УДК 004.75

ЛУЦКИЙ Г.М.,
ВОЛОКИТА А.Н.,
ИВАНОВ Д.Г.

МЕТОД ОРГАНИЗАЦИИ ЗАЩИЩЕННЫХ ФАЙЛОВЫХ СИСТЕМ НА ОСНОВЕ КРИПТОГРАФИЧЕСКИХ ПРИМИТИВОВ КОНТРОЛЯ ДОСТУПА

Современные организации все больше ориентированы на облачные вычисления, одним из направлений которых являются удаленные системы хранения данных. Среди ключевых вопросов использования этого сервиса можно выделить обеспечение безопасности данных организации. В данной статье предложен метод организации защищенных файловых систем, отличающийся совместным использованием симметричного шифрования и внутренней схемы управления ключами, поддерживающий расширенную модель контроля доступа, и обеспечивающий более высокую производительность при необходимой степени безопасности.

Modern organizations are increasingly focused on cloud computing, one of the commonly used service of cloud computing is remote storage. Among the key issues surrounding the use of this service is provide security of organization's data. In this paper we present the method of secured file system organization based on cryptographic primitives for access control supporting enhanced access control model, using the internal key management and provides the necessary security and productivity.

Введение

В последнее время все более популярным становится использование удаленных систем хранения данных (СХД), в которых организации используют услуги внешнего поставщика сервиса хранения данных (Storage Service Provider, SSP). На данный момент основными мировыми компаниями по предоставлению данного типа услуг являются: Amazon, BC Grid, Arsenal Digital и Iron Mountain Digital.

Использование удаленных СХД для организации имеет следующие преимущества: экономия затрат на обслуживание СХД, простота управления, повышение гибкости и отказоустойчивости.

Однако широкому распространению данного сервиса препятствует целый ряд технических проблем, основной из которых является проблема обеспечения безопасности удаленных СХД. Так как, во-первых, данные находятся на удаленной системе, доступ к которой осуществляется через глобальную сеть, по умолчанию являющейся ненадежной средой обмена данными, а во-вторых, система хранения данных находится под управлением сторонней компании, от которой информация также должна быть защищена.

Реализация задач безопасности для удаленных СХД выполняется с помощью шифрования данных организации. Шифрование данных может быть реализовано на двух уровнях: уровне приложений и на уровне файловой системы. Если говорить о реализации на уровне приложений, то среди преимуществ можно выделить селективность шифрованных данных, а если говорить о реализации на уровне файловой системы, то однообразность работы и более высокий уровень безопасности.

На данный момент преобладают решения на уровне файловых систем, которые называются криптографическими файловыми системами (КФС) [1] – файловые системы с встроенными функциями криптографических преобразований.

В данной статье предложен метод организации криптографических файловых систем, который позволит разграничить права доступа и обеспечить целостность данных в удаленной СХД за счет использования криптографических примитивов контроля доступа.

Модели организации взаимодействия с удаленными системами хранения данных

Модели организации взаимодействия с удаленными системами хранения данных отличаются путями передачи данных и метаданных, централизованным или децентрализованным доступом, а также сложностью управления ключами. Под метаданными будем понимать структурированные данные, представляющие собой характеристики (атрибуты) описываемых сущ-

ностей для целей их идентификации, поиска, оценки и управления ими [2]. Существует три модели организации взаимодействия с удаленным СХД: централизованная, децентрализованная и модель с централизованным доступом к метаданным и децентрализованным доступом к данным. Будут рассмотрены особенности централизованной и децентрализованной моделей.

Централизованная модель доступа к СХД

В централизованной модели используется единая точка доступа для записи и чтения данных, хранящихся в уделенной СХД. Как показано на рис. 1, узлы организации используют защищенную файловую систему для шифрования/дешифрования данных, хранящихся на стороне SSP. Доступ к защищенной файловой системе осуществляется с помощью шлюза (файл-сервера), который обслуживает запросы от пользователей, находящихся на других узлах организации.

