DOI：10.7652/xjtuxb201603019

一种 Android 平台下高性能透明文件加密机制

胡炀[1]，陶敬[1,2]，刘林峰[1]，李剑锋[1]，胡文君[1]，梁肖[1]
（1. 西安交通大学智能网络与网络安全教育部重点实验室，710049，西安；
2. 西安交通大学苏州研究院，215123，江苏苏州）

摘要：针对 Android 平台当前采用的文件加密机制不能兼顾高安全性和高 I/O 性能的问题，提出了一种新型的基于映射表加密和物理块加密的透明文件加密机制。该机制使用 10 轮 AES-128 块加密算法加密文件的逻辑块与物理块的映射关系，使用改进的 7 轮 AES-128 块加密算法加密存储文件数据的物理块，从而在安全性和 I/O 性能之间寻求平衡。通过密码学理论推导证明了该机制的密码强度不低于 Android 平台当前采用的加密机制，通过 I/O 性能理论分析得出该机制的 I/O 性能优于 Android 平台当前采用的文件加密机制。此外，在 Google Nexus 4 智能手机上分别对该机制和 Android 平台当前采用的文件加密机制进行了 108 种不同测试条件下的 I/O 性能测试，测试结果验证了 I/O 性能理论分析得出的结论，并且发现：该机制的 I/O 写入性能比率比 Android 平台当前采用的文件加密机制平均高出 13.12%，该机制的 I/O 读取性能比率比 Android 平台当前采用的文件加密机制平均高出 16.16%。

关键词：文件加密机制；Android 平台；I/O 性能
中图分类号：TP393 **文献标志码**：A **文章编号**：0253-987X(2016)03-0120-07

A High-Performance Transparent File Encryption Mechanism on Android Platform

HU Yang[1], TAO Jing[1,2], LIU Linfeng[1], LI Jianfeng[1], HU Wenjun[1], LIANG Xiao[1]
(1. Ministry of Education Key Lab for Intelligent Networks and Network Security, Xi'an Jiaotong University, Xi'an 710049, China; 2. Xi'an Jiaotong University Suzhou Academy, Suzhou, Jiangsu 215123, China)

Abstract: It is regarded as a design defect that high I/O performance is not compatible to high security in the file encryption mechanism used nowadays in Android platform. To solve this problem, we presented a new transparent file encryption mechanism based on the mapping chart encryption and physical block encryption. It applies 10-round AES-128 block cipher algorithm to the mapping chart encryption and applies revised 7-round AES-128 block cipher algorithm to physical block encryption, which contributes to the balance between I/O performance and high security. Then we performed cryptanalysis and proved that the cipher strength of our mechanism is not lower than Android's. In addition, we have conducted I/O performance theoretical analysis and concluded that the I/O performance of this mechanism is higher than Android's. Moreover, we also performed 108 I/O performance tests in different test conditions on Google Nexus 4 smart phone. The result of these tests not only verifies the conclusion proved by I/O performance

收稿日期：2015-06-04。 作者简介：胡炀(1992—)，男，硕士生；陶敬(通信作者)，男，高级工程师。 基金项目：国家自然科学基金资助项目(91418205)；江苏省自然科学基金资助项目(SBK2014021758)；苏州应用基础研究计划资助项目(SYG201311)；江苏省未来网络创新研究院未来网络前瞻性研究资助项目(BY2013095-4-13)；江苏省产学研联合创新资金前瞻性联合研究资助项目(BY2014074)。
网络出版时间：2015-12-28 网络出版地址：http://www.cnki.net/kcms/detail/61.1069.T.20151228.1956.004.html

analysis, but also shows that the I/O writing performance of our mechanism is 13.12% higher than Android's and the I/O reading performance is 16.16% higher than that of Android system.
Keywords: file encryption mechanism; Android platform; I/O performance

