© Р.Т. ФАЙЗУЛЛИН, И.Р. ФАЙЗУЛЛИН, О.Т. ДАНИЛОВА
frt@omgtu.ru, blackildar@list.ru, olgdan56@gmail.com

УДК 519.7

АЛГОРИТМЫ РАЗДЕЛЕНИЯ СЕКРЕТА С ИСПОЛЬЗОВАНИЕМ ПРИНЦИПИАЛЬНО МАЛОЙ ЧАСТИ СЕКРЕТА В КАЧЕСТВЕ КЛЮЧА

АННОТАЦИЯ. Предложены аппаратно-эффективные алгоритмы разделения секрета, основанные на применении стеганографической ключевой информации для использования в дата-центрах.

SUMMARY. The given article presents hardware effective algorithms of secret division for data center applications based on steganography-like approach.

КЛЮЧЕВЫЕ СЛОВА. Схема разделения секрета, дата-центр.
KEY WORDS. Secret division scheme, data center.

Консолидация обработки и хранения информации в **центрах обработки данных (ЦОД)** — одна из самых перспективных областей совершенствования инфраструктуры корпоративных систем. Применение и внедрение в ЦОД позволяет оптимально использовать вычислительные ресурсы, уменьшает общее число серверного оборудования, снижает расходы на их поддержку [1], .[2]. Целью компонента по обеспечению информационной безопасности автоматизированных систем и баз данных ЦОД является защита информации и поддерживающей ее инфраструктуры от случайных или преднамеренных воздействий естественного или искусственного характера, чреватых нанесением ущерба системе и ее пользователям. Информационная безопасность автоматизированных систем и баз данных ЦОД обеспечивается за счет реализации многоуровневой системы безопасности и контроля, выполняющей следующие задачи:

• обеспечение равнопрочной защиты информации и контроля доступа к ней на всех этапах ее сбора, передачи, обработки, хранения и предоставления пользователю системы;

• обеспечение конфиденциальности и целостности информации;

• обеспечение защиты программных и технических средств, используемых для сбора, передачи и обработки информации от утечки и разрушающего воздействия.

В связи с вышесказанным защите подлежит, наряду с другими компонентами, и статистическая и аналитическая информация, формируемая в результате взаимодействия пользователей между собой. А основными направлениями защиты информации здесь являются: защита от вирусных атак; обеспечение безопасности процесса взаимодействия информационных систем ЦОД с внешними источниками информации; защита информации от несанкционированного доступа.

Представляется возможным предложить решение обеспечения компонента информационной безопасности с помощью схемы разделения секрета, когда клиент центра обработки данных передает на хранение данные, но не инфор-

мацию как таковую. Простейшим примером такого рода протокола может служить передача ЦОД набора бит, зашифрованного одноразовым ключом, длина которого больше или равна длине набора передаваемых бит. В этом случае математически доказано, что набор бит принципиально недешифруем, но также очевидно, что данная схема непригодна практически, т.к. теряется сам смысл сервиса, предоставляемого ЦОД. Набор ключа и данных в этом случае должны быть равны по длине. Все попытки использовать малую часть секрета, как начальное значение для генератора ключа, являются аппроксимациями схемы с одноразовым ключом, которые трудно оценить по стойкости.

Также выбор генератора возлагается на клиента, который не является специалистом в области защиты информации. Какое же решение может удовлетворить всех участников протокола и учесть компетенции сторон?

Обратим внимание на то, что, учитывая уже имеющиеся технические возможности ЦОДов, критичным является размер секрета, хранимого у клиента, но не размер данных, передаваемых клиентом в ЦОД. Поэтому увеличение размеров данных по сравнению с исходным информационным блоком на порядок или более не представляется чем-то критичным.

Это позволяет привлечь к решению задачи стеганографические алгоритмы, например, встраивая в поток случайно сгенерированных данных блоки зашифрованного текста, статистически не отличимые от окружения.

Но опять же возникают вопросы об эффективности и доказуемой стойкости протоколов. Например, бремя кодирования и декодирования информации ложится уже на клиентскую часть схемы и поэтому требует максимально простых и эффективных алгоритмов, работающих «на лету».

