КОМПЬЮТЕРНОЕ ОБЕСПЕЧЕНИЕ И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА

УДК 681.3.04

Ш. Г. Магомедов

ПЕРЕДАЧА И ПРИЕМ ДАННЫХ В ВЫЧИСЛИТЕЛЬНЫХ УСТРОЙСТВАХ С ИСПОЛЬЗОВАНИЕМ СИСТЕМЫ ОСТАТОЧНЫХ КЛАССОВ

Традиционная технология обеспечения защиты информации ограниченного доступа обычно предполагает использование методов шифрования. Однако требования, предъявляемые к подобным системам, достаточно строгие и, как следствие, громоздкие в реализации и затратные в эксплуатации, что часто делает их использование малоприемлемым и затруднительным. Предложен иной подход к организации защищенного обмена данными между процессором и оперативной памятью, включающий две особенности: использование системы остаточных классов в качестве основы технологии закрытия данных и подход к организации процесса обмена, исключающий использование специальных подсистем в рамках системы защиты. При таком подходе к процессу защиты передаваемых данных потребность в наличии указанного центра отпадает, что лишает потенциальных злоумышленников возможности выбора диспетчерского центра в качестве объекта атаки и тем самым повышает безопасность процесса обмена данными. Предлагается при обмене данными между процессором и оперативной памятью возложить процедуру формирования ключей непосредственно на участников обмена данными (процессор и оперативную память), но при этом необходимо, чтобы минимальное время раскрытия выбранных ключей было меньше, чем период смены системы остаточных классов, которые используются в процессе шифрования. Таким образом, если даже злоумышленник вскроет текущие параметры системы остаточных классов, то окажется, что они уже заменены на другие, и в этой гонке за ключами злоумышленник всегда будет отставать от системы защиты информации. Применительно к системе защите, основанной на использовании системы остаточных классов, данная задача приобретает еще большую актуальность, поскольку уровень стойкости системы закрытия данных, основанной на системе остаточных классов, ниже стойкости многих современных систем шифрования.

Ключевые слова: система остаточных классов, процессор, оперативная память, защита информации, вычислительные устройства, безопасность данных.

Введение

Традиционная технология обеспечения защиты информации ограниченного доступа обычно предполагает использование методов шифрования. Однако требования, предъявляемые к подобным системам, достаточно строгие и, как следствие, громоздкие в реализации и затратные в эксплуатации, что часто делает их использование малоприемлемым и затруднительным. Ниже предлагается процедура закрытия информации, опирающая на использование системы остаточных классов (СОК). Кроме того, традиционная схема защиты линий обмена данными обычно, как необходимый этап, включает задачу распределения заданного ключа шифрования между участниками обмена данными, и при передаче данных они шифруются (или подписываются, если речь идет об электронной подписи) с помощью этого ключа. Контроль за процессом использования ключей, обновлением, формированием и распределением ключей, по существующим технологиям, должен осуществлять специальный диспетчерский центр в составе системы, в частности специальная подсистема (или утилита) при взаимодействии процессора с

оперативной памятью. Поскольку вся ключевая информация, включая шифры, сосредоточена в этом центре, то центр становится потенциальным объектом злоумышленных атак. В связи с этим необходимо предпринимать особые усилия по защите этого центра, что является достаточно сложной и затратной задачей. Специфика данной проблемы применительно к задаче обмена данными между процессором и оперативной памятью (эта задача и является предметом наших исследований) заключается, в частности, в использовании СОК в качестве основы технологии закрытия данных, поскольку в [1–3] мы рассматриваем процессоры, в которых СОК является вычислительной основой процессора. Именно поэтому использование СОК также в качестве основы технологии закрытия данных в процессе обмена рассматривается как естественное продолжение технологий выполнения вычислительных операций в процессоре.

Работ по тематике использования СОК в качестве основы технологии закрытия данных нам найти не удалось. Наиболее близкой является наша работа по использованию СОК в процессе обмена данными между удаленными субъектами [1]. Близкая к рассматриваемой процедура организации защиты передаваемых данных при сетевом обмене на основе частой смены ключей шифрования рассмотрена в [4].

I. Процедура закрытия данных в оперативной памяти в процессе ее обработки

Итак, нами предлагается другой подход к организации защищенного обмена данными между процессором и оперативной памятью, включающий следующие две особенности. Первая – использование СОК в качестве основы технологии закрытия данных. Вторая – подход к организации процесса обмена, исключающий использование специальных подсистем в рамках системы защиты. Конкретизируем: предлагается подход к процессу защиты передаваемых данных, при котором потребность в наличии указанного центра отпадает, что лишает потенциальных злоумышленников возможности выбора диспетчерского центра в качестве объекта атаки и тем самым повышает безопасность процесса обмена данными. Предлагается при обмене данными между процессором и оперативной памятью процедуру формирования ключей возложить непосредственно на участников обмена данными (процессор и оперативную память), но при этом необходимо, чтобы минимальное время раскрытия выбранных ключей было меньше, чем период смены СОК, которые используются в процессе шифрования. Таким образом, если даже злоумышленник вскроет текущие параметры СОК, то окажется, что они уже заменены на другие, и в этой гонке за ключами злоумышленник всегда будет отставать от системы защиты информации. Применительно к системе защиты, основанной на использовании СОК, данная задача приобретает еще большую актуальность, поскольку уровень стойкости системы закрытия данных, основанной на СОК, ниже стойкости многих современных систем шифрования.

