplus(K_8^r,K_9^r,K_{10}^r,K_{11}^r)\\(x_{12}^r,x_{13}^r,x_{14}^r,x_{15}^r)=T_0[x_{12}^{r-1}]\oplus T_1[x_1^{r-1}]\oplus T_2[x_6^{r-1}]\oplus T_3[x_{11}^{r-1}]\oplus(K_{12}^r,K_{13}^r,K_{14}^r,K_{15}^r)\end{cases}\tag{15}$$

$$\begin{cases}(C_0,C_1,C_2,C_3)=(T_4[x_0^9]\oplus K_0^{10},T_4[x_5^9]\oplus K_1^{10},T_4[x_{10}^9]\oplus K_2^{10},T_4[x_{15}^9]\oplus K_3^{10})\\(C_4,C_5,C_6,C_7)=(T_4[x_4^9]\oplus K_4^{10},T_4[x_9^9]\oplus K_5^{10},T_4[x_{14}^9]\oplus K_6^{10},T_4[x_3^9]\oplus K_7^{10})\\(C_8,C_9,C_{10},C_{11})=(T_4[x_8^9]\oplus K_8^{10},T_4[x_{13}^9]\oplus K_9^{10},T_4[x_2^9]\oplus K_{10}^{10},T_4[x_7^9]\oplus K_{11}^{10})\\(C_{12},C_{13},C_{14},C_{15})=(T_4[x_{12}^9]\oplus K_{12}^{10},T_4[x_1^9]\oplus K_{13}^{10},T_4[x_6^9]\oplus K_{14}^{10},T_4[x_{11}^9]\oplus K_{15}^{10})\end{cases}\tag{16}$$

其中,K^{10} 为最后一轮 AES 扩展密钥;C 为密文;x^9 为最后一轮输入状态变量,也是查 T_4 表输入索引.

近年来,密码界针对 OpenSSL v.0.9.8a 中 AES 快速实现提出了多种 Cache 计时攻击方法[5,7,9,10,12].OpenSSL v.0.9.8j 中,AES 实现为抵御上述中最后一轮攻击[7,10],最后一轮放弃使用 T_4 表,取而代之的是使用 T_0,T_1,T_2,T_3 这 4 个表进行查表操作,该补丁可有效防御上述攻击.这样,最后一轮分别查找 T_0,T_1,T_2,T_3 表各 4 次,共 16 次查表操作,其前 9 轮加密与公式(15)相同,最后一轮加密见公式(17):

$$\begin{cases}(C_0,C_1,C_2,C_3)=(T_2[x_0^9]\oplus K_0^{10},T_3[x_5^9]\oplus K_1^{10},T_0[x_{10}^9]\oplus K_2^{10},T_1[x_{15}^9]\oplus K_3^{10})\\(C_4,C_5,C_6,C_7)=(T_2[x_4^9]\oplus K_4^{10},T_3[x_9^9]\oplus K_5^{10},T_0[x_{14}^9]\oplus K_6^{10},T_4[x_3^9]\oplus K_7^{10})\\(C_8,C_9,C_{10},C_{11})=(T_2[x_8^9]\oplus K_8^{10},T_3[x_{13}^9]\oplus K_9^{10},T_0[x_2^9]\oplus K_{10}^{10},T_4[x_7^9]\oplus K_{11}^{10})\\(C_{12},C_{13},C_{14},C_{15})=(T_2[x_{12}^9]\oplus K_{12}^{10},T_3[x_1^9]\oplus K_{13}^{10},T_0[x_6^9]\oplus K_{14}^{10},T_4[x_{11}^9]\oplus K_{15}^{10})\end{cases}\tag{17}$$

改进后的 AES 最后一轮使用了多个查找表,可有效降低查表访问碰撞发生率,进而有效防御攻击[7,10],目前,国内外尚未见到针对 OpenSSL v.0.9.8j 版本中 AES 最后一轮成功实施的攻击.下面将详细描述针对 OpenSSL v.0.9.8a,v.0.9.8j 中 AES 的访问驱动 Cache 计时攻击.

6.2 信息采集与分析

6.2.1 信息采集

将第 2 节攻击模型应用到针对 OpenSSL 密码库 AES 攻击中,在本地攻击实验时,为攻击端 ATP 分配与 $L1$ 数据 Cache 大小相等的字节数组 $A[0,\ldots,S\times W\times B-1]$,通过调用 OpenSSL 中加密库函数触发 AES 进程 AP 执行加密,并在加密前后对 A 进行访问来模拟间谍进程 SP;而在远程攻击实验时,将 ATP 和 AES 加密服务端 AP 部署在网络中不同 PC 机上,令间谍进程 SP 和 AP 运行在同一 PC 机上,为 SP 分配与 $L1$ 数据 Cache 大小相等的字节数组 $A[0,\ldots,S\times W\times B-1]$,信息采集过程如图 11 所示,采集步骤如下:

步骤 1. ATP 通知 SP 从 A 中每隔 B 字节读取数组数据清空 Cache,初始化 Cache 为已知状态;

步骤 2. ATP 向 AP 发送随机明文 P,触发 AP 执行加密,并将密文 C 发送给 ATP;

步骤 3. ATP 收到 C 后,通知 SP 对数组 A 每隔 B 字节对数据进行再次访问采集一次 AES 加密过程中查 T_4 表访问的 Cache 组集合 S_l^{CS};

步骤 4. ATP 根据查 T_l 表访问 Cache 组和查 T_l 表索引之间映射关系将 S_l^{CS} 转换为加密查表索引集合 S_l^y.