根据 SandDroid 恶意代码分析平台提供的恶意代码数据,近年 Android 平台下的恶意代码数量激增,对用户手机上的私密信息构成严重威胁[1]。对用户私密信息进行加密存储是保护用户隐私信息的有效手段。目前,Android 平台已构建了一套基于磁盘加密机制 dm-crypt 的文件加密机制。然而,由于该机制需要在磁盘操作之前执行大量的加密或解密指令,该机制会延长 I/O 操作的完成时间,从而严重降低 I/O 性能,增加系统响应时间,严重影响移动设备的用户体验。针对该问题,文献[2]通过简化加密解密算法降低对 I/O 性能的影响,但算法已被破解[3]。文献[4-5]通过引入可信芯片提升加密速度,从而降低对 I/O 性能的影响,但目前可信芯片技术尚不成熟[6],大多数移动设备没有装配此类芯片。文献[7]通过调整 I/O 模式和策略提升 I/O 性能,实验表明此类方法并不能显著提升 I/O 性能。究其根源,加密机制的性能瓶颈在于加密与解密算法过于复杂,调整 I/O 模式和策略不能帮助移动设备跨越性能瓶颈的核心障碍。

通过深入探究 Android 平台的文件系统工作原理,发现该文件系统通过特有的索引机制检索文件数据,这种索引机制会把文件包含的块的逻辑地址和物理地址进行关联,被称为逻辑地址与物理地址的映射表,如果映射表被彻底打乱,那么文件的数据就很难按顺序读取或写入,因此可以通过加密映射表以提高攻击者破解文件数据的难度。然而,仅仅保护映射表并不能彻底保护文件数据。这是因为:同一个文件中的数据常常有较强的上下文关联,即便攻击者不知道映射表的具体内容,也能够根据数据间的上下文关联分析出正确的数据组织顺序。为了降低基于上下文关联攻击方法实施成功的可能性,应当在加密映射表的基础上大幅减弱甚至消除数据间的上下文关联。

基于上述观点,提出了一种 Android 平台下新型透明文件加密机制,即通过 10 轮 AES-128 加密算法[8]加密映射表并适时随机交换物理块中的数据,通过改进后的 7 轮 AES-128 加密算法[8]加密物理块中的文件数据。这种双层加密的设计思路不仅不会降低密码强度,而且与 Android 当前采用的透明文件加密机制相比减少了加密、解密时所需的计算量,从而减少了对 I/O 性能的影响。

1　透明文件加密机制的设计思路

1.1　透明文件加密机制的层次结构

本文提出的文件加密机制依赖于 Android 平台的文件系统。图 1 展示了本文提出的加密机制和 Android 平台的文件系统集成后组成的层次调用关系,灰色部分属于本文提出的文件加密机制,白色部分属于 Android 平台的文件系统。本文提出的文件加密机制包括两个层次:在图 1 中的文件系统 ext4 之上引入映射表加解密层,负责维护、加密、解密文件逻辑地址与物理地址的映射表;在图 1 中的文件系统 ext4 之下引入物理块加解密层,负责对每个物理块加密和解密。

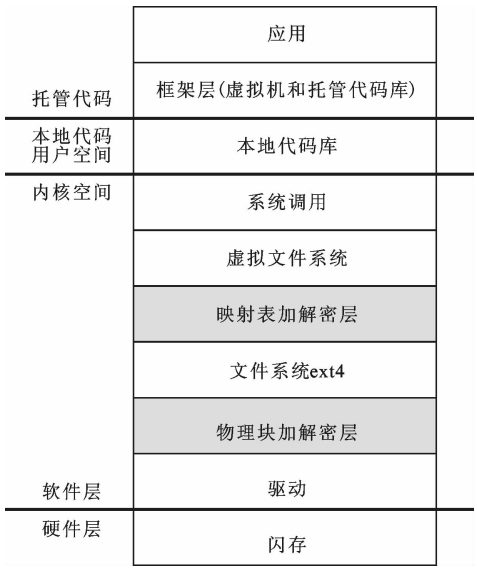

图 1　本文提出的文件加密机制的层次结构

1.2　映射表加解密层的原理与设计

映射表加解密层负责对映射表进行加密和解密。映射表的作用是记录逻辑块和物理块的映射关系,从而为文件操作提供检索服务。图 2 为映射表加密的原理示意图,文件 a.pdf 和 b.pdf 分别按顺序对应若干逻辑块,每个逻辑块又对应若干物理块。映射表被加密后,文件系统将无法获取其数据索引,文件系统就不能检索到目标物理块中的数据。