Предлагаемая ниже процедура может быть использована в процессе взаимодействия процессора с оперативной и внешней памятью. Она обеспечивает защиту текущих обрабатываемых данных, и, следовательно, даже если злоумышленник будет похищать данные непосредственно из оперативной или другой памяти, он не сможет воспользоваться похищенной информацией, поскольку у него нет параметров дешифровки этих данных. Отметим, что предлагаемый алгоритм позволяет организовать настолько быстрое изменение параметров обработки, что даже если злоумышленник будет оперативно и быстро дешифрировать похищаемые данные, он не будет успевать за обновляемыми параметрами обработки.

Отметим другие преимущества предлагаемой процедуры по сравнению с существующими системами шифрования:

– отсутствие необходимости обмена ключами (или ключом) шифрования;

– доступность для использования всеми физическими и юридическими объектами и субъектами, которые желают обменяться сообщениями именно с данным физическим или юридическим субъектом и знают или могут получить необходимые общедоступные параметры указанного субъекта (например, для юридического субъекта – наименование, юридический адрес, сфера деятельности и др.; для физического субъекта – фамилия, имя, отчество, возраст и др.). Список этих параметров согласовывается предварительно и может включать параметры, которые добавляет сам субъект к списку общесогласованных параметров.

Данная операция является одной из наиболее трудоемких и обычно выполняется после завершения всех вычислений и преобразований, связанных с приемом/передачей данных. Для по-

вышения быстродействия во многих алгоритмах используются преимущества табличных методов. Характерная особенность известных алгоритмов – хранение констант СОК в памяти (модули, веса позиционных представлений, базисы и др.).

Отметим, что при вычислении базисов СОК $\pi(n, \vec{P}) = (P_1, P_2, \ldots, P_n)$ наибольшие временные затраты связаны с операциями нахождения обратных весов в уравнении $m_i\delta_i \equiv 1 \pmod{P_i}$, где m_i – целое положительное число, называемое весом P_i; δ_i – остаток от деления полученной величины на модуль P_i [5, 6].

Таким образом, основные характеристики определяются свойствами набора простых чисел $P_1, P_2, \ldots, P_n$. Предлагается следующая процедура их выбора.

0. Предварительно формируется база всех простых чисел, не превосходящих заданного числа *N*, где *N* определяется максимальной величиной чисел, которые предполагается обрабатывать каждым ядром микропроцессора.

Опишем теперь процедуру закрытия результирующих данных процессором перед отправкой их в шину.

1. В виде строки записываются следующие данные: выбранная команда; адреса операндов в порядке их следования; данные, содержащиеся в адресах операндов. К этим данным, в зависимости от требований по безопасности, могут быть добавлены другие данные, в частности отдельные данные, связанные со средой обработки, временем обработки, характеристиками микропроцессора, на котором будет идти обработка, параметрами владельца используемой программы. Период обновления последних данных является свободным параметром системы, выбирается субъектом (пользователем, разработчиком) и может вообще не изменяться либо меняться перед каждым этапом или сеансом работы. Полученная строка оцифровывается любым способом, например путем сопоставления каждому знаку его ASCII-кода. В результате получаем число *D*, однозначно соответствующее данной конкретной ситуации по обработке данных.

2. Полученное число *D* разбивается на блоки, каждый из которых как число не превосходит числа *N*. Все блоки для данного числа складываются по модулю *N*; в результате получается число *M*. Если $M \le \left[\frac{N}{2}\right]$, то *M* заменяется на $N - M$, так что всегда выполняется неравенство $\left[\frac{N}{2}\right] \le M < N$. Очевидно, значение *M* зависит от параметров ситуации обработки данных.

3. Из базы простых чисел выбирается наибольшее простое число *P*, меньшее числа *M*. Полагаем $P_0 = M$, $P_1 = P$, вычисляем $R_1 = 1 - 4\frac{P_0 - P_1 - 1}{n_0\,(P_0 - P_1)^2}$ и находим $Q_2 = P_0 \bmod\left(P_1\left[R_1\right]\right)$, где [ ] – знак целой части числа (коэффициент R_1 введен для того, чтобы, с одной стороны, число *n* простых чисел в окончательном наборе $P_1, P_2, \ldots, P_n$ не оказалось малым, а с другой – чтобы соседние простые числа в наборе достаточно сильно различались. Дополнительные пояснения по выбору R_1 и n_0 приводятся ниже). Выбираем из базы простых чисел наибольшее простое число P_2, меньшее Q_2. Процедура формирования простых чисел продолжается аналогичным образом по формулам $Q_{j+1} = P_{j-1} \bmod\left(P_j\left[R_j\right]\right)$, где $R_j = 1 - 4\frac{P_{j-1} - P_j - 1}{n_0 (P_{j-1} - P_j)^2}$ и P_{j+1} – наибольшее целое число, меньшее Q_{j+1}. Процедура продолжается до тех пор, пока не достигнем $n = n_0$ либо при некотором $j = n + 1$ не получим $P_j = 1$. Так как $R_j < 1$, то очевидно $P_{j+1} < P_j$ для всех *j*.

