基于安全操作系统的透明加密文件系统的设计*)

魏丕会　卿斯汉　刘海峰

（中国科学院软件研究所　北京100080）（中国科学院信息安全技术工程研究中心　北京100080）

Design and Implementation of a Transparent Cryptographic File System Based on Secure Operating System

WEI Pi-Hui　QING Si-Han　LIU Hai-Feng

(Institute of Software Chinese Academy of Sciences, Beijing 100080)

(Engineering Research Center for Information Security Technology Chinese Academy of Sciences, Beijing 100080)

E-mail: wphui@sina.com

Abstract　Almost all the important information is saved on physical media as files and managed by file system. So filesystem's security is an important promise to information security. We present a transparent cryptographic file system based on secure operating system(SecTCFS). The users do not aware the exist of the encrypting process. Authentication promises that valid user can access the files in the system.

Keywords　Information security, File system, Transparent encryption, Authentication

1　引言

随着Internet的飞速发展，信息资源的共享程度进一步加强，随之而来的信息安全问题也日益突出。但是现在的系统越来越复杂，系统中总是存在着各种各样的漏洞[1,2]，以及一些人为的因素（如：没有合理配置防护墙规则，口令比较弱等）。这些都有可能被黑客利用[2]，入侵到系统中去。

在分布式文件系统中，这种安全问题尤为严重。这种系统中存在交换信息的两端：服务器端有本地文件系统，服务器直接访问本地文件系统中的文件；客户端要访问服务器本地文件系统中的文件。服务器端和客户端通过通信网络相连。以Sun的网络文件系统(NFS)[3]为例，NFS对信息实施很低级的保护，客户端向服务器提出请求，服务器将文件信息以明文方式传送给客户端，在客户端和服务器之间的搭线窃听者会很容易地得到文件信息[4]。另外，服务器仅仅通过客户端传送的uid或者gid来判断客户端是否有权获得数据，这样任何非法客户端只要设法构造出正确的uid或者gid就可以获得数据，这时的信息没有安全性。

防止信息泄漏的措施可以在3个级别上进行：用户级、应用级和系统级。在系统级防漏，可以更好地对信息进行保护[5,6]。传统的文件系统对于入侵者也是很脆弱的，只要入侵者能够欺骗操作系统，系统中的信息就不再是秘密的。另外，如果入侵者能够得到计算机系统的物理硬盘，那么盘上的所有明文数据就全部泄露了。对文件系统加密，服务器的本地文件系统只存放文件密文，能够大大加强系统的安全性。对文件系统的加密应该对用户透明，也就是用户感觉不到是在加密文件系统上工作。只要是合法用户，就能和访问普通文件系统一样进行访问。

Matt Blaze在1993年实现了加密文件系统(CFS)，文件内容在通过不可信媒介传输前要先进行加密，在可信端进行解密。但是，CFS系统对用户是不透明的，用户必须显式地创建加密文件系统。CFS的加密粒度是在目录级，用户必须为每个目录记住一个密码，这就限制了使用的灵活性。CFS不能实现对加密文件的共享。2001年，G. Cattaneo等实现了基于UNIX的透明文件加密系统(TCFS)[8]，很好地克服了CFS的缺点，服务器端的文件加密存放，客户端的内核通过远程系统调用，从服务器端得到密文数据，然后在客户端解密。但是，TCFS要求用户信任客户端，文件在客户端以明文方式存在。这样就会在客户端的临时区里存在大量明文数据，如果入侵者能够读取临时区，就能得到数据。本文在TCFS的基础上提出了基于安全操作系统的透明文件加密系统(SecTCFS)，系统中每个文件都有一个密钥，每个文件的密钥保存在文件密钥服务器中，用户不必相信客户机，只要求相信密钥服务器。

本文介绍基于操作系统的加密文件系统的体系结构、文件服务器的访问控制、密钥管理、加密引擎。

2　基于安全操作系统的透明加密文件系统体系结构

本加密文件系统基于的安全操作系统(SecLinux)是基于Linux资源自主开发的符合GB17859-1999第三级“安全标记保护级”[7]安全功能要求的一个安全操作系统。