由于映射表所占存储空间通常不超过文件数据所占存储空间的 0.5%,采用复杂的加密算法加解密映射表对 I/O 性能影响是微弱的,故应采用较多

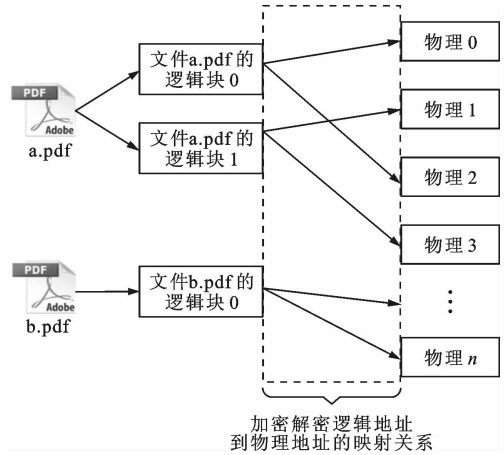

图 2 映射表加解密原理

轮的块加密算法加解密映射表。本文提出的文件加密机制采用 10 轮 AES-128 块加密算法[8]加解密该映射表。此外,由于文件系统在分配物理块时往往会把地址临近的物理块分配给同一个文件,攻击者可以利用该特性推测出映射表的部分内容。为了阻断这种攻击模式,周期性地伪随机挑选两个存储文件数据的物理块并对换它们的数据,对文件系统的物理块分配算法进行伪随机化处理,即分配物理块时待分配的物理块被分配的概率基本相等。该机制采用 ANSI X9.17 算法[9]生成伪随机串并用于物理块对换和分配。

图 3 展示了映射表加解密层的设计。当用户程序读写文件时,虚拟文件系统会向映射表加解密层发送文件操作请求;文件操作请求解析模块会解析出该请求对应文件的相关信息,生成控制命令,并把文件操作请求和控制命令发送给解密模块;解密模块会根据这些信息从下层的文件系统 ext4 获取该文件对应的映射表密文,解密映射表后得到映射表明文;最终映射表明文连同文件操作请求传递给下层文件系统 ext4。

当文件操作请求为读取请求时,映射表的内容

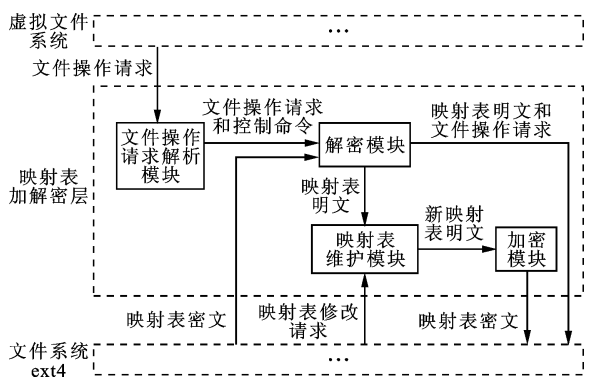

图 3 映射表加解密层的设计

不变;当文件操作请求为写入操作时,映射表的内容需要更新。文件系统 ext4 会向映射表加解密层发送映射表修改请求;受理该请求的模块是映射表维护模块,该模块会从解密模块获得当前文件操作的映射表明文,然后根据更新请求更新映射表并把更新后的映射表传递给加密模块;加密模块由加密更新后的映射表得到映射表的密文,并把密文传递给文件系统 ext4;文件系统 ext4 会把映射表的密文同步到持久化存储器上。

1.3 物理块加解密层的设计

物理块加解密层负责对物理块中的数据进行透明加密与解密。如图 4 所示,当文件系统 ext4 对某个物理块进行操作时,会向物理块加解密层发送物理块操作请求,物理块操作请求分析模块会分析该请求的类型。当该请求的请求类型是写入请求时,物理块操作请求分析模块会把请求中附带的写入数据明文后传递给加密模块;加密模块加密这些数据得到密文,并把密文传递给物理块数据同步模块;物理块数据同步模块会调用下层的驱动程序把密文写入物理块。当该请求的请求类型是读取请求时,物理块操作请求分析模块会把请求传递给物理块数据提取模块,该模块根据请求内容调用下层驱动程序提取物理块的密文,然后把密文传递给解密模块进行解密;解密模块把物理块数据明文化并反馈给文件系统 ext4。