4. Набор чисел $P_1, P_2, \ldots, P_n$ и есть искомый. Отметим, что в полученном наборе простые числа расположены не в порядке возрастания, как описано выше, а в порядке убывания.

5. Передаваемое сообщение, аналогично пункту 2, записывается в виде текстовой строки, оцифровывается (получаем число *C*) и разбивается на блоки $C_1, C_2, \ldots, C_r$, каждый из которых

по величине не превосходит значения $P = \prod_{i=1}^{n} P_i$. Затем каждое из чисел записывается в СОК. Получаем множество наборов $\Lambda = \{(c_{i1}, c_{i2}, ..., c_{in},), i = \overline{1,r}\}$, где $c_{ij} = C_i \mod P_j$. Сформированный набор Λ с приписанным именем передающей стороны и посылается в шину данных для размещения в памяти.

II. Процедура обработки закрытых данных микропроцессором

Рассмотрим теперь действия микропроцессора при получении закрытого набора данных вышеописанным алгоритмом. Микропроцессор имеет все необходимые данные для расшифровывания закрытых данных, ему известен алгоритм формирования набора чисел $P_1, P_2, \ldots, P_n$, а также необходимые конфиденциальные данные, если таковые используются в сообщении. Напомним, что микропроцессор получает файл, содержащий $\Lambda = \{(c_{i1}, c_{i2}, ..., c_{in},), i = \overline{1,r}\}$ и адресные характеристики данных, поэтому микропроцессор выполняет следующие действия.

1. По адресным характеристикам данных формируются все данные, необходимые для построения набора $P_1, P_2, \ldots, P_n$, который и формируется на основе описанного выше алгоритма – все данные, необходимые для его построения, у микропроцессора имеются.

2. На основе обратного преобразования из СОК в позиционную систему счисления восстанавливаются блоки C_i, $i = \overline{1,r}$ и формируется число $C = C_r, C_{r-1}, ..., C_1$.

3. Полученное число *C* переводится в символьную форму на основе преобразования, обратного использованному в пункте 1 в процессе оцифровывания текста. Например, если использовались ASCII-коды для оцифровывания, то для восстановления текста число *C* записывается в двоичной (восьмеричной или шестнадцатиричной) форме, и каждый блок из 8 битов заменяется на соответствующий ASCII-символ. Полученный набор символов и есть исходный текст.

III. Частота изменения основания СОК

Реализация описанной во введении концепции защиты обмена данными на основе использования СОК уже на стадии ее формирования ставит вопрос о частоте обновления и даже полной замене основания СОК, используемых для закрытия данных. Частая смена СОК потребует расходования значимых вычислительных ресурсов процессора, что нежелательно. При редкой смене основания СОК существенно взрастает опасность вскрытия основания и, как следствие, несанкционированное проникновение в систему обработки данных. Следовательно, есть некоторое оптимальное значение интервала между последовательными моментами смены оснований. Вследствие этого возникает следующая задача: как часто следует менять основание СОК с тем, чтобы суммарные издержки были минимальными?

Ниже предлагается формализованная модель, в рамках которой и предлагается решение задачи поиска оптимальной величины интервала между последовательными моментами смены ключей. Проведем анализ проблемы смены основания СОК на основе построения формализованной модели смены ключей и решения, в рамках этой модели, задачи частоты смены оснований. Приведем формализованное описание задачи.

В качестве критерия оптимальности возьмем вероятность хищения ключа (т. е. основания СОК) за промежуток времени, не превосходящий интервала между последовательными моментами смены или обновления оснований СОК (назовем этот промежуток периодом обновления), минус минимальное время, необходимое для активизации этого ключа. Рассмотрим вначале, от каких факторов зависит вероятность хищения ключа в рамках поставленной задачи.

Вероятность хищения зависит прежде всего от стойкости процедуры закрытия данных (шифрования), которая, в свою очередь, определяется числом чисел в основании и их величиной: чем больше чисел в основании и чем больше их значения, тем в целом более стойкой будет процедура шифрования. Кроме того, среди чисел в основании СОК не должно быть относительно малых, т. е. все числа в основании должны быть приблизительно одной длины. Однако степень близости отдельных значений может рассматриваться как параметр процедуры. Отметим, что при этом обычно несоразмерно увеличивается время шифрования.

Таким образом, основными параметрами модели являются:

а) величина простых чисел, входящих в основание СОК;

б) количество чисел в основании СОК;

в) минимальные затраты времени на восстановление данных при наличии основания СОК;

г) границы наиболее предпочтительной зоны возможных значений простых чисел в основании СОК;

д) промежуток времени обновления ключа.

Одним из основных параметров, который необходимо оценить в рамках модели, является степень стойкости ключа закрытия данных. В связи с этим вначале опишем основные положения концепции выбора функции, оценивающей стойкость ключа системы. Оценка стойкости должна включать оценку длины ключа n и оценку стойкости каждого из чисел P_i в основании СОК. Для простоты оценки ограничимся аддитивными оценками по указанным параметрам ключа. Далее, стойкость чисел P_i предлагается оценивать с помощью функций, удовлетворяющих следующим эвристическим условиям, способствующим повышению стойкости системы СОК.