Для перехода к использованию удаленной СХД, согласно централизованной модели, организации необходимо настроить защищенную файловую систему, которая будет шифровать все данные при их записи на удаленную СХД и расшифровывать при чтении. При этом для пользователей доступ к данным является прозрачным. Обычно используются протоколы сетевых файловых систем Network File System (NFS) и Common Internet File System (CIFS). В этой модели конфиденциальность данных обеспечивается за счет использования простых схем управления ключами: данные могут быть зашифрованы единственным ключом, доступным только защищенной файловой системе шлюза. Кроме того, система контроля доступа функционирует на стороне организации, что позволяет использовать существующую локальную базу пользователей.

Основным недостатком данной модели является наличие шлюза доступа к данным, так как может привести к перегрузке канала передачи данных при больших рабочих нагрузках, не обладает достаточной масштабируемостью, кроме того, шлюз становится местом, при отказе которого прекращается работа всей системы.

Таким образом, использование централизованной модели является целесообразным при доступе к корпоративным данным из одного узла организации. Централизация и дополнительные расходы на криптографические операции снижают её эффективность при множественном доступе к данным.

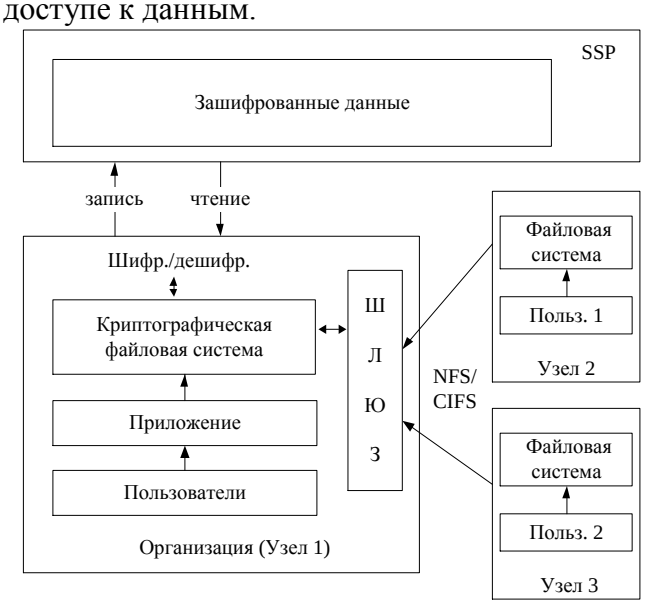

Рис. 1. Централизованная модель доступа к СХД

Децентрализованная модель доступа к СХД

В децентрализованной модели взаимодействия с удаленной СХД пользователи при доступе к данным или метаданным обращаются непосредственно с SSP (рис. 2).

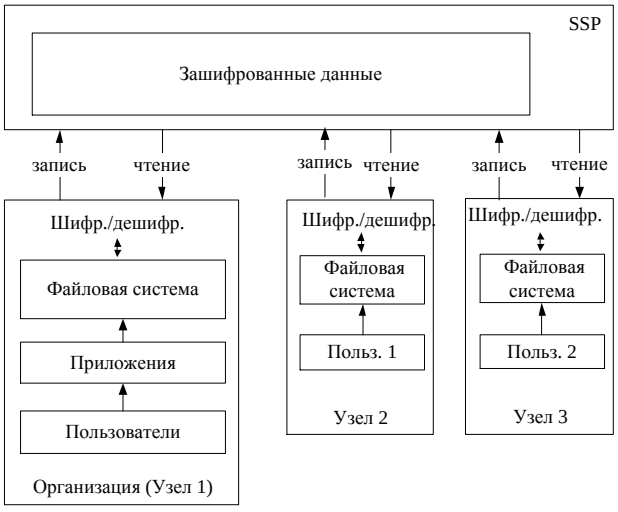

Рис. 2. Децентрализованная модель доступа к СХД

Использование данной модели исключает все узкие места, связанные с каналами передачи данных и метаданных, т.к. каждый пользователь имеет непосредственный доступ к SSP. Преимуществом децентрализованной модели является отсутствие необходимости управления шлюза доступа к данным, недостатком – проблема управления ключами, так как каждый

файл необходимо зашифровывать новым ключом.

Проведенный анализ моделей взаимодействия показывает, что использование централизованной модели в StaaS [3] нецелесообразно, т.к. снижается производи-тельность и увеличиваются расходы. Таким образом, в статье будут рассмотрены методы обеспечения безопасности удаленных СХД для децентрализованной модели организации взаимодействия.

