УДК 004.056.53 DOI: 10.17587/it.27.259-266

Л. В. Аршинский, д-р техн. наук, проф., e-mail: larsh@mail.ru,
Г. Н. Шурховецкий, аспирант, e-mail: gshn5@yandex.ru,
Иркутский государственный университет путей сообщения

Особенности применения метода рассечения—разнесения для безопасного хранения данных во внешних хранилищах

Рассматриваются вопросы применения метода рассечения—разнесения для защищенного хранения информации во внешних, в первую очередь облачных, хранилищах данных. Анализируются различные подходы к реализации метода, включая патентный материал. Показывается, что метод наиболее эффективен при побитовом рассечении файлов и случайном размещении битов в потоках данных, направляемых в отдельные хранилища.

Ключевые слова: *защита информации, хранилища данных, облачные технологии, метод рассечения—разнесения*

Введение

Безопасность хранения данных во внешних, в первую очередь облачных, хранилищах данных — одно из активно обсуждаемых сегодня направлений информационной безопасности (см., например, работы [1—9]). При этом одной из основных проблем применения облачных технологий для хранения данных является доверие к поставщикам этих услуг [2—4]. Согласно работе [5] несанкционированный доступ персонала и кража конфиденциальной информации являются весьма значимой угрозой при облачном хранении. К сожалению, сегодня ни один поставщик не гарантирует в полной мере защиту клиентских данных от постороннего вмешательства. Угроза вполне реальная, поскольку на некоторых ресурсах не обеспечивается даже шифрование [1, 4]. Если злоумышленнику удастся обойти систему защиты, он получит возможность работать со сведениями, которые клиент предпочел бы сохранить в тайне. Причем шифрование на стороне поставщика также не освобождает от рисков доступа посторонних к клиентским данным, например, в силу недобросовестности персонала, договоренностей с третьими лицами и т.п. Это существенно снижает интерес к таким технологиям со стороны потенциальных клиентов [6, 7].

Несмотря на комплексность проблемы [7—9], одним из наиболее распространенных элементов защиты сегодня является шифрование [3—9]. Надежные алгоритмы плюс не скомпрометировавший себя ключ существенно повышают защищенность данных, хотя по-прежнему остаются сомнения в их полной безопасности в силу, например, неудачного выбора ключа, его случайной компрометации, недокументированных проблем в алгоритмах и т. п. Основная уязвимость здесь — хранение всего объема информации в одном месте, хотя и в защищенном виде. В связи с этим возникает вопрос поиска иных, в том числе некриптографических, способов защиты, которые снизят риск внешнего доступа к важной для клиента информации.

Среди некриптографических методов наибольший интерес вызывает процедура рассечения исходной информации на части с раздельной обработкой частей. В телекоммуникации и стеганографии известен метод пространственно-временного распыления, заключающийся в дроблении исходного сообщения на возможно мелкие составляющие (предложения, слова, символы, блоки символов, группы байтов, байты, группы битов, биты) с последующей их передачей частями, распределенными по нескольким каналам связи (см., например, работы [10—12]). Также известен метод рас-

сечения—разнесения, также представляющий собой метод дробления (рассечения) исходной информации на части с последующей передачей частей по разным каналам связи или размещением в разных хранилищах (см., например, работы [13—15]). В отличие от передачи информации как достаточно кратковременного процесса при ее помещении в хранилище у "заинтересованных лиц" появляется, условно говоря, неограниченное время доступа к ней, что расширяет возможности атакующего. В связи с этим возникает проблема анализа особенностей подобного хранения с точки зрения информационной безопасности.

1. Описание метода

Метод пространственно-временного распыления основан на защите передаваемых данных путем их дробления на блоки, которые затем направляются от источника A к получателю B в определенные моменты времени [12]. Блоки могут помещаться в контейнеры, среди которых могут быть и пустые (ложные). Этот прием, в частности, используется в стеганографии [10]. Пространственно-временное распыление — хороший способ защиты при передаче данных, однако характер информации, помещаемой в хранилища, накладывает свои требования, в частности, большой объем и недопустимость частой замены (отсутствует "временная" часть), продолжительное хранение информации, размещение ее "открытым" образом и т. д. Как результат, приходится ограничиваться пространственным"распылением" в виде рассечения файлов на отдельные фрагменты и разнесения их по различным, в том числе географически-удаленным, местам (хранилищам). На этом основан известный метод рассечения—разнесения. Идея метода сама по себе проста. Информация делится на части, которые помещаются в разные, в том числе географически удаленные, места хранения так, что владение одним куском не дает представление об информации в целом. Это снижает ущерб от постороннего вмешательства. Наличие сегодня большого числа сервисов облачного хранения дает такую возможность, и ею следует пользоваться. Поскольку принцип рассечения и места хранения частей в идеале знает только владелец информации, он может быть относительно спокоен за нее, особенно, если она не имеет явного текстового или иного подобного характера, позволяющего воспользоваться сведениями, хранящимися в одной отдельной части. Примером могут быть фото- и видеоданные, звуковые файлы и т. п. Но даже текстовая информация может быть достаточно надежно сохранена при надлежащем использовании метода. Рассмотрим его возможности при двух основных подходах к рассечению—разнесению:

1. Рассечение "монолитное", когда информация делится на равные или не равные части с разнесением в различные, в том числе географически удаленные, хранилища так, что каждая часть представляет собой слитный — единый и непрерывный — фрагмент исходной информации.

2. Рассечение "диффузное", когда отдельная часть (назовем ее потоком) содержит фрагменты различных участков исходного материала, так что при завладении одним или даже частью потоков смысл исходного сообщения оказывается трудноустановим.

Проанализируем возможности такого метода при разных способах реализации.

2. Анализ особенностей

Рассмотрим *первый подход.*

Самый простой случай — это размещение исходной информации в единственном потоке с направлением ее в хранилище. По сути это означает ее незащищенное хранение и упомянуто для полноты изложения.

О защищенности можно говорить, если потоков более одного. Тогда атакующий владеет лишь частью информации и оказывается перед задачей восстановления всего содержимого по имеющемуся фрагменту. Здесь возможны два варианта.

I. Для работы с файлом необходимы все фрагменты. В этом случае обладание единичным фрагментом ничего не дает. Такая реализация метода обеспечит надежную защиту, во всяком случае в той степени, в какой информация защищена отсутствием других фрагментов. Однако далеко не все пользовательские файлы обладают такой возможностью.

II. Имеющийся фрагмент позволяет выявить хранящуюся в нем информацию. Например, владея куском текста, мы воспринимаем его, хотя недостающие части придется восстанавливать. Последнее можно попытаться сделать или подбором, или додумывая содержимое по полученному фрагменту. И то и другое достаточно проблематично, но тем не менее, частью информации при таком подходе атакующий владеет. Если же он владеет всеми фрагментами, то восстановление информации потребует $S!$ комбинаций потоков, где S — их

число. При небольшом числе потоков (реальный сценарий) это небольшая величина. Иными словами, владение всеми потоками делает такую реализацию метода уязвимой.

При *втором подходе* исходная информация делится на части существенно меньше длины потока. Части складываются в потоки внешне случайным, неизвестным атакующему образом. О ключе — порядке формирования потоков — осведомлен только владелец информации, и только он, в идеале, способен восстановить исходный файл из потоков. Алгоритм в общем случае может выглядеть следующим образом:

1. Задать закон рассечения—разнесения, для чего:

1.1. Определить характер фрагментации исходного сообщения, разбив его на фрагменты, существенно меньшие, чем длина будущих потоков, например, на слова, символы, группы символов определенной длины и т. п.

1.2. Задать ключ рассечения—разнесения в виде набора $P = \{(p_1,a_1), ..., (p_l,a_l), ..., (p_L,a_L)\}$, где p_l — идентификаторы (к примеру, номера) потоков, куда будут направлены соответствующие фрагменты (идентификаторы могут повторяться), a_l — относительный (в пределах действия ключа) адрес l-го фрагмента в соответствующем потоке (фрагмент помещается в позицию с номером a_l), L — длина ключа. К примеру, для $P = \{..., (p^*,2), ..., (p^*,1), ...\}$ к потоку p^* сначала будет присоединен фрагмент, помеченный $(p^*,1)$, а затем фрагмент $(p^*,2)$. Таким образом, произойдет изменение порядка следования (перемешивание) фрагментов в потоке. Порядок, в котором нет перемешивания, рассматриваем как "нормальный". В данном примере это $P = \{..., (p^*,1), ..., (p^*,2),...\}$. Если порядок нормальный, адрес a_l можно не указывать и записывать ключ как $P = \{p_1, p_2,..., p_L\}$. В этом случае очередной фрагмент присоединяется в конец соответствующего потока.

2. Поставить в соответствие каждому фрагменту двойку (p_l, a_l) или идентификатор p_l, если порядок нормальный. Очевидный способ — последовательное "наложение" ключа на исходное сообщение.

3. Сформировать потоки согласно ключу. Например, для трех потоков, ключей {1, 2, 3} и фрагментов-слов текст "Однажды, в студеную зимнюю пору, я из лесу вышел; был сильный мороз." разобьется следующим образом:

3.1. Однажды, зимнюю из был

3.2. в пору, лесу сильный

3.3. студеную я вышел; мороз.