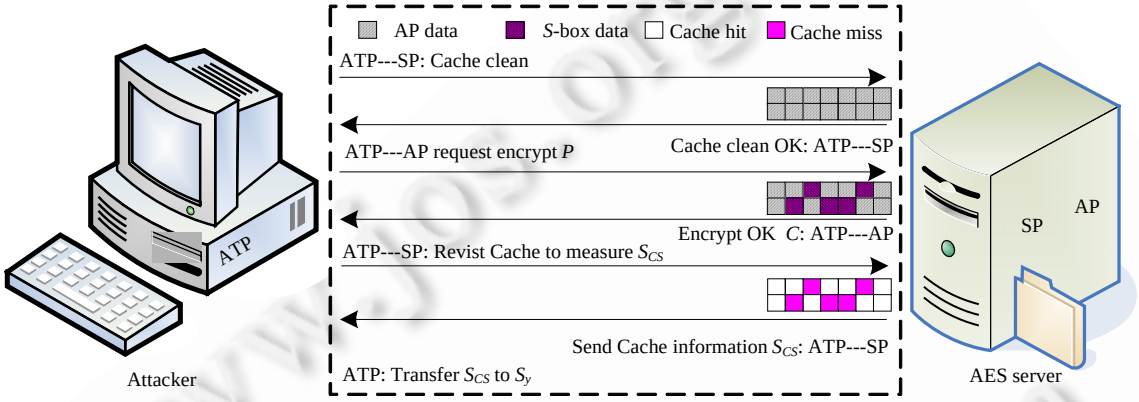


Fig.11 Measurement process of the attack

图 11 攻击采集过程

6.2.2 信息分析

OpenSSL v.0.9.8a 和 OpenSSL v.0.9.8j 中 AES 实现第 1 轮分析时,公式(15)中 AES 加密第 1 轮输入查表索引 x_i^0 可表示为

$$x_i^0 = P_i \oplus K_i \Rightarrow K_i = P_i \oplus x_i^0 \tag{18}$$

参考公式(16),给出 OpenSSL v.0.9.8a 中 AES 最后一轮加密公式等价表示:

$$C_i = T_4[x_j^9] \oplus K_i^{10} \Rightarrow K_i^{10} = C_i \oplus T_4[x_j^9], j=(5i) \bmod 16 \tag{19}$$

同样,参考公式(17),给出 OpenSSL v.0.9.8j 中 AES 最后一轮加密公式等价表示:

$$C_i = T_l[x_j^9] \oplus K_i^{10} \Rightarrow K_i^{10} = C_i \oplus T_l[x_j^9], l=(i+2) \bmod 4, j=(5i) \bmod 16 \tag{20}$$

将公式(18)~公式(20)与公式(4)进行比较不难发现,公式(18)中的 P_i 和 x_i^0、公式(19)中的 C_i 和 $T_4[x_j^9]$、公式(20)中的 C_i 和 $T_l[x_j^9]$ 都分别等价于公式(4)中的 G_i, U_j^{r-1} 变量,只不过 G_i 在第 1 轮中表示明文字节,最后一轮表示密文字节;U_j^{r-1} 在第 1 轮表示查表输入索引值,在最后一轮表示查表结果值.这 3 种情况都可以根据第 3.2 节直接分析、第 3.3 节排除分析两种方法对密钥进行推断.

需要说明的是:

第 1 轮分析时,如果加密明文字节的取值范围为 a~z 这 26 个字母,最后一轮分析时由于 AES 加密的雪崩性,密文字节的取值范围为 0~255 之间随机的字节,那么在使用排除分析方法对密钥进行预测时,最后一轮攻击排

除的不可能密钥字节值比较随机,所需的样本量要远小于第 1 轮攻击,后面攻击实验也证明了这一点.

同时,查找表第 1 个字节在 Cache 组中的起始位置 O_l 对第 1 轮分析样本量也有一定影响:

(1) O_l=0

如果查找表第 1 个字节在 Cache 组中是对齐的,即其恰好对应 Cache 组的第 1 个字节,那么利用所采集的样本加密没有访问的每个 Cache 组将对应着 16 个高 4 位相同、低 4 位连续的索引字节,其每次将排除掉 16 个高 4 位相同、低 4 位连续的密钥字节不可能值,而正确的密钥字节值是不可能被排除掉的.同样,与正确的密钥字节高 4 位相同、低 4 位不同的另外 15 个密钥字节也是不可能被排除掉的.这种情况下进行第 1 轮排除分析,只能将密钥搜索空间降低到 2^{64},如果要进一步缩小密钥搜索空间,可通过对第 2 轮进行分析来完成.

