第36卷 第15期 计 算 机 工 程 2010年8月
Vol.36 No.15 Computer Engineering August 2010
·安全技术· 文章编号：1000—3428(2010)15—0137—03 文献标识码：A 中图分类号：TP309.2

基于读写优化的内核级加密文件系统

裴灿浩[1]，谢长生[1,2]，黄建忠[2]
(1. 华中科技大学武汉光电国家实验室，武汉 430074；2. 华中科技大学数据存储教育部重点实验室，武汉 430074)

摘 要：为增强数据的机密性，在对存储读写流程优化的基础上提出一种内核级加密文件系统(KCFS)，该内核级加密文件系统能克服加密应用程序的使用不便利性和用户级加密文件系统的低效性，在内核级文件系统层提供加/解密功能，从而保护存储系统中的数据。对比测试结果表明，KCFS比用户级加密文件系统CFS具有更好的读写性能。

关键词：存储安全性；内核级加密文件系统；加密策略

Kernel-level Cryptographic File System Based on Read/Write Optimization

PEI Can-hao[1], XIE Chang-sheng[1,2], HUANG Jian-zhong[2]
(1. Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan 430074;
2. Key Lab of Data Storage System, Ministry of Education, Huazhong University of Science and Technology, Wuhan 430074)

【Abstract】In order to strengthen the confidentiality of data, this paper proposes a kind of Kernel-level Cryptographic File System(KCFS) combining the optimized data read/write flow. The kernel-level cryptographic file system can overcome the inconvenience of encryption application and the low efficiency of user-level encryption file system, so as to provide the encryption/decryption function in the kernel-level file system, protecting the data stored in the storage system. The result of comparative experiment shows that KCFS can reach better read/write performance compared to user-level encryption file, such as CFS.

【Key words】storage security; Kernel-level Cryptographic File System(KCFS); encryption policy

1 概述

数据已经成为企业、单位和个人最重要的财富，而作为数据存放平台的存储系统也逐渐成为恶意攻击者的主要目标，如果存储系统被攻击者成功入侵，存储数据遭篡改、删除和破坏的可能性将变得很大，对企业、机构或个人造成的损失可能是致命的[1]。

为了更好地应对这种情况，思路之一是将数据进行加密，并将加密后数据存放到存储设备上，从而降低数据的破解概率。目前的加密方案主要有 WinRAR 之类加密应用程序和 CFS 之类用户级加密文件系统[2]。为了有效地克服加密应用程序的使用不便利性和用户级加密文件系统的低效性，增强存放数据的机密性，本文提出并实现了一种内核级加密文件系统(Kernel-level Cryptographic File System, KCFS)。该文件系统可以在文件系统层提供加/解密功能，从而保护存储系统(如 NAS)中的数据不被篡改或破坏。其中采用的读写优化流程将提高数据读写的访问性能。

2 存储读写流程

网络存储系统的系统结构一般如下：本地磁盘通过主机总线适配器(HBA)挂接到存储系统上，同时通过网络协议(如TCP/IP)将存储系统连接到网络中，客户端通过网络对存储系统进行访问，以 NAS 系统为例，为 NAS 系统增加一个加密模块(即本文实现的加密文件系统 KCFS)，其软件层次结构如图 1 所示，图中标记出 NAS 系统的读写流程。其 I/O 路径经历几个文件系统：客户端利用 NFS/CIFS 通过网络访问 NAS 服务器，再利用服务器本地的文件系统如 Ext2 访问底层的磁盘设备；而在虚拟文件系统(VFS)层和 NFS 层之间加入加密文件系统后，对经过 I/O 路径的数据进行加密处理，达到加密效果。模块 KCFS 对经过的数据进行加密或解密。图中粗箭头表示远程客户进行写数据时的流向，当客户向 NAS 服务器提出写文件请求，数据通过 NFS 传输到 NAS 服务器，数据经过 KCFS 模块加密后再传给 VFS，VFS 调用 NAS 服务器本地物理文件系统(即 Ext2)将数据写入存储设备，读文件操作的流程则与之相反，如图中虚线所示。

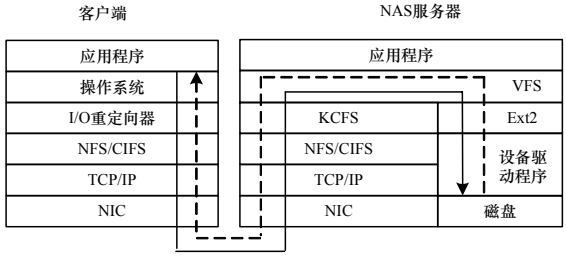

图 1 含加密模块的 NAS 系统读写流程

3 KCFS 的设计

3.1 堆叠式文件系统

传统文件系统驻留在内核中，并直接与设备驱动程序通

基金项目：国家“863”计划基金资助项目(2009AA01A402)；国家自然科学基金资助项目(60603074, 60603075, 60933002)
作者简介：裴灿浩(1971－)，男，博士，主研方向：网络存储，存储安全；谢长生，教授、博士生导师；黄建忠，副教授、博士
收稿日期：2010-01-06 **E-mail：**chpei5678@163.com