Современные методы организации защищенных файловых систем

Рассмотрим основные методы организации КФС в зависимости от типов алгоритмов, используемых для шифрования данных и метаданных [4,5,6,7,8,9]:

1. Метод организации КФС без шифрования метаданных.

Идея метода заключается в том, что не выполняется шифрование метаданных, а сами данные шифруются с помощью симметричных алгоритмов шифрования. Среди реализаций данного метода можно выделить криптографические файловые системы CFS, CryptFS и NCryptFS.

2. Метод организации КФС с шифрованием метаданных. Метаданные шифруются с использованием асимметричных алгоритмов, а данные – симметричных алгоритмов. Криптографические файловые системы, получившие наибольшее распространение: SiRiUS и SNAD.

3. Метод организации КФС с шифрованием ключа метаданных. Отличительной чертой данного метода является шифрование данных и метаданных симметричными алгоритмами, а ключ метаданных шифруется ассиметрично.

В табл. 1 представлен сравнительный анализ методов и средств обеспечения информационной безопасности удаленных систем хранения данных.

Среди реализаций данного метода можно выделить криптографическую файловую систему Plutus.

Как видно из таблицы, ни одно из существующих средств не объединяет в себе безопасность и производительность: CFS, CryptFS и NCryptFS не поддерживают шифрование метаданных, SiRiUS и SNAD требуется большие временные затраты для работы с зашифрованными данными, так как используются медленные алгоритмы RSA, а Plutus предполагает наличие отдельно реализованного сервиса управления ключами, что повлияет на стоимость управления системы и повысит затраты време-

Табл. 1. Сравнительная характеристика современных методов и средств защиты информации в удаленных СХД

Характеристики	**Без шифрования метаданных**			**С шифрованием метаданных**		**С шифрованием ключа метаданных**
	CFS	**Cryptfs**	**NCryptfs**	**SiRiUS**	**SNAD**	**Plutus**
Обеспечение целостности	-	-	-	+	+	+
Внутренняя схема управления ключами	-	-	-	+	+	-
Проверка времени использования ключей	+	-	+	-	-	-
Алгоритм шифрования метаданных	-	-	-	RSA	RSA	3DES CBC/CTS
Длина ключа шифрования метаданных	-	-	-	2048	1024	168
Алгоритм шифрования данных	DES OFB+ ECB	Blowfish CBC	Blowfish CFB	AES CBC	Blowfish CFB	3DES CBC/CTS
Длина ключа шифрования данных, бит	56	128	128	128	128	168
Размер шифруемых блоков данных	8B	4KB/8KB	4KB/16KB	4KB/16KB	4KB/16KB	4KB
Поддержка групп пользователей	-	-	+	-	+	+
Поддержка разделения прав пользователей	-	-	+	+	+	+
Поддержка разделённого доступа к файлам	-	-	+	+	+	+
Метод реализации	На основе NFS	Стековая ФС	Стековая ФС	На основе NFS	Полноценная ФС	Полноценная ФС

ни на выполнения операции работы с данными.

Поэтому необходима разработка нового метода организации КФС с высоким уровнем безопасности и производительности, а также внутренней схемой управления ключами.

Метод организации защищенных файловой системы на основе криптографических примитивов контроля доступа

Предложен метод организации КФС на основе криптографических примитивов контроля доступа, максимально использующий преимущества симметричного шифрования, одним из которых является отсутствие временных затрат. Данный метод организации защищенных файловых систем, отличающийся совместным использованием симметричного шифрования и внутренней схемы управления ключами, поддерживающий расширенную модель контроля доступа, и обеспечивающий более высокую производительность при необходимой степени безопасности.

Модель контроля доступа метода является дискреционной, в которой каждый объект (файл, каталог или ссылка) имеет связанного с ним владельца, контролирующего к нему доступ. При этом доступ может быть предоставлен на основе субъектов трех типов: владелец (owner), группа-владелец (group) и все остальные (other).

Каждый пользователь для идентификации имеет пару ключей (открытый/секретный), которая обозначается <Bu, Pu>, где Bu является открытым ключом, и Pu – секретным ключом, который известен только пользователю.У каждой группы пользователей также есть аналогичная пара ключей <Bg, Pg>.