整个透明加密系统存在3个实体：文件服务器、客户端和文件密码服务器。文件服务器是保存客户文件的系统，操作系统是Seclinux，它的本地文件系统是透明加密文件系统，用户不必相信文件服务器；客户端是用户访问文件服务器的远程终端，也是不安全的，客户也不必相信客户端；文件密码服务器保存文件服务器上所有文件的密码，对这些密码需要加以特别保护。

*)本课题受中科院知识创新工程重大项目(YC2K5609)和国家保密局科研项目共同支持，同时是国家“863”委托开发课题(863-301-06-04)。魏丕会　硕士生，研究方向：信息系统安全理论与技术。卿斯汉　研究员，博士导师，研究方向：信息系统安全理论与技术。刘海峰　博士生，研究方向：信息系统安全理论与技术。

2.1 文件服务器简介

文件服务器使用 Seclinux 安全操作系统。SecLinux 安全操作系统除了具有一般操作系统的功能，在安全方面主要由以下几部分组成：

·标识与鉴别(Identification & Authentication)。不仅检查用户的登录名和口令，赋予用户唯一标识 uid、gid，还检查用户的安全级、计算特权集，赋予用户进程安全级和特权集标识，从而保证了只有合法用户以系统允许的安全级和特权集存取系统资源。

·自主存取控制(Discretionary Access Control)。是按用户意愿进行的一种存取控制机制。使用该机制，属主可以自由地决定其资源由系统中哪些用户以何种权限进行访问，粒度可达到每个用户。

·强制存取控制(Mandatory Access Control)。将系统中的信息分密级和类进行管理。SecLinux 实现的强制存取控制对系统中的每个客体(进程、文件、设备、管道、IPC 客体)都赋予了相应的安全级。当一进程访问一个客体时，依据相应的安全规则，确定是否允许访问。

·最小特权管理(Privilege Access Control)。SecLinux 将原超级用户的特权细化为32个特权，分别授予4个不同的角色：系统安全管理员(SSO)、安全审计员(AUD)、安全操作员(SOP)、网络管理员(NET)，每个角色只具有完成其任务所需的特权，从而满足最小特权原理。

·安全审计(Security Audit)。对系统中安全相关事件进行记录、检查及审查的过程。

·可信通路(Trusted Path)。提供给用户可信的登录通道，以防止伪造的登录序列。

·密码服务(Cryptogram Service)。SecLinux 中内含自主设计的密码算法(QC 算法)，并提供所需的 API。

·网络安全(Security Network)。SecLinux 的安全性在网络上的拓展，是强制存取控制机制和自主存取控制机制在网络安全服务上的应用。

每次用户需要从文件服务器读写文件，服务器都要负责对用户的身份进行验证，合法用户才能访问。同时，客户端和文件服务器之间的所有往来信息都是密文信息，即使存在旁路窃听者，也不能理解它们之间的对话。

2.2 文件密码服务器

传统的透明加密文件系统或者每个目录使用一个密码，整个目录下的所有文件使用统一密码，或者每个用户使用一个密码，属于同一个用户的文件使用同一个密码加密，这样就会在磁盘上存在很多同一个密码加密的文件，而且入侵者很容易知道哪些文件是同一个密码加密出来的，入侵者拥有了大量的密文就能进行唯密文攻击。SecTCFS 中每个文件使用一个密码加密，每个文件的密码保存在密码服务器上，对密码服务器进行保护。每次用户要从密码服务器上得到一个密码，必须向密码服务器进行严格的认证。只有通过认证，服务器才会提供密码。这就将对文件密码的保护独立出来，有利于模块化设计以及对系统密码进行更为有效的保护。

2.3 文件访问

在这个加密文件系统中，对文件的访问包括客户端写文件和客户端读取文件两部分。文件服务器负责保存加密的文件，而且对文件的访问采取严格的控制，只有合法用户才能有权利进行读写。

