УДК 681.3.06

ИССЛЕДОВАНИЕ МЕТОДОВ КРИПТОАНАЛИЗА ПОТОЧНЫХ ШИФРОВ

д.т.н., проф. Стасев Ю.В., к.т.н. Потий А.В, Избенко Ю.А.

Одной из составляющих схем симметричного шифрования являются поточные шифры. В общем виде данные шифры состоят из одного или нескольких регистров сдвига и нелинейной функции. Схемы, состоящие из одного линейного рекуррентного регистра (ЛРР) и нелинейной функции, расположенной на выходе данного регистра, называются фильтр-генератором, схемы, в которых нелинейная функция комбинирует выходы нескольких ЛРР, называются комбинирующим генератором. Задачей нелинейных функций (НЛФ) является внесение нелинейности в выходную последовательность ЛРР, поскольку сам по себе ЛРР является линейным устройством. В классической интерпретации ЛРР представляется в виде последовательности битовых ячеек, а в качестве нелинейных функций используют булевы функции. Все арифметические операции выполняются в $GF(2)$.

Поскольку при разработке любых криптографических схем их стойкость определяется прежде всего стойкостью к известным на текущий момент криптографическим атакам, направленных на выявление в рассматриваемой схеме слабостей различного рода, представляется целесообразным рассмотрение наиболее распространенных и упоминаемых методов (алгоритмов) криптоанализа для выработки рекомендаций относительно конструирования криптографически стойких схем криптопреобразований.

При рассмотрении методов криптоанализа схем поточного шифрования все методы можно условно разделить на три класса: аналитические атаки, статистические и силовые. К аналитическим атакам относят атаки, в которых алгоритм построения атаки основан на аналитических принципах вскрытия криптосхемы. К статистическим атакам относятся атаки, основанные на оценке статистических свойств гаммы шифрующей. К силовым атакам относятся атаки, основанные на принципе полного перебора всех возможных комбинаций ключа; теоретически, при попытке вскрытия криптосхемы данный вид атаки должен быть наиболее эффективным по сравнению с остальными предлагаемыми видами атак.

Класс аналитических атак можно разбить на два подкласса: методы криптоанализа гаммы шифрующей и методы криптоанализа процедуры ключевой инициализации и реинициализации. В силу специфики принципов построения поточных шифров основным видом атак на данные схемы в первом подклассе являются корреляционные атаки, основная идея которых состоит в нахождении корреляции между гаммой шифра и различными линейными комбинациями ключа (регистра сдвига). В качестве объекта исследования корреляционные атаки рассматривают нелинейную функцию, вносящую нелинейность в выходную последовательность регистра сдвига – таким образом, каждый раз, в зависимости от устройства применяемой нелинейной функции, реализации корреляционных атак будут различны и будут основаны на специфическом устройстве анализируемой функции.

Класс статистических атак делится на два подкласса: методы криптоанализа статистических свойств гаммы шифрующей и методы криптоанализа сложности последовательности. Методы первого подкласса направлены на выявление возможного дисбаланса в выходной последовательности криптосхемы с целью нахождения способа предположения следующего бита выходной последовательности с вероятностью лучшей, чем при случайном выборе. Данные методы оперируют различными статистическими тестами, выбор необходимого и достаточного количества тестов – прерогатива криптоаналитика. Методы второго подкласса направлены на выявление возможности генерации последовательности, аналогичной гамме шифрующей, каким-либо другим способом, сложность реализации которого была бы меньшей по сравнению со способом генерации гаммы шифрующей; в идеале, найденный способ должен быть применимым на практике.

Данные методы используют концепции линейной сложности, профиля линейной сложности, квадратичного размаха.

При проведении криптоанализа схем подразумевается, что атака произошла успешно, если ее вычислительная сложность меньше, чем вычислительная сложность полного перебора всех ключевых комбинаций данного шифра.

Целью данной статьи является рассмотрение аналитических атак, выработка рекомендаций относительно конструирования криптографически стойких схем поточного шифрования.

Аналитические атаки

Все аналитические атаки происходят при допущении, что криптоаналитику известно описание генератора (образующие полиномы, вид нелинейного преобразования), он обладает открытым и соответствующим ему закрытым текстом. Атаки данного класса эквивалентны атакам по известному открытому тексту. Задачей криптоаналитика является определение применяемого ключа (начального заполнения). На рисунке 1 представлены наиболее известные криптоаналитические атаки, применяемые к синхронным поточным шифрам.

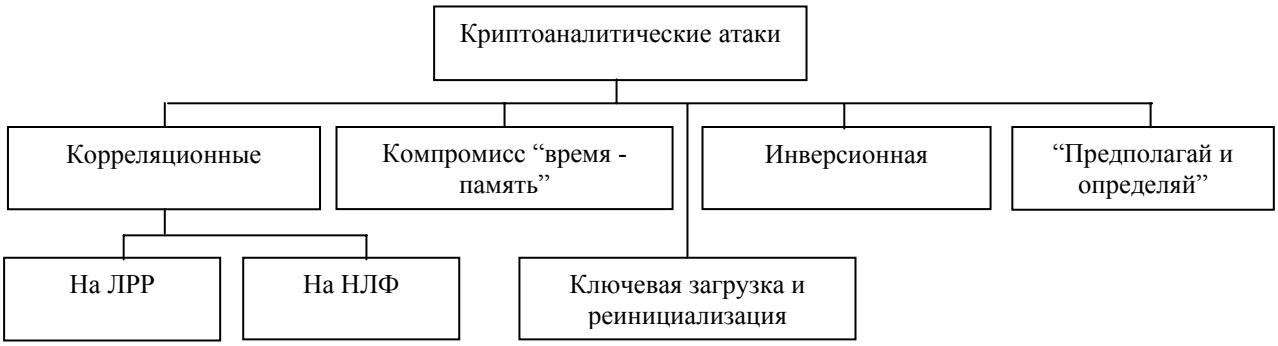


Рис. 1.

