eCryptfs v0.1 Design Document

目录

1. 引言
2. 威胁模型
3. 功能概述
   1. VFS对象……………………………………………………………………………………………………………………….3
   2. VFS操作……………………………………………………………………………………………………………………….4

3.2.1 挂载

3.2.2打开文件

3.2.3页读取模式

3.2.4页写入模式

3.2.5文件截断

3.2.6 文件关闭

4. 密码学特性

4.1 密钥管理

4.2 密码保密执行

4.3 文件格式

4.3.1 标记

4.4 发展与想法

4.5 密码学概要

1 引言

此文档详述了关于eCryptfs的设计： eCryptfs 是一个基于linux平台下兼容POSIX的企业级别的文件加密系统。它起源于Erez Zadok's Cryptfs，并通过FiST框架实现层叠式文件系统的产生。它将加密元数据存储在每一份被写过的[文件头](http://baike.baidu.com/view/1606305.htm)中，从而实现文件在不同[主机](http://baike.baidu.com/view/23880.htm)之间的拷贝。该文件将被用合适的秘钥解密，此外，没有必要去跟踪其他更多的信息（除了已经在加密文件中的）。

eCryptfs是一个本地的linux平台下文件系统， 它是为linux内核而构建的独立的内核模块，而没有必要去使用其他任何的内核补丁。

设计者在阶段上实现了eCryptfs的特性。第一阶段（版本0.1）包含了mount wide 密码支持和密码保密执行。第二阶段（版本0.2）将会包括mount wide 公钥支持和数据完整性执行。第三阶段（版本0.3）将会包括每个策略文件（per-file policy）支持。这个文档提供了关于eCryptfs文件系统（修订版本0.1）的技术说明书，并且设计者推荐eCryptfs应被归并到linux主线内核中。

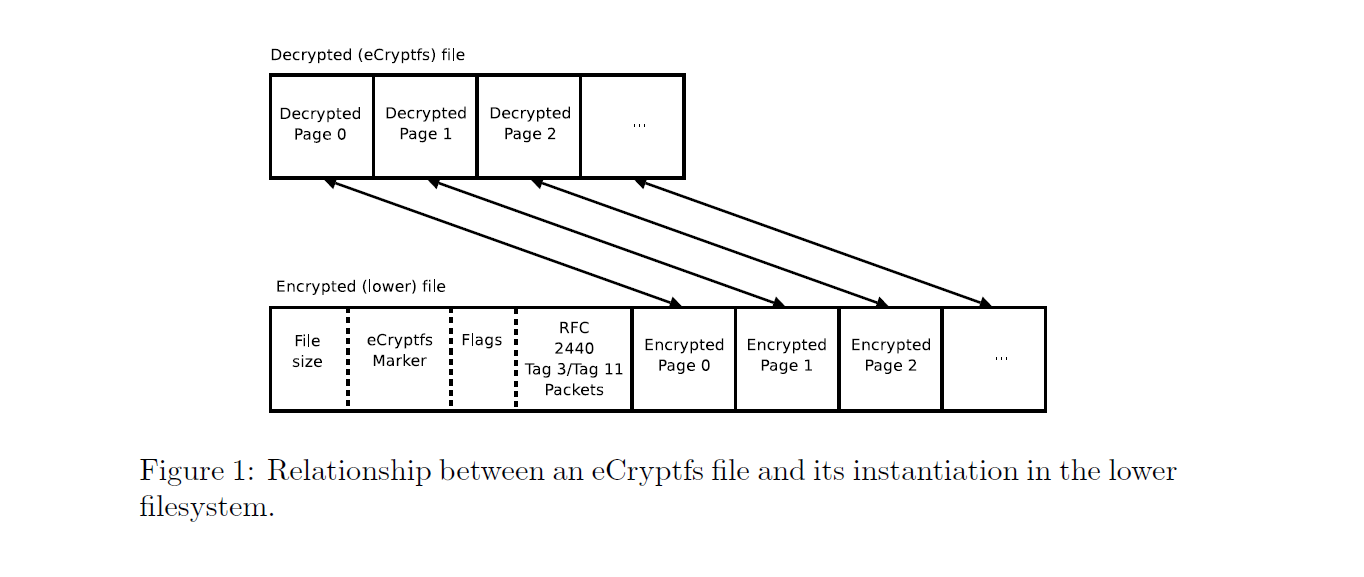
Michael Halcrow出版了两个涉及了Ottawa Linux Symposium(2004 and 2005)平台下的eCryptfs的文献。 这些文献除了提供了eCryptfs高层次的概述，还提供了一些不同主题的关于基于linux下文件系统安全的的广泛深入讨论。