用户要从客户端读取文件服务器上的数据，必须从文件密码服务器得到这个文件的密码，然后文件服务器将文件密文传送给客户端。密文在客户端进行初步解密，得到第一步解密的密码。客户端进程对这些信息采取严格的保护，就是一边使用一边彻底解密，保证客户端机器上不会有大量的明文数据，从而减小了信息泄漏的可能性。

当用户要从客户端写文件到文件服务器，客户端先向文件密码服务器申请一个密码，这个密码由文件密码服务器随机生成，并且由文件密码服务器保护存放，客户端使用这个密码将文件加密并且传输密文到文件服务器。

这样的设计具有以下优点：

· 对用户几乎没有什么影响，用户不必记住每次使用的密码，实际上用户自己也不知道自己使用什么密码对文件进行加密，减轻了用户的负担。

· 用户不必相信客户端，因为客户端仅仅处理第一步解密的数据，这些数据仍然是加密的，只不过考虑到处理的速度，使用了比较简单的加密方法。

· 客户端和文件服务器端的数据传输都是密文传输，不怕中间有入侵者截获数据。

· 文件服务器管理员意识不到加密系统的存在，文件服务器对于管理员来说是一个普通的系统，减轻了管理员的负担。

· 通过对每个文件分配一个加密密码，保证了服务器上不会存在同一个密码加密的大量数据，使得入侵者不可能进行唯密文攻击。

· 对文件密码进行集中的管理，有利于系统的安全。传统的文件加密系统要求用户记住每次加密的密码，而且用户要相信客户端是绝对安全的。这样，实际上将对文件密文的保护交给了单个用户，用户如果不小心泄漏密码，那么他的文件将会不具有安全性了。通过一个文件服务器管理文件系统中所有文件的密码，有利于减轻用户负担，而且增大了系统的安全性。

3 文件服务器的访问控制

文件服务器是基于安全操作系统 Seclinux[6]的主机系统，对每个要访问系统的用户都要进行详细的访问控制，只有合法用户才能存取系统的文件。传统的文件加密系统仅仅依靠加密进行文件的保护[5]，用户只要能够登录文件服务器就能得到文件服务器中的加密文件。非法用户没有文件的密码，不能解密文件，但这样的设计使得加密的密文文件可以被任何用户获得，用户就能针对文件进行攻击试验，从而有可能解密文件，泄漏信息。在 SecTCFS 中，文件服务器保证了文件不能被非法用户获得，大大加强了对文件的保护。

文件服务器对用户的访问控制包括用户身份认证和安全检查两部分。

·用户身份认证。通过标识与鉴别机制保证只有合法用户才能存取系统中的资源。它识别每个用户的真实身份，并为每个用户取一个名称——唯一的标识符。唯一标识符必须是唯一的且不能被伪造，这样一个用户不能冒充另一个用户。将唯一标识符与用户联系的动作称为鉴别，用以识别用户的真实身份。在 Linux 中，鉴别是在用户登录时发生的，系统提示用户输入口令，然后判断用户输入的口令是否与系统中存在的该用户的口令一致。

用户登录时，可选择输入安全级。用户标识与鉴别机制检查该安全级是否在安全文件档中定义的该用户安全级范围之

内。若是，则认可，否则拒绝该次登录。若用户没有选择安全级，使用缺省安全级。

当标识与鉴别机制确认用户的登录名、口令和安全级后，便允许用户登录。

·安全检查。安全检查控制包括自主存取控制、强制存取控制[8]，在用户合法登录，请求读写文件的时候进行检查。

SecLinux 安全操作系统的自主存取控制机制是在原 Linux 系统保护模式“文件主/同组用户/其他用户”的基础上进一步细化而成，它保持了与原系统的兼容性，并将用户粒度细化到系统中的单个人，即能够赋予或排除系统中某一个用户对一文件或目录的存取权限，克服了原 Linux 系统中只能将存取权限分到组或所有其他用户这样一种较粗粒度的局限性。

SecLinux 安全操作系统的强制存取控制(MAC)机制，需要对系统中的每个进程、每个文件、每个 IPC 客体赋予相应的安全级。当一进程访问一个客体时，调用 MAC 机制，根据进程的安全标识和访问方式，比较进程的安全级和文件的安全级，从而确定是否允许进程对文件的访问。