1. Корреляционные атаки

Наиболее распространенными и упоминаемыми атаками являются корреляционные атаки в силу специфики построения поточных шифров. В [1] показано, что если нелинейная функция пропускает на выход информацию о своих внутренних компонентах, то работа по вскрытию такой системы может быть существенно сокращена. Более того, такая функция существует всегда, даже если она задействует память. В силу данной аксиомы корреляционные атаки используют корреляцию выходной последовательности схемы шифрования с выходной последовательностью регистров для восстановления начального заполнения последних.

Среди атак данного класса можно выделить следующие атаки:

1. Базовые корреляционные атаки.
   - Базовая корреляционная атака Зигенталера;
   - Корреляционная атака Зигенталера.
2. Атаки, базирующиеся на низковесовых проверках четности.
   - Быстрая корреляционная атака Майера-Штаффельбаха;
   - Быстрая корреляционная атака Форре;
   - Быстрый итеративный алгоритм Михалевича-Голича;
   - Быстрая корреляционная атака Чепыжова-Смитса.
3. Атаки, базирующиеся на использовании конволюционных кодов.
4. Атаки, использующие технику турбо кодов.
5. Атаки, базирующиеся на восстановлении линейных полиномов.

6. Быстрая корреляционная атака Чепыжова, Йоханссона, Смитса;

Под быстрыми корреляционными атаками подразумеваются атаки, вычислительная сложность которых значительно меньше вычислительной сложности силовых атак.

Бóльшая часть рассматриваемых атак основана на некотором итерационном исправляющем ошибке алгоритме. Основываясь на допущении, что выходная последовательность генератора $\{z\}$ коррелирует с выходной последовательностью регистра $\{x\}$, для описания и исследования корреляции моделируется поведение выходной последовательности регистра как последовательности, проходящей через некоторый канал. В качестве модели, имитирующей прохождение, рассматривается двоичный симметричный канал (ДСК) с некоторой вероятностью корреляции $1 - p$, где $1 - p = P(x_i = z_i)$, p определено как вероятность перехода (вероятность ошибки) в ДСК, полагается $p = 1/2 - \delta$. Проблема криптоанализа рассматривается как проблема декодирования некоторого кода с присутствующим в ДСК сильным шумом. Длина регистра полагается ℓ, длина перехваченной последовательности – N. Обобщенная схема корреляционной атаки для восстановления начального состояния представлена на рисунке 2.

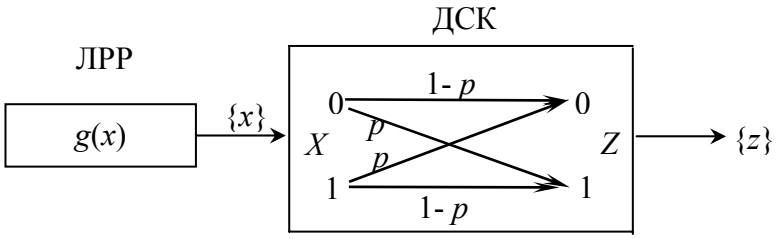


Рис.2.

1.1. Базовые корреляционные атаки

Базовая корреляционная атака Зигенталера [1]. Данная атака именуется в литературе также как “разделяй и вскрывай”, применима к комбинирующим генераторам и построена на расстоянии Хэмминга между двумя двоичными последовательностями одинаковой длины. Суть алгоритма основана на идее, что всякая комбинирующая функция f в той или иной мере выдает в гамму шифра $\{z\}$ некоторую информацию о последовательности $\{x^{(j)}\}$, сгенерированной j-тым регистром сдвига. Для выделения воздействия j-го ЛРР на гамму шифра $\{z\}$ моделируется часть генератора как двоичный симметричный канал (ДСК).

Алгоритм атаки состоит из двух фаз: фазы вычисления корреляционной ("утекающей") вероятности $p_j = P(z=x^{(j)})$ исходя из комбинирующей функции f и фазы перебора начальных заполнений регистра. В фазе перебора посредством вычисления функции кросс-корреляции

$$C_{\{x^{(j)}\},\{z\}}(d) = \frac{1}{n}\sum_{i=1}^{n}(-1)^{z_i}(-1)^{x_{i+d}^{(j)}} \qquad (1)$$

и последующего сравнения с некоторым пороговым значением T определяется наличие корреляции данного фрагмента гаммы z с соответствующим фрагментом из $x_d^{(j)}$. В случае успешного прохождения теста d_j считается верной фазой и осуществляется переход к j+1 - му ЛРР, иначе переходим к $d = d + 1$, $d = 1\ldots 2^{L_j}$. Выходом алгоритма является набор начальных состояний $\{x_0^{(j)}\}$ для ЛРР, выдающих информацию в гамму.