В качестве базового алгоритма можно рассмотреть следующую процедуру.

Рассмотрим поток данных, состоящий из бит, полученных с помощью физического генератора псевдослучайных чисел или с помощью программного датчика псевдослучайных чисел. Задача состоит в том, чтобы встроить в этот поток маркер сообщения, который будет содержать само сообщение. Схожая задача была рассмотрена в работе [3], но там маркер предваряет сообщение и зависит от него, как хэш-код, что представляется не совсем эффективным, т.к. требует оффлайн-обработки больших объемов данных.

Рассмотрим последовательность байтов Q^1, ..., Q^N, каждый из которых состоит из K бит и соответствующий им кортеж бит q^1, ..., q^N. Будем называть объединение этих кортежей мастер-ключом.

Передающая сторона, или Алиса, генерирует кортеж X^1, ..., X^N с условиями:

$$X^j \geq Q^j \text{ если } q^j=0 \qquad (1)$$
$$X^j < Q^j \text{ если } q^j=1$$

и маркирует начало сообщения операцией побитового сложения байтов.

$$X^j,\ Q^j: \qquad (2)$$
$$Z_i^j=X_i^j \oplus Q_i^j \quad i=1, \ldots, K$$

Получатель сообщения, или Боб, производим аналогичную операцию *XOR* над байтом Z^j и байтом Q^j, получая в итоге $\bar{X}^j$. Если некоторая последовательность битов $\bar{X}^m$, ..., $\bar{X}^{m+N-1}$ удовлетворяет условиям (1), то Боб делает вывод о том, что с вероятностью $1-2^{-N}$ эта последовательность байтов маркирует начало сообщения.

Но заметим, что Алиса может специально генерировать некоторые из X^j так, что условия (1) для них не будут выполняться. Если число таких «ошибок» (будем далее называть их «кодо», т.е. кодирование ошибками) будет относительно мало, например, не более четверти и окаймлено заранее заданным числом не кодо, то Боб, проверяя входной поток, может сделать хорошо обоснованный вывод о наличии специально встроенных Алисой символов «ошибки». Более того, X_j, отвечающие кодо, могут быть не сгенерированы, а нести уже полезную информацию. Боб должен выбрать из двух гипотез: он получает случайный поток данных или поток данных, созданный Алисой. Учитывая, что случайный поток данных, например, созданный с помощью генератора псевдослучайных чисел BBS, хорошо описывается схемой Бернулли, а при увеличении N хорошо описывается нормальным законом, то выделение кортежа $\bar{X}^m$, ..., $\bar{X}^{m+N-1}$ не представляет большого труда, с учетом, например, того, что доверенное лицо проводит тест на открытый текст для наиболее вероятного фрагмента потока данных. В этом случае Боб может интерпретировать маркер как битовую строку $\bar{q}^{-j}$, j=1, ..., N, где нулями являются те из q^{-j}, которые совпадают с q^{-j}, а единицами места кодо. Длина слова в этом случае будет равна N, а «полезное пространство сообщений» 2^{N-l}, где l сравнимо с N. Насколько стойкой является предложенная схема шифрования?

Предположим, что аналитик каким-то образом смог выделить несколько кортежей Z^{jS}, j=1, ..., N, s=1, ..., M $M \geq 2$, отвечающих маркерам, и ему известно, что в маркерах нет кодо. Предположим, что мы точно знаем расположение M кортежей Z^j в потоке данных и тем самым знаем $z^{jS}{}_i = x^{jS}{}_i \oplus Q^{jS}{}_i$ i=1, ..., K, s=1, ..., M, j=1, ..., N для каждых j, s. Зафиксируем некоторое j и оценим трудоемкость определения битов Q_1^j, ..., Q_K^j при неизвестном q.

Оказывается, что в этом случае верна следующая лемма [4].

Лемма 1. *В наилучшем для атакующего случае трудоемкость определения мастер-ключа Q_1^j, ..., Q_K^j и q^1, ..., q^N не меньше, чем $CNK2^K$.*

Из леммы следует важный практический вывод о том, что гарантированная на сегодняшний день стойкость, равная 2^{80}, без учета операций записи и считывания из памяти, будет обеспечиваться выбором длины байта K большей или равной 64. Также можно сформулировать очевидную лемму [5] о классе сложности, к которому принадлежит задача криптографического анализа данного алгоритма.