2 威胁模型

如果有一个没有授权的用户在无法控制主机操作环境的情况下而获得了进入读取数据的权力，eCryptfs（版本0.1）就可以保护数据的机密性。一个秘钥predicates没有加密内容的个人文件对象的权力。一个用户没有与指定的文件相关的密码，就不能识别关于指定加密文件内容的重要（strategic）信息，除了那些可以从文件名，文件大小，或者其他关于该文件的元数据中推断出来的信息。攻击一个加密的eCryptfs文件和通过GnuPG(使用相同的密码、秘钥等等)攻击一个加密的文件应该是几乎一样困难。

磁盘上的文件不是中间状态的会比终态更加容易被攻击；在eCryptfs操作中系统错误和电源故障的情况下，不会有特别的已写内容会减弱文件的保密性。攻击者不能够通过水印攻击来察觉eCryptfs的使用者是否存储了特别的纯文档。我们认为一个攻击者潜在的会有进入每一个中间状态的加密文件（在辅助存储器上）的权力。

除了通过了标准POSIX文件的准许或者Mandatory Access Control之类的并且已经实现的，eCryptfs没有再提供其他控制功能的权力。0.1的版本没有包含完整性认证，这个特性将会被包含在之后的版本中。



1. 功能概述

eCryptfs是一个基于linux内核的VFS上实现的堆栈式文件系统。因为eCryptfs是堆栈式的，它不直接向块设备中写入，而是通过更低层的文件系统上挂载在目录的最上层。大部分兼容POSIX的文件系统的可以充当低层文件系统使用；比较熟知的有EXT2,EXT3,JFS可以和eCryptfs一起工作运行。eCryptfs文件系统的对象（包括i节点，目录项和文件对象）与在低层文件系统的对象有着一对一的联系。

eCryptfs可以对文件的内容进行加密或者解密，修订版本0.1通过文件的其他属性可以不用加密，如文件大小，文件名，进入权限，时间戳，扩展属性。目录内容也被可以不用加密。

eCryptfs是来自于Cryptfs【2】，而Cryptfs【2】是FIST框架的一部分，由Erez Zadok【3】发展和维护。

3.1 VFS对象

eCryptfs维护了在eCryptfs文件系统上的对象和低层文件系统上的对象的引用（reference）。在eCryptfs上，这个指向低层文件系统对象的引用是通过文件对象的private\_data指针、i节点对象u.generic\_ip指针、目录对象d\_fsdata指针、超级块对象s\_fs\_info指针来维护的。这个指向eCryptfs目录项、文件、超级块对象的指针只引用了相应的低层文件系统对象。

这个i节点u.generic\_ip指针引用了一个包含了关于密码操作的状态信息的数据结构和一个低层i节点对象的引用。ecryptfs\_crypt\_stat结构是一个i节点密码状态结构；这个结构的内容见图2。eCryptfs向ecryptfs\_crypt\_stat结构添加了存储在低层文件的头部分（已经存在的文件）或者来自mount-wide policy的信息（最近创建的文件）。

3.2 VFS操作

3.2.1 挂载

在挂载的时候，一个辅助应用会为用户指定的密码产生一个认证令牌。eCryptfs在linux内核中使用秘钥环\*把认证令牌存储在用户会话的秘钥环中。一个挂载参数包含了认证令牌的标识符。eCryptfs从使用该标识符的会话秘钥环中检索认证令牌。它为了最近创建的文件，使用了认证令牌的内容去建立密码系统的环境，也使用了认证令牌的内容来进入之前创建的文件的内容。

3.2.2页打开

此论文4.3节中包含了文件对于低层文件的格式。

当一个存在的文件在eCryptfs中被打开时，eCryptfs要打开低层文件然后读取文件的头部。eCryptfs标志的存在是核实的，标志被语法分析，补丁也被语法分析。

在挂载时候，头部包含了的秘钥标识符与被指定的mount-wide秘钥标识符是相匹配的。如果eCryptfs不能在挂载的时候将秘钥标识符与指定的匹配的话，这个打开操作将会失败（返回EIO错误码）。eCryptfs通过by taking the MD5 sum of the file encrypted key产生根初始化向量。这个根初始化向量（IV）是第一个N字节的MD5 sum，N是一个由IV构成的字节

当处理头部信息的时候，eCryptfs修改了关于eCryptfs的i节点对象的ecryptfs\_crypt\_stat结构。对ecryptfs\_crypt\_stat结构的修改包括如下：