Пороговое значение T и необходимое для успешного криптоанализа количество бит гаммы n определяется исходя из следующих соображений. Пусть $P_f = P(C_{\{x^{(j)}\},\{z\}}(d) \geq T \mid$ неверная фаза) определена как вероятность “ложной тревоги”, и пусть $P_m = P(C_{\{x^{(j)}\},\{z\}}(d) < T \mid$ верная фаза) определена как вероятность пропуска истинного начального заполнения. Тогда T и n следуют из вероятности P_m, которая выбирается свободно в соответствии с тем риском,

на который можно пойти в пропуске верной фазы, например P_m=0,01, и вероятности P_f, которую рекомендуется выбирать как $P_f = 2^{-L}$.

Вычислительная сложность алгоритма составляет в среднем $O(\sum_{j=1}^{N} 2^{L_j})$, что выглядит намного привлекательней по сравнению со сложностью $O(\prod_{j=1}^{N} 2^{L_j})$, требующейся при поиске ключа тотальным перебором. Алгоритм применим лишь при $L_j \leq 50$ и в случаях, когда нелинейный комбинирующий узел f допускает утечку информации об отдельных входных переменных.

Корреляционная атака Зигенталера [2]. Суть данной атаки состоит в рассмотрении некоторого фильтр-генератора (рисунок 3), генерирующего гамму шифрующую z_k, и нахождении эквивалентной системы, которая бы генерировала аналогичную выходную последовательность при условии, что криптоаналитику известен примитивный образующий полином. Количество точек съема n, позиции ячеек точек съема i_1, i_2, …,i_n, вид нелинейной функции f и начальное заполнение считаются неизвестными. Данная атака позволяет представить любую криптосхему в виде ее эквивалента и по сути своей является универсальной атакой на криптосистемы различного рода.

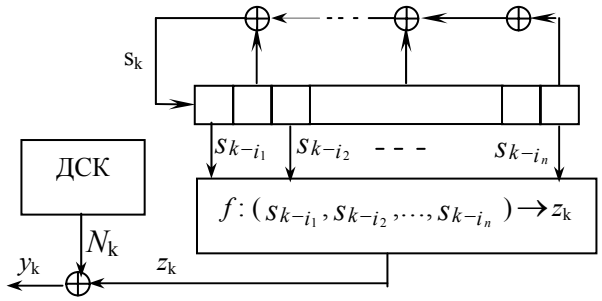


Рис.3. Схема фильтр-генератора

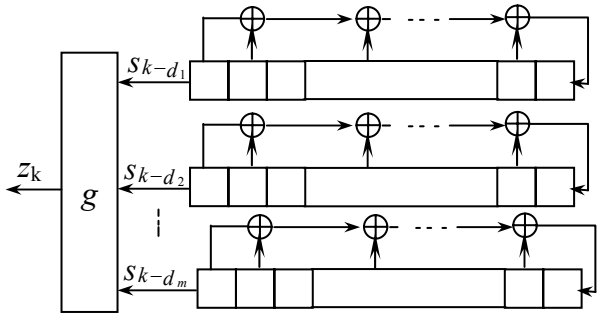


Рис.4.Схема эквивалентной системы

Знание образующего полинома позволяет криптоаналитику предположить эквивалентную систему, представленную на рисунке 4. Такие эквивалентные системы будут существовать всегда. Данная схема будет содержать m ЛРР, построенных согласно известному полиному, с различными начальными состояниями, $1 \leq m \leq n$; n фаз, снимаемых функцией f, загружаются в отдельные регистры, полагается $g = f$. При анализе схемы рассматривается функция кросс-корреляции

$$C_{s,z}(d) = \frac{1}{T}\sum_{i=0}^{T}(-1)^{z_k}(-1)^{s_{k-d}} \quad \text{для } d = 0, 1, \ldots, p\text{-}1 \qquad (2)$$

между последовательностями $\{s_k\}$ и $\{z_k\}$.

Вследствие использования алгоритмом корреляции со значительной частью периода ЛРР метод применим при $L \leq 50$.

Как видно из приведенных атак, они являются малопригодными для практической реализации в силу накладываемых ограничений на длину регистра.

Атаки, базирующиеся на низковесовых проверках четности

Быстрая корреляционная атака Майера и Штаффельбаха [3]. Атака данного типа применима как к фильтр-генераторам, так и комбинирующим генераторам и является базовой для всех остальных быстрых корреляционных атак данного типа. Идея атаки базируется на использовании уравнений проверки четности для полинома обратной связи линейного регистра. Авторами представлено два алгоритма, A и B, для восстановления начального заполнения регистров по выходной последовательности генератора.