信，其性能很高，但它与底层设备驱动紧密结合使得开发和移植的难度很大。用户层文件系统虽然具有容易开发、容易调试、移植性好等优点，却挂接在用户层，要么作为一个进程运行，要么被实现为一个运行时库，过多的上下文切换和数据拷贝降低了该类文件系统的性能。

VFS 从文件系统操作的实现中抽象出文件系统操作，每个文件系统类型提供了特定文件系统操作的实现，为了访问某一物理文件系统(如 Ext2)，进程发出的系统调用请求先被翻译为 VFS 调用，然后 VFS 将调用请求发送给适当的物理文件系统，此时 VFS 被认为是上层文件系统，物理文件系统被认为是下层文件系统，上下层文件系统的调用是通过虚拟节点接口实现的。虚拟节点的提出促进了文件系统功能的模块化，并产生了堆叠式虚拟节点(Stackable Vnode)[3]。堆叠式文件系统是一种面向对象的文件系统，其对象是堆叠式虚拟节点，而堆叠特性是指该文件系统能堆叠在别的文件系统之上。堆叠式文件系统的执行流程如下：从上一层文件系统接收虚拟节点对象，执行与虚拟节点相关的操作，然后调用下一层文件系统。因此，堆叠式虚拟节点相对于上下层文件系统是透明的。

3.2 加密文件系统的设计

在文件系统层实现数据加/解密提高数据的安全性对于网络存储系统的应用具有很重要的意义：(1)可以保护任何应用程序所使用的任意数据，最大限度地保护已有投资；(2)允许用户透明地加密数据，为应用程序提供统一的加密方式，降低用户介入开销。在 NFS 和 VFS 层之间加入一个数据加密层对流经的数据进行加密和解密是透明的加密方式，这种方式避免了用户级加密文件系统(如 CFS、TCFS)性能效率低的缺点[4-5]。

该文件系统主要考虑以下 4 个因素：

(1)实现层次。在文件系统层实现加密功能不仅能提高该网络存储系统的安全性，而且提供了一种对用户透明的加密方式。

(2)性能。用户级文件系统(如 CFS)具有性能方面的缺陷，而内核层文件系统能较好地克服这些缺点。

(3)密钥管理。本文将密钥存放在用户所关联的文件系统根目录下，当用户进入加密文件系统时，文件加密模块会提示输入密码，该密码会和已设置的密码进行比较，如果匹配，则进入该文件系统根目录下，用户在该目录下操作的文件都将进行加密[6-7]。

(4)机密性。为了增强文件数据的机密性，本文采用的密钥不是由用户 ID 及其密码生成的，而是根据会话 ID 和用户 ID 进行生成。会话 ID 的随机性也增加了数据机密性，达到数据的随机加密。

3.3 读写操作流程的设计

(1)读操作

文件读请求的数据操作如图 2 所示。在操作系统中，文件读请求往往先将文件数据从二级存储设备(如 IDE 硬盘)读入内存中，并以内存页面的方式呈现；对于文件写请求，文件数据也要经过内存，然后在系统的调度下，再写入二级存储设备中，读请求需要读取的地址是第 11 000 Byte－第 27 000 Byte 的数据，从图中可知，该数据段分布于 3 个内存页面(Page1、Page2、Page3)，KCFS 模块先读取 3 个页面的数据并解密，如图 2 中(1)所示，然后从解密数据中截取第 11 000 Byte-第 27 000 Byte 范围的数据，如图 2 中(2)所示，最后将截取的数据传递给上层的 VFS。在读请求时，KCFS 完成了解密功能。

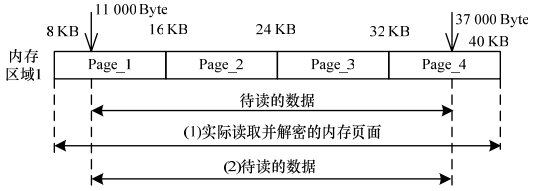

图 2 文件读请求的数据操作

(2)写操作

对于写操作，假设待写入的数据位于第 49 154Byte-第 78 056 Byte 之间(图 3)，该数据位于 4 个内存页面(Pg_1、Pg_2、Pg_3、Pg_4)。加密模块先加密上述 4 个页面，如图中(1)所示，然后截取第 49 154 Byte－第 78 055 Byte 中的密文数据，如图 3 中(2)所示，并将截取的密文数据传递给下层的物理文件系统(如 Ext2)，写操作完成了加密过程。