Пример структуры распределения ключей для группы пользователей student (<Bstud, Pstud>) показан в Табл. 2. Таблица группы состоит из двух частей: хеш-свертки идентификатора пользователя и зашифрованных ключей группы открытым ключом пользователя.

Зашифрованные ключи группы хранятся на стороне SSP. Когда пользователь выполняет вход в систему (то есть, монтирует файловую систему SSP), он получает зашифрованные ключи групп, к которым принадлежит, и использует свой секретный ключ для их расшифровки.

Табл.2. Таблица группы пользователей student

Пользователь	Ключи группы
Хэш-свертка (bob)	E_{Bbob}\{<Bstud, Pstud>\}
Хэш-свертка (alice)	E_{Balice}\{<Bstud, Pstud>\}

Для эффективной обработки файлы большого размера, как правило, разделяются на несколько блоков данных, которые шифруется отдельно. Это позволяет выполнять обновление данных более эффективно, т.к. нет необходимости повторно шифровать весь файл после внесения изменений в одну из его частей. В дальнейшем, для простоты описания, предполагается, что файлы помещаются в один блок данных.

Каждый блок данных имеет набор связанных с ним ключей:

1. Ключ шифрования данных, DEK (Data Encryption Key): DEK является уникальным симметричным ключом шифрования, используемым для шифрования блоков данных.

2. Ключ подписи данных, DSK (Data Signing Key) и ключ проверки данных, DVK (Data Verification Key): DSK и DVK являются парой асимметричных ключей, так что любые подписанные с использованием DSK данные, могут быть проверены только с DVK и, наоборот.

Аналогично, блоки метаданных также имеют связанные с ними ключи:

1. Ключ шифрования метаданных, MEK (Metadata Encryption Key): MEK является уникальным симметричным ключом, используемым для шифрования блоков метаданных.

2. Ключ подписи метаданных, MSK (Metadata Signing Key): MSK доступен только для пользователей, которые могут выполнить операцию записи в блоке метаданных.

3. Ключ проверки метаданных, MVK (Metadata Verification Key): MVK используется для проверки целостности блока метаданных.

Все типы ключей показаны в Табл. 3.

Табл. 3. Типы ключей доступа

Субъект	Обозначение
Пользователь	Bu
	Pu
Группа пользователей	Bg
	Pg
Блок данных	DEK
	DSK
	DVK
Блок метаданных	MEK
	MSK
	MVK

Блоки данных шифруются симметричными алгоритмами с использованием заранее сгене-

рированных ключей, поэтому пользователю необходимы соответствующие ключи. Для этого в блок метаданных было добавлено три новых поля: DEK, DSK и DVK. Таким образом, метаданные не только указывают на блоки данных, но и предоставляют пользователю ключи для работы с ними.

Также дополнительно выполнены следующие модификации. Во-первых, добавлено поле, содержащее MSK, доступное исключительно владельцу файла (каталога) и позволяющее подписывать блок метаданных при его обновлении, например при изменении прав доступа. Во-вторых, из блока метаданных удалены поля, изменяющиеся при записи модифицированного файла (каталога).

Блок данных каталога содержит таблицу, которая состоит из двух колонок, содержащих номер I-узла [10] и имена файлов и подкаталогов, содержащихся в этом каталоге. В таблицу каталога были добавлены два новых столбца, содержащие MEK и MVK для метаданных файлов и подкаталогов. Таким образом, таблица каталога предоставляет информацию о расположении блока метаданных, а также ключи для расшифровки и проверки.

Доступ к полям метаданных и таблиц каталога, содержащие ключи, является избирательным. Такая избирательность необходима для обеспечения контроля доступа и поддерживается с помощью криптографических примитивов CAP (Cryptographic Access Control Primitives) [11].

CAP объекта файловой системы заменяет контроль доступа в удаленной системе хранения данных, селективно предоставляя ключи доступа в метаданных и таблице каталога. Например, для установки разрешения только на чтение, CAP файла содержит поля DEK и DVK, а ключ DSK – недоступен.

Варианты метаданных и записей таблицы каталога с использованием CAP представлены на рис 3. Слева приведены права доступа, в среднем столбце – поля метаданных, а в правом столбце – таблица каталога. Для поддержки «нулевого» права доступа, все поля метаданных недоступны.

