|  |  |
| --- | --- |
| 学号 | 2017282110217 |
| 密级 |  |

**海量存储课程作业报告**

|  |
| --- |
| **DEFY: A Deniable, Encrypted File System for Log-Structured Storage** |

|  |  |
| --- | --- |
| 院（系）名 称 ： | 计算机学院 |
| 专 业 名 称 ： | 计算机技术 |
| 学 生 姓 名 ： | 李江琪 |

二〇一七年十二月

摘 要

当前保护文件系统最常用的方式就是对文件进行加密，但是当用户需要隐藏数据时，加密不仅无法解决问题。文件加密仅仅能够防止攻击者获取文件的内容，却无法阻止攻击者发现加密文件的存在，甚至有的时候对文件进行加密反而会使需要保护的文件变得引人注意。因此文件系统单单靠加密进行保护是不够的。而本文设计了DEFY——一种日志结构的deniable file system。所谓的deniable file system是指攻击者在不知道文件系统加密密钥的情况下，无法证明在该文件系统上存在任何的超越自己阅读权限的文件，是一个允许使用者进行“抵赖”的文件系统。但是已有的deniable file system均不适用于现在的移动设备，因为当前主流的移动设备使用的存储介质都是诸如NAND这样的基于闪存的存储器，这样存储器无法提供deniable file system所要求的原地擦除数据的功能。而日志结构可以满足固态驱动的限制，因此就可以运用在移动设备中。

关键词：信息安全；加密存储；文件系统

目 录

[1 研究背景 1](#_Toc19808)

[1.1 deniable file system 1](#_Toc16770)

[1.2 NAND 1](#_Toc12513)

[1.3 YAFFS概述 2](#_Toc7331)

[2 系统关键技术及设计 4](#_Toc5165)

[2.1 防御模型 4](#_Toc971)

[2.2 系统设计需求 5](#_Toc24114)

[3 系统设计概要 7](#_Toc3944)

[4 系统分析 12](#_Toc17907)

1研究背景

由于移动设备的普及，它们变得无处不在，并且功能也越来越强大。因此一个人的移动设备中通常收集和存储有大量个人敏感信息。而移动设备上的这些敏感数据也需要像PC平台上的一样需要保护其免受未经授权的访问。现在在移动端上已经有很多相关的应用程序，如ChatSecure （用于短信安全），WhisperYAFFS （加密文件系统），RedPhone （通话加密），TextSecure （用于短信安全），Orbot （tor移动安全软件），Lookout（数据备份和反病毒工具），等等。防止未经授权访问移动设备上的信息的标准方法与一般的安全通信相同：加密。虽然加密可以用于限制对某些文件的访问，但并没有隐藏文件的存在本身。而一些文件信息（如大小）还是可以在加密的情况下被攻击者获取。

## deniable file system

deniable file system，即可抵赖文件系统，是一种能够掩盖存储数据的相关信息的文件系统，用户可以利用它去否定（抵赖）文件在系统内的存在。这种系统一般是依靠通过基于数据的敏感性的用不同的密钥来加密数据。关于deniable file system目前已经有一些研究成果，但是现有的这些成果在应用于基于闪存的存储设备时是不适用的。为此本文设计了基于YAFFS的加密文件系统DEFY。 DEFY专为基于闪存的固态驱动器而设计，这是大多数移动设备中的主要存储设备。闪存的物理特性给deniable file system带来了巨大的挑战，而为此就得使用日志结构进行结构。

## NAND

NAND是目前主流移动设备使用的存储介质。NAND向上层提供页(page)级别的读写和块(block)级别的删除操作。当一个页上被写入内容后，除非擦除整个页，否则无法再向上写入数据。对于NAND来说，一个很重要的指标就是可擦写次数，当一个页的擦写次数超过了设计寿命时，整个存储介质就报废了。所以对于大部分的上层驱动来说，会尽可能的平衡每一个页面的读写次数，以保证最长的使用寿命。

由于不断增长的移动设备市场的需求，固态存储器借由其耗能小、速度快等特点，已逐步取代了传统硬盘驱动器。而闪存的发展在很大程度上是在成本，容量，性能，使用寿命和访问/擦除粒度之间所做的平衡。最新的闪存技术是NAND闪存。