1. Функция должна давать минимальные значения для оценок P_i, выходящих за пределы некоторой зоны значений. Нижняя граница зоны определяется минимально допустимым значением числа P_i – при меньших значениях сильно увеличивается вероятность вскрытия числа P_i. Верхняя граница зоны определяется вычислительными возможностями процессора и долей допустимого времени на закрытие данных в процессе их обработки – эта доля определяется при проектировании процессора.

2. Внутри зоны значений распределение значений P_i при полной замене одной СОК на другую СОК может определяться линейной (горизонтальной) функцией, что соответствует равновероятному выбору любого из значений, входящего в эту зону. Однако, если числа в основании СОК заменяются не полностью, а частично, то желательно (с точки зрения обработки текущих данных) при замене, по возможности, сохранять приближенно длину заменяемого и заменяющего чисел P_i. В этом случае равномерное распределение не подходит, в связи с чем предлагается использовать функции, имеющие зоны скопления данных – многовершинные функции. Ниже предлагается для этих целей класс двухвершинных функций.

3. Необходимо иметь параметры, определяющие степень размытости, неопределенности данных вокруг каждого из горбов многовершинной функции. Это будет дополнительным параметром повышения стойкости, осложняя, в приемлемой мере, злоумышленнику возможность ограничить зону перебора возможных вариантов ключа.

4. Желательно, чтобы функции были попроще (для уменьшения объема вычислений при оценке стойкости) и относительно гладкими (для обеспечения устойчивости вычислений).

Применительно к функции, оценивающей стойкость длины ключа, можно сформулировать аналогичные пожелания.

С учетом вышесказанного предлагается следующая формула для первичной оценки стойкости выбранного основания СОК в качестве ключа шифрования:

$$S = S(\pi(n, \vec{P})) = \varphi(n) + \sum_{i=1}^{n} f(P_i), \tag{1}$$

где

$$f(P) = y(P, a_1, b, c, d, \sigma, r) + y(P, a_2, b, c, d, \sigma, r),\ \varphi(P) = y(P, A, B, C, 0, s, R),$$

$$y(P, a, b, c, d, s, r) = \frac{a}{1 + b(P - (c - c \cdot d))^k}\left(1 - e^{-r \cdot \max(0; P - s)}\right) \tag{2}$$

и либо $k = 2$, либо $k = 4$, причем все коэффициенты – положительные числа.

Поясним содержание всех коэффициентов, входящих в определение функций $f(P)$, $\varphi(n)$ и $y(P, a, b, c, d, \sigma, r)$. Функция $y(P, a, b, c, d, \sigma, r)$, а вместе с ней и функция $\varphi(P)$, при любых положительных значениях коэффициентов являются одновершинными. В частности, на рис. 1 приведен пример функции $\varphi(P)$ со следующими значениями параметров: $A = 3$; $B = 0{,}01$; $C = 7$; $s = 3$; $R = 1$, $k = 4$.

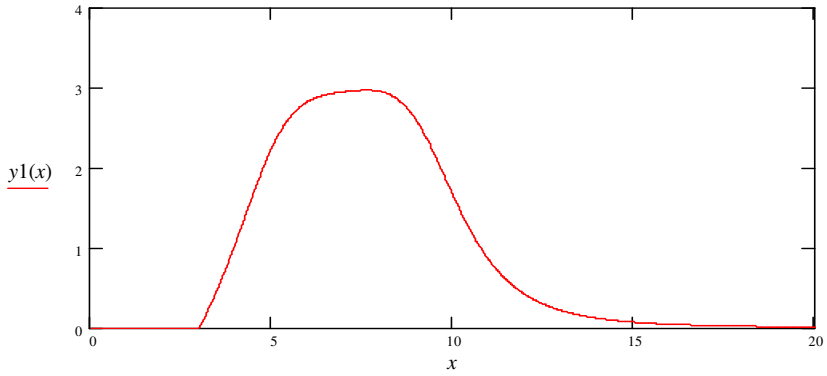

Рис. 1. Пример графика функции $\varphi(P)$

A является масштабным параметром, характеризующим максимальное значение функции $\varphi(P)$ – выбирается произвольно, исходя из соображений наглядности; B определяется степенью «растянутости» функции $\varphi(P)$ – чем больше B, тем более одновершинной является функция $\varphi(P)$ и тем у́же интервал допустимых эффективных (т. е. наиболее часто встречающихся) значений переменной x (= n); в примере B выбран так, чтобы интервал эффективных значений длины ключа n изменялся от 3 до 15; C определяет центральное значение длины ключа n (в данном случае это n = 10), вокруг которого и формируется интервал эффективных значений; s – нижняя граница допустимых значений длины ключа n; в примере наложено ограничение – количество чисел в основании СОК не должно быть меньше 3 (соответствует s = 3); R предназначено для обеспечения относительной гладкости функции в точке $n = s$ (точнее, для уменьшения величины скачка функции в граничной точке $x = s$); при увеличении R в точке $x = s$ увеличивается величина скачка графика; k определяет, во-первых, крутизну подъемов графика к вершине (слева и справа), а во-вторых, длину верхней, относительно ровной верхней части графика – при k = 4 верхняя часть длиннее и подъемы круче, при k = 2 – функция более остроконечна.

