А. М. Коротин

О СПОСОБАХ РЕАЛИЗАЦИИ ПРОЗРАЧНОГО ШИФРОВАНИЯ ФАЙЛОВ НА БАЗЕ СЕРТИФИЦИРОВАННОГО СКЗИ ДЛЯ ОПЕРАЦИОННОЙ СИСТЕМЫ LINUX

В настоящее время одним из решений задачи обеспечения конфиденциальности информации, хранящейся в файловой системе, является шифрование. Существуют различные методы и способы шифрования. Например, пользователь может зашифровать свои файлы и хранить их в защищенном виде. При необходимости использования этих файлов пользователь должен расшифровать их, провести с ними нужные ему операции и снова их зашифровать. Неудобство данного способа шифрования состоит в том, что при каждом обращении к файлам пользователь сам должен расшифровывать, а затем зашифровывать их. Гораздо удобнее для него, если бы система сама делала данные операции.

Прозрачное шифрование, известное также как шифрование в режиме реального времени, является методом шифрования, при котором данные зашифровываются и расшифровываются без участия пользователя с помощью драйвера, работающего в фоновом режиме и следящего за всеми обращениями к данным [1]. Основной целью этого метода шифрования является защита от атак, направленных на получение данных в обход операционной системы, т. е. путем загрузки через другую ОС или использования средства прямого доступа к жесткому диску. Среди преимуществ прозрачного шифрования, помимо преимуществ, присущих всем остальным видам шифрования файлов, можно выделить следующие:

а) простота использования — пользователю достаточно один раз отметить для файла использование шифрования, и при всех дальнейших обращениях к файлу со стороны пользователя он будет расшифровываться автоматически;

б) данные расшифровываются в оперативной памяти, и не требуется сохранения расшифрованных данных на диск.

Существует два вида прозрачного шифрования: шифрование разделов (самый простой пример — шифрование всего диска) и шифрование на уровне файлов, которое, как правило, обеспечивается за счет надстройки над файловой системой, поддерживаемой операционной системой. Сравнение их характеристик представлено в таблице 1.

Таблица 1. Сравнение шифрования разделов и шифрования на уровне файлов

Шифрование разделов	**Шифрование на уровне файлов**
Простота реализации	Высокий уровень сложности: необходимо изучение поддерживаемой файловой системы
Шифрование всех файлов внутри раздела, независимо от их уровня конфиденциальности	Избирательный метод шифрования — только выбранные файлы
Не предназначено для шифрования по сети	Может использоваться для шифрования файлов в сети на стороне сервера
Полностью шифрует раздел с именами файлов, папок, метаданными	Шифрует исключительно данные внутри файла
Возможность скрытия факта, что раздел находится в зашифрованном виде	Наличие зашифрованных данных очевидно для наблюдателя

Работает на любой файловой системе	Работает на определенном наборе файловых систем

Как правило, в средствах прозрачного шифрования используется технология цифрового конверта: данные шифруются симметричным алгоритмом, а соответствующий им симметричный ключ — асимметричным, с помощью открытого ключа пользователя. Принцип работы средства прозрачного шифрования выглядит следующим образом [2]. Пусть пользователь работает с некоторым приложением и через это приложение пытается открыть какой-нибудь файл. Тогда приложение отправляет запрос файловой системе на открытие файла *(в ОС Windows — API вызов CreateFile)*. Файловая система по атрибутам файла определяет, зашифрован он или нет. Если зашифрован, то файловая система через службу операционной системы *(в ОС Windows — lsass.exe)* отправляет запрос на расшифровку симметричного ключа агенту, который может взаимодействовать с пользователем, например, он может запросить токен с закрытым ключом пользователя или ввод пароля. Этот агент расшифровывает симметричный ключ с помощью соответствующего закрытого ключа пользователя и через службу операционной системы отправляет его в ядро, где с его помощью будут расшифрованы уже сами данные.