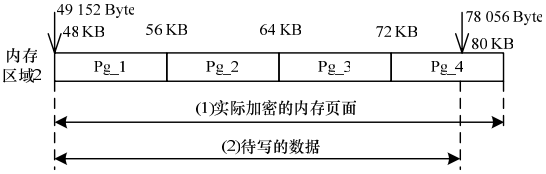

图 3 文件写请求的数据操作

(3)修改操作

修改操作的过程如图 4 所示，需修改长度为 26 000 Byte 的密文数据，该数据从二级存储设备读入内存区域 3 后，仍处于密文状态，存放地址为第 111 000 Byte－第 136 999 Byte。通常情况下，可采用与读操作相对应的方法：读取 PP_1、PP_2、PP_3、PP_4 并进行解密，然后将修改请求的数据段覆盖相应的位置(第 111 000 Byte－第 136 999 Byte)，对 4 个内存页(第 110 592 Byte－第 143 360 Byte)进行加密。从中可以发现，位于第 111 000 Byte－第 136 999 Byte 的数据段先被解密，后被覆盖，而加密是一项耗费 CPU 的操作，如果需要写入的数据段很长，那么上述操作将对系统性能产生很大影响。因此，本文采用局部解密-覆盖-全局加密-局部截取这一操作顺序，修改操作在 2 段内存区完成，修改后的数据仍为密文数据。

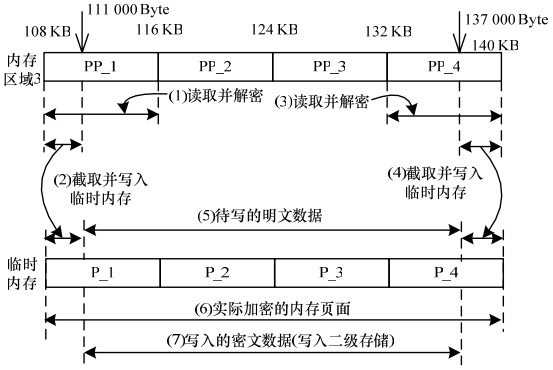

图 4 文件中数据的修改过程

4 实验及结果

4.1 测试环境

按照堆叠式文件系统规范生成一个文件系统模板，然后在该堆叠式文件系统模板基础上实现上述的读、写、修改流程，最终生成 KCFS 加密文件系统模块。生成的 KCFS 模块

将在 NAS 存储系统中进行功能测试和性能测试。为了了解 KCFS 相对于用户级加密文件系统 CFS 的性能差异和使用时的便利情况，采用了对比测试。

具体的实验环境是以一台装有Linux操作系统(内核版本是2.6.18)的P4 2.4 GHz/1 024 MB内存的PC作为客户端，NAS 服务器的软硬件配置为：Intel Core 2 Duo CPU(E6300 1.86 GHz)，1 GB内存和80 GB IDE 希捷硬盘(ST380812A)，操作系统是Fedora Core 6(内核版本为2.6.20)；2台机器通过千兆以太网进行连接。

4.2 实验方法

测试对象是 3 组文件系统(NFS、NFS＋KCFS、NFS＋CFS)，其中，NFS 指未加载加密文件系统；NFS+KCFS 指将 KCFS 模块加载在 NAS 服务器端的内核层；而 NFS+CFS 指将 CFS 加载在用户层；然后用测试软件通过网络文件系统 NFS 对服务器端进行测试。

为了增强比较性，测试时CFS和KCFS都采用了Blowfish算法[8]。而测试软件是 Bonnie++，它在顺序读、顺序写和随机定位3种模式下对文件系统的传输速率和CPU占用率进行测试，反映的是文件系统的输入输出性能。测试参数如下：文件大小设为 300 MB，内存设为 60 MB。并选择顺序块读取、顺序块写入、顺序块读写作为测试项。

4.3 实验结果

从功能角度看，由于 CFS 只能装载在本地 NFS 之上，用户使用时必须在客户端进行装载。而 KCFS 装载在服务器端，对客户端用户是透明的，相对而言具有更好的操作便利性；同时，当用户进入 KCFS 文件系统根目录后，用户在该工作目录下操作的文件都将进行加/解密。