(2) $O_l\neq 0$

如果查找表第 1 个字节在 Cache 组中不是对齐的,如图 10 所示,那么加密没有访问的每个 Cache 组将对应着两个查表行:第 1 个查表行对应 16–O_l 个索引,其高 4 位 y^{H4} 相同、低 4 位 ($y_a^{L4}\in\{O_l,...,16\}$) 连续;第 2 个查表行对应 O_l 个索引,其高 4 位 ($y_b^{H4}=y_a^{H4}+1$) 相同、低 4 位 ($y_b^{L4}\in\{0,...,O_l\}$) 连续.这样,每次排除的 16 个密钥值和索引值一样,也对应着两个密钥行:第 1 个行 a 对应 16–O_l 个值,其高 4 位相同、低 4 位连续;第 2 个行对应 O_l 个值,其高 4 位相同、低 4 位连续.易见,这两个行的数量越接近,即当 O_l=8 时,排除效果越好,攻击所需样本量也小.

在最后一轮排除分析时,由于 AES 查表的非线性,不论此时查找表第 1 个字节在 Cache 组中是不是对齐的,那么利用所采集的样本加密没有访问的每个 Cache 组对应的 16 个查表结果值都是很随机的,其高 4 位和低 4 位基本不相同,那么进行排除分析十分有效.因此,此时,O_l 取值对排除分析所需样本量大小没有影响.

6.3 实验环境

6.3.1 本地攻击环境

为了能够快速、准确地定位 AES 查找表对应的 Cache 组区域,采集到加密查表访问 Cache 信息,我们选用 $L1$ 数据 Cache 较大、相连度较低的 AMD 64 位 CPU 作为被攻击机器平台,同时选用 OpenSSL v.0.9.8a,v.0.9.8j 密码库中两个版本的 AES 快速实现作为被攻击加密软件平台,攻击实验环境配置见表 4.

Table 4 Environment configuration of AES local attack

表 4 AES 本地攻击实验环境配置

Configuration	Specific configuration	
Operating system	Windows XP Professional	
OpenSSL	OpenSSL v.0.9.8a, v.0.9.8j	
CPU	Athlon 64 3000+ 1.81 GHz	
$L1$ Cache	Cache size: 64KB	Associative size: 2 way
	Cache line size: 64B	
$L2$ Cache	Cache size: 512KB	Associative size: 16 way
	Cache line size: 64B	

6.3.2 远程攻击环境

在远程攻击实验中,攻击端和加密服务端被配置在局域网和校园网两种环境中.其中,攻击对象和被攻击端机器配置不变,攻击端的配置则比较灵活.具体配置见表 5.

Table 5 Environment configuration of AES remote attack

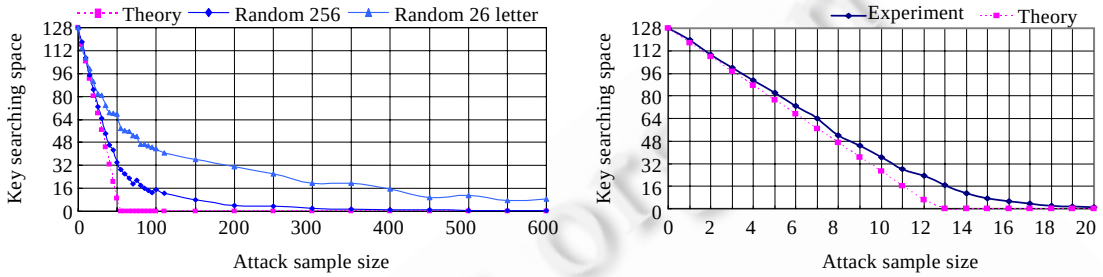
表 5 AES 远程攻击实验环境配置

Configuration	Operating system	CPU	L1 Cache			IP
			Cache size	Associative size	Line size	
Campus network attack client	Windows XP	Intel(R)Celeron(TM)1.3GHz	16KB	4 way	32B	10.10.153.4
Local network attack client	Windows 2000	Intel(R)Pentium(R) 3.00GHz	16KB	8 way	64B	192.168.2.242
AES server	Windows XP	Athlon 64 3000+ 1.81 GHz	64KB	2 way	64B	23.104.223.205

6.4 OpenSSL v.0.9.8a中AES攻击实验

6.4.1 本地攻击实验

OpenSSL v.0.9.8a 中,AES 本地攻击排除分析本地攻击样本量 N 和密钥搜索空间关系如图 12 所示.实验结果表明:第 1 轮攻击使用 a~z 这 26 个字母产生的随机明文,直接分析和排除分析可分别在 1200 个、400 个样本左右快速恢复 AES 完整密钥,攻击所需样本量和明文发散度成反比;最后一轮攻击直接分析和排除分析可分别在 2000 个、13 个样本左右恢复 AES 密钥.由于密文字节和查表结果的扩散性,最后一轮排除分析攻击样本量远小于第 1 轮.随机 256 明文条件下,AES 实际攻击的样本量和密钥搜索空间关系基本接近于理论值,证明了第 4 节模型和样本量分析正确性.