与之前的EEPROM、NOR技术相比，NAND的制造成本更加低廉存储密度也有所增大。当前的NAND芯片最大有56GB。 NAND在页面级提供随机访问读取和写入，而擦除在块级别发生。例如，一个有212个块的8GB NAND设备可以写一个4KB的页面，但是必须以一个256KB块的粒度进行擦除。一旦页面被编程，它们必须在再次写入之前被擦除;这被称为程序擦除周期。每个页面的Outof-Bound（OOB）区域包含页面数据的元数据和纠错码。

闪存在多次擦除后性能就会下降。许多固态驱动器采用损耗均衡技术来延长其使用寿命：即驱动程序通过尝试分散擦除/写入流量以避免介质磨损。在动态损耗均衡中，数据将根据可用性和最少写入次数写入位置。在静态耗损均衡中，可以移动一些现有的未充分利用的（静态）块，以在页面写入请求期间在设备上分配磨损。大多数设备为了简单高效而使用动态损耗均衡。

 闪存设备可以使用Linux的内存技术设备（MTD）子系统接口访问，实质上为NAND闪存设备提供“原始”接口。 MTD提供从逻辑块到物理块的一致映射，它不提供写入均衡，因此不会阻止单元过度使用。未分类的块图像（UBI）接口基于MTD构建，提供了与Linux逻辑卷管理器相媲美的抽象。 UBI跟踪数据结构中的逻辑块，根据使用情况决定重新映射逻辑块，实现损耗均衡。闪存翻译层（FTL）可以构建在UBI之上，为闪存提供简化的块级接口，以换取对数据放置和严格覆盖的低级控制的损失。

## YAFFS概述

YAFFS是针对NAND而设计的轻量级日志结构文件系统（log-structured file system），因为其简单、轻便、低消耗而被移动设备广泛采用(Android2.3之后的版本使用ext4代替YAFFS)。作为一个日志结构文件系统，YAFFS写入请求是在逻辑地址空间内按顺序分配的，此外一个显著的特点在于，当一个页面的内容被修改时，该页面原来的内容并没有被删除，而是在另一个页中写入新的数据，然后将文件的指针指向新的页。这是由之前提到NAND的读写和删除操作粒度的不同造成的，当修改一个页面的内容时，由于包含该页的块中存在其他有用的数据，所以无法在这个块上执行删除操作，也就不能向这个块中写入新的数据。这个特性同样导致了YAFFS文件系统中删除操作并不能真正删除存储介质上的数据，而这些特性也被YAFFS的后继者EXT4文件系统集成。在这种情况下，磁盘上废弃的数据会在系统定期的垃圾回收(garbage collection)过程中被擦除，在此之前，废弃的数据会一直残留在磁盘上。

YAFFS中分配单位是页面（在YAFFS术语中称为块），大小从512字节到32KB不等。擦除的单位是块，每个块由32到128个页面组成，具体取决于NAND块的容量。 YAFFS使用闪存设备提供的OOB空间来存储页面元数据和纠错码。

现在已有两个版本的YAFFS：YAFFS1和YAFFS2。它们之间的关键区别在于：（1）YAFFS1只能处理大小不超过1KB的页面，而YAFFS2则支持更大的页面;（2）YAFFS2实现了一个真正的日志结构文件系统，写入新数据时不会覆盖。

每个YAFFS实体（文件，目录，链接等）都作为一个对象来维护，并带有一个对象头部。每个对象头都存储关于其关联对象的元数据，包括其名称，大小和页面的位置。目录的标题包含其子项（文件和子目录）的标题的位置。

# 系统关键技术及设计

## 防御模型

我们需要完成一个安全可靠的文件系统向敌人隐藏信息的存在。它需要隐藏所有可能指示文件是否存在的信息，但是它并不需要执行全系统的信息流动政策。 当前已经有人演示过如何使用所显示的文件的内容来推断隐藏文件的存在，比如，由桌面搜索服务产生的索引来找到文件。