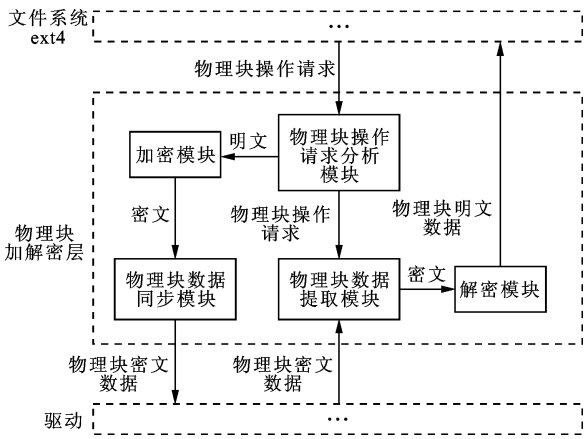

图 4 物理块加解密层的设计

由于映射表加解密层对映射表的有效保护,攻击者获取物理块明文密文对的难度提升,这使得物理块加解密层抗选择明文攻击的能力大幅提升。在映射表加解密层的保护下,图 4 中的加密模块和解密模块使用的加密算法可以适当简化,即在加密和解密物理块时可以采用轮数较少的块加密算法。然而,轮数较少会造成伪随机化程度不足,上下文关联

可能会被攻击者发现。为了解决该问题,本文在 7 轮 AES-128 块加密算法[8]的基础上增加一个伪随机轮,并把该算法作为图 4 中加密模块和解密模块的核心算法。伪随机轮使用预先准备好的伪随机串按位异或 AES-128 块加密算法[8]第 4 轮的输出结果作为第 5 轮的输入。该伪随机串由 ANSI X9.17 算法[9]生成。

1.4　映射表加解密层在文件系统 ext4 中的实现与优化

虽然映射表加解密层和文件系统 ext4 属于两个不同的层次,但是从上文描述的设计中可以看出,两者交互颇为密切。因此,从高内聚、低耦合的设计出发,本文将映射表加解密层和文件系统 ext4 合并,即通过对文件系统 ext4 的改造实现映射表加解密层的全部功能。

在文件系统 ext4 中,每一个文件对应一个被称为 ext4_inode 的数据结构,其中有一个字段存放了在文件系统 ext4 中被称为 extent tree 的根节点指针。extent tree 是一棵索引树,它存储了对应文件中所有块的逻辑和物理地址的映射关系。对于文件系统 ext4,extent tree 相当于上文提及的映射表。按照 1.2 节介绍的映射表加解密层的设计,应当加密全部映射表。然而,特殊的树形结构为性能优化提供了空间,extent tree 的叶节点存放了逻辑块对应的物理块地址,而非叶节点不会存放这些内容,只要加密全部叶节点就可以满足设计要求。假设某 extent tree 有 x 个节点,其中 y 个节点是叶节点。对于树形结构而言,y 通常小于 x,这无疑减少了加密任务量。图 5 是一个优化示例,灰色节点表示需要被加密的节点,白色节点表示不需要加密的节点。在图 5 中,优化前需要加密 9 个节点,而优化后只需要加密 5 个节点,加密的任务量减少了近一半。

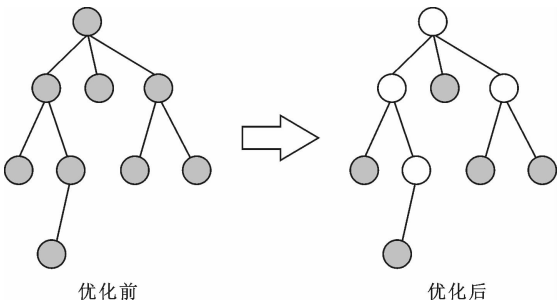

图 5　extent tree 加密优化效果对比

2　I/O 性能的理论分析与测试

2.1　密码强度分析

密码强度分析是为了衡量本文所提文件加密机制的安全性。通常采用破解所需的计算复杂度来衡量文件加密机制的安全性,破解所需的计算复杂度越高,安全性越好。Android 平台当前使用的加密机制采用 10 轮 AES-128 块加密算法[8],由于目前关于 10 轮 AES-128 块加密算法[8]破解方法的计算复杂度为 $2^{126.1}$[10],故只要本文提出的文件加密机制被破解的计算复杂度达到甚至超过 $2^{126.1}$,就可以说明本文所提文件加密机制的安全性不低于 Android 平台当前使用的文件加密机制。