4. Поместить потоки в соответствующие места хранения.

5. Стоп!

Сборка исходного сообщения выполняется обратной процедурой:

1. Собрать потоки из мест хранения.

2. Пользуясь законом рассечения—разнесения:

1.1. Выполнить последовательное считывание фрагментов из потоков.

1.2. Собрать фрагменты в исходное сообщение.

3. Стоп!

Видно, что в принципе процедура легко выполнима и обеспечивает более высокую степень защиты. Здесь каждый поток содержит "произвольные" фрагменты сообщения, в результате чего сложнее установить его смысл по отдельному потоку. Хотя, если фрагментами являются слова или достаточно большие группы символов, а информация текстовая, можно предположить, чему она посвящена. Однако для некоторых файлов, например, содержащих фото-, видео-, аудиоданные и файлов программ (это типичные случаи [4]) этот прием может быть вполне подходящим.

Если у атакующего в распоряжении имеется только часть потоков, ему необходимо:

1. Подобрать закон рассечения—разнесения, для чего:

1.1. Предположить, сколько потоков было всего (гипотеза 1).

1.2. Предположить, как выполнялось рассечение на фрагменты (гипотеза 2).

1.3. Предположить, какое место занимали фрагменты имеющихся потоков в исходном информационном массиве (гипотеза 3).

2. Последовательно разместить имеющиеся фрагменты согласно гипотезам 1—3 в едином сообщении, оставляя пропуски для отсутствующих фрагментов.

3. Заполнить пропуски так, чтобы получилось осмысленное сообщение (гипотеза 4).

Число вариантов здесь может быть крайне велико. В частности, если $P(L,S)$ — ключ рассечения для S потоков, а $\boldsymbol{P(L,S)}$ — множество таких ключей, то мощность множества равна числу возможных цепочек $\{(p_1,l_1),(p_2,l_2),...,(p_L,l_L)\}$:

$$|\boldsymbol{P}(\boldsymbol{L},\boldsymbol{S})| = \frac{(L-1)!}{(S-1)!(L-S)!}L!. \qquad (1)$$

Здесь $\frac{(L-1)!}{(S-1)!(L-S)!} = C_{L-1}^{S-1}$ — число композиций длиной S числа L, т.е. наборов $\{n_1, ..., n_s, ..., n_S\}_j$ таких, что $n_1 + ... + n_s + ... + n_S = L$, где s — номер потока, j — номер композиции; $L!$ — число перестановок пар (p_l,a_l) в ключе.

Несложно заметить, что мощность $|\boldsymbol{P(L,S)}|$ растет с увеличением L. Наибольших значений она достигает при $S \approx \frac{L}{2}$, однако при большой длине ключа это может создать проблемы с размещением. Того же результата можно добиться простым увеличением L при малом числе потоков S.

Для нормального порядка вместо соотношения (1) работает формула

$$|P(L,S)| = \sum_{j=1}^{C_{L-1}^{S-1}} \frac{L!}{(n_1!\cdot\ldots\cdot n_s!\cdot\ldots\cdot n_S!)_j}. \qquad (2)$$

Так как $n_1 + ... + n_s + ... + n_S = L$, выражение под суммой можно оценить как $q(S,L)^L$, где $q(L,S) > 1$. Это позволяет представить формулу (2) как

$$|\boldsymbol{P(L,S)}| = \frac{(L-1)!}{(S-1)!(L-S)!} q(L,S)^L. \qquad (3)$$

Функция (3) растет существенно медленнее, чем (1), однако является функцией показательного роста. В частности, при $S = 2$ имеем:

$$|\boldsymbol{P(L,S)}| = 2^L - 2.$$

Наконец, частным случаем (1) и (2) при $S = L$ будет

$$|\boldsymbol{P(L,S)}| = S!. \qquad (4)$$

Последнее означает, что самый плохой вариант использования метода рассечения—разнесения — совпадение длины ключа с числом потоков.

В целом из соотношений (1)—(4) следует, что:

I. Защищенность в большей мере зависит от длины ключа, чем от числа потоков.

II. Длина ключа должна превышать число потоков: $L > S$, причем высокая степень защищенности (число подбираемых вариантов закона рассечения) может быть достигнута уже одним увеличением L. Увеличение числа потоков S также повышает защищенность, но это усложняет процедуру размещения информации в хранилищах и зависит от внешних условий, тогда как L определяется владельцем информации. Хотя рассечение—разнесение — это не криптологическая парадигма защиты, тем не менее *принцип максимизации длины ключа* сохраняется и здесь.

III. Метод рассечения—разнесения обеспечивает наилучшую защиту, когда закон с точки зрения атакующего выглядит *случайным*. В противном случае она обеспечивается только сложностью получения всех потоков.