假设用户，操作系统和应用程序的使用方式都是合乎规范的，这种情况下假定两种类型的攻击者：一次性访问设备（单视图对手）和那些具有定期访问（快照对手）的攻击者。在这种情况下，攻击者的“访问”产生磁盘的完整副本和文件系统的完整描述（比如通过其源代码的副本）。而且，攻击者可以强制用户显示一些隐藏文件。假设这里，对手不能访问设备的RAM内容，也不能捕获设备的运行状态。这种情况下DEFY可以通过一个接口以在紧急情况下立即清零RAM数据结构（包括加密密钥），即快速锁定功能，这可能导致有限的数据丢失。

单视图攻击模式下敌人只能访问文件系统及其用户一次。这种攻击方式既包括了许多常见情景：如设备被盗用，或被没收，用户被扣留进行质疑。快照对抗模型描述了在不同时间点对设备的访问授权的情况，并收集文件系统映像。例如，进入和退出警卫设施或在过境点。快照对手然后可以使用所收集的图像中的差异来识别设备上的改变的数据块。

这两种模式都包括对手使用各种手段（即威胁和人身暴力）来迫使用户展示一些文件的能力。这两种模式都允许攻击者在磁盘映像上执行高级计算机取证，使用密码破解程序，使用统计测试等。我们的工作就是在这样的威胁模型下开展的。以上攻击环境总结成：

文件系统，操作系统，应用程序均是可信的

攻击者能够获得操作系统的源码

攻击者能够获取任意时刻磁盘的完整拷贝

从对抗攻击的角度来说，如果攻击者无法证明用户隐瞒了数据，那么这个可否认系统就是成功的。也就是说，用户必须能够说服对手除了她已经选择揭示的内容，系统里已经没有别的数据了。换句话说，磁盘上所有其他的块都不得包含有效的数据。另外，面对存在快照对手，文件系统也会提供快照阻止，如果它具有可信的可否认性。特别地，用户可能已经在快照之间向磁盘写入数据，并且对手可以确定在访问之间已经修改了哪些块这种情况下被发现修改了的部分也都不能包含任何数据。

## 系统设计需求

 抵赖等级：在以前的deniable file systems实现中引入了抵赖等级的概念[，抵赖级别是形成敏感性等同级别的文件的集合（例如，情书对、商业秘密）。用户可以通过对应密码来显示所选的抵赖级别的所有文件。依据已有设计的规则，当解开一个层次时，也应该打开所有比它低的层次文件。该系统应该足够灵活，以适应多变的动态创建，而不是在初始化时预先指定和定义总分层水平。

安全删除：系统在可抵赖的基础上还有安全删除机制。安全删除确保在设备密钥被敌人获得的情况下，删除的对象也永久不可访问。我们希望安全的删除是高效的、细化的。精细删除意味着删除整个文件系统或一组文件与删除单个文件一样复杂。最后，为了保持可否认性，删除的数据不应该被能被辨认出来：它应该与系统未使用的数据区域和未被声明存有数据的位置一样混在一起不可区分。

现有两种可以完全删除数据并且使存储设备在擦除后可用的技术：安全覆盖和密钥处理。在安全覆盖中，旧数据被新数据覆盖，使得旧数据不可恢复。 但是针对从硬盘驱动器安全删除数据的技术已被证明对基于闪存的存储无效。而对于使用加密的系统，通过“忘记”相应的加密密钥可以安全地删除数据。没有密钥，就无法再次解密数据。加密密钥可能通过安全覆盖进行处理。 相比之下，密钥处理可能是最适合的闪存存储技术。

认证加密：认证加密提供机密性和数据完整性。保密性对于将数据隐藏在未公布的数据中至关重要，而当用户可能被迫放弃拥有他们的移动设备时即在物理安全性丧失的情况下数据完整性也是相当重要的。但是，经过认证的加密需要消息扩展 ，这使得密文大于原始明文，这也是将其集成到传统文件系统中的障碍。当前大部分已有的加密文件系统中仅使用未经身份验证的分组密码，其保留消息大小以满足基于块的存储设备的对齐限制。在我们的实践中，必须使用额外的存储器来管理与密文扩展相关的额外位。