Для восстановления ЛРР-последовательности $\{x\}$ из гаммы шифрующей $\{z\}$ используется следующий принцип: каждый бит x_n из $\{x\}$ удовлетворяет нескольким линейным зависимостям, основанных на принципе обратной связи регистра, данные зависимости обусловлены другими t битами последовательности $\{x\}$, где t – количество ненулевых элементов в полиноме. Подстановкой соответствующих бит из $\{z\}$ в эти зависимости вычисляются уравнения для каждого бита z_n. Уравнения проверки четности для каждого бита x_n и соответствующего ему бита гаммы z_n записываются как

$$\begin{array}{rcrcl} x_n & + & b_1 & = & 0, \\ x_n & + & b_2 & = & 0, \\ & & \vdots & & \\ x_n & + & b_m & = & 0, \end{array} \quad \text{и} \quad \begin{array}{rcrcl} z_n & + & y_1 & = & L_1, \\ z_n & + & y_2 & = & L_2, \\ & & \vdots & & \\ z_n & + & y_m & = & L_m, \end{array}$$

соответственно, где b_j и y_j – сумма t различных символов в ЛРР последовательности и последовательности гаммы соответственно, $L_i \in GF(2)$, $1 \le i \le m$.

Для проверки того, равно ли $z_n = x_n$, высчитывается количество всех зависимостей, которые содержат z_n, после чего отбираются зависимости, в которых вероятность того, что $z_n = a_n$, выше, чем в других зависимостях, и ищется корреляция со всеми фазами, близкими к этим битам.

Алгоритм A имеет вычислительную сложность $O(2^{c\ell})$, $c<1$, где $c(p,t,N/\ell)$ – функция от p, t и N/ℓ, и эффективен при малых t и $p \le 0.75$; при $t \ge 16$ следует $c=1$, вследствие чего алгоритм имеет малое преимущество над исчерпывающим поиском.

Алгоритм B является модификацией алгоритма A, является полиномиальным и имеет вычислительную сложность $O(\ell)$ при фиксированных p, t и N/ℓ, и эффективен при малых t и $p \approx 0{,}5$. Ожидаемое количество уравнений проверок четности $E[m]$

$$E[m] \approx (t+1)\log_2\left(\frac{N}{2l}\right). \tag{3}$$

Вероятность p^* того, что $x_n \ne z_n$ для h уравнений из m, определяется как

$$p^* = \frac{p \cdot s^{m-h} \cdot (1-s)^h}{p \cdot s^{m-h} \cdot (1-s)^h + (1-p)(1-s)^{m-h} s^h} \tag{4}$$

где $s = P(y_j = b_j) = 1/2 + 2^{t-1}\delta^t$ - вероятность того, что $y_j = b_j$.

Использование алгоритма A целесообразно при $c<<1$ и $p \approx 0{,}75$, алгоритм B является более эффективным при $p \approx 0{,}5$. Алгоритмы применимы при малом количестве обратных связей в образующем полиноме ЛРР. При использовании большого количества ячеек обратных связей алгоритмы неприменимы в силу того, что уравнения проверки четности будут давать очень малую исправляющую коррекцию: увеличение p^* влечет уменьшение $(1 - p^*)$ – корректирующей способности алгоритма. Другими словами, максимальная вероятность корреляции $1-p$, при которой алгоритмы будут эффективны, будет значительно ниже, если ЛРР будут иметь большое количество ячеек обратной связи (т.е. около $\ell/2$). Для противостояния атакам данного класса рекомендуется соблюдение двух условий: избежание корреляции последовательности регистра с менее чем 10 точками съема и выбор ЛРР длиной не менее 100 бит.

Быстрая корреляционная атака Форре [4]. Алгоритм является модификацией алгоритма A Майера и Штаффельбаха [3] и использует идею рассмотрения кросс-корреляции последовательностей, предложенную Зигенталером [2]. Предложенная модификация

ориентирована на применение к фильтр-генераторам, поскольку доказано [4], что применение алгоритма A к фильтр-генераторам не гарантирует успеха.

Имея в качестве базового алгоритма алгоритм [3], предлагается после присвоения каждому биту z_n вероятности p_n соответствия z_n и x_n выбирать не k наиболее вероятных бит согласно базовому алгоритму, а некоторое множество S из M высоко вероятных бит z_n, $M>k$. Далее случайным выбором k бит из множества S принимается гипотеза H_0 и вычисляется соответствующее начальное состояние ЛРР, после чего тестируются модификации гипотезы H_0 с весом Хэмминга 0,1,2,… корреляцией ЛРР-последовательностей с последовательностью $\{z\}$ и запоминаются начальные состояния, дающие достаточно высокую корреляцию. После определения подмножества линейно независимых начальных состояний ЛРР предлагается согласно [2] построить эквивалентную фильтр-генератору систему.

Ограничения на атаку накладывают количество ячеек обратной связи регистра (чем больше ячеек задействовано, тем большее количество бит привлечено в каждую линейную зависимость и тем меньшая надежность назначенных вероятностей p_n соответствия z_n и x_n), значения пиков корреляции рассматриваемых последовательностей и количество бит в множестве S. В качестве добавочных критериев при конструировании фильтр-генераторов рекомендуется использовать регистры с числом точек съема не менее 10 (аналогично [3]) и выбирать нелинейную функцию f таким образом, чтобы пики кросс-корреляции между последовательностью генератора $\{z\}$ и ЛРР-последовательностью $\{x\}$ имели значения менее чем 75%.