(a) First round
(a) 第 1 轮

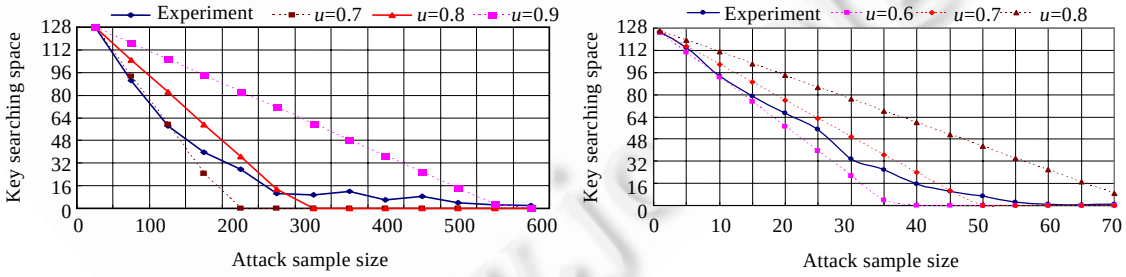
(b) Last round
(b) 最后一轮

Fig.12 Relationships between N and key searching space by elimination method in local attack

图 12 排除分析本地攻击样本量 N 和密钥搜索空间关系

6.4.2 远程攻击实验

OpenSSL v.0.9.8a 中,AES 远程攻击排除分析本地攻击样本量 N 和密钥搜索空间关系如图 13 所示.实验结果表明:随机 256 明文条件下,局域网环境中 AES 加密第 1 轮和最后一轮攻击可分别在 250 个、40 个样本左右恢复密钥;校园网环境下,最后一轮攻击 100 个样本即可恢复密钥;远程攻击所需样本量比本地攻击要稍大些,但只是线性增长;局域网攻击实验中,噪声因子为 0.6~0.8 左右,实际攻击样本量和密钥搜索空间关系曲线也介于 u 为 0.6~0.9 之间的远程攻击理论值曲线之间,证明了第 4 节远程攻击 Cache 信息泄露模型和攻击所需样本量分析理论正确性.



(a) First round
(a) 第 1 轮

(b) Last round
(b) 最后一轮

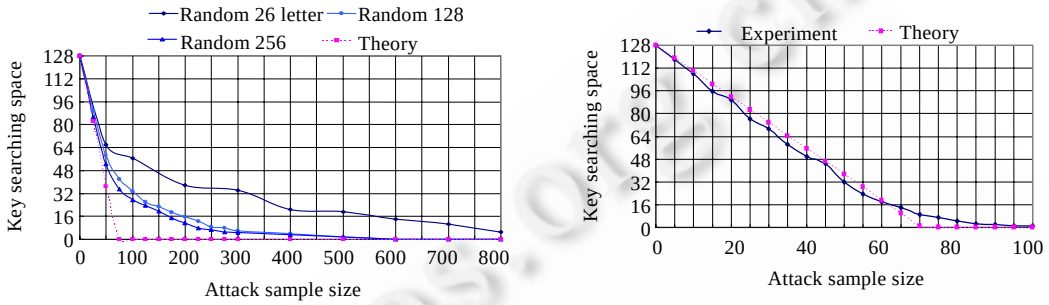
Fig.13 Relationships between N and key searching space by elimination method in remote attack

图 13 排除分析远程攻击样本量 N 和密钥搜索空间关系

6.5 OpenSSL v.0.9.8j中AES攻击实验

6.5.1 本地攻击实验

OpenSSL v.0.9.8j 中,AES 本地攻击排除分析本地攻击样本量 N 和密钥搜索空间关系如图 14 所示.实验结果表明,第 1 轮攻击中,随机 26 字母明文条件下直接分析可在 1600 个样本左右恢复 AES 密钥,随机 256、128、26 字母明文条件下排除分析可分别在 170、200、550 个样本恢复 AES 密钥;最后一轮攻击中,直接分析和排除分析可分别在 12000 个、64 个样本左右恢复 AES 完整密钥;由于 v.0.9.8j 中采集的是 40 次查 T_0~T_3 表后的 Cache 组信息,同 v.0.9.8a 中第 1 轮攻击中的查 T_0~T_3 表 36 次和最后一轮攻击中的 16 次查 T_4 表相比,排除分析利用的加密没有访问 Cache 组数量更小,进而 v.0.9.8j 中 AES 攻击所需样本量要大于 v.0.9.8a;同时,实验结果也再次验证了第 4 节 Cache 信息泄露模型和攻击所需样本量分析理论正确性.