最大限度地减少数据丢失：当隐藏数据被覆盖时，会发生数据丢失，因为文件系统安装在较低的级别上，更高层的隐藏文件就会被覆盖，这是不可靠的，但对于所有deniable file system这都是一个无法避免的问题。防止覆盖的一个策略是维护一个全局可自由写入的内存块列表（不被任何更高或更低的级别使用）。但这样使得系统失去了可抵赖性，敌方可以根据它来判断是否存在隐藏文件。 一种补救措施是创建废弃的块或者被错误标记为正在使用的块。这造成了似是而非的可否认性，以永久牺牲能力为代价。还有方案通过块复制防止系统中的数据丢失，但是这种方案需要花费相当大的开销。虽然NAND驱动器的容量在不断增加，价格也在下降，但闪存的每字节成本仍然是硬盘设备的两倍，因此我们一般使用开销较小的方案。

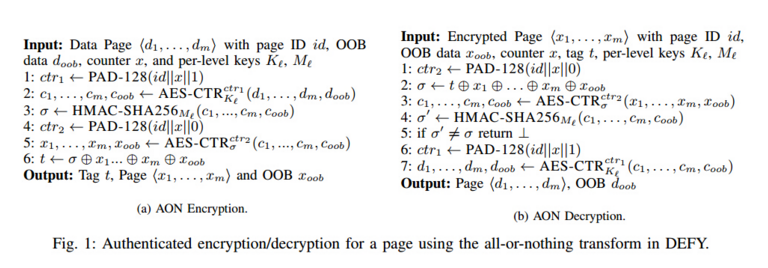
磨损平准化：NAND闪存在数据写入块失败之前的次数是有限制的。为此许多设备通过磨损均衡，使其中所有的写入被系统地写入到新的位置，从而防止一些块的过早失效。这对加密和抵赖文件系统都有影响：磨损均衡机制可能使得旧数据长期保持不变，从而表明了磁盘的活动区域，为对手更改磁盘提供了时间。磨损平衡否定了那些安全性源于覆盖数据能力的文件系统。任何为基于闪存的存储设计的安全文件系统都应该是安全的，并且与那些有或没有管理自己的损耗均衡的驱动器兼容。

易于部署：为了产生广泛使用，我们的设计应该能实现文件的轻松分配，并且与当前流行的操作系统（例如Android和Linux）兼容。使用可加载的内核模块来扩展现有的内核，可以修改系统而无需从源代码重建。

# 系统设计概要

DEFY被设计成YAFFS文件系统的扩展，其安全特性受到WhisperYAFFS的启发。选择YAFFS，是因为它设计用于NAND，支持耗损均衡，适用范围广，并且是开源的。对于YAFFS，我们添加了认证加密，加密安全删除的功能，并支持多抵赖级别，能够抵抗强大的对手。以下提供了DEFY主要设计特征的高级描述。

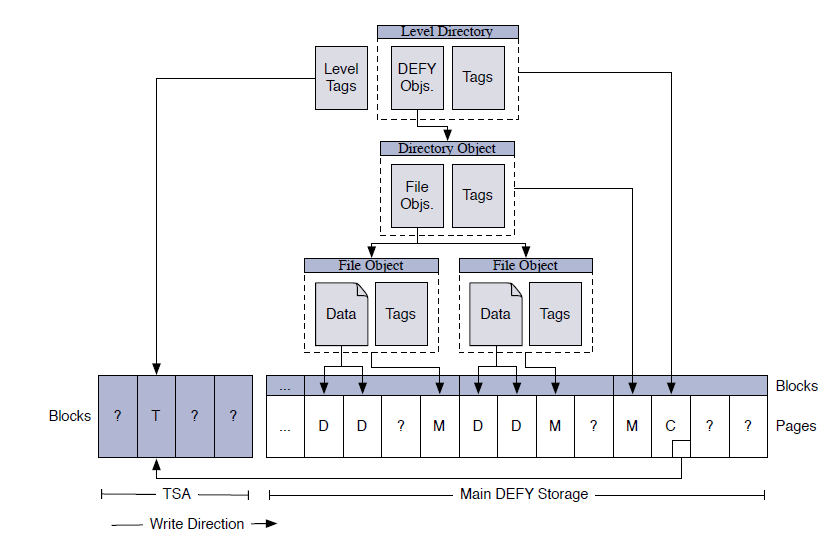
拒绝级别：DEFY支持一至多个抵赖级别，每个级别与级别目录相关联，并允许动态创建级别。每个级别的目录都存在于文件系统的根目录下。抵赖级别的所有文件都位于其级别目录下。每个抵赖级别都与唯一ID和加密密钥相关联，这些密钥由用户提供的密码派生而来。系统没有记录系统中存在的级别;它只能知道当前打开哪个级别。当用户显示一个级别时，所有较低的级别也显示出来。这样有助于尽量减少重写的机会（因为只有未知的几率会发生覆盖）。