Применительно к функции $f(P)$ появляется еще один параметр – d, а параметр a заменяется на два других – a_1 и a_2. Поясним и эти параметры. Отметим, что, как видно из приведенных ниже примеров графиков функции $f(P)$, эта функция является в наиболее интересных для нас случаях двухвершинной («двугорбой»). Расстояние между горбами определяется величиной $2cd$, и, следовательно, d определяется (при фиксированном c) степенью размытости горбов. Вследствие этого параметр. Вследствие этого параметр d всегда меньше 1; его можно оценивать в процентах: насколько следует размыть зону эффективных значений переменной x вокруг ее центра c. Параметры a_1 и a_2 определяют максимальное значение каждого из горбов – левого и правого.

Отметим, что формула (1) и функция (2) удовлетворяют перечисленным выше условиям 1–3. При этом, изменяя параметр b, можно получить функции с желаемой глубиной ямы между горбами – вплоть до случая отсутствия ямы и даже с выступом вместо ямы (при малых значениях b). Относительно выбора параметра укажем следующее: значение k = 4 более предпочтительно, т. к. обеспечивает большую размытость данных в каждом горбе, но в то же время требует большего числа вычислений, что существенно при многократных (массовых) вычислениях. Для иллюстрации указанного положения ниже (на рис. 2 и 3) приведены два примера функции $fi(P)$ при k = 2 и k = 4; остальные параметры имеют одинаковые значения. Значения параметров: a_1 = 3,5; a_2 = 3; $b = 10^{-6}$; c = 400; d = 0,3; r = 0,005; s = 500.

Функция $f(P)$ описывает степень неожиданности (даже неожидаемости) для злоумышленника найти число P в составе основания СОК: малые значения P маловероятны, поскольку они менее устойчивы по отношению к взлому; большие значения P также маловероятны, поскольку сильно увеличивают затраты времени на обработку данных при их использовании в составе СОК; срединные значения также в большей степени ожидаются злоумышленником, поскольку в области срединных значений обычно и расположены значения оснований, оптимальные по компромиссным требованиям одновременно к стойкости ключей и приемлемости времени обработки данных. Графики на рис. 2 и 3 и удовлетворяют указанным интуитивным соображениям, поэтому именно по указанным причинам коэффициент S и рассматривается выше как оценка стойкости ключа.

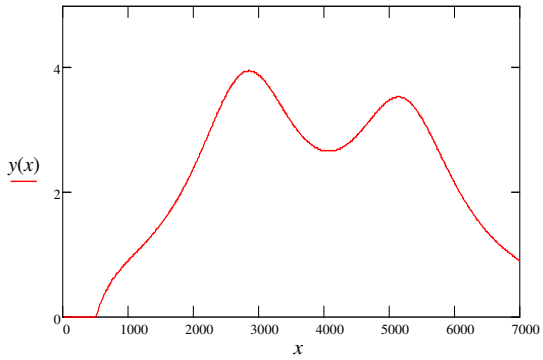

Рис. 2. Пример функции оценки с $k = 2$

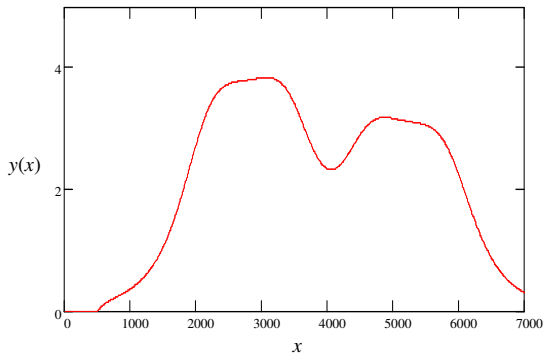

Рис. 2. Пример функции оценки с $k = 4$

Исходя из выбора показателя S в качестве показателя стойкости ключа, можно предложить выбирать ключ случайно – в соответствии с его стойкостью, т. е. вероятность выбора ключа $\pi(n,\vec{P})$ считать равной пронормированному значению S:

$$q(\pi(n,\vec{P})) = \frac{S(\pi(n,\vec{P}))}{\sum\limits_{\pi(n,\vec{U})} S(\pi(n,\vec{U}))}, \tag{3}$$

где сумма в знаменателе берется по ключам $\pi(n,\vec{U})$ таким, значения параметров n и P_i которых находятся в эффективных зонах значений этих параметров. Для того чтобы исключить повторное использование данного ключа, а также ключей, близких к нему, предлагается до использования формулы (3) перед обновлением ключа умножить функцию $f(P)$ на выражение

$$\theta(P) = \prod_{i=1}^{n} (1 - y(P, 1, \Delta, P_i, 0, s, R).$$

График функции $1 - y(P, 1, \Delta, P_i, 0, s, R)$ (назовем ее исключающей функцией) при $\Delta = 10^{-7}$, $P_i = 4000$, $s = 500$ и $R = 1$ приведен на рис. 4.