当运行VFS操作时，eCryptfs之后就会使用这些（如下）信息。

当打开一个文件而这个文件并不存在的时候，ecryptfs\_crypt\_stat结构会根据mount-wide policy（版本0.1中）来初始化。在以后的VFS操作前，这些信息是（如下）用来产生和写入文件头部：

一旦ecryptfs\_crypt\_stat结构被写入，eCryptfs为i节点会初始化内核的加密应用程序接口【crypto API】环境。密码环境在CBC模式中初始化，被使用在连续的页读取和页写入中。

3.2.3 页读取

读取只发生在打开的文件中，当可应用的认证令牌存在于用户会话的钥匙环中的，且在VFS系统调用（该调用有效的打开了文件）的时候，一个文件可以被打开。

在一个读取页中，eCryptfs页索引被插入到相应的低层文件索引，并考虑文件中的页头部。eCryptfs产生了初始向量，为了给出的页目录索引，通过将一系列的ASCII的页偏移量的文档呈现与根IV字节联系起来，并计算字符串的MD5 sum值。

然后eCryptfs从低层文件读取加密的页后解密该页。eCryptfs刚开始建立了密码结构去运行解密，这个导致对内核应用程序接口的调用（在版本0.1中，长度相当于一页）。这个解密的页是在来自于VFS页read系统调用。

3.2.4 页写入

在写入页中，eCryptfs运行了一系列的发生在读取页中的操作（见3.2.3节），只有数据被加密而不是解密。低层索引的插入，初始化向量的导出，页被文件加密秘钥加密，通过内核应用程序接口完成，然后加密后的页被写入到低层文件中。

3.2.5 页截断

当一个文件被截断成一个更小的文件或者扩展成一个更大的文件，eCryptfs会相应的更新文件大小区域（即低层文件的前8个字节）。当搜寻一个文件至末尾时，eCryptfs向加密的0字符串写入，该字符串是存在先前文件的末端和新文件的末端。

3.2.6 文件关闭

在版本0.1的eCryptfs中，在文件最初被创建的时候，头部的数据包集是不会改变的。当一个文件不再被进入时，VFS内核释放与相关的文件、目录项、以及i节点对象根据在VFS中标准的的资源存储单元分配集的过程。eCryptfs不会在文件上运行任何更进一步的加密操作。

1. 加密特性

4.1 秘钥管理

RFC 2440（openPGP）[1]很严重的影响了eCryptfs的设计，尽管RFC的绝对偏差是有必要支持随机进入文件系统的权力。eCryptfs在每个文件的头部都存储了RFC 2440-兼容的数据包。数据包类型使用了包括Tag3（密码）和Tag11（文字错误）。每个文件有个一个与之有关的独特的文件加密秘钥。这个文件加密秘钥相当于加密或者解密文件内容的对称秘钥。在文件被创建的时候，eCryptfs将通过linux内核get\_random\_bytes()功能调用去产生其加密文件秘钥，这个加密文件秘钥的长度是由被使用的密码而定的。eCryptfs会默认选择AES-128位的。

有效的eCryptfs i节点包含了加密环境和一个独特的对应i节点对应的环境。这个环境存在于数据结构中，该数据结构包含了一些东西像文件加密秘钥、加密名称、根初始化向量、与i节点有关的认证令牌签名、各种各样的指示i节点加密属性的标志、指向cryptoAPI结构的指针等等。ecryptfs\_crypt\_stat结构定义是存放在ecryptfs\_kernel.h头文件中，由图2中的元素组成。

文件加密的秘钥是被加密的，然后存储在低层（已加密的）文件的第一个数据库区，文件加密秘钥被认证令牌的秘钥加密，该秘钥也是加密文件的秘钥。认证令牌的类型反应了加密机制。这里有一个“通用”的密码认证令牌，是由eCryptfs在挂载的时候从用户特定的密码而来从而产生的。由密码转化成秘钥根据S2K审查（在RFC 2440中已描述），因为密码是有一点联系的。那个数据块就会被MD5-hash迭代的运行65536次，目的是产生加密文件加密秘钥的秘钥。

eCryptfs存储了在用户秘钥环中的认证令牌（一个linux内核秘钥服务的成员）。在挂载的时候，辅助脚本将包含mount-wide密码的认证令牌放入到用户的秘钥环中。