由有序四元组(A,M,N,S)表示图 1 描述的文件加密机制,其中:A 代表该加密机制采用的块加密算法的类型;M 代表映射表加密层使用加密算法的轮数,$M=0$ 表示该加密机制没有映射表加解密层;N 代表物理块加密层使用加密算法的轮数,$N=0$ 表示该加密机制没有物理块加解密层;S 代表持久化存储的容量上限,单位为字节。

在上述规定下,Android 平台当前采用的文件加密机制可以被近似表示为(AES-128,0,10,S),而本文所提加密机制可以被近似表示为(AES-128,10,7,S)。

为了说明(AES-128,10,7,S)的安全性,给出定理 1 并加以证明。

定理 1　当 $A=1$、$N=7$、$S\geqslant 2^{30}$ 时,加密机制(AES-128,M,N,S)被破解的计算复杂度不低于 $2^{126.1}$。

证明　映射表加解密层采用 10 轮的 AES-128 块加密算法[8],由文献[10]可知,本文所提文件加密机制的映射表加解密层被破解的计算复杂度为 $2^{126.1}$。物理块加解密层采用 7 轮的 AES-128 块加密算法,由文献[10]可知,目前最佳破解方法的计算复杂度为 2^{116},然而破解条件是获取若干明文密文对,由于每个物理块的大小为 512 B,故 $S\geqslant 2^{30}$ 的块设备存储包含至少 2^{21} 个物理块。又因映射表已被保护,攻击者难以根据物理块密文判定其是否与选择明文相对应,唯一的方法是遍历所有物理块,计算复杂度至少为 2^{21},破解物理块加解密层的计算复杂度 $\min(2^{21+116},2^{128})=2^{128}$,因此本文所提加密机制被破解的计算复杂度 $\min(2^{128},2^{126.1})=2^{126.1}$。

当 $S\geqslant 2^{30}$ 时,(AES-128,10,7,S)被破解的计算复杂度不低于(AES-128,0,10,S)的计算复杂度。$S\geqslant 2^{30}$ 在现实生活中容易满足,现在大量的移动设备持久化存储容量超过 2^{30} B。因此,本文所提加密机制的安全性在现实生活中能够得到保证。

2.2　I/O 性能分析

I/O 性能分析是为了从理论上分析本文所提文

件加密机制的 I/O 性能优于 Android 平台当前采用的文件加密机制，通常采用 I/O 操作中单位字节所要付出的时钟周期或指令周期来衡量 I/O 性能。关于 I/O 性能给出定理 2 并加以证明。

定理 2 当 $S\geqslant 2^{30}$ 时，(AES-128,10,7,S)的 I/O 性能优于(AES-128,0,10,S)的性能。

证明 对于一个开启加密机制的移动设备而言，I/O 操作包含 3 部分内容：磁盘操作和文件系统调度；映射表的加解密；物理块的加解密。下面分别计算在读写一个物理块时 3 部分所要付出的时钟周期。

对于磁盘操作和文件系统调度，不妨假设磁盘操作和文件系统调度的单位字节所需的时钟周期为 c，则磁盘操作和文件系统调度需要消耗的时钟周期为 $512c$；对于物理块的加解密，不妨假设 AES-128 块加密算法[8] 每一轮所需的时钟周期为 q。由于 AES-128 块加密算法[8] 的明文输入大小为 16 B，则对于文件加密机制(A,M,N,S)，加解密一个物理块消耗的时钟周期为 $32Nq$。对于映射表的加解密，一个物理块仅对应于 extent tree 的一个叶节点。因此，在读写一个物理块时，映射表部分的加解密等价于对该物理块对应的叶节点进行加解密。由文献[11]知，Android 平台的文件系统中 extent tree 的叶节点大小为 12 B，则对于文件加密机制(A,M,N,S)，映射表加解密消耗的时钟周期为 $32Mq/4$。

因此，读写一个物理块付出的时钟周期总和为 $512c+32Nq+3Mq/4$，需要被加解密的数据大小为 524 B，文件加密机制(A,M,N,S)单位字节付出的时钟周期为

$$128c/131+(8q/131)(3M/128+N) \quad (1)$$

由于 $8q/131$ 和 $128c/131$ 均为大于 0 的常数，不影响大小比较，可以采用式(1)中的 $3M/128+N$ 作为 I/O 性能比较指标，即对于任意加密机制(A,m,n,S)、(A,m^*,n^*,S)，(A,m,n,S)的 I/O 性能优于(A,m^*,n^*,S)的性能，当且仅当 $3m/128+n<3m^*/128+n^*$。因此，定义加密机制(A,M,N,S)的 I/O 性能参数为

$$3M/128+N \quad (2)$$

根据式(2)，(AES-128,10,7,S)的 I/O 性能参数为 7.2，(AES-128,0,10,S)的性能参数为 10.0。由于 $7.2<10.0$，故(AES-128,10,7,S)的 I/O 性能优于(AES-128,0,10,S)的性能。