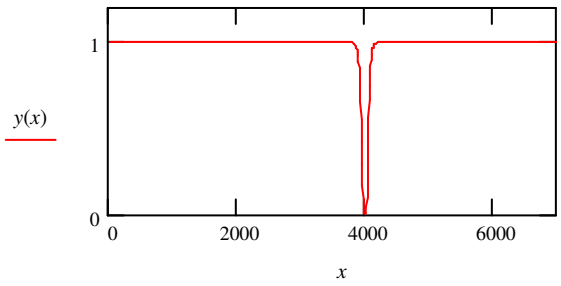

Рис. 4. График исключающей функции

Из рис. 4 видно, что функция $y(P, 1, \Delta, P_i, 0,)$ почти всюду равна 1 и лишь в узкой полосе шириной порядка $1/(P_i \cdot \Delta) \approx 250$ резко опускается вниз до нулевого значения при $P = P_i$ и затем так же резко поднимается вверх. Таким образом, при умножении функции $f(P)$ на $y(P, 1, \Delta, P_i, 0,)$ в точке $P = P_i$ функция $f(P)$ опускаться до нуля, тем самым исключая значение $P = P_i$ и близкие к нему значения при случайном выборе очередного значения P_j на основе плотности распределения $q(\pi(n,\vec{P}))$.

Отметим, что вероятность $q(\pi(n, \vec{P}))$ можно рассматривать как вероятность стойкости ключа, т. е. нераскрытия ключа за регламентное время T. Тогда в первом приближении можно предположить, что $\Pr(\pi(n, \vec{P}), T) = 1 - q(\pi(n, \vec{P}))$. Считая, что зависимость от времени вероятности нераскрытия ключа имеет экспоненциальный характер, т. е. $1 - \Pr(\pi(n, \vec{P}), t) = \alpha \cdot e^{-\beta t}$, где α и β – константы, с учетом условий $\Pr(\pi(n, \vec{P}), 0) = 0$ и $\Pr(\pi(n, \vec{P}), T) = 1 - q(\pi(n, \vec{P}))$, получаем следующее соотношение: $\alpha = 1$ и $q(\pi(n, \vec{P})) = e^{-\beta \cdot T}$, откуда выводим $e^{-\beta} = \left(q(\pi(n, \vec{P}))^{1/T}\right)$ и

$$\Pr(\pi(n, \vec{P}), t) = 1 - \left(q(\pi(n, \vec{P}))\right)^{t/T}. \quad (4)$$

Для построения модели введем следующие обозначения: $\Pr(\pi(n, \vec{P}), t)$ – вероятность раскрытия ключа $\pi(n, \vec{P})$, т. е. основания СОК, за время t; $C(\pi(n, \vec{P}))$ – затраты ресурсов (прежде всего, времени) на формирование ключа; $\tau(\pi(n, \vec{P}))$ – затраты времени на активизацию ключа $\pi(n, \vec{P})$; $L(\pi(n, \vec{P}))$ – средние потери от раскрытия ключа $\pi(n, \vec{P})$ при обмене данными; $D(\pi(n, \vec{P}))$ – затраты на обновление ключа $\pi(n, \vec{P})$; T – регламентный период работы компьютера по обработке данных с учетом повторяемости обрабатываемых данных (например, месяц, неделя, день); λ – интенсивность обмена данными между процессором и оперативной памятью; δ – промежуток между последовательными моментами обновления ключей; N – максимально приемлемое количество чисел в основании СОК.

$$x(\pi(n, \vec{P})) = \begin{cases} 1, \text{ если используется ключ } \pi(n, \vec{P}); \\ 0 \text{ в противном случае.} \end{cases}$$

Оценим суммарные потери, связанные с использованием заданного набора ключей шифрования. За регламентное время T среднее количество обновлений ключа равно $\frac{T}{\delta}$. Тогда суммарные средние издержки, связанные с обновлением ключей за регламентный период, равны:

$$\rho_1 = \frac{T}{\delta} \sum_{n=s}^{N} \left(\sum_{\pi(n, \vec{P})} (1 - q(\pi(n, \vec{P}))) C(\pi(n, \vec{P})) x(\pi(n, \vec{P})) \right).$$

Вторая группа потерь связана с раскрытием ключа в процессе передачи, что привело к потерям. Поскольку вероятность раскрытия ключа обычно является очень малой величиной, то вероятность того, что за время жизни ключа произойдет более одного раскрытия ключа, практически равна нулю. Считая, что момент вторжения злоумышленника в узел сети является случайным, естественно предположить, что среднее время, которым располагает злоумышленник для того, чтобы воспользоваться раскрытым ключом, равно $\frac{\delta}{2}$. Тогда для величины потерь, связанных с раскрытием ключа, ввиду (4), можно записать выражение

$$\rho_2 = \frac{T}{\delta} \sum_{n=s}^{N} \left(\sum_{\pi(n, \vec{P})} (1 - \left(q(\pi(n, \vec{P}))^{\delta/(2T)}\right)) L(\pi(n, \vec{P})) x(\pi(n, \vec{P})) \right).$$