Лемма 2. *Задача определения мастер-ключа при известных кортежах Z^j не принадлежит классу NP.*

Заметим, что реализация кодирования информации, или создание последовательности X^1, ..., X^N, может быть осуществлена инвертированием бит, например, применением операций И, ИЛИ с битами байта Q и содержимым некоего регистра сдвига с линейной обратной связью. Если, например, q^j=0, то некоторые нулевые биты Q^j необходимо инвертировать на единичные (ИЛИ), если же q^j=1, то наоборот, необходимо инвертировать единичные биты Q^j (И).

Принимающая сторона должна определить набор q^1, ..., q^N пропуская его через регистр и тестируя байты на предмет проверки условия $(X^{jS}\text{-}Q^j)(-1)^{gj} \geq 0$, желательно с помощью автомата без памяти. В работе [6] показано, что КНФ, которая решает задачу распознавания ситуации $A \geq B$ за один такт, требует экспоненциального числа входов в зависимости от размера битовой записи чисел. Там же показано, что предложенная схема может быть сведена с помощью преобразований Цейтлина, т.е. введением новых вспомогательных переменных, к многоярусной релейной схеме, глубина которой равна K. Это неявно

предполагает, что в автомате могут реализоваться гонки, и усложняет структуру схемы, требуя введения задержек сигналов.

Оказывается, можно произвести сравнения всех бит Q^j и X^j за один-два такта работы автомата без памяти. Рассмотрим задачу сравнения однобитовых чисел x и y. Можно записать базовые эквивалентности, которые кодируют релейную схему:

$$x \geq y \Leftrightarrow x \vee \bar{y} \qquad (3)$$

$$x \neq y \geq \Leftrightarrow (x \wedge \bar{y}) \vee (\bar{x} \wedge y)$$

Используя эти соотношения, можно далее построить релейные схемы и для сравнения чисел с длиной записи в несколько бит. Например, для числа с длиной записи в два бита можно записать:

$$a \geq b \Leftrightarrow \{a_2 \geq b_2\} \wedge (\{a_2 \neq b_2\} \wedge \{a_1 \geq b_1\}) \qquad (4)$$

Нетрудно заметить, что длина записи логического выражения растет квадратично, как сумма арифметической прогрессии, в зависимости от длины записи исходных чисел. Это делает реалистичной техническую реализацию схемы для сравнения чисел длины K=64, и заметим, что нам не совсем обязательно сводить полученное выражение к КНФ, нас интересует лишь истинность логического выражения. Программная реализация КНФ или полученного логического выражения на параллельных вычислительных устройствах возможна, но принципиально не так эффективна, как аппаратная. Это позволяет сделать вывод о перспективности использования данного подхода в практике корпораций и неэффективности программных реализаций частными лицами.

Рассмотрим приложение предложенного алгоритма к исходной задаче разделения секрета. Если предположить, что последовательность q^1, ..., q^N или весь мастер-ключ это секрет, хранимый у клиента ЦОД, то как выбрать этот секрет так, чтобы он удовлетворял статистическим тестам на проверку случайности и чтобы он был стоек к частотному анализу или к тесту на биграммы?

Рассмотрим блок данных, предназначенных для хранения, как набор байт, каждый из которых требует для своей записи L бит, F_1, ..., F_N. Образуем суммы следующего вида:

$$\Omega_i = F_i + F_{i+1},\ \Omega_N = F_N + F_1 \qquad (5)$$

где для записи Ω_i требуется уже в примерно половине случаев $L+1$ бит. Образуем из старших бит Ω_i битовую часть мастер-ключа, т.е. последовательность q_1,..., q_N и закодируем информацию о F_1, ..., F_N с помощью приведенной выше процедуры, т.е. с помощью кодо. Рассмотрим последовательность $\bar{q}_1$, ..., $\bar{q}_N$, где переменная с чертой равна нулю, если нет кодо и равна единице, если кодо есть. Будем считать, что $N=2P + K$ и будем располагать значимую информацию строго «по-середине», т.е. окаймляя строку длиной K бит строками по P бит, которые не будут содержать ошибок. Тогда с помощью кодо можно будет кодировать биты F_i или остальные биты всех Ω_i.