IV. Для повышения защищенности рекомендуется соблюдать следующее условие. Если F — длина файла, а f — характерный размер фрагмента, отношение $C = \frac{F}{f}$, которое является своеобразным показателем или коэффициентом дробления, должно быть *максимально*:

$$C = \frac{F}{f} \to \max,$$

или в более слабом варианте, $f \ll F$. Действительно, с учетом того, что выполняется условие $L \leqslant C$, величина L может быть тем больше, чем больше C. Это особенно важно для случаев, когда защищаемые файлы невелики. Для файлов большого размера ключ P может применяться периодическим образом.

V. При должном подборе ключа метод обеспечивает защиту против угрозы "атакующий владеет всеми потоками".

В результате можно заключить:

1. *При достаточно большой длине ключа и его случайном характере обладание даже всеми потоками не позволит атакующему восстановить сообщение.* Отметим, что эффект достигается даже при небольшом числе потоков, например, равном 2 или 3.

2. Рост L обеспечивается ростом коэффициента дробления C.

Далее. Если атакующий владеет частью потоков и знает закон рассечения—разнесения, ему необходимо заполнить пропуски (отсутствующие фрагменты — гипотеза 4). Если файл текстовый, а фрагменты — символы, можно попробовать подбирать исходя из контекста частотности пар, троек и т. д. символов. Для нетекстовых файлов это проблематично, потому в общем случае считаем, что сложность подбора пропорциональна числу вариантов выбора. Его можно оценить величиной 2^E, где E число отсутствующих битов. Например, если у атакующего имеется в распоряжении два из трех потоков, потоки имеют равную длину, а размер исходного файла 1Мбайт (довольно типичная величина), число вариантов $\sim 2^{280000}$. Даже для маленького по сегодняшним меркам файла 1Кбайт это составит $\sim 2^{2730}$ вариантов. Более чем астрономическая величина! Это означает, что незнание всех потоков при отсутствии закона рассечения еще более (можно сказать драматически) ухудшает ситуацию, так как наравне с подбором ключа атакующий должен заполнить пропуски.

Если атакующий владеет всеми потоками, а ключ имеет длину, равную S, число проверочных комбинаций составляет вполне приемлемую величину порядка $S!$. Незнание длины ключа усложнит процедуру подбора в L раз:

$$|\boldsymbol{P}(\boldsymbol{L},\boldsymbol{S})| = L \cdot S!,$$

но это не принципиально. Как только получится осмысленная комбинация начальных фрагментов потоков, ее можно применить к оставшейся части данных и восстановить искомый файл. Таким образом, регулярность рассечения—разнесения в случае владения всеми потоками при $L = S$ делает метод уязвимым к атаке "владение всеми потоками". Защитой может служить только низкая вероятность или невозможность получения всех потоков. Повысить защищенность можно, меняя ключ вдоль всего исходного сообщения, а также изменяя длину фрагментов, но это приводит к усложнению процедуры и чрезмерному росту длины ключа.

В целом для "диффузного" варианта метода рассечения—разнесения защищенность D информации можно оценить величиной

$$D \sim \frac{2^E}{\varepsilon^S} \cdot |\, \boldsymbol{P}(\boldsymbol{L},\boldsymbol{S}) \,| - 1, \qquad (5)$$

где E — число неизвестных битов сообщения; ε — вероятность получения доступа к потоку.

Из вышесказанного следует, что наилучшим способом реализации метода служит рассечение на минимально возможные фрагменты при достаточно большой длине случайно сгенерированного ключа. Даже при ε, близком к 1, и небольшом числе потоков это надежно защищает данные. Наименьшим фрагментом информации является бит.

3. Побитовое рассечение

Рассечение—разнесение как способ защиты информации известен достаточно давно. Информацию в целях ее сокрытия можно не только шифровать, но и делить на части так, чтобы она нигде не была представлена полностью [16]. На этом строятся многие современные подходы к работе с закрытыми сведениями. Для всей полноты сведений фрагменты нужно получить и сложить в определенном порядке. Защищенность здесь определяется главным образом трудностью получения всех фрагментов, хотя при монолитном рассечении владение даже одним из них может дать какое-то представление об общем содержимом.

Монолитное рассечение—разнесение — подходящий прием, когда дробление на слишком мелкие фрагменты с их последующими хранением, получением и сборкой трудноосуществимо, а вероятность доступа к потокам мала. С развитием вычислительных и информационно-коммуникационных технологий такое рассечение целесообразно заменять диффузным, при котором фрагменты могут предельно минимизироваться, а затем передаваться, храниться и собираться в любой последовательности. Сами потоки при этом становятся внешне бессмысленными, что придает защите новое качество.