Наконец, последняя группа потерь связана с затратами на обновление ключей. Так как число обновлений за один регламентный период равно в среднем $\frac{T}{\delta}$, то суммарные затраты за регламентный период на обновление ключей в среднем равны:

$$\rho_3 = \frac{T}{\delta} \sum_{n=s}^{N} \left(\sum_{\pi(n, \vec{P})} (1 - \left(q(\pi(n, \vec{P}))^{\delta/(2T)}\right)) D(\pi(n, \vec{P})) x(\pi(n, \vec{P})) \right).$$

На основе полученных соотношений выводим следующее выражение для суммарных издержек и потерь за регламентный период T:

$$w(T) = \rho_1 + \rho_2 + \rho_3 = \frac{T}{\delta}\sum_{n=s}^{N}\left(\sum_{\pi(n,\vec{P})}\left[(1-\left(q(\pi(n,\vec{P}))^{\delta/(2T)}\right))(L(\pi(n,\vec{P})) + D(\pi(n,\vec{P}))) + \right.\right.$$

$$\left.\left. + (1-q(\pi(n,\vec{P}))C(\pi(n,\vec{P}))\right]\right)x(\pi(n,\vec{P})).$$

Поскольку ключ всегда один (единственный), то справедливо равенство

$$\sum_{n=s}^{N}\left(\sum_{\pi(n,\vec{P})} x(\pi(n,\vec{P}))\right) = 1. \qquad (5)$$

На основе полученных соотношений может быть формализована задача выбора оптимального количества чисел в основании СОК, а также состава этих оснований: выбрать длину ключа n, числа P_i ($i = \overline{1;n}$) и период обновления ключа τ таким образом, чтобы суммарные издержки $w(T)$ были минимальными, т. е. чтобы выполнялось равенство (5).

$$w(T) \to \min \text{ по } x(\pi(n,\vec{P})) \text{ таким, что } x(\pi(n,\vec{P})) = 0 \vee 1. \qquad (6)$$

В теоретическом плане задача (6) является классической задачей булевского программирования, где разработаны свои специфические методы решения [7]. Решение задачи может быть найдено также на основе методов целочисленного программирования [8]. Вне рассмотрения данной постановки осталась задача частичного обновления основания СОК.

Одной из наиболее сложных проблем, связанных с реализацией описанной выше процедуры обновления ключей закрытия данных с использованием СОК, является задача выбора параметров модели, в частности параметров функций $\varphi(P)$ и $f(P)$. Решение данной проблемы целесообразно реализовать на основе использования экспертных процедур.

Заключение

Предложенный алгоритм закрытия данных, размещаемых в памяти (прежде всего, оперативной) в процессе обработки микропроцессором, обесценивает эти данные для злоумышленника, поскольку не позволяет воспользоваться ими даже при их хищении. Практическая реализация предложенной процедуры предполагает наличие микропроцессоров, поддерживающих режим работы процедуры.

Основные результаты исследований.

1. Описаны процедуры закрытия данных при их передаче и раскрытия полученных данных при приеме в процессе обмена данными между различными компонентам компьютера, и прежде всего между процессором и оперативной памятью, основанные на использовании СОК.

2. Проведен анализ и сформирован набор параметров, значения которых в решающей степени влияют на эффективность обеспечения безопасности данных при обработке в компьютере с использованием СОК.

3. Сформулирована модель задачи выбора оптимального интервала обновления оснований СОК, решение которой позволит минимизировать суммарные издержки, связанные со злонамеренным хищением ключей закрытия данных, а также с затратами ресурсов на обновление ключей.

СПИСОК ЛИТЕРАТУРЫ

1. *Магомедов Ш. Г.* Использование системы остаточных классов для организации передачи данных морскими судами / Ш. Г. Магомедов // Вестн. Астрахан. гос. техн. ун-та. Сер.: Морская техника и технология. 2010. № 2. С. 44–46.

2. *Магомедов Ш. Г.* Вариант архитектуры защищенного микропроцессора на основе системы остаточных классов / Ш. Г. Магомедов // Прикаспийский журнал. Управление и высокие технологии. 2013. № 4. С. 118–125.

3. *Магомедов Ш. Г.* Алгоритм и схема сложения чисел в арифметико-логическом устройстве с использованием системы остаточных классов / Ш. Г. Магомедов // Вестн. Астрахан. гос. техн. ун-та. Сер.: Управление, вычислительная техника и информатика. 2014. № 1. С. 62–68.

4. *Кальнов М. И.* Защита сетевых коммуникаций в распределенной образовательной системе на основе интенсивной смены ключей шифрования. / М. И. Кальнов, В. В. Лаптев, Г. А. Попов // Вестн. Астрахан. гос. техн. ун-та. Сер.: Управление, вычислительная техника и информатика. 2012. № 2. С. 94–98.

5. *Червяков Н. И.* Модулярные параллельные вычислительные структуры нейропроцессорных систем / Н. И. Червяков, С. А. Ряднов, П. А. Сахнюк, А. В. Шапошников. М.: Физматлит, 2003. 288 с.

6. *Omondi A.* Advances in Computer Science and Engineering: Texts / A. Omondi, B. Premkumar. Vol. 2. Residue number system. Theory and Implementation. London, Imperial College Press, 2007. 296 p.