只有当用户进程同时通过了自主存取控制和强制存取控制才能访问文件。这样，Seclinux 的访问控制比普通操作系统的访问控制有了很大的加强，能够保护数据不被非法访问。

4 密钥管理

在 SecTCFS 中，密钥的分配都是由文件密码服务器提供的。对密钥的分配分成3部分设计：用户向文件服务器写入文件、用户从服务器读取文件以及用户从文件服务器删除文件。当用户向文件服务器写入文件时，用户需要向密码服务器索取一个加密密钥。密码服务器给用户密钥以后，需要保存这个密钥，以备将来用户读取文件的时候解密。在用户从文件服务器读取文件的时候，需要同时向密码服务器索要文件的密钥，对文件解密。在密钥管理过程中，要用到公钥密码体制[9~12]的数字签名技术。

用户向文件服务器写入文件的过程如图1所示。

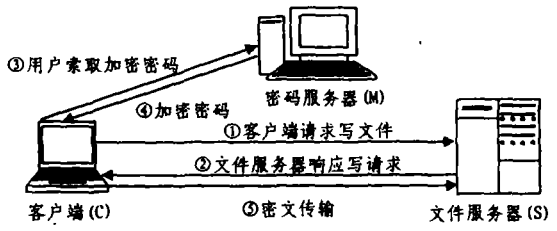

图1 用户写文件过程

①用户在客户端请求写文件。用户将自己的身份、签名、时间戳以及要写入文件的位置使用文件服务器的公钥加密，然后传送给文件服务器。

②文件服务器响应写请求。文件服务器使用自己的私钥解密客户端传送的信息，检查用户是否拥有写入文件权限。如果有，文件服务器将自己的签名、时间戳、用户的标志、允许写标志以及文件服务器给密码服务器的分配密钥信息(带有文件服务器的签名、请求写文件用户的标志信息、时间戳，并且使用密码服务器的公钥加密)用用户的公钥加密，传送给用户。

③用户索取加密密钥。用户用自己的私钥解密文件服务器的信息。如果文件服务器允许写，就将自己的请求密钥信息、文件服务器给密码服务器的分配密码信息(此时仍然是使用文件服务器公钥加密的)、时间戳以及自己的签名用密码服务器的公钥加密，传送给密码服务器。

④密码服务器分配加密密码。密码服务器用自己的私钥解密信息。如果服务器给自己的请求写用户的身份与向自己发送消息的用户身份一致，并且文件服务器指示自己分配密钥，密码服务器就产生一个密钥，将密钥、自己的签名以及时间戳用用户的公钥加密，传给用户。同时，文件服务器记录下这个文件信息以及文件的密码。

⑤密文传输。用户用自己的私钥解密密码服务器的信息，得到密码，然后用户使用这个密码加密文件，将密文传给文件服务器。

在上述过程中，时间戳用来防止入侵者的重放攻击。在每个时刻，入侵者截取了某一步传输信息，以后重放的时候，接收这个重放信息的一端立刻就能根据时间戳发现这是一个非法信息，从而直接抛弃。公钥密码系统在这个过程中起着很重要的作用。

用户从文件服务器读取文件的过程如图2所示。

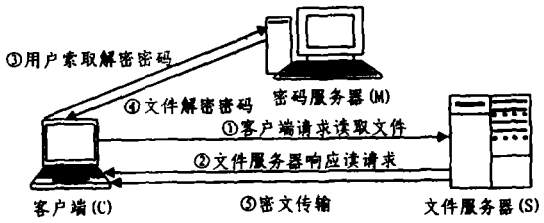

图2 用户读取文件过程

①用户在客户端向文件服务器请求读取文件。用户将自己的身份信息、签名、时间戳以及要读取的文件信息用文件服务器的公钥加密，传给文件服务器。

②文件服务器响应读请求。文件服务器用自己的私钥解密客户端的信息，得到客户端用户的身份、签名以及要读的文件信息。文件服务器根据这些信息检查用户是否具有读权限。如果用户可以读，文件服务器将自己的签名、时间戳、用户的标志、允许读标志以及文件服务器给密码服务器的传送密码指令信息(带有文件服务器的签名、请求读文件用户的标志信息、时间戳，并且使用密码服务器的公钥加密)用用户的公钥加密，传送给用户。