Быстрая корреляционная атака Михалевича-Голича [5]. Алгоритм основан на итеративном декодирующем методе с нахождением низковесовых проверок четности и улучшает результаты Майера-Штаффельбаха. Алгоритм базируется на матричном представлении ЛРР. Полагаются известными отрезок гаммы шифрующей длиной N, длина ЛРР ℓ и вид образующего полинома $p(x) = c_0 + c_1x + c_2x^2 + \ldots + c_\ell x^\ell$. Пусть u_N является состоянием ЛРР в момент времени N, т.е. $u_N = (u_{N+1}, u_{N+2}, \ldots, u_{N+\ell})$. Тогда существует $\ell \times \ell$ матрица A

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & \ldots & 0 \\ 0 & 0 & 1 & 0 & \ldots & 0 \\ \ldots & \ddots & \ddots & \ldots & \ldots & \ldots \\ 0 & \ldots & \ldots & 0 & 0 & 1 \\ c_l & c_{l-1} & \cdots & c_3 & c_2 & c_1 \end{pmatrix},$$

такая, что состояние в момент времени N может быть выражено как $u_N = A^j u_{N-j}^T$. Следовательно, взятием всех степеней A^j, $j = 1, \ldots, N$ получаем множество из N уравнений проверок четности всех улучшенных u_{N+1}. Сохраняем все ортогональные уравнения проверки четности с весом t. Ожидаемое количество уравнений проверок четности $E[m]$

$$E[m] = \frac{N}{l \cdot 2^l} \cdot \binom{l}{t}. \qquad (5)$$

Проблемой является необходимость наличия большого значения N для нахождения достаточного количества уравнений при увеличении ℓ. Так, для нахождения 100 уравнений с весом 3 при длине регистра 64 требуется $N \approx 2^{61}$ бит.

Быстрая корреляционная атака Чепыжова и Смитса [6]. Предложенный метод базируется на известной проблеме в теории кодирования. Известно, что выходная последовательность ЛРР может рассматриваться как кодовое слово из $[\ell, N]$ линейного кода $\hat{C}$. Существует такой код $\hat{H}$, что

$$c \cdot h^T = 0, \qquad \forall c \in \hat{C},\ h \in \hat{H}.$$

Код $\hat{H}$ называется кодом, двойственным к $\hat{C}$. Из приведенного выражения видно, что нахождение уравнений проверок четности с низким весом эквивалентно нахождению кодового слова в $\hat{H}$ с низким весом. Предлагается использовать декодирующее множество информации для нахождения кодового слова в $\hat{H}$ низкого веса. Вычислительная сложность алгоритма составляет $O(m \cdot 2^{(1-r/n)r} + (n H^{-1}(1-R)+n) m^2)$, где n – длина кода, r – степень образующего полинома, m – количество уравнений проверки четности, R – оценка кода, $H(p)$ является бинарной функцией энтропии $H(p) = -p \log_2 p - (1-p)\log_2(1-p)$. Атака имеет высокую вычислительную сложность для ЛРР с образующим полиномом высокой степени и применима к низковесовым полиномам малой степени.

1.3. Атаки, основанные на конволюционных кодах

Основным препятствием при практической реализации выше перечисленных атак состоит в невозможности их применения к регистрам произвольной длины. Нижеописанные атаки позволяют производить криптоанализ схем, основанных на регистрах произвольной длины с произвольным количеством точек съема.

В [7] представлен алгоритм, основанный на преобразовании части кода $\hat{C}$, полученного из последовательности ЛРР, в конволюционный код с использованием более передового алгоритма декодирования, использующего память, при этом сложность декодирования является низкой. Алгоритм состоит из двух фаз. В первой фазе алгоритм преобразовывает часть кода $\hat{C}$ из последовательности ЛРР в конволюционный код. Кодер этого конволюционного кода создан нахождением подходящих уравнений проверки четности кода $\hat{C}$. Во второй фазе осуществляется декодирование посредством кода Витерби.

Анализ алгоритма показывает, что ожидаемое число уравнений проверки четности составит

$$E[m] \approx \frac{\binom{N-l-B}{t}}{2^{l-B}} \approx \frac{N^t}{t!} 2^{B-l}. \qquad (6)$$

Кроме того, с вероятностью $1 - p_e$, $p_e < 1$, атака является успешной, если криптоаналитик располагает последовательностью длиной

$$N \geq \frac{1}{4\delta^2}(4(1+d)\ln 2 \cdot t!)^{\frac{1}{t}} \cdot 2^{\frac{l-B}{t}}, \qquad (7)$$

где корреляционная вероятность $P(x_i = z_i) = 1/2 + \delta$,

$$p_e \leq l \frac{1}{2^d - 1} 2^{-(1+d)B}, \qquad (8)$$

и d – любое фиксированное положительное целое, B обозначает фиксированный размер памяти, t – некоторое число символов (обычно $t = 2$ или $t = 3$).

Вычислительная сложность первой фазы алгоритма, составляющей уравнения проверки четности, составляет $O(\frac{N^t}{t!} 2^{B-l})$ при резервировании памяти размером $O(N^{\lceil t/4 \rceil})$. Вычислительная сложность фазы декодирования составляет $O(\ell \cdot m \cdot 2^B)$ при резервировании

памяти размером $O(m2^B)$. Общая вычислительная сложность алгоритма составляет $O(l\frac{N^t}{t!}2^{2B-l})$.