2.3 I/O 性能测试

本节将通过实验来测量与评估所提文件加密机制和 Android 平台当前采用的文件加密机制对系统 I/O 性能的影响。

2.3.1 *实验设计* 在 Google Nexus 4 智能手机上，使用 iozone 测试工具，测试不开启任何文件加密机制的 Android 平台、开启 Android 平台当前所用文件加密机制的 Android 平台、开启本文所提文件加密机制的 Android 平台这 3 组系统的 I/O 性能。

为了让实验能够反映客观规律，每组实验需遵守如下测试条件：读写文件的大小分别为 2^0 MB，2^1 MB，2^2 MB，…，2^{11} MB；文件由若干条相同大小的记录构成，记录的大小分别取 2^2 kB，2^3 kB，2^4 kB，…，2^{10} kB；避免使用内核磁盘、C 库、Java 文件缓存，且采用 Direct I/O 读写数据。此外，为了评估 I/O 性能，本文把 I/O 性能划分为 I/O 写入性能(又称为文件写入速度)和 I/O 读取性能(又称为文件读取速度)，并测量不同测试条件下分别开启两种文件加密机制时的文件写入速度和文件读取速度以及不开启任何文件加密机制时的文件写入速度和文件读取速度。

2.3.2 *实验结果* 由于大部分的 I/O 性能变化趋势类似，故只展示部分典型的测试结果。图 6、7 分别展示了文件大小为 16 MB 时的写入和读取速度，其中 L 表示记录长度，单位为 kB；R 表示文件读取速度，单位为 MB/s；W 表示文件写入速度，单位为 MB/s。由图 6、7 可知：在记录长度相等的情况下，开启本文所提加密机制的 Android 平台的文件写入速度和文件读取速度高于开启 Android 平台当前采用的文件加密机制的 Android 平台。

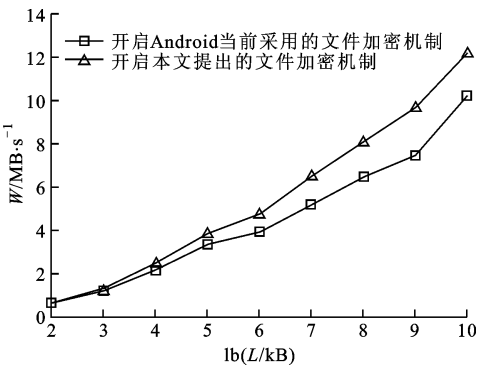

图 6 文件大小为 16 MB 时文件写入速度与记录长度的变化关系

为了进一步分析与对比两种加密机制对 I/O 性能的影响，本文把相同测试条件下开启加密机制的 Android 平台的文件写入速度与不开启任何加密机制的 Android 平台的文件写入速度的比值作为衡量文件加密机制对系统 I/O 写入性能的影响指标，

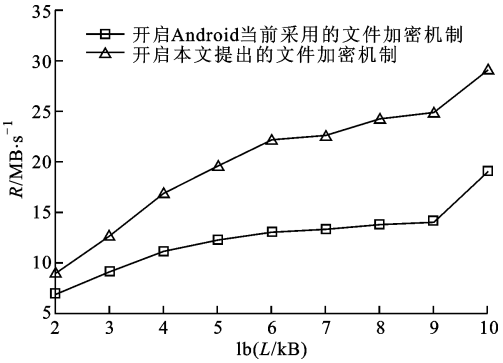

图 7　文件大小为 16 MB 时文件读取速度与记录长度的变化关系

把开启加密机制的文件读取速度与不开启任何加密机制的文件读取速度的比值作为衡量文件加密机制对系统 I/O 读取性能的影响指标，这两个性能指标称为 I/O 性能比率，取值范围为[0，1]。I/O 性能比率越大，加密文件系统对系统 I/O 的影响就越小。