③用户索取解密密码。用户使用自己的私钥解密文件服务器发来的响应信息。如果文件服务器允许自己读取文件，就向密码服务器发送请求密码信息。用户将自己的用户标志、签名、时间戳和文件服务器发送给密码服务器的传送密码指令信息(带有文件服务器的签名、请求读文件用户的标志信息、时间戳，并且使用密码服务器的公钥加密)用密码服务器的公钥加密，传给密码服务器。

④密码服务器传送解密密钥。密码服务器使用自己的私钥解密客户端传来的数据，然后进行验证。如果验证通过，就从自己的数据库中取出文件密码，连同自己的签名、时间戳使用用户的公钥加密，传给用户。

⑤文件服务器传输文件密文。同时，文件服务器将文件传送给客户端，客户端使用解密密钥解密文件，进行读取。

用户从文件服务器删除文件的过程如图3所示。

①客户端请求删除文件。客户端将自己的用户标志、签名、时间戳、请求删除文件信息使用文件服务器的公钥加密，

传给文件服务器。

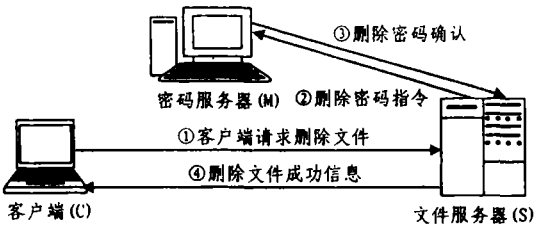

图3 删除文件过程

②文件服务器向密码服务器发送删除密码指令。首先文件服务器根据用户传送的请求检查用户是不是有权删除文件。如果有,文件服务器先向密码服务器下达删除文件密码指令,即文件服务器将自己的签名、时间戳、文件信息用密码服务器的公钥加密后传给密码服务器。

③密码服务器向文件服务器发送删除密码确认。密码服务器根据文件服务器的指示,删除自己保存的对应文件的密码,然后向文件服务器发送确认信息,即密码服务器把自己的签名、时间戳、密码删除信息用文件服务器公钥加密,传给文件服务器。

④文件服务器向客户端传送删除文件成功信息。得到密码服务器的删除密码,确认信息以后,文件服务器将文件删除,同时向客户端传送删除文件成功信息。即文件服务器将自己的签名、时间戳、文件已删除信息用用户公钥加密,传给用户。这样就完成了文件删除任务。

从上面论述可以看出,文件服务器在整个系统中占有中心地位。它核实用户身份,确定用户是否能够进行所请求的操作,同时让密码服务器协助用户完成指定的操作。这样,在文件服务器上只保存文件的密码形式,而且文件解密密码和文件分开保存,特别保护,保障了系统安全。文件服务器和客户端之间仅仅传送密码信息,防止了旁路窃听带来的危害。

5 加密引擎

在SecTCFS中,所有对文件加密和解密的操作对用户都是透明的。无论是在客户端还是在文件服务器端,操作员都感觉不到加密/解密动作的存在。密码是作为系统核心的一部分,我们参考了TCFS[5]的做法,把加密引擎设计成核心的可加载模块。这样,用户就能把自己的加密模块加入系统,从而提高系统的安全。

在我们的设计中,使用QC算法作为系统的加密模块。QC算法是我们自主设计并且通过国家验证的高强度对称加密算法。由于高强度的密码算法要受到国家的管制,从国外进口的软件产品其安全级别都是很低的,因此使用我们自行设计的高强度加密算法具有深远的意义。