(a) First round
(a) 第 1 轮

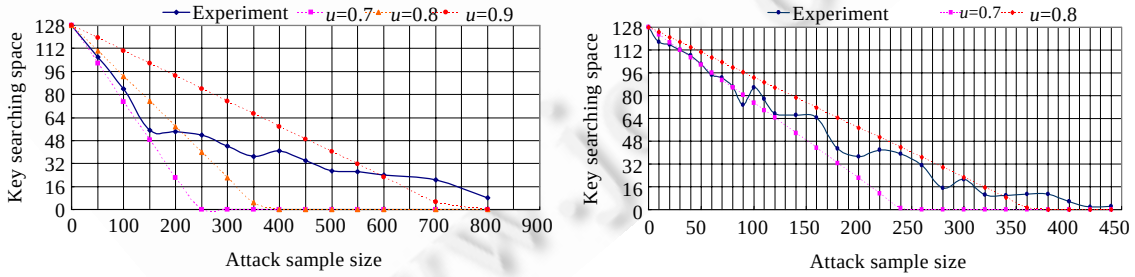
(b) Last round
(b) 最后一轮

Fig.14 Relationship between attack sample size and key searching space

图 14 攻击样本量和密钥搜索空间关系

6.5.2 远程攻击实验

OpenSSL v.0.9.8j 中,AES 远程攻击排除分析本地攻击样本量 N 和密钥搜索空间关系如图 15 所示.实验结果表明:随机 256 明文条件下,局域网环境中,第 1 轮和最后一轮远程攻击可分别在 750 个、280 个样本左右恢复密钥;校园网环境中,最后一轮攻击 500 个样本即可恢复密钥;v.0.9.8j 远程攻击所需样本量比本地攻击要稍大些,但只是线性增长;远程实验中噪声因子为 0.7~0.9 左右,实际攻击样本量和密钥搜索空间关系曲线也介于 u 为 0.7~0.9 之间的远程攻击理论值曲线之间,证明了第 4 节远程攻击 Cache 信息泄露模型和攻击所需样本量分析理论正确性.



(a) First round
(a) 第 1 轮

(b) Last round
(b) 最后一轮

Fig.15 Relationship between attack sample size and key searching space

图 15 攻击样本量和密钥搜索空间关系

需要说明的是,在对远程攻击样本量和密钥搜索空间进行采样时,由于噪声位置的不确定性,有时某个查找表对应的所有 Cache 组都被噪声干扰掉了,这样往往会出现该表对应的几个密钥字节的错误候选值一个也排除不掉或者很少被排除掉这种情况.所以,有时随着样本量的增加,其密钥搜索空间不但没有降低反倒增加了,如图 15(b)中攻击样本量为 100,220,300 时的采样数据.

6.6 本地、远程攻击比较分析

与本地攻击相比,远程攻击由于信息采集过程中网络发包拆包需多次访问 Cache,带来一定的噪声,攻击样本量比本地大.本地和局域网环境下,AES 访问驱动 Cache 计时攻击一个样本中间谍程序对所有 Cache 组采样分别如图 16、图 17 所示.其中,横坐标表示 Cache 组序号,纵坐标表示 Cache 组访问时钟周期.易见,本地攻击中由系统进程和其他用户进程对 Cache 访问带来的噪声比较小,对攻击影响不大;而远程攻击中网络发包拆包确实给攻击带来很大影响,但 AES 加密过程中查找多个表访问 Cache 组信息(方框选择区域中访问时钟周期较小的 Cache 组)仍可被采集到,攻击仍可成功实现,只是需要更多的样本量而已.

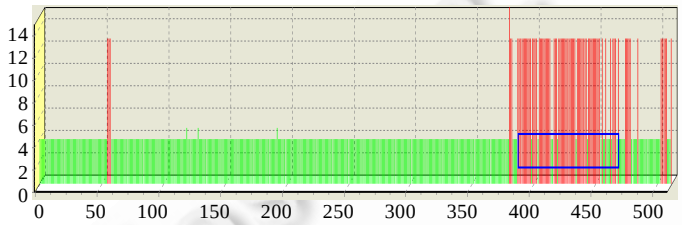


Fig.16 One sample Cache set chart of spy program in local attack

图 16 本地攻击间谍程序一个样本 Cache 组采样图

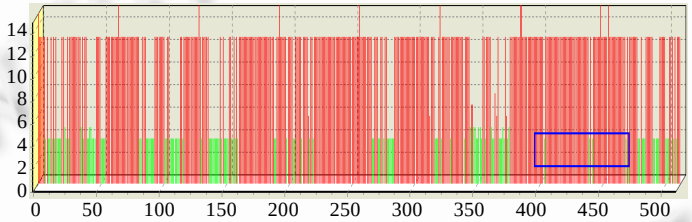


Fig.17 One sample Cache set chart of spy program in remote attack

图 17 远程攻击间谍程序一个样本 Cache 组采样图

6.7 与国内外攻击比较

我们的 AES 访问驱动攻击结果与国内外现有 Cache 计时攻击比较见表 6(其中样本量为将密钥搜索空间降低到 2^{16} 对应的样本大小).