认证加密：在DEFY中实现认证加密的两个关键是：（1）新的文件系统能适应由认证产生的数据扩展;（2）设计一个支持高效和细粒度安全加密删除方案。针对前一个重点，我们设计将该算法将输入的数据页面分为m个128位的消息（d1，...， dm），OOB数据（doob），唯一的页面标识符（id），唯一的全局计数器（x），每级加密密钥（K`）和每级MAC密钥（M`）。该算法实现加密。MAC方案：首先使用计数器模式AES（AES-CTR）加密页面和OOB数据，然后使用基于SHA的消息认证码（HMAC-SHA256）对产生的密文进行MAC编码。执行使用AES-CTR使用认证器作为密钥的额外加密以完成全有或全无转换（稍后描述）。标签（t）通过将密文块（x1,..., xm,xoob）与认证符（？）异或来产生。这个小标签不是秘密的;相反，它是加密数据的扩展，受制于全有或全无特性。将加密的页面（x1;：：xm）作为数据写入磁盘，将加密的OOB数据（xoob）写入OOB区域，并将标签（t）作为元数据存储在父对象中。加密的具体步骤如下

总之DEFY对每一个页进行单独加密，每个页的数据和OOB都会被单独加密，同时计算出一个额外的信息的tag，tag本身无法被用来解密数据，但是缺失了tag会导致无法正确的解密数据，tag存储在DEFY的metadata数据区域中。

基于加密的删除：提供认证加密的相同AON转换也提供了高效安全删除的手段。AON变换是一个加密函数，只有部分输出，它没有显示任何输入。没有一个密文的单个消息可以在不需要解密整个密文的情况下被孤立地解密。转换的初衷是为穷举搜索攻击提供额外的复杂性，这样就能使得攻击者为每个关键猜测对整个消息进行解密。 基于AON的加密安全删除方案，可用于安全删除版本化数据，即不允许删除密文块的任何子集（例如通过重写）以删除整个密文;没有所有的密文块，页面永远不能被解密。与经过身份验证的加密结合使用时，AON转换会创建一个绑定到相同全部或全部属性的消息扩展。这个小的扩展成为标签，可以高效地覆盖，以安全删除相应的页面。事实上，消息扩展是我们删除模型的基础，而且AON转换是DEFY提供高效安全删除的一个自然构造，因为它最大限度地减少了需要重写的数据量，不会使密钥管理复杂化，并且符合我们的分层删除模型。

DEFY的元数据：YAFFS中的元数据已被这里重新用于支持认证加密和安全删除。每个DEFY元数据对象都支持为其子对象存储标记：如果是文件对象，则为数据页;如果是目录对象，则为文件对象。当一个子对象被修改时，父对象被更新一个新的标签，覆盖以前的标签，安全地删除旧的对象。作为存储新标签的结果，父对象被修改。因此，在DEFY中创建，删除或修改对象将触发该对象路径中所有目录对象的标记级联，直至文件系统根目录。下图简要概述了DEFY的分层元数据设计。