采用对称加密方法每次对一个数据块(如8bytes)加密,存在一个问题。因为对于同一个密码,同样的明文,加密后的密文也是相同的。这样,对于一些明文多次重复出现的文件(比如C源代码文件),加密后的密文片断会重复出现。如果非法用户对这样的文件进行结构分析,将会极大地损害文件的安全性。对此,Matt Blaze在CFS[6]中提出了一种解决方法:对每个文件产生两个不同的密码,一个用于对文件进行预处理,一个用于加密。对于第一个密钥(假设其长度是56bits),对文件中的每个字节(假设其在文件中的偏移量是i)的第i%56位与密钥的第i%56位进行异或。对文件进行这样的预处理以后,文件中就不会有大量重复出现的内容了,然后对这个预处理后的文件进行加密。解密的时候,先根据第二个密码将密文解密,然后用第一个密码的第i%56(i是密文在文件中的偏移量)和密文字节的第i%56进行异或,得到明文。

在对文件进行加密的时候,加密端要在密文文件的头部产生校验块。采用MD5算法根据文件的内容产生文件校验码。解密的时候,如果文件校验不通过,就认为是非法文件,用户端立刻就会发现错误,进行错误处理。

这样的处理有以下优点:

①密文不会出现大量重复出现的内容。

②对密文可以随机存取,对任何一段密文都可以根据密码恢复出明文。

③文件认证块用于检测文件密文在文件服务器上是不是被非法改动了。这样,任何对文件的非法改动都会被用户发觉。

结束语 本文提出了一种基于安全操作系统的文件透明加密系统的设计方法。在这个系统中,每个文件使用单独的密码加密,用户意识不到加密过程的存在,减少了对用户的依赖性;同时,系统中所有文件的密码存放在密码服务器,不必让用户记忆文件的密码,密码的这种集中管理有利于系统运行的安全。客户端和文件服务器之间仅仅存在密文的交换,而且文件服务器上只存放文件的密文,系统安全性有很大增强。文件服务器负责对用户身份进行认证,只有合法用户才能访问文件。

参考文献

1 Miller B P, Koski D, Lee C P, et al. Fuzz Revisited: A Re-examination of the Reliability of UNIX Utilities and Services: [Technical report, CS-TR-95-1268]. Computer Sciences Department, University of Wisconsin, April 1995

2 Lee W, Stolfo S J. Data mining approaches for intrusion detection. In: Proc. of the 7th USENIX Security Symposium, 1998

3 Sandberg R, Goldberg D, Kleimann S, Walsh D, Lyon B. Design and implementation of the Sun Network Filesystem. USENIX Association Conf. Proc. June 1985. 119～130

4 McCanne S, Jacobson V. The BSD Packet Filter: a new architecture for user-level packet capture. In: Proc. of the 1993 winter USENIX conf. San Diego CA, 1993. 259～269

5 Cattaneo G, Catuogno L, Sorbo A D, Persiano P. The Design and Implementation of a Transparent Cryptographic File System for UNIX. USENIX Annual Technical Conf. 2001, Freenix Track. (Boston MA, June 29, 2001)

6 Blaze M. A Cryptographic File System for Unix. First ACM Conference on Communication and Computing Security, Fairfax VA 1993. 158～165

7 中华人民共和国国家标准:计算机信息系统安全保护等级划分准则,GB 17859-1999,1999年9月13日

8 刘海峰,卿斯汉,刘文清. 安全操作系统的设计与实现. 计算机研究与发展,2001,38(10):1262～1268

9 Diffie W, Hellman M E. New Directions in Cryptography. IEEE Transactions on Information Theory, 1976, IT-22(6): 644～654

10 Rabin M O. Digital Signatures. Foundations of Secure Communication, New York: Academic Press, 1978. 155～168

11 Schaumuller-Bichl. On the Design and Analysis od New Cipher Systems Related to the DES: [Technical Report]. Linz University, May 1981

12 Adams C, Lloyd S. Understanding Publickey Infrastructure: Concepts, Standards, and Deployment Considerations, Macmillan Technical Publishing, 1999

13 Schneier B. Applied Cryptography Second Edition: protocols, algorithms, and source code in C. ISBN 7-111-07588-9