Пример диффузного подхода представлен в работах [13—15]. Здесь предлагается посимвольная фрагментация сообщений с разнесением символов по потокам согласно некоторой таблице с вычислением идентификатора (номера) потока по формуле

$$p_s = n(r_i - 1) + c_j,$$

где n — число столбцов; r_i — числовой идентификатор строки; c_j — идентификатор столбца в таблице. Порядок формирования потоков нормальный. Например, для двух строк с идентификаторами $r = \{2,1\}$ и четырех столбцов с идентификаторами $c = \{4,1,3,2\}$ получаем табл. 1.

Таблица 1

	4	1	3	2	4	1	3	2	4	1	3	2	4	1	3	2
2	О	д	н	а	в	—	с	т	у	ю	—	з	ю	п	о	р
1	ж	д	ы	—	у	д	е	н	и	м	н	ю	у			

Формируемые потоки:
$p = 1$: ддм
$p = 2$: _нюп
$p = 3$: ыен
$p = 4$: жуиу
$p = 5$: д_юп
$p = 6$: атзр
$p = 7$: нс_о
$p = 8$: Овую

Такой же прием демонстрируется во многих других источниках. К сожалению, как уже говорилось, это наихудшая реализация метода с защищенностью $S!$ при получении всех потоков (увеличение числа строк и столбцов не меняет дела, поскольку длина ключа всегда совпадает с числом потоков). В вышеприведенном примере, в частности, закон рассечения является посимвольным с ключом {8,5,7,6,4,1,3,2}, т. е. $L = S$.

Таблица 2

H	e	l	l	o	,	w	o	r	l	d	!
01001000	01100101	01101100	01101100	01101111	00101100	01110111	01101111	01110010	01101100	01100100	00100001

Таблица 3

$	f	v	W	V	D
00100100	01100110	01110110	01010111	01010110	01000100

Таблица 4

Л	к	■	√	╩	б
10001011	10101010	10110010	11111011	11001010	10100001

Более интересным примером является метод защиты, представленный в патенте US 10216822B2 от 26.02.2019 "Data distribution methods and systems" [17, 18]. Согласно патенту исходная информация обрабатывается как последовательность *битов* — минимально возможных фрагментов информации. Исходные байты рассекаются на биты, которые направляются в различные потоки, например, по принципу "чет—нечет" для двух потоков. В результате бессмысленным становится не только "текст", но случайный характер приобретают и образующиеся байты. В простейшем варианте, когда потоки нормальны, получается следующий результат.

Исходное сообщение представлено в табл. 2, поток 1 — в табл. 3, поток 2 — в табл. 4.

Предлагаются и более сложные разнесения на несколько потоков с перемешиванием (рис. 1, 2). Как видно из рис. 1 и 2, рассечение—разнесение здесь не является нормальным, что повышает защищенность.

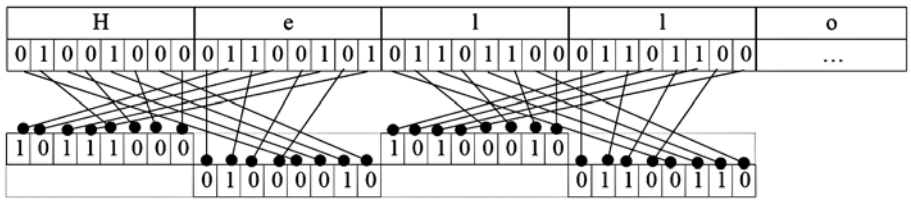

Рис. 1. Усложненное формирование потоков при побитовом рассечении по принципу "чет—нечет" с перемешиванием

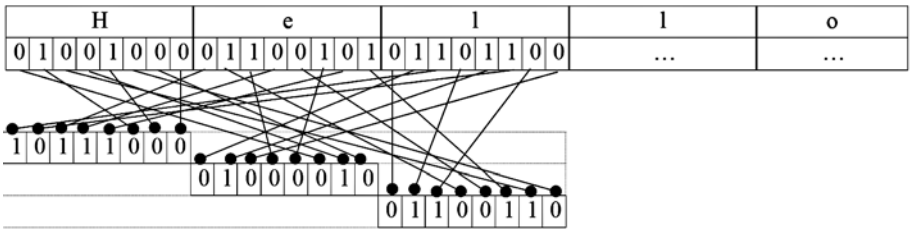

Рис. 2. Формирование потоков при побитовом рассечении с тремя потоками и перемешиванием

Для рис. 1 закон рассечения—разнесения выглядит как побитовое рассечение с 16-элементным ключом:

$$P(16,2) = \\ = \{(2,5),(1,5),(2,6),(1,6),\ (2,7),(1,7),(2,8),(1,8),\\ (2,1),(1,1),(2,2),(1,2),(2,3),(1,3),(2,4),(1,4)\}.$$