При приемлемой сложности атаки существенным недостатком алгоритма являются высокие требования к памяти при использовании регистров значительной длины – обе фазы атаки требуют значительные объемы памяти.

1.4. Атаки, использующие технику турбо кодов

С целью улучшения вышеописанной атаки представлен метод уменьшения оценки конволюционного кода без увеличения памяти на основе алгоритма, использующего технику турбо кодов [8]. Основным принципом использования турбо кодов является параллельное использование двух конволюционных кодов с некоторым видом преобразований между ними. Информационные символы поступают на вход обоих кодеров. Перед поступлением последовательности информационных символов на вход второго кодера выполняется некоторое перемешивание π для повышения вероятности обнаружения ошибки. Для декодирования турбо кодов используется итеративный декодирующий алгоритм. Декодирующий алгоритм является условно оптимальным декодирующим алгоритмом, основанным на апостериорной вероятности декодирования составных кодов. Временная сложность декодирующего алгоритма для турбо кодов является точно такой же, как и максимальная временная сложность для декодирующих составных кодов. Следовательно, имеется возможность уменьшить оценку без увеличения временной сложности декодирующего алгоритма.

Предложенный алгоритм является синтезом двух вышеописанных алгоритмов: первая фаза использует конволюционный код, вторая основана на идее алгоритма *B* Майера-Штаффельбаха и использует апостериорную вероятность декодирующим алгоритмом для обеспечения апостериорной вероятности для каждого символа в определенной части принятой последовательности. Достоинством данного алгоритма по сравнению с предыдущим состоит в увеличении возможности успешного декодирования, при росте используемого количества кодов, символов при наличии более высокой вероятности ошибки в ДСК.

При фиксированных параметрах используемой памяти и гаммы турбо алгоритм имеет (грубо) аналогичные требования к памяти при увеличении вычислительной сложности по меньшей мере в $6M$ раз, где M – количество используемых кодов. В целом, при уменьшении временной сложности, выполнение алгоритма, как и в предыдущем случае, не ограничивается вычислительной сложностью, но существенным недостатком алгоритма являются высокие требования к памяти.

1.5.Атаки, базирующиеся на восстановлении линейных полиномов

Основным препятствием к широкому применению быстрых корреляционных атак, основанных на конволюционных кодах, является большой объем памяти, требующийся в процессе декодирования. Предшествующий алгоритм, основанный на применении сразу нескольких конволюционных кодов с целью ослабления требований к требуемому объему памяти, по прежнему требует слишком много памяти для того, чтобы быть применимым к длинным регистрам. Ограничения по памяти, налагаемые упомянутыми алгоритмами, послужили толчком к нахождению новых быстрых корреляционных атак. В [9] был представлен метод, рассматривающий проблему криптоанализа не как проблему декодирования в двоичном симметричном канале [1-8], а как проблему изучения линейного многочлена от нескольких переменных. Достоинством данного алгоритма являются очень низкие затраты памяти в декодирующей фазе.

Полагается, что криптоаналитику известен некоторый отрезок гаммы шифрующей длиной N бит $z_1, z_2, \ldots, z_N$, образующий полином $g(x)$ и регистр длиной ℓ бит. Неизвестное

начальное состояние ЛРР полагается как $u_0 = (u_1, u_2, \ldots, u_l)$. Известно, что существует $\ell \times N$ матрица, $G_{ЛРР}$, вида

$$G_{ЛРР} = (h_0 \quad h_1 \quad \ldots \quad h_N), \tag{9}$$

такая, что $u_i = u_0 h_i$. Следовательно, каждое u_i можем выразить как известную линейную комбинацию начального состояния u_0

$$u_i = h_{i1}u_1 + h_{i2}u_2 + \ldots + h_{i\ell}u_\ell \tag{10}$$

где h_{ij} является элементом j-го столбца i-й строки матрицы $G_{ЛРР}$.

Начальное состояние полинома обозначается как

$$U_0(x) = U_0(x_1, x_2, \ldots, x_\ell) = u_1x_1 + u_2x_2 + \ldots + u_\ell x_\ell \,. \tag{11}$$

Введением переменных $x_1, x_2, \ldots, x_N$, где $x_i = h_i^T$, $1 \le i \le N$, т.е. $x_i = (h_{i1}, h_{i2}, \ldots, h_{i\ell})$, можем выразить каждое u_i как начальное состояние полинома в некоторой точке x_i: $u_i = U_0(x_i)$, $i \ge 1$. Корреляция между u_i и z_i описывается введением вектора шума $e = (e_1, e_2, \ldots, e_N)$, где $e_i \in GF(2)$ является независимой случайной величиной для $1 \le i \le N$ и $P(e_i = 0) = 1/2 + \varepsilon$. В конечном виде модель корреляции $z = u + e$ имеет вид

$$z = (U_0(x_1) + e_1, U_0(x_2) + e_2, \ldots, U_0(x_N) + e_N), \tag{12}$$

где x_i известный ℓ - кортеж для всех $1 \le i \le N$. Таким образом, выходной вектор z состоит из некоторого количества зашумленных составляющих неизвестного полинома $U(x)$, рассматриваемых в точках $\{x_1, x_2, \ldots, x_N\}$. Задачей криптоаналитика является определение неизвестного полинома $U(x)$.