7. *Анри-Лобардер К.* Модели и методы исследования операций / К. Анри-Лобардер, А. Кофман. Т. 3. Целочисленное программирование. М.: Мир, 1977. 432 с.

8. *Схрейвер А.* Теория линейного и целочисленного программирования. Т. 2. М.: Мир, 1991. 342 с.

Статья поступила в редакцию 27.11.2014

ИНФОРМАЦИЯ ОБ АВТОРЕ

Магомедов Шамиль Гасангусейнович – Россия, 367015, Махачкала; Дагестанский государственный технический университет; канд. техн. наук; старший преподаватель кафедры «Программное обеспечение вычислительной техники и автоматизированных систем»; msgg@list.ru.

Sh. G. Magomedov

TRANSMISSION AND RECEPTION OF DATA IN COMPUTATIONAL DEVICES USING SYSTEMS OF RESIDUAL CLASSES

Abstract. Traditional technology of information security restricted usually involves the use of encryption methods. However, the requirements for such systems are quite strict, and as a result, in the implementation of a bulky and expensive to operate, which often makes it barely acceptable and cumbersome to use. We propose a different approach to secure communications between the processor and memory, comprising the following two features. The first – the use of residual classes as the basis of the closing data. Second – the approach to the organization of the exchange, precludes the use of specific subsystems within the defense. More specifically, the proposed approach to the protection of the transmitted data leaves no need for a specified center, thus depriving potential intruders of possibility to choose a dispatch center as an object of attack, and thus improving the security of data exchange. It is offered while exchanging data between the processor and memory, to assign the procedure of formation of keys directly on the participants in the data exchange (CPU and memory), but at the same time to produce a change of residual classes, which are used in the encryption process, faster than the minimum opening time of the selected keys. Thus, even if an attacker reveals the current parameters of the system of residual classes, they are turned out to be replaced by the others and in this race for the keys an attacker will always keep up with the information protection system. With regard to the system of protection based on the use of residual classes, this problem becomes even more urgent, as the level of resistance of the closing data, based on the system of residual classes, is below the resistance of many modern encryption systems.

Key words: residue number system, processor, memory, information security, computing devices, data security.

REFERENCES

1. Magomedov Sh. G. Ispol'zovanie sistemy ostatochnykh klassov dlia organizatsii peredachi dannykh morskimi sudami [Use of the system of residual classes for organization of data transmission at the marine vessels]. *Vestnik Astrakhanskogo gosudarstvennogo tekhnicheskogo universiteta. Seriia: Morskaia tekhnika i tekhnologiia*, 2010, no. 2, pp. 44–46.

2. Magomedov Sh. G. Variant arkhitektury zashchishchennogo mikroprotsessora na osnove sistemy ostatochnykh klassov [Type of architecture of the protected microprocessor based on the systems of residual classes]. *Prikaspiiskii zhurnal. Upravlenie i vysokie tekhnologii*, 2013, no. 4, pp. 118–125.

3. Magomedov Sh. G. Algoritm i skhema slozheniia chisel v arifmetiko-logicheskom ustroistve s ispol'zovaniem sistemy ostatochnykh klassov [Algorithm and scheme of addition of numbers in arithmetic-logical device using the system of residual classes]. *Vestnik Astrakhanskogo gosudarstvennogo tekhnicheskogo* universiteta. *Seriia: Upravlenie, vychislitel'naia tekhnika i informatika*, 2014, no. 1, pp. 62–68.

4. Kal'nov M. I., Laptev V. V., Popov G. A. Zashchita setevykh kommunikatsii v raspredelennoi obrazovatel'noi sisteme na osnove intensivnoi smeny kliuchei shifrovaniia [Protection of network communications in distributed educational system based on the intense change of encryption keys]. *Vestnik Astrakhanskogo gosudarstvennogo tekhnicheskogo universiteta. Seriia: Upravlenie, vychislitel'naia tekhnika i informatika*, 2012, no. 2, pp. 94–98.

5. Cherviakov N. I., Riadnov S. A., Sakhniuk P. A., Shaposhnikov A. V. *Moduliarnye parallel'nye vychislitel'nye struktury neiroprotsessornykh sistem* [Modular parallel computational structures of neuro-processor systems]. Moscow, Fizmatlit Publ., 2003. 288 p.

6. Omondi A., Premkumar B. *Advances in Computer Science and Engineering: Texts. Vol. 2. Residue number system. Theory and Implementation*. London, Imperial College Press, 2007. 296 p.

7. Anri-Lobarder K., Kofman A. *Modeli i metody issledovaniia operatsii. Vol. 3. Tselochislennoe programmirovanie* [Models and methods of the operational studies. Vol. 13. Integer programing]. Moscow, Mir Publ., 1977. 432 p.

8. Skhreiver A. *Teoriia lineinogo i tselochislennogo programmirovaniia* [Theory of linear and integer programming]. Vol. 2. Moscow, Mir Publ., 1991. 342 p.

The article submitted to the editors 27.11.2014

INFORMATION ABOUT THE AUTHOR

Magomedov Shamil Gasanguseinovich – Russia, 367015, Makhachkala; Dagestan State Technical University; Candidate of Technical Sciences; Senior Lecturer of the Department "Software of Computers and Automated Systems"; msgg@list.ru.