当eCryptfs打开了一个加密文件的时候，它会尝试匹配保存在文件头部的认证令牌，不是在挂载点上实体化的认证令牌。如果在挂载点上的认证令牌匹配了在文件头部的认证令牌，然后认证令牌使用实体化的认证令牌，去将在页写入和读入的操作时被使用加密/解密文件内容进行加密和解密。

* 1. 密码保密执行

在主机环境控制之外，eCryptfs执行了保密数据，是通过加密包含了该数据的文件对象的方法，eCryptfs通过分区如extents，使用linux内核密码API去运行加密解密的文件内容。

在0.1版本中，每个extent的长度是根据页大小而定的（通常是4096个字节）。由于每个文件包含了一个页的头部，都是被eCryptfs加密，这个在低层文件系统中加密文件将会一直是比未加密的大一个页。在eCryptfs文件和低层文件的读入和写入操作中，eCryptfs半透明的map页面指标。每个扩展将会以CBC模式独立的加密。

eCryptfs为每个来自riv（根初始化向量）的扩展派生出IV（初始化向量），并且每个文件中的riv都是独立的。这个根IV是一个MD5 hash的加密文件秘钥的子集。这个扩展的IV产生过程需要计算秘密的根IV的MD5总和，此IV与代表扩展索引（extent index）的ASCII二进制字符有联系。

当一个readpage()请求通过作为VFS系统调用的结果时候，将在低层（加密）文件中插入页索引去寻找相应的扩展。eCryptfs读入此扩展，然后将它加密。每个扩展被以eCryptfs为文件选择的密码加密，同时这个文件被创建（在0.1版本中，这个是AES-128的默认）。每个扩展区域是独立于其他扩展区域，它们没有以任何方式被链接。

当一个writepage()请求通过作为VFS系统调用的结果，eCryptfs将会读入来自低层文件（使用描述在优先的段落中的过程）的目标扩展。根据页请求，这个页上的数据会被修改。整个（修改后的）页将会被以相同的IV和秘钥来重新修改（以CBC模式），该IV和秘钥是最初加密该页的，然后最近加密的页会被写入低层文件。

未来的版本将会支持完整性认证。

* 1. 文件格式

这个版本只支持mount-wide密码，数据包集只包括一个单独的Tag3跟着一个单独的Tag11数据包。这些数据包存储了加密文件的加密秘钥，也遵守在RFC 2440中给出的说明。

前20个字节包含了文件大小，eCryptfs标记，还有一系列的状态标志。从第20个字节开始，只有RFC 2440-兼容的数据包是有效的。

在RFC 2440数据包集中，每个Tag3（密码）数据将会立即被在Tag3上包含密码标识符的Tag11（字面上）继承。这个标识符是由哈希这个秘钥（该秘钥是从S2K操作中的密码产生的）组成的。版本0.1只支持一对与mount-wide密码有关的Tag3/Tag4。

4.3.1标记

每个文件下的eCryptfs标记是由产生的一个32字节的随机数（X）和在底层文件头部中8字节文件大小后面立即写入它组成的。

* 1. 发展与想法

在辅助存储器中（在信任主机环境控制之外），eCryptfs是与保护数据机密性有关。

eCryptfs在VFS层面上操作，但是它将不会加密写入交换辅助存储器中的数据。推荐在机器上（在某些点上敏感数据可能会被装载到内存），使用者使用dm-crypt去加密交换空间。

密码的选择需要遵守标准的强密码准则。在杂项设备/目录中，eCryptfs包含了各种的辅助应用。使用对你任何方便的工具去产生一个强密码串。用户应该存储字符串在安全的地方，然后当出现提示的时候，使用它作为密码。

* 1. 密码学概要
  2. 在版本0.1中，为eCryptfs设计秘钥的元素有：

页头部：包括了纯文本文件大小，eCryptfs标志，版本，标记，和RFC 2440数据包。

Mount-wide密码：已一种认证令牌的形式储存在用户秘钥环中。

每个文件有一个独立的、随机产生的文件加密秘钥。这个文件加密秘钥被加密，然后作为Tag3数据包（定义在RFC 2440）储存在头文件中。

认证令牌标识是由用来加密文件秘钥的hash秘钥组成的。

秘钥是根据在RFC 2440中描述的S2K而产生，也用于加密文件秘钥。

页大小扩展被在CBC模式中默认的密码（AES-128）加密。

每个文件根IV是在每个文件中的MD5的总和。

每个扩展的IV由根IV和页索引的ASCII和那个字符串的总和联系起来而产生的。

\*3.2：秘钥环：记录了应用程序需要的各种密码,方便用户使用。

CBC:

IV初始化向量：