Право доступа «только для чтения» позволяет просматривать содержимое каталога (команда ls), но не позволяет выполнять переход или изменения (команды cd, rm и т.д.). Поэтому, CAP содержит ключи DEK и DVK в метаданных и поле «имя» в таблице каталога. Право «чтение-запись» каталога имеет одинаковое значение с «только для чтения», так как право на запись невозможно использовать без права на выполнение.

Разрешение «чтение-выполнение» позволяет пользователю переходить в каталог и получать доступ к содержимому, но не допускает изменений. В конструкции CAP для этого разрешения в метаданных доступны ключи DEK и DVK. В таблице каталога все четыре поля являются доступными, поскольку с разрешением на выполнение пользователи имеют возможность перехода в каталог и получения доступа ко всем метаданным файлов и подкаталогов. Используя номер I-узла, пользователи могут получить метаданные, используя MEK расшифровать их, после чего проверить целостность с помощью MVK.

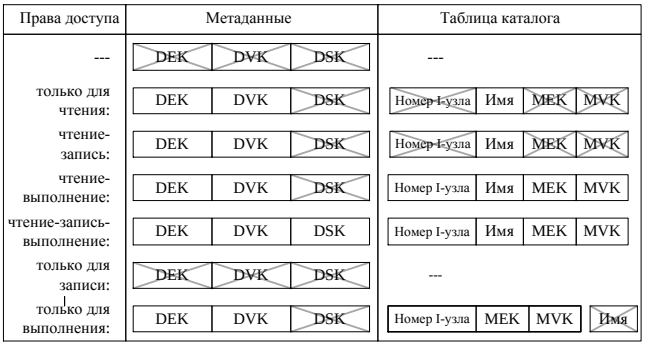

Рис. 3. Варианты CAP для каталогов

Право доступа «чтение-запись-выполнение» разрешает пользователям также изменять содержимое таблицы каталога, при этом поле с ключом DSK доступно.

Конструкция CAP для разрешения «только для записи» соответствует «нулевому» разрешению, так как такое разрешение не может работать без разрешенного права на «выполнение».

Наиболее интересна конструкция CAP разрешения «только для выполнения». Данное разрешение позволяет пользователям переходить в каталог и подкаталоги, однако пользователь не может просматривать их содержимое. Поскольку пользователи могут переходить в подкаталоги, то необходимо обеспечить доступ к таблице каталога, для этого DEK и DVK должны быть доступными, а поле «имя» недоступно. Это достигается за счет шифрования полей номера I-узла, MEK и MVK с помощью нового ключа, полученного из имени файла или подкаталога. Новый ключ генерируется следующим образом: с помощью ключевой хэш-функции такой как MD5 или SHA-1, где входным сообщением является имя, а ключом DEK. Таким образом, любой пользователь, знающий

имя файла или подкаталога, может сгенерировать ключ, а затем использовать его для расшифровки соответствующей строки в таблице каталога и получения доступ к метаданным.

Существует вариант прав доступа, который не поддерживается в описываемом методе – «запись-выполнение». Это связано с использованием симметричных ключей для шифрования блоков данных. Любой пользователь, который имеет право записи, может расшифровать блоки данных, используя тот же ключ и, следовательно, сможет прочитать содержимое. Тем не менее, такое разрешение очень редко встречается в реальных системах, и данный недостаток не является существенным.

Примеры CAP для всех вариантов прав доступа к файлам представлены на рис 4. В случае с файлами, блоки данных не используются в модели доступа, поэтому они опущены. При «нулевом» разрешении, все поля метаданных, содержащие ключи, недоступны. При разрешении «только для чтения» DEK и DVK доступны, что позволяет расшифровывать и проверять блоки данных. CAP для разрешения «чтение-запись» предоставляет также доступ к DSK. Разрешение «чтение-выполнение» предоставляет доступ к тем же ключам, как и «только для чтения», поскольку после расшифровки файла пользователь может выполнить его как программу. CAP «чтения-записи-выполнения» соответствует CAP «чтение-запись».