易见,本文提出并实现了多例针对 AES 的本地和远程访问驱动 Cache 计时攻击,且攻击样本量要远小于国内外攻击.可以看出,访问驱动 Cache 计时攻击在本地和远程均具有较好的可行性.另外,国内外并没有针对 OpenSSL v.0.9.8j 中 AES 实现的攻击,其 AES 实现方式可有效防御上述国内外 Cache 计时攻击,尤其是针对 AES 最后一轮的攻击.然而,本文研究结果表明,OpenSSL v.0.9.8j 中 AES 实现仍然易遭受访问驱动 Cache 计时攻击的威胁.

在远程攻击方面,目前国内外已公布针对 AES 的 Cache 远程计时攻击只有 Bernstein[5]和 Acıicmez[12]两例,其攻击均属时序驱动 Cache 计时攻击.Bernstein 远程攻击[5]中,攻击端和加密服务端虽部署在不同的 PC 机上,但负责采集从加密开始到完成时间信息并回传给攻击端的却是远程加密端,绝对消除了网络传输时延.使用 $2^{27.5}$ 个样本恢复出 128 位 AES 密钥,这在实际攻击中是不现实的,攻击实际上仍属本地计时攻击.Acıicmez[12]远程攻击中,攻击端和加密服务端部署在同一台 PC 机上,攻击端负责采集从加密开始到加密完成时间,也是在消

除了网络传输时延的条件下,使用 $2^{26.66}$ 个样本才恢复出完整密钥,验证了理想条件下进行时序驱动远程计时攻击的可行性.在真实远程条件下,网络传输时延本身甚至其抖动时延一般远大于加密时间,时序驱动 Cache 远程计时攻击信息的精确采集十分困难.

我们在访问驱动远程攻击实验中发现,虽然攻击过程中网络发包、拆包要对 Cache 进行一定数量访问给攻击带来一定噪声,但其噪声并没有完全屏蔽掉实验要采集的与 AES 加密相关的 Cache 组信息,多样本采集后仍可快速恢复完整 AES 密钥.本文实现的访问驱动 Cache 远程计时攻击在很大程度上避开了时序驱动远程计时攻击中网络传输时延干扰比较大的问题,在局域网甚至不同教学楼之间的远程环境下将攻击端和加密服务端分别部署在不同的 PC 机上,同时将间谍程序植入到 AES 加密服务器上,受远程攻击端操控执行正常的数据访问操作采集 AES 加密过程中的 Cache 旁路信息,结合明文/密文对信息开展密钥分析,在有限样本下快速恢复完整 AES 密钥,攻击样本量和所需时间相对较小.

Table 6 Improvements of the attacks in this paper over several previous attacks

表 6 攻击实验结果和国内外已有 AES Cache 计时攻击比较

Attack	Attack setting	OpenSSL	Attack round	Attack model	Sample size	Recoverd key	Note
Bernstein[5]	Remote	v.0.9.8a	First round	Time driven	$2^{27.5}$	128-bit	
Tsunoo[4]	Local	v.0.9.8a	First round	Time driven	2^{26}	128-bit	
Bonneau[7]	Local	v.0.9.8a	First round	Time driven	$2^{14.58}$	60-bit	
	Local	v.0.9.8a	Last round	Time driven	2^{13}~2^{15}	128-bit	
Osvik[9]	Local	v.0.9.8a	First two rounds	Time driven	$2^{18.93}$	128-bit	
	Local	v.0.9.8a	First two rounds	Acecss driven	$2^{8.22}$	128-bit	
Neve[10]**	Local	v.0.9.8a	Last round	Acecss driven	$2^{7.53}$ $2^{3.80}$~$2^{4.32}$	128-bit	
Acıicmez[12]	Remote	v.0.9.8a	First two rounds	Time driven	$2^{26.66}$	128-bit	
Deng[16,17]	Local	v.0.9.8a	Last round	Time driven	2^{21}~2^{25}	128-bit	
	Local	v.0.9.8a	Last round	Time driven	$2^{15.29}$	30-bit	
Li[18]	Local	v.0.9.8a	First round	Time driven	2^{24}	128-bit	
This paper	Local	v.0.9.8a	First round	Acecss driven	$2^{10.23}$	128-bit	Non-Elimination
	Local	v.0.9.8a	First round	Acecss driven	$2^{6.32}$~$2^{8.64}$	128-bit	Elimination
	Remote	v.0.9.8a	First round	Acecss driven	$2^{8.45}$	128-bit	Elimination
	Local	v.0.9.8j	First round	Acecss driven	$2^{10.64}$	128-bit	Non-Elimination
	Local	v.0.9.8j	First round	Acecss driven	$2^{7.40}$~$2^{9.10}$	128-bit	Elimination
	Remote	v.0.9.8j	First round	Acecss driven	$2^{9.55}$	128-bit	Elimination
	Local	v.0.9.8a	Last round	Acecss driven	$2^{10.97}$	128-bit	Non-Elimination
	Local	v.0.9.8a	Last round	Acecss driven	$2^{3.70}$	128-bit	Elimination
	Remote	v.0.9.8a	Last round	Acecss driven	$2^{5.32}$	128-bit	Elimination
	Local	v.0.9.8j	Last round	Acecss driven	$2^{13.55}$	128-bit	Non-Elimination
	Local	v.0.9.8j	Last round	Acecss driven	2^{6}	128-bit	Elimination
	Remote	v.0.9.8j	Last round	Acecss driven	$2^{8.12}$	128-bit	Elimination