На сегодняшний день существует большое разнообразие средств прозрачного шифрования файлов на диске. Практически все они используют иностранные криптографические алгоритмы. Согласно перечню средств защиты информации, сертифицированных ФСБ России, по состоянию на 1 октября 2011 г. [3], сертифицированные средства прозрачного шифрования существуют только для операционной системы Windows (например, КриптоПро CSP 3.6.1 [4]). Для Linux-систем таких средств нет. Для реализации сертифицированного средства прозрачного шифрования было необходимо изучить существующие средства для ОС Linux. Наиболее известные средства и их сравнение представлены в таблице 2. В силу большого размера таблицы пронумеруем ее столбцы и напишем, что означает каждый столбец:

- столбец 1: свободное программное обеспечение;
- столбец 2: доступность исходного кода;
- столбец 3: скрытые тома;
- столбец 4: тома, не имеющие идентификаторов;
- столбец 5: изменение размера тома;
- столбец 6: поддержка токенов, ключей, смарткарт;
- столбец 7: возможность доступа только для чтения;
- столбец 8: шифрование файла подкачки.

Таблица 2. Сравнение средств прозрачного шифрования в ОС Linux

Издатель	Название	Тип ПШ	1	2	3	4	5	6	7	8
Jetico	BestCrypt	Раздел	-	+	+			-	-	
RiTech International Ltd	BioSlim Disk	Раздел	-	-		-		-	-	
RiTech International Ltd	BioSlim Disk Siganture	Раздел	-	-		-		-	-	
darkstar@frog.org	CFS	Файл	+	+	-			-	-	

Matt Blaze/ Tt&T	CFS	Файл	+	+	-			-	-	
Встроенные	Cryptoloop	Раздел	+	+	-	+		-	-	+
Встроенные	Dm-Crypt	Раздел	+	+	-	+		-		+
V. Gough	EncFS	Файл	+	+	-	-		-		
C. Fruhwirth/ M. Broz	LUKS	Раздел	+	+	-	-		-		
Check Point Software	Pointsec PC	Раздел	-	-				+		
J. Assange,R. P. Weinman, S. Dreyfus	Rubber hose	Раздел	+	+	+	+		-		
Juettner, Beinert	Scramdisk For Linux	Раздел	+	+		+		-		
Matthias Withopf	SecurStick	Раздел	+	-	-		-	-		
Marco Alporone	TCFS	Файл	+	+				-		
TrueCrypt Foundation	TrueCrypt	Раздел	+	+	+	+	-	+		
Protengrity Corporation	VPDisk	Раздел	-	-			-			

Для операционной системы Linux на сегодняшний день существует около 20 решений. Однако ни одно из них не является сертифицированным в России. Начиная со сборки ядра 2.6.19 для осуществления технологии прозрачного шифрования используется шифрующая файловая система eCryptFS. Наиболее перспективным представляется внедрение российских криптографических алгоритмов в уже существующее и работающее средство, каким является eCryptFS. Аналогичным образом было спроектировано средство прозрачного шифрования КриптоПро EFS для ОС Windows [5], которое по сути является надстройкой над файловой системой EFS с набором дополнительно поддерживаемых функций, таких как контроль целостности информации.

Архитектура eCryptFS представлена на рис. 1 [6]. Пользователь с помощью интерфейса среды рабочего стола, например KDE или GNOME, обращается через виртуальную файловую систему к файловой системе eCryptFS, в которой находится нужный пользователю файл. Файловая система eCryptFS берет криптографические данные этого файла, которые хранятся так же как в PGP-зашифрованных файлах [7], и через хранилище ключей в пространстве ядра отправляет их в службу eCryptFS, работающей в режиме пользователя. Эти метаданные могут быть расшифрованы с помощью одного из PKI-модулей, в зависимости от установленной политики. Связь между службой eCryptFS и PKI-модулем обеспечивается за счет PKI API, установленных стандартом PKCS#11 [8]. PKI-модуль перебирает все закрытые ключи пользователя, пока не найдет нужный. Если такого не существует, этот модуль выдаст сообщение, что файл не может быть расшифрован. Если пользователь обладает нужным закрытым ключом, то метаданные расшифровываются

в PKI-модуле и через службу eCryptFS отправляются в хранилище ключей в ядре. Затем файловая система eCryptFS, основываясь на метаданных, с помощью криптографического API ядра выбирает нужный алгоритм шифрования и расшифровывает данные в файле, после чего вызывающее приложение или пользователь может открыть и прочитать его. Служба eCryptFS, работающая в пространстве пользователя, обеспечивает работу PKI-модулей, согласно политике. Например, данная служба может обеспечить запрос пароля пользователя для расшифровки закрытого ключа, если это нужно.