Для рис. 2 ключ 24-элементный:

$$P(16,2) = \{(2,6),(1,6),(3,7),(2,7),(1,7),(3,8),(2,8),(1,8),\\ (1,3),(3,4),(2,4),(1,4),(3,5),(2,5),(1,5),(3,6),\\ (3,1),(2,1),(1,1),(3,2),(2,2),(1,2),(3,3),(2,3)\}.$$

Особенностью реализации метода в работах [17, 18] является создание параллельно основным потокам потоков четности, получаемых из основных, их сложением посредством исключающего ИЛИ: *x*XOR*y*. Это повышает надежность, позволяя восстанавливать информацию при потере части потоков, но требует, чтобы потоки имели равную длину, что уменьшает вариативность ключа (хотя в случае перемешивания защищенность остается по-прежнему высокой). При нормальном порядке разнесения битов по потокам число возможных ключей равно (сравните с (2))

$$|\boldsymbol{P}(\boldsymbol{L},\boldsymbol{S})| = \frac{L!}{((L/S)!)^S} \sim q\left(L,S\right)^L, \text{ где } q > 1.$$

Для ключей с перемешиванием:

$$|\boldsymbol{P}(\boldsymbol{L},\boldsymbol{S})| = L!.$$

В обоих случаях работает только одна композиция числа L, а именно такая, что $n_1 = \ldots = n_S = \frac{L}{S}$. Несложно видеть, что требование равенства потоков снижает защищенность в $C_{L-1}^{S-1} = \frac{(L-1)!}{(S-1)!(L-S)!}$ раз.

Заметим, что регулярное (не случайное) разнесение битов по потокам при небольшой длине ключа и равных долях идентификаторов снижает защищенность при атаке "владение всеми потоками".

Если длина ключа не известна, показатель защищенности содержит умножение

на L (перебор всех вариантов ключей разной длины).

Общий принцип побитового рассечения—разнесения можно сформулировать как способ защищенной передачи и хранения информации, размещаемой в облачных и внешних хранилищах данных, который включает в себя формирование и преобразование первичной цифровой информации, отличающийся тем, что биты первичной цифровой информации образуют несколько битовых последовательностей так, *что существует не менее одного байта первичной цифровой информации, биты которого были помещены в разные последовательности*, с последующей передачей и размещением последовательностей в разных хранилищах данных, причем обеспечивается обратимый и контролируемый процесс. Здесь отсутствуют какие-либо ограничения на характер рассечения, позволяя генерировать ключи любой сложности и потоки различной длины. Например, фраза "Hello, world!" может быть разбита на два неравных потока:

1) Qп8у€{н9ў@

2) °¬Ђ

Программная реализация такого подхода даже в простейшем случае двух потоков с нормальным порядком согласно фоормуле (5) дает защищенность

$$D = \frac{2^E}{\varepsilon^S}(2^L - 2) - 1.$$

Главным отличием такого более общего взгляда служит то, что потоки могут создаваться путем побитового рассечения исходного файла любым образом. Принцип равенства потоков при этом может не соблюдаться.

Заключение

Подытоживая, можно заключить следующее:

1. В условиях повсеместного внедрения технологий облачного хранения данных остро встал вопрос защищенности размещаемой в них информации.

2. Облачные провайдеры недостаточно внимания уделяют вопросам защиты клиентских данных, что существенно снижает интерес к этой технологии.

3. В случае криптографической защиты на стороне провайдера по-прежнему остается вопрос доверия к нему в случае размещения критически важной информации.

4. Одним из перспективных способов защиты информации в облаках является метод рассечения—разнесения, заключающийся в том, что информация делится на фрагменты, из которых по определенному алгоритму формируются потоки данных, направляемые в различные хранилища так, что владение одним потоком даже в незашифрованном виде не позволяет атакующему воспользоваться полученными сведениями.

5. Защищенность тем выше, чем выше коэффициент дробления информации и больше длина ключа. Число потоков при этом существенной роли не играют.

6. Наибольшую защищенность обеспечивает побитовое рассечение, притом так, что потоки формируются из битов исходной информации случайным образом. Какой-либо регулярности следует избегать. Это обеспечивает высокий уровень защиты даже при получении атакующим всех потоков. При завладении лишь частью потоков перед ним стоит вообще труднореализуемая задача заполнения неизвестных битов, особенно если файлы имеют нетекстовый характер.

Список литературы

1. **Ступина М. В., Шпаков Д. В.** К вопросу безопасности облачных технологий // Молодой ученый. 2016. № 9(113). С. 85—88.