Права доступа	Метаданные		
---	~~DEK~~	~~DVK~~	~~DSK~~
только для чтения:	DEK	DVK	~~DSK~~
чтение-запись:	DEK	DVK	DSK
чтение-выполнение:	DEK	DVK	~~DSK~~
чтение-запись-выполнение:	DEK	DVK	DSK

Рис. 4. Варианты CAP для файлов

Как и в каталогах, в CAP файлов не поддерживается разрешение «только для записи». Кроме того, для удаленных систем хранения данных разрешение «только для выполнения» является недопустимым, так как при выполнении программы предварительно необходимо прочитать её файл.

При выполнении операции монтирования файловой системы пользователю необходимо считать суперблок, содержащий описание базовой структуры и атрибутов файловой системы, и номер I-узла корневого каталога. В данном случае также с номером I-узла хранятся MEK и MVK корневого каталога, что позволяет выполнить расшифровку его блока метаданных. Для каждого зарегистрированного пользователя суперблок зашифрован с помощью открытого ключа и хранится на стороне SSP. Таким образом, когда пользователь монтирует файловую систему, он расшифровывает суперблок, используя свой секретный ключ, и получает доступ к метаданным корневого каталога.

В данном методе присутствует недостаток избыточного хранения метаданных объектов файловой системы, так как двум пользователям с различными правами доступа необходимо предоставить два варианта CAP. Для решения этой задачи была разработана схема дифференциации CAP, предусматривающая использование для каждого файла трёх типов CAP, определяющих права доступа владельца, группы-владельца и всех остальных соответственно.

Выводы

В данной статье предложен метод организации защищенных файловых систем на основе криптографических примитивов контроля доступа, отличающийся совместным использованием симметричного шифрования и внутренней схемы управления ключами, и обеспечивающий более высокую производительность при необходимой степени безопасности. Данный метод имеет ряд существенных преимуществ:

1. Поддержка расширенной модели контроля доступа, позволяющая более точно описывать права доступа к объектам файловой системы.

2. Использование внутреннего управле-ния ключами. В предложенном методе все ключи находятся внутри самой файловой системы и хранятся в зашифрованном виде на стороне SSP.

3. Совмещение уровня безопасности метода с шифрованием метаданных и уровня производительности метода без шифрования метаданных.

Одним из существенных недостатков данного метода является сложная структура метаданных объектов файловой системы и таблиц каталогов, что является предметом дальнейших исследований в данной области.

Список литературы

1. Muller, R . "How IT Works: Encrypting File System". TechNet Magazine. Microsoft. 2006-05
2. Воройский Ф.С. Информатика. Новый систематизированный словарь-справочник (Вводный курс по информатике и вычислительной технике в терминах). – 2-е изд., перераб. и доп.. – М.: Издательство Либерия, 2001. – С. 536.
3. Roettgers, Janko. "Piracy Beyond P2P: One-Click Hosters", Retrieved: 5 January 2008.
4. Blaze, M., “A Cryptographic File System for Unix”, in Proceeding of the ACM Conference on Computer and Communications Security, 1993
5. Zadok, Erez, Ion Badulescu and Alex Shender, "Cryptfs: A Stackable Vnode Level Encryption File System", CUCS-021-98.
6. Wright, C., Martino, M., and Zdok, E., “Ncryptfs: A secure and Convenient Cryptographic File System.” In Proceedings of the USENIX AnnualTechnical Conference, 2003
7. Goh, E., Shacham, H., Modadugu, N., and Dohen,D., “SiRiUS: securing reote untrusted storage,” in Proceedings of the Network and Distributed System Security Symposium (NDSS), 2003
8. Ethan Miller “Strong Security for Distributed File Systems”, 20th International Performance, Computing and Communications Conference, April 2001
9. Kallahalla M., Reidel, E.,Swaminathan, R., Wang, Q., and Fu, K., “Plutus: Scalable secure file sharing on untrusted storage,” in Proceedings of the USENIX Conference on File and Storgae Technologies (FAST), 2003
10. Робачевский А. Н., Немнюгин С. А., Стесик О. Л. Индексные дескрипторы / Базовая файловая система System V / Глава 4. Файловая система // Операционная система UNIX. – 2-е изд. – СПб.: БХВ-Петербург, 2008. – С. 334-. – 656 с.
11. Singh, A., and Liu, L., “Cryptographic Access Primitives in cryptographic file systems,” Georgia Tech CERCS Technical Report GIT-CERCS-07-08, 2008.