Суть алгоритма состоит в следующем. В предвычислительной фазе выбираются n различных (ℓ - B)-кортежей $s_1, s_2, \ldots, s_n$ и для каждого s_i принимается гипотеза, что ($u_1, u_2, \ldots, u_B$) бит являются начальным состоянием. Выбором определенного (ℓ - B)-кортежа s_i и нахождением всех линейных комбинаций из t векторов в $\{x_1, x_2, \ldots, x_N\}$,

$$\hat{x}(i) = \sum_{j=1}^{t} x_{a_j}, \tag{13}$$

имеющих специальный вид $\hat{x}(i) = (\hat{x}_1, \ldots, \hat{x}_B, s_i)$ для произвольных величин $\hat{x}_1, \ldots, \hat{x}_B$, осуществляется проверка того, является ли значение $(\hat{u}_1, \ldots, \hat{u}_B)$ корректным или нет. Аналогично для каждого s_i находятся все $\hat{z}(i) = \sum_{j=1}^{t} z_{a_j}$. Пусть множество таких пар равно S_i.

Ожидаемое количество линейных комбинаций составляет

$$E[S^{(k)}] = \frac{\binom{N}{t}}{2^{l-B}} \approx \frac{N^t}{t!}2^{B-l} \tag{14}$$

Поскольку взаимосвязь между $U(\hat{x}(i))$ и $\hat{z}(i)$ может быть записана как $U(\hat{x}(i)) = \hat{z}(i) + e$, можем данное выражение записать как

$$\sum_{j=1}^{B}(u_j + \hat{u}_j)\hat{x}_j + \sum_{j=B+1}^{l} u_j s_j + e = \sum_{j=1}^{B} \hat{u}_j \hat{x}_j + \hat{z}(i) \tag{15}$$

Заметим, что величина $W = \sum_{j=B+1}^{l} u_j s_j$ в (15) является фиксированной бинарной случайной величиной для всех линейных комбинаций специального вида, т.е. $W = 0$ либо $W = 1$ для всех $\hat{x}(i)$. Далее для всех 2^B значений $(u_1, \ldots, u_B) = (\hat{u}_1,...,\hat{u}_B)$ делаем следующее. Для каждого s_i, имея пары $\{\hat{x}(i), \hat{z}(i)\}$, вычисляем количество пар, для которых

$$\sum_{j=1}^{B} \hat{u}_j \hat{x}_j = \hat{z}(i), \tag{16}$$

и обозначаем его как *num*. Обновляем *dist*←*dist* + $(S_i - 2num)^2$. Если текущее *dist* имеет большее значение, чем предыдущее, $(\hat{u}_1,...,\hat{u}_B)$ запоминается и *dist*←0.Выходом алгоритма является $(\hat{u}_1,...,\hat{u}_B)$, имеющее наибольшее значение *dist.*

Сложность предвычислительной фазы алгоритма составляет $O(\frac{N^t}{t!}2^{B-l})$ при использовании $O(N^{\lfloor t/2 \rfloor})$ памяти. Вычислительная сложность фазы декодирования составляет $O(n \cdot 2^B)$ при резервировании памяти размером $O(2^B)$.Общая вычислительная сложность алгоритма может быть представлена как $O(n\frac{N^t}{t!}2^{2B-l})$.

1.6. Быстрая корреляционная атака Чепыжова, Йоханссона, Смитса

Проблема криптоанализа [10] рассматривается как проблема декодирования в двоичном симметричном канале [1-8] (рис.2.) и моделируется как декодирование случайного (N, ℓ) линейного кода. Алгоритм вычисляет [m, B] линейный код, ассоциируемый с искомым ЛРР, где $1 < B < \ell$. Далее посредством *ML*-декодирования этого нового кода восстанавливаются первые B символов начального состояния ЛРР, после чего восстанавливаются остальные биты.

В предвычислительной фазе используется улучшенный метод нахождения всех m уравнений проверки четности, имеющий вид

$$\sum_{j=1}^{B} a_j u_j = u_{i_1} + u_{i_2} + \ldots + u_{i_t}. \tag{17}$$

Множество всех уравнений проверки четности имеет вид

$$\begin{aligned} &\textstyle\sum_{j=1}^{B} a_j^{(1)} u_j = u_{i_1^{(1)}} + u_{i_2^{(1)}} + \ldots + u_{i_t^{(1)}}, \\ &\textstyle\sum_{j=1}^{B} a_j^{(2)} u_j = u_{i_1^{(2)}} + u_{i_2^{(2)}} + \ldots + u_{i_t^{(2)}}, \\ &\qquad\qquad \ldots \\ &\textstyle\sum_{j=1}^{B} a_j^{(m)} u_j = u_{i_1^{(m)}} + u_{i_2^{(m)}} + \ldots + u_{i_t^{(m)}}. \end{aligned} \tag{18}$$

На основе множества из m составленных уравнений формируется матрица G_2

$$G_2 = \begin{pmatrix} a_1^{(1)} & a_1^{(2)} & \dots & a_1^{(m)} \\ a_2^{(1)} & a_2^{(2)} & \dots & a_2^{(m)} \\ \dots & \dots & \dots & \dots \\ a_B^{(1)} & a_B^{(2)} & \dots & a_B^{(m)} \end{pmatrix},$$

называемая матрицей генерации [*m, B*] линейного кода C_2, первые B символов начального состояния (u_1, …, u_B) являются информационными символами. Посредством декодирующего алгоритма вычисляется (u_1, …, u_B) · G_2 и вычисляется расстояние Хэмминга к принятому слову (z_1, z_2, …, z_m). Выходными информационными символами являются символы с наилучшим принятым вектором. Представлено два алгоритма атаки.