为了减少噪声对 I/O 性能比率的影响，采用数理统计中参数估计的方法求解文件加密系统的性能比率。本文用 X_1 表示 Android 平台当前采用文件加密机制的 I/O 写入性能比率，用 X_2 表示本文所提的写入性能比率，用 Y_1 表示 Android 平台当前采用的读取性能比率，用 Y_2 表示本文所提的读取性能比率。本文中 X_1、X_2、Y_1、Y_2 为随机变量，且假设其服从正态分布。根据 I/O 性能比率的定义和实验中测出的文件写入、读取速度，可以计算出 X_1、X_2、Y_1、Y_2 的观察值。分别计算 X_1、X_2、Y_1、Y_2 的期望值 $E(X_1)$、$E(X_2)$、$E(Y_1)$、$E(Y_2)$作为 I/O 性能比率的真值，用两种文件加密机制的 I/O 性能比率期望值的差值 $E(X_2)-E(X_1)$和 $E(Y_2)-E(Y_1)$来衡量 I/O 性能的提升幅度。

X_1、X_2、Y_1、Y_2 的分布如图 8、9 所示。其中，$E(X_2)-E(X_1)=13.12\%$，$E(Y_2)-E(Y_1)=16.16\%$。因此，本文所提文件加密机制的 I/O 写入性能比率比 Android 平台当前采用的文件加密机制平均高出 13.12%，本文所提文件加密机制的读取性能比率比 Android 平台当前采用的文件加密机制平均高出 16.16%。这些实验结果充分说明了本文所提文件加密机制在 I/O 性能方面的优势。

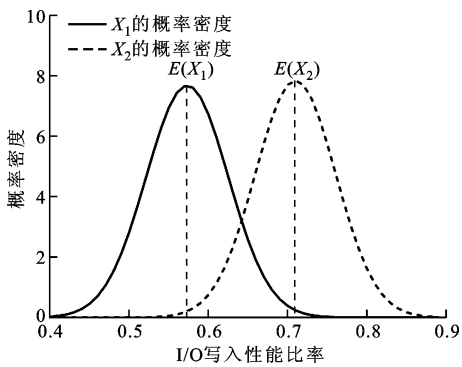

图 8　I/O 写入性能比率分布

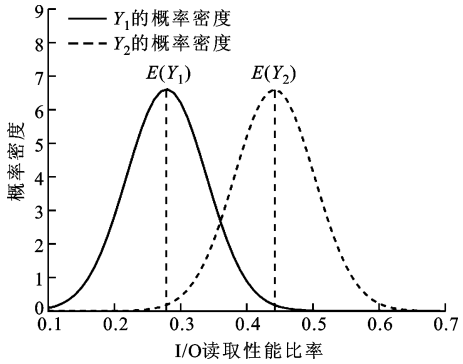

图 9　I/O 读取性能比率分布

3　结　论

针对 Android 平台当前使用的文件加密机制不能兼顾高安全性和高 I/O 性能的问题，提出了一种基于映射表加解密和物理块加解密的透明文件加密机制。本文详细介绍了该文件加密机制的设计思路，从理论上证明了该机制的安全性不低于 Android 平台当前采用的文件加密机制，采用 I/O 性能理论分析得出了该机制在 I/O 性能上优于 Android 平台当前采用的文件加密机制，通过在 Google Nexus 4 智能手机上分别对该机制和 Android 平台当前采用的文件加密机制进行了 108 种 I/O 性能测试。测试结果验证了 I/O 性能理论分析的结论，并且发现：该机制的 I/O 写入性能比率比 Android 平台当前采用的文件加密机制平均高出 13.12%，该机制的 I/O 读取性能比率比 Android 平台当前采用的文件加密机制平均高出 16.16%。

参考文献：

[1]　胡文君，赵双，陶敬，等. 一种针对 Android 平台恶意代码的检测方法及系统实现 [J]. 西安交通大学报，2013，47(10)：37-43.
HU Wenjun，ZHAO Shuang，TAO Jing，et al. A detection method and system implementation for Android malware [J]. Journal of Xi'an Jiaotong University，2013，47(10)：37-43.

[2]　CROWLEY P. Mercy：a fast large block cipher for disk sector encryption [C]// Fast Software Encryption. Berlin，Germany：Springer，2001：49-63.