2. **Cloud Security:** взгляд на российский рынок // Information Security = Информационная безопасность [Электронный ресурс]. URL: https://lib.itsec.ru/articles2/cloud-security/cloud-security-vzglyad-na-rossiiskii-rinok (дата обращения: 02.02.2021).

3. **Довгаль В. А.** Методы повышения безопасности в сфере "облачных" технологий // Вестник АГУ. 2014. № 4(147). С. 170—174.

4. **Шпаков Д. В.** Изучение проблемы развития и безопасности облачных технологий [Электронный ресурс]. URL: https://docplayer.ru/57291877-Izuchenie-problemy-razvitiya-i-bezopasnosti-oblachnyh-tehnologiy.html (дата обращения: 02.02.2021).

5. **Вишняков А. С., Макаров А. Е., Уткин А. В.** и др. Обеспечение защиты данных, представленных в облачных сервисах [Электронный ресурс]. URL: https://cyberleninka.ru/article/n/obespechenie-zaschity-dannyh-predstavlennyh-v-oblachnyh-servisah/viewer (дата обращения: 02.02.2021).

6. **Дремач К.** Облачные сервисы: проблемы в доверии // Information Security = Информационная безопасность [Электронный ресурс]. URL: https://lib.itsec.ru/articles2/cloud-security/oblachnie-servisi-problemi-v-doverii (дата обращения: 02.02.2021).

7. **Прудникова А. А., Садовникова Т. М.** Анализ облачных сервисов с точки зрения информационной безопасности. [Электронный ресурс]. URL: https://cyberleninka.ru/article/n/analiz-oblachnyh-servisov-s-tochki-zreniya-informatsionnoy-bezopasnosti/viewer (дата обращения: 02.02.2021).

8. **Подколозина Л. А., Коваленко Д. К.** Обеспечение безопасности мобильных облачных хранилищ на основе методов классификации данных [Электронный ресурс]. URL: https://cyberleninka.ru/article/n/obespechenie-bezopasnosti-mobilnyh-oblachnyh-hranilisch-na-osnove-metodov-klassifikatsii-dannyh/viewer (дата обращения: 02.02.2021).

9. **Сахаров Д. В., Левин М. В., Фостач Е. С., Виткова Л. А.** Исследование механизмов обеспечения защищенного доступа к данным, размещенным в облачной инфраструктуре // Наукоемкие технологии в космических исследованиях Земли. 2017. Т. 9, № 2. С. 40—46.

10. **Алексеев А. П., Орлов В. В.** Сокрытие сообщений путем их распыления в пространстве // Инфокоммуникационные технологии. 2008. Т.6, N. 3. С. 52—56.

11. **Алексеев А. П., Макаров М. И.** Принципы многоуровневой защиты информации // Инфокоммуникационные технологии. 2012. Т. 10, № 2. С. 88—93.

12. **Алексеев А. П.** Многоуровневая защита информации. Самара: ПГУ-ТИ-ИУНЛ, 2017. 128 с.

13. **Безбогов А. А., Яковлев А. В., Шамкин В. Н.** Методы и средства защиты компьютерной информации: Учеб. пособ. Тамбов: Издательство ТГТУ, 2006. 196 с.

14. **Лыгин Е. А.** Тайнопись. Практическое пособие по ручному шифрованию. Саратов: Новый ветер, 2010. 206 с.

15. **Виды** и способы криптографических преобразований [Электронный ресурс]. URL: https://www.securitylab.ru/blog/personal/aguryanov/29980.php (дата обращения: 01.02.2021).

16. **Древние** информационные системы: защита информации [Электронный ресурс]. URL: https://zen.yandex.ru/media/info_law_society/drevnie-informacionnye-sistemy-zascita-informacii-5c83d8eae2ca0400b31c154f (дата обращения: 02.02.2021).

17. **Data** distribution methods and systems [Электронный ресурс]. URL: http://appft.uspto.gov/netacgi/nph-Parser?Sect1=PTO1&Sect2=HITOFF&d=PG01&p=1&u=%2Fnetahtml%2FPTO%2Fsrchnum.html&r=1&f=G&l=50&s1=%2220150293986%22.PGNR.&OS=DN/20150293986&RS=DN/20150293986 (дата обращения: 02.02.2021).

18. **Data** distribution methods and systems [Электронный ресурс]. URL: https://patents.google.com/patent/US10216822B2/en?oq = US + 10216822B2 (дата обращения: 02.02.2021).