7 结 论

本文就当前最安全的 AES 分组密码算法访问驱动 Cache 计时攻击进行了一些相关研究,研究结果表明,此类针对 AES 的计时攻击手段对信息安全将带来突出威胁:首先,AES 已经被广泛地应用并被期望在未来 20 余年内作为支配性的分组密码算法,因此,攻击产生的影响将是广泛而深远的;其次,实施攻击并不需要物理地获得密码执行部件以测量泄漏信息,在网络环境下也可成功获取远程加密服务器密钥;还有,此类攻击能够作用于一切实现于 Cache-Memory 层次存储结构计算机设备上软件形式的 AES 算法,从而危害到服务器、桌面以及嵌入式等各种领域的主流计算机系统,因此应对这类攻击予以充分的关注.

需要特别指出的是,基于 Cache 的 AES 计时攻击是 AES 查表操作实现方式与现有主流计算机硬件系统特征所固有决定的,具有难于规避的特点.防御这种攻击可采取使用较小查找表、去除查找表、进程控制、Cache

** Michael Neve 仅给出了理论上最后一轮 AES 本地访问驱动攻击所需样本量.理论上来说,直接分析方法需 $2^{7.53}$ 个样本,排除分析方法需 $2^{3.80}$~$2^{4.32}$ 个样本,但其并没有公布详细的实验细节和结果.

预热、调整 Cache 结构甚至移除 Cache 等策略[5,23−26],但所有的防御措施都是要以牺牲加密速度为代价的.所以,在速度和安全性这一对矛盾的指标上进行平衡选择,是目前密码程序实现所面临的巨大挑战.

致谢 在此,特别对实验室李华博士、帅建宁、慕平同学在本文撰写过程中给予的支持和帮助表示衷心感谢.

References:

[1] Kocher PC. Timing attacks on implementations of Diffie-Hellman, RSA, DSS, and other systems. In: Proc. of the CRYPTO'96. LNCS 1109, Berlin: Springer-Verlag, 1996. 104−113. [doi: 10.1007/3-540-68697-5_9]

[2] Kelsey J, Schneier B, Wagner D, Hall C. Side channel cryptanalysis of product ciphers. Journal of Computer Security, LNCS 1485, Berlin: Springer-Verlag, 1998. 97−110.

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

[4] Tsunoo Y, Saito T, Suzaki T, Shigeri M, Miyauchi H. Cryptanalysis of DES implemented on computers with Cache. In: Proc. of the Cryptographic Hardware and Embedded Systems (CHES 2003). LNCS 2779, Berlin: Springer-Verlag, 2003. 62−76. [doi: 10.1007/978-3-540-45238-6_6]

[5] Bernstein DJ. Cache-Timing attacks on AES. 2005. http://cr.yp.to/papers.html\#cachetiming

[6] OpenSSL the open-source toolkit for SSL/TLS. 2005. http://www.openssl.org/

[7] Bonneau J, Mironov I. Cache-Collision timing attacks against AES. In: Goubin L, Matsui M, eds. Proc. of the Cryptographic Hardware and Embedded Systems (CHES 2006). LNCS 4249, Berlin: Springer-Verlag, 2006. 201−215. [doi: 10.1007/11894063_16]

[8] Percival C. Cache missing for fun and profit. In: Proc. of the Technical BSD Conf. 2005. 2005. 1−13.