图 5 所示为 3 组文件系统 NFS、NFS＋KCFS、NFS＋CFS 在运行 Bonnie++得出的平均传输率。以顺序块读取为例，NFS 的传输率为 6 345 KB/s，NFS＋KCFS 的传输率为 5 723 KB/s，相对 NFS 传输率下降了 9.8%；NFS＋CFS 的传输率为 5 067 KB/s，相对 NFS 下降了 20.1%，这是因为内核层存在更少的上下文切换和数据拷贝，因此，文件系统装载在内核层比挂接在用户层对系统性能的影响更低。装载了 KCFS 模块，在顺序块读取时系统性能下降不到 10%，说明文件系统模块装载在内核层对系统的性能不会产生很大的影响。在 3 个测试项中，NFS＋KCFS 的传输率相对于 NFS 下降了 9.2%~13.2%；而 NFS＋CFS 的传输率相对于 NFS 下降了 18.6%~30.1%，性能明显下降。

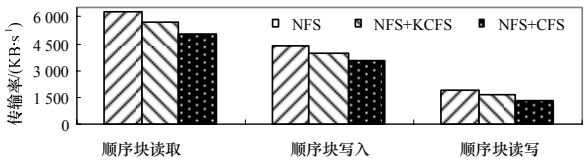

图 5 不同组合文件系统的数据传输率

5 结束语

加密文件系统本质上是一种文件系统过滤器，对发送到物理文件系统上的数据进行加密操作，从而达到保密的目的。本文在堆叠式文件系统模板上具体实现了存储数据的读写流程，从而得到一个加/解密读写过程优化的内核级加密文件系统，即 KCFS。KCFS 具有随机的数据机密性和较强的操作便利性。从实验结果可知，KCFS 模块在内核级运行，具有良好的读写性能，可以增强一些网络存储系统(如 NAS 系统)的数据安全性。

参考文献

[1] 舒继武. 网络存储安全[J]. 中国教育网络, 2007, (10): 33-35.

[2] Blaze M. A Cryptographic File System for Unix[C]//Proceedings of the 1st ACM Conference on Communications and Computing Security. [S. l.]: ACM Press, 1993: 9-16.

[3] Heidemann J S. File System Development with Stackable Layers[J]. ACM Transactions on Computer Systems, 1994, 12(1): 58-89.

[4] Huang Jianzhong, Xie Changsheng, Cai Bin. Research and Implement of an Encrypted File System Used to NAS[C]// Proceedings of the 2nd IEEE Int'l Conf. on Security in Storage. [S. l.]: IEEE Press, 2003: 73-77.

[5] Zadok E. Cryptfs: A Stackable Vnode Level Encryption File System[R]. Columbia, USA: Computer Science Department, Columbia University, Tech. Rep.: CUCS-021-98, 1998-06.

[6] 郑晓林, 荆继武. 基于身份加密的密钥管理方案研究[J]. 计算机工程, 2006, 32(21): 145-147.

[7] 赵铭伟, 毛 锐, 江荣安. 基于过滤驱动的透明加密文件系统模型[J]. 计算机工程, 2009, 35(1): 150-152.

[8] Schneier B. The Blowfish Encryption Algorithm[J]. Dr. Dobb's Journal, 1994, 7(3): 38-40.

编辑 张正兴

（上接第 136 页）

参考文献

[1] Greg H, James B. Rootkits: Subverting the Windows Kernel[M]. [S. l.]: Addison-Wesley Professional, 2006.

[2] 陈文钦. BIOS 研发技术剖析[M]. 北京: 清华大学出版社, 2001.

[3] 杨 彦, 黄 皓. Windows Rootkit隐藏技术研究[J]. 计算机工程, 2008, 34(12): 181-183.

[4] 彭 毅. BIOS Rootkit 及其检测技术的研究[D]. 重庆: 重庆大学, 2008.

[5] Thimbleby H, Anderson S, Cairnsp A. Framework for Modeling Trojans and Computer Virus Infection[J]. The Computer Journal, 1998, 41(7): 444-458.

[6] 张新宇, 卿斯汉. 特洛伊木马隐藏技术研究[J]. 通信学报, 2004, 25(7): 153-159.

[7] 胡和君. Windows Bootkit 的技术研究与应用[D]. 成都: 电子科技大学, 2009.

[8] Russinovich M E, Solomon D A. 深入解析 Windows 操作系统[M]. 4 版. 潘爱民, 译. 北京: 电子工业出版社, 2007.

[9] Intel. Intel Architecture Software Developer's Manual, Volume 3: System Programming Guide[Z]. 2003.

[10] Richard C, Detmer R C. Introduction to 80x86 Assembly Language and Computer Architecture[M]. 北京: 机械工业出版社, 2006.

[11] Icelord. BIOS RootKit[EB/OL]. (2008-12-23). http://www.xfocus.net/articles/200705/918.html.

[12] Cheng5103. BIOS 中隐藏 Telnet 后门[EB/OL]. (2009-02-17). http://www.xfocus.net/articles/200903/992.html.

编辑 张正兴