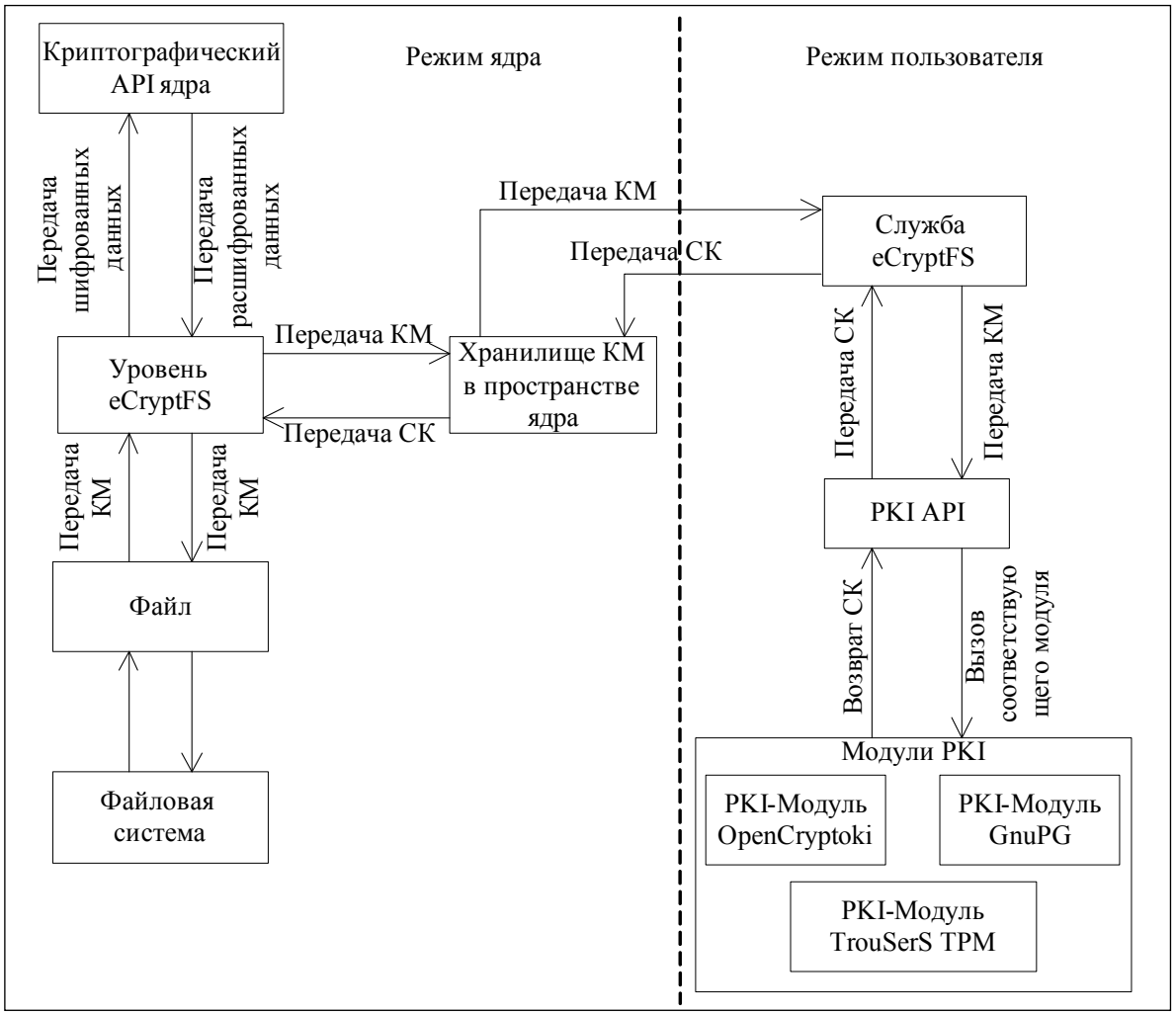

Рис. 1. Архитектура файловой системы eCryptFS

Рассмотрим более подробно составные части eCryptFS. Криптографический API ядра можно разделить на следующие составляющие [9]:

а) реализованные алгоритмы, т. е. те, для которых в ядре существует соответствующий исполняемый код;

б) интерфейс ядра — набор функций, служащих непосредственно для выполнения шифрования или расшифровки данных и выбора алгоритма и его характеристик;

в) интерфейс управления алгоритмами — набор функций, служащих для добавления алгоритмов в список уже реализованных или удаления оттуда.

Для реализации средства прозрачного шифрования на базе сертифицированного СКЗИ необходимо обеспечить работу алгоритма ГОСТ 28147-89 для шифрования и расшифрования

самих данных в файле. Для этого можно воспользоваться существующим сертифицированным СКЗИ и обеспечить его связь с eCryptFS.

PKI-модуль GnuPG использует пользовательский пароль для расшифровки закрытого ключа пользователя, который хранится в кольце ключей GnuPG и который необходим для расшифровки симметричного ключа файла.

PKI-модуль TriuSerS TPM используется для поддержки Trusted Platform Module. Это дает возможность использовать закрытый ключ, который вшит в аппаратное средство, например токен.

PKI-модуль OpenCryptoiki обеспечивает механизм работы с открытым ключом с помощью различных аппаратных устройств, например криптографического ускорителя.

Служба eCryptFS поддерживает возможность написания дополнительных модулей. Такие дополнительные модули могут взаимодействовать с существующими средствами PKI, которые используют стандарт сертификатов X.509.

Таким образом, для обеспечения работы с ключевым материалом на базе ГОСТ Р 34.10 2001 можно спроектировать дополнительный PKI-модуль на базе сертифицированного СКЗИ и обеспечить его связь со службой eCryptFS, работающей в режиме пользователя.

Для реализации средства прозрачного шифрования файлов на базе сертифицированного СКЗИ для ОС Linux целесообразно доработать штатное средство eCryptFS таким образом, чтобы файлы шифровались с помощью российского симметричного алгоритма в режиме ядра, а соответствующий симметричный ключ шифровался с помощью российского асимметричного алгоритма в режиме пользователя. Для этого необходимо обеспечить работу алгоритма ГОСТ 28147-89 в режиме ядра и работу алгоритма ГОСТ Р 34.10 2001 в режиме пользователя, причем обеспечивающий его работу PKI-модуль должен использовать стандарт сертификатов X.509. Это возможно осуществить с помощью сертифицированного СКЗИ, которое может работать как в режиме ядра, так и в режиме пользователя.

СПИСОК ЛИТЕРАТУРЫ:

1. *Шайдеманн Ф.* Прозрачное шифрование файлов // Журнал сетевых решений/LAN. 2005. № 15. С. 6.
2. *Иванов А., Смирнов П.* Соответствие требованиям российских нормативных документов по защите информации. Microsoft Платформа 2011. IS 304.
3. Перечень средств защиты, сертифицированных ФСБ России (по состоянию на 30 декабря 2011 г.). URL: http://clsz.fsb.ru/certification.htm. С. 97.
4. ЖТЯИ. 00050-02 30 01 Средство криптографической защиты информации «КриптоПро CSP». Формуляр. С. 18.
5. ЖТЯИ. 00051-01 30 01 Средство хранения конфиденциальной информации «КриптоПро EFS». Формуляр. С. 17.
6. *Halcrow M. A.* eCryptFS: An Enterprise-class Cryptographic Filesystem for Linux. URL: http://ecryptfs.sourceforge.net/ecryptfs.pdf. 2008. С. 18.
7. RFC 2440 OpenPGP Message Format. November 1998. С. 66.
8. PKCS #11 v2.30: Cryptographic Token Interface Standard. 16 April 2009. С. 7.
9. *Morris J.* The Linux kernel cryptographic API // Linux Journal. 2003. April 01. С. 10.