级别目录的标签位于标签存储区（TSA）中，该区域与文件系统的其余部分分开管理。当等级标签被更新时，它们被写入到一个新的块中，并且先前的版本被擦除并用伪随机数据重新写入。即使用专门管理的闪存存储区域来实现安全删除。这种分层体系结构具有实现细粒度和高效安全删除的许多优点。单个对象，无论是页面，文件还是目录，都可以通过覆盖相应的标签并执行标签级联来安全删除。这种粒度扩展到级别目录，允许用户通过覆盖标签存储区域安全地删除整个级别或整个文件系统。由于YAFFS和DEFY将所有元数据对象存储在内存中，因此标记级联仅影响内存中的结构，并且不需要额外的设备I / O，从而限制了计算新标签的性能开销。

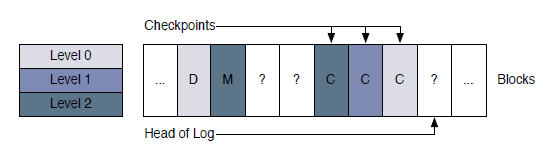
检查点：当DEFY卸载时，DEFY就会在文件操作过程中定期写一个检查点，代表当前日志头并且标记最近的一个特殊状态。检查点在日志结构文件系统中很常见，如在YAFFS中，它能够在崩溃后将文件系统恢复到一致状态。这通常是通过将所有脏数据和元数据写入磁盘，随后是包含当前时间和指向最新inode映射的指针的特殊标记来实现的。在崩溃之后，文件系统会“向后”扫描日志，以查找最近的检查点对象使得二者处于一致状态。

检查点在DEFY中的工作方式大概是：对于每个显示级别，特殊标记可以指示临时写入的日至头部。检查点的内容包括加密级目录对象以及指向标记存储区中的级别标记的指针。一个检查点用其各自的关键字进行加密，当揭示一个文件等级时可以对其进行解密和识别。

最小数据丢失：DEFY不会将所有跟踪已使用页面或空闲页面的数据结构保存到磁盘;任何在安装过程中无法解密的页面都被页面分配器认为是可以存放数据的。这会导致系统覆盖存储在未被解锁的级别中的数据。 DEFY防止覆盖的主要挑战来自维护日志结构文件系统的目标。磨损均衡设备不利于DEFY利用以前工作中使用的解决方案，这需要修改数据或写入完全随机的位置。DEFY采用三种策略来缓解可能由覆盖造成的数据丢失。

首先，DEFY强制执行一个政策，当一个级别被打开时，所有的较低级别也被揭示出来。因此，意外的覆盖效应只会在更高层次上阻塞。如果用户在日常操作（私密）期间始终显示最高级别，则不会发生意外覆盖。

其次，DEFY强制实施一个一级的政策。即对每一个块仅用于一个可抵赖性级别使用。因此，在显示块中标记为空闲的页面将保证是可以用的。此外，这简化了分配策略并防止在子块级别的数据丢失。

第三，DEFY以这样的方式写入检查点，以防止当文件系统被安装在较低级别时立即覆盖较高级别。当安装多个级别时，检查点被写入独立块，从最高权限到最低权限。因此，挂载在较低级别时写入的数据将避免较高级别的检查点，直到日志完全覆盖。

页面分配：DEFY使用在挂载文件系统时创建的内存中空闲页面位图来分配页面。任何显示级别密钥都无法解密的页面被标记为空闲。 DEFY只管理一个自由页面位图，但每个块都有一个关卡标记，确保分配给一个块的所有页面处于同一层次。如果在卸载文件系统时没有完全分配一个数据块，则会使用临时密钥填充由我们的AON转换生成的伪随机数据。

# 系统分析

对于可抵赖文件系统来说，衡量安全性最为关键的因素在于三种不同状态的基本块是否能够被攻击者所区分出来：未分配的块、已删除的块、未打开的块。而经过系统处理后这三种块在系统中看上去都是随机无意义的乱码。

在性能方面，本系统在读方面基本上没有额外的损耗，因为文件系统打开时已经完成了解密工作。在写方面和传统的可抵赖文件系统没有明显区别，和正常文件系统存在10-30倍的额外性能开销。同时，多个可抵赖级别并不会对系统性能造成额外的影响。