[9] Osvik DA, Shamir A, Tromer E. Cache attacks and countermeasures: The case of AES. In: Proc. of the Topics in Cryptology (CT-RSA 2006). LNCS 3860, Berlin: Springer-Verlag, 2006. 1−20. [doi: 10.1007/11605805_1]

[10] Neve M, Seifert JP. Advances on access-driven Cache attacks on AES. In: Proc. of the Selected Areas in Cryptography (SAC 2006). LNCS 4356, Berlin: Springer-Verlag, 2007. 147−162. [doi: 10.1007/978-3-540-74462-7_11]

[11] Ac_icmez O, Gueron S, Seifert JP. Micro-Architectural cryptanalysis. Crypto Corner, 2007,5(4):62−64. [doi: 10.1109/MSP.2007.91]

[12] Ac_icmez O, Schindler W, Koc CK. Cache-Based remote timing attack on the AES. In: Proc. of the Topics in Cryptology (CT-RSA 2007). LNCS 4377, Berlin: Springer-Verlag, 2007. 271−286. [doi: 10.1007/11967668_18]

[13] Wu WL, He YP, Feng DG, Qing SH. Power attack of mars and rijndael. Journal of Software, 2002,13(4):532−536 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/13/532.htm

[14] Zhou YB, Feng DG. Side-Channel attacks: Ten years after its publication and the impacts on cryptographic module security testing. In: Proc. of the NIST Physical Security Workshop. 2005. 1−34.

[15] Hou FY, Gu DW, Lin XY. Cache-Based attacks against AES: Research progress. Information Security and Communications Privacy, 2007,(8):41−43 (in Chinese with English abstract).

[16] Deng GM, Zhao Q, Zhang P, Chen KY. Cache hit side channel attack based on AES. Computer Engineering, 2008,34(13):113−114 (in Chinese with English abstract).

[17] Deng GM, Zhang, Zhao Q, Liu XQ. Difference timing attack against AES based on Cache timing character. Geomatics and Information Science of Wuhan University, 2008,33(10):1087−1091 (in Chinese with English abstract).

[18] Li B, Hu YP, Zhong MF. Time-Based Cache attacks on AES. Computer Engineering, 2008,34(17):141−143 (in Chinese with English abstract).

[19] Stallings W, Wrote; Zhang KZ, *et al* Trans. Computer Organization and Architecture: Designing for Performance. Beijing: Tsinghua University Press, 2006. 78−110 (in Chinese).

[20] Hu XD, Wei QF. Applied Cryptography. Beijing: Electronics Industry Press, 2006. 83−105 (in Chinese).

[21] Aclicmez O, Koc CK, Trace-Driven Cache attacks on AES. In: Proc. of the ICICS 2007. LNCS 4307, Berlin: Springer-Verlag, 2006. 112−121. [doi: 10.1007/11935308_9]

[22] Zhao XJ, Wang T, Mi D. Robust first two rounds access driven Cache timing attack on AES. In: Proc. of the Int'l Conf. on Computer Science and Software Engineering (CSSE 2008), Vol.3. 2008. 785−788. [doi: 10.1109/CSSE.2008.633]

[23] Blömer J, Krummel V. Analysis of countermeasures against access driven Cache attacks on AES. In: Proc. of the Selected Areas in Cryptography (SAC 2007). LNCS 4876, Berlin: Springer-Verlag, 2007. 96−109. [doi: 10.1007/978-3-540-77360-3_7]

[24] Brickell E, Graunke G, Neve M, Seifert JP. Software mitigations to hedge AES against Cache-based software side channel vulnerabilities. Cryptology ePrint Archive, Report 2006/052, 2006. http://eprint.iacr.org/2006/052.pdf

[25] Page D. Partitioned Cache architecture as a side-channel defense mechanism. Cryptology ePrint Archive, Report 2005/280, 2005. http://eprint.iacr.org/2005/280.pdf

[26] Krummel V. Tamper resistance of AES—Models, attacks and countermeasures [Ph.D. Thesis]. Department of Computer Science University of Paderborn, 2007. 71−108.

附中文参考文献:

[15] 侯方勇,谷大武,李小勇.基于 Cache 的 AES 攻击:研究进展.信息安全与通信保密,2007,(8):41−43.

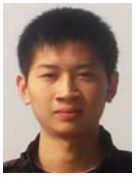
[16] 邓高明,赵强,张鹏,陈开颜.针对 AES 算法的 Cache Hit 旁路攻击.计算机工程,2008,34(13):113−114.

[17] 邓高明,张鹏,赵强,刘晓芹.基于 Cache 时间特性的 AES 差分时间分析攻击.武汉大学学报,2008,33(10):1087−1091.

[18] 李波,胡予濮,钟名富.针对 AES 的基于时间的缓存攻击.计算机工程,2008,34(17):141−143.

[19] Stallings W,著;张昆藏,等译.计算机组织与体系结构性能设计.北京:清华大学出版社,2006.78−110.

[20] 胡向东,魏琴芳.应用密码学.北京:电子工业出版社,2006.83−105.



赵新杰(1986－),男,河南开封人,博士生,主要研究领域为分组密码旁路分析,故障分析.

郭世泽(1969－),男,博士,研究员,博士生导师,主要研究领域为信息安全,密码学.

王韬(1964－),男,博士,教授,博士生导师,主要研究领域为信息安全,密码旁路分析.

郑媛媛(1981－),女,博士生,主要研究领域为分组密码安全,代数旁路分析.