Лемма 3. *Для дешифрования сообщения с помощью атаки грубой силы требуется не менее $CNK2^K$ операций.*

Выберем в качестве ключа q_1, ..., q_N последовательность бит, в которой будем считать некоторую часть уже известной, например, определенной с помощью переборных алгоритмов, приведенных выше, т.е. будем считать, что q_1,..., q_P и q_{P+K+1}, ..., q_N

уже известны. Для определения оставшихся неизвестных бит будем использовать следующий тривиальный алгоритм:

1. Выберем произвольный битовый вектор q_{P+1}, ..., q_{P+K}, который будем считать частью ключа.

2. Используя предполагаемые q_i и кодо, восстановим все F_1, ..., F_N.

3. Построим Ω_i и восстановим для них битовый вектор t_1, ..., t_N представляющий запись старших бит слагаемых.

4. Проверим, совпадают или нет биты последовательностей q_1, ..., q_N и t_1, ..., t_N.

Если последовательности совпадают, то мы, возможно, дешифровали сообщение и нашли ключ, зависящий от сообщения. Если нет, то следует обращение к пункту 1.

Отсюда можно сделать вывод, что аналогично предыдущему, при $K \geq 64$ и $N \approx 100$ алгоритм обеспечивает должную степень стойкости. Также приведенная процедура может служить контролем ошибок при передаче данных.

Обратим внимание, что в качестве секрета мы выбрали битовую строку длины N. Это при малых K увеличивает часть секрета, хранимого у клиента ЦОД.

Можно предложить модификацию, когда на клиентской части вычислительной системы будет храниться только битовая часть мастер ключа длины K.

Рассмотрим в качестве операции сложения побитовое сложение и будем считать, что на серверной части хранятся байты вида:

$$Z_1=F_1 \oplus F_2^1, \ldots, Z_K=Z_{K-1} \oplus F_K^K, Z_{K+1}=Z_K \oplus F^1_{K+1} \quad (6)$$

где верхний индекс j соотнесен с битом и слагаемые задаются следующим образом: $F_{1j}=0$, $F_r^j=F_1$, $F_{1j}=1$, $F_r^j=\bar{F}_r$.

Число операций, необходимых для определения F_1 при атаке грубой силы оценивается величиной $C2^K$, что требует использования уже двойного слова. Но в отличие от предыдущего алгоритма объем данных, располагаемых на серверной части приложения, ограничен разумными пределами и возможна эффективная реализация алгоритма с помощью простейших релейных схем.

Заключение. Предложены алгоритмы разделения секрета для использования в сфере услуг центров обработки данных, предполагающие использовать меньшую часть секрета как ключ шифрования для большей части. Даны оценки криптографической стойкости и предложена схема аппаратной реализации алгоритма.

СПИСОК ЛИТЕРАТУРЫ

1. URL: http://www.bytemag.ru/articles/detail.php?ID=14070.
2. URL: http://www.modetel.ru.
3. Файзуллин И.Р. Криптостеганографический алгоритм с использованием хэш-функции для маркировки начала сообщения // Тез.докл. Научная сессия МИФИ-2007. Т. 16 Компьютерные науки. Информационные технологии. С. 149-150
4. Файзуллин И.Р., Файзуллин Р.Т. Аппаратно-эффективный алгоритм формирования маркера начала сообщения // Компьютерная оптика. Т. 34. № 4. С. 552-554.
5. Файзуллин И.Р., Файзуллин Р.Т. Алгоритм кодирования и передачи информации базирующийся на стеганографическом подходе // Вестник ТюмГУ. 2010. № 6. Физ.-мат. науки. Информатика. С. 147-152.
6. Дулькейт В.И. Сведение задач факторизации, дискретного логарифмирования и логарифмирования на эллиптической кривой к решению ассоциированных задач «Выполнимость»// Компьютерная оптика. 2010. Январь-март. Т. 34. № 1. С. 118-123.