[3] FLUHRER S R. Cryptanalysis of the mercy block cipher [C]// Fast Software Encryption. Berlin, Germany: Springer, 2002: 28-36.

[4] AGOSTA G, BARENGHI A, DE SANTIS F, et al. Fast disk encryption through GPGPU acceleration [C]// International Conference on Parallel and Distributed Computing, Applications and Technologies. Piscataway, NJ, USA: IEEE, 2009: 102-109.

[5] LI Jun, YU Huiping. Trusted full disk encryption model based on TPM [C]// 2010 2nd International Conference on Information Science and Engineering. Piscataway, NJ, USA: IEEE, 2010: 1-4.

[6] LV Y Q, ZHOU Q, CAI Y C, et al. Trusted integrated circuits: the problem and challenges [J]. Journal of Computer Science and Technology, 2014, 29(5): 918-928.

[7] WANG Z, MURMURIA R, STAVROU A. Implementing and optimizing an encryption filesystem on Android [C]// 13th International Conference on Mobile Data Management. Piscataway, NJ, USA: IEEE, 2012: 52-62.

[8] DAEMEN J, RIJMEN V. The design of Rijndael: AES-the advanced encryption standard [M]. Berlin, Germany: Springer, 2013.

[9] DESAI A, YIN Y, HEVIA A. Enhanced ANSI X9.17 pseudorandom number generators with forward security: US 7, 227, 951 [P]. 2007-06-05.

[10] BOGDANOV A, KHOVRATOVICH D, RECHBERGER C. Advances in Cryptology-ASIACRYPT 2011 [M]. Berlin, Germany: Springer, 2011: 344-371.

[11] MEDIA W. Ext4 disk layout [EB/OL]. [2015-05-21]. https://ext4.wiki.kernel.org/index.php/Ext4_Disk_Layout, 2014.

(编辑 赵炜)

(上接第 93 页)

3 结 论

本文对具有相对滑动粗糙表面上的单个微凸体,在相互接触过程中产生的闪点温度的变化趋势及影响因素进行了分析研究,得到以下结论。

(1)接触面中心位置处的闪点温度随着接触时间的延长而升高,在总接触时间的 82.5%时刻,闪点温度达到最大值,之后随着接触时间的延长,闪点温度逐渐降低。

(2)本文预测的单个微凸体接触过程中的最高温度比 Archard 预测的高 10%。

(3)最大闪点温度与滑动速度 V 的平方根、$d_0^{3/4}$ 及 μE^* 成正比;与热导率 K、密度和比热容的乘积 $\frac{K}{k^{1/2}}$ 成反比。

(4)在微凸体的峰顶半径和最大干涉深度一定的情况下,对于相同的滑动速度,以铝合金和铸铁为材料的单个微凸体,在接触过程中的最高温度低于以钢和铸铁为材料的单个微凸体接触的最高温度。

参考文献:

[1] LING F F, PU S L. Probable interface temperatures of solids in sliding contact [J]. Wear, 1964, 7(1): 23-34.

[2] ARCHARD J F. The temperature of rubbing surfaces [J]. Wear, 1959, 2(6): 438-455.

[3] KENNEDY F E. Surface temperatures in sliding systems: a finite element analysis [J]. Journal of Tribology, 1981, 103(1): 90-96.

[4] GREENWOOD J A, WILLIAMSON J B P. Contact of nominally flat surfaces [J]. Proceedings of the Royal Society of London: Series A Mathematical and Physical Sciences, 1966, 295(1442): 300-319.

[5] CARSLAW H S, JAEGER J C. Conduction of heat in solids [M]. Oxford, UK: Clarendon Press, 1959.

[6] BERGMAN T L, INCROPERA F P. Fundamentals of heat and mass transfer [M]. [s.l.]: John Wiley & Sons, 2011.

[7] WANG S, KOMVOPOULOS K. A fractal theory of the interfacial temperature distribution in the slow sliding regime: part I Elastic contact and heat transfer analysis [J]. Journal of Tribology, 1994, 116(4): 812-822.

[8] JOHNSON K L, JOHNSON K L. Contact mechanics [M]. Cambridge, UK: Cambridge University Press, 1987.

[9] LIU Y, BARBER J R. Transient heat conduction between rough sliding surfaces [J]. Tribology Letters, 2014, 55(1): 23-33.

(编辑 赵炜 苗凌)