L. V. Arshinskiy, Professor, e-mail: larsh@mail.ru,
G. N. Shurkhovetsky, Postgraduate Student, e-mail: gshn5@yandex.ru,
Irkutsk State Transport University, Irkutsk, Russian Federation

Features Application of the Dissection-Placing Method for Secure Data Storage in External Data Warehouses

The article considers the application of the dissection-placing method for secure storage of information in external, primarily cloud-based data warehouses. Various approaches to the implementation of the method, including patent material, are analyzed. It is shown that the method is most effective for bitwise dissection of files and random placement of bits in data streams sent to separate warehouses.

Keywords: *information security, data storage, cloud computing, dissection-placing method*

DOI: 10.17587/it.27.259-266

References

1. **Stupina M. V., Shpakov D. V.** On the issue of cloud security, *Young Scientist*, 2016, no. 9 (113), pp. 85—88 (in Russian).

2. **Cloud Security:** a look at the Russian market, *Information Security*, https://lib.itsec.ru/articles2/cloud-security/cloud-security-vzglyad-na-rossiiskii-rinok (date of access: 01/02/2021) (in Russian).

3. **Dovgal V. A.** Methods of safety increase in the sphere of "cloud" technologies, *The Bulletin of the Adyghe State University*, 2014, no. 4(147), pp. 170—174 (in Russian).

4. **Shpakov D. V.** Study of the problem of development and security of cloud technologies, https://docplayer.ru/57291877-Izuchenie-problemy-razvitiya-i-bezopasnosti-oblachnyh-tehnologiy.html (date of access: 02.02.2021) (in Russian).

5. **Vishnyakov A. S., Makarov A. E., Utkin A. V.** et. al. Ensuring the protection of data presented in cloud services, available at: https://cyberleninka.ru/article/n/obespechenie-zaschity-dannyh-predstavlennyh-v-oblachnyh-servisah/viewer (date of access: 02.02.2021) (in Russian).

6. **Dremach K.** Cloud services: Problems in trust, *Information Security,* available at: https://lib.itsec.ru/articles2/cloud-security/oblachnie-servisi-problemi-v-doverii (date of access: 02.02.2021) (in Russian).

7. **Prudnikova A. A., Sadovnikova T. M.** Analysis of cloud services from the point of view of information security, available at: https://cyberleninka.ru/article/n/analiz-oblachnyh-servisov-s-tochki-zreniya-informatsionnoy-bezopasnosti/viewer (date of access: 02.02.2021) (in Russian).

8. **Podkolozina L. A., Kovalenko D. K.** Ensuring the security of mobile cloud warehouses based on data classification methods, available at: https://cyberleninka.ru/article/n/obespechenie-bezopasnosti-mobilnyh-oblachnyh-hranilisch-na-osnove-metodov-klassifikatsii-dannyh/viewer (date of access: 02.02.2021) (in Russian).

9. **Sakharov D. V., Levin M. V., Fostach E. S., Vitkova L. A.** Research of mechanisms of the protected access problem to cloud data storage, *H&ES Research*, 2017, vol. 9, no. 2, pp. 40—46 (in Russian).

10. **Alekseev A. P., Orlov V. V.** Hiding messages by scattering them in space, *Infocommunication Technologies*, 2008, vol. 6, no. 3, pp. 52—56 (in Russian).

11. **Alekseev A. P., Makarov M. I.** Principles of multilevel protection of the information, *Infocommunication Technologies*, 2012, vol. 10, no. 2, pp. 88—93 (in Russian).

12. **Alekseev A. P.** Multilevel protection of the information. Samara, PGU-TI-IUNL, 2017, 128 p. (in Russian).

13. **Bezbogov A. A., Yakovlev A. V., Shamkin V. N.** Methods and means of protection of computer information systems: textbook. Tambov, TSU, 2006. 196 p. (in Russian).

14. **Lygin E. A.** Secret writing. A practical guide to manual encryption. Saratov, Novyy veter, 2010, 206 p. (in Russian).

15. **Types** and methods of cryptographic transformations, available at: https://www.securitylab.ru/blog/personal/aguryanov/29980.php (date of access: 01.02.2021) (in Russian).

16. **Ancient** information systems: Information security, available at: https://zen.yandex.ru/media/info_law_society/drevnie-informacionnye-sistemy-zascita-informacii-5c83d8eae2ca0400b-31c154f (date of access: 01.02.2021) (in Russian).

17. **Data** distribution methods and systems, available at: http://appft.uspto.gov/netacgi/nph-Parser?Sect1=PTO1&Sect2=HITOFF&d=PG01&p=1&u=%2Fnetahtml%2FPTO%2Fsrchnum.html&r=1&f=G&l=50&s1=%2220150293986%22.PGNR.&OS=DN/20150293986&RS=DN/20150293986(date of access: 02/02/2021).

18. **Data** distribution methods and systems, available at: https://patents.google.com/patent/US10216822B2/en?oq = US + 10216822B2 (date of access: 02/02/2021).