С заданными B, ℓ, ε для алгоритма $A1$ и B, ℓ, ε, t для алгоритма $A2$ требуемая длина N перехваченной последовательности $\{z\}$ должна быть

$$N \approx 1/2 \cdot \sqrt{B \cdot (\ln 2)} \cdot \varepsilon^{-2} \cdot 2^{(l-B)/2} \quad \text{и} \tag{19}$$

$$N \approx 1/4 \cdot (2B \cdot t! \cdot \ln 2)^{1/t} \cdot \varepsilon^{-2} \cdot 2^{(l-B)/t}, \; N >> n_t \tag{20}$$

соответственно. Вычислительные сложности алгоритмов $A1$ и $A2$ составляют O(N log N) + O($2^B \cdot B \cdot \frac{\log_2 2}{8e^4}$) и O($N$ log N) + O($2^B \cdot B \cdot \frac{2\log_2 2}{(2e)^{2t}}$) соответственно. Требуемая память составляет $n_2(B + 2\log_2 N)$ и $n_t(B + t\log_2 N)$ бит соответственно, где n_2, n_t – длина кодового слова. Общая вычислительная сложность алгоритма может быть представлена как $O(\frac{N^t}{t!} 2^{2B-l})$.

В таблице 1 представлены сравнительные характеристики атак, применимых к регистрам произвольной длины (АКК - атаки, базирующиеся на конволюционных кодах, АЛП - атаки, базирующиеся на восстановлении линейных полиномов, АЧЙС - быстрая корреляционная атака Чепыжова, Йоханссона, Смитса; атака, построенная на турбо алгоритме, не рассматривается в силу тождественности конволюционным кодам).

Таблица 1.

Атаки	Вычислительная сложность		Требуемая память		Сложность алгоритма	Требуемая гамма
	Предвычисл. фаза	Фаза декодирования	Предвычисл. фаза	Фаза декодирования		
АКК	$\frac{N^t}{t!} 2^{B-l}$	$\ell \cdot m \cdot 2^B$	$N^{\lceil t/4 \rceil}$	$m2^B$	$l\frac{N^t}{t!} 2^{2B-l}$	$N \geq \frac{1}{4\delta^2}(4(1+d)\ln 2 \cdot t!)^{\frac{1}{t}} \cdot 2^{\frac{l-B}{t}}$
АЛП	$\frac{N^t}{t!} 2^{B-l}$	$n \cdot 2^B$	$N^{\lfloor t/2 \rfloor}$	≈ АЧЙС	$n\frac{N^t}{t!} 2^{2B-l}$	не приведена
АЧЙС	$\frac{N^t}{t!} 2^{B-l}$	2^B	$N^{\lfloor (t-1)/2 \rfloor}$	$n_t(B + t\log_2 N)$	$\frac{N^t}{t!} 2^{2B-l}$	$N \approx 1/4 \cdot (2B \cdot t! \cdot \ln 2)^{1/t} \cdot \varepsilon^{-2} \cdot 2^{(l-B)/}$

Как видно из приведенной таблицы, вычислительная сложность алгоритмов является приблизительно одинаковой, различие составляют объемы памяти, необходимые для фазы декодирования. Высокие требования к памяти в фазе декодирования делают алгоритм, основанный на конволюционных кодах (турбо кодах), практически малопригодным и ограничивают размер памяти величиной B = 20-30, поскольку декодирующий алгоритм Витерби имеет 2^B состояний. Атаки, базирующиеся на восстановлении линейных полиномов и быстрая корреляционная атака Чепыжова, Йоханссона, Смитса, имеют низкие и

приблизительно одинаковые затраты объемов требуемой памяти и при увеличении B остаются постоянными; размеры требуемой памяти пропорциональны количеству уравнений проверки четности, используемых в алгоритмах декодирования. В силу накладываемых ограничений на объем используемой памяти в фазе декодирования АКК данный алгоритм нельзя признать практичным для регистров с большими длинами – при $\ell = 100$ требуемая память является недостижимой, вследствие чего при наименьшей вычислительной сложности, низких требованиях к памяти и простоте алгоритма наиболее оптимальным следует считать АЧЙС.

При проведении криптоанализа особый интерес представляет определение нижней границы длины гаммы шифрующей, необходимой для успешного проведения криптоанализа. Анализируя выражение (20), можно констатировать, что для АЧЙС при известной корреляции $P(x_i = z_i)$ параметрами, влияющими на N, являются память B и t. Манипулируя B и t, добиваются необходимого значения N: при увеличении обоих параметров значение N уменьшается. Согласно [7-10], при проведении криптоанализа используют значения $t = 2$ и $t = 3$, $6 \le B \le 35$. Поскольку минимальная длина применяемых на практике регистров должна составлять не менее 100 бит, произведем теоретический расчет параметров для данной длины регистра. В таблице 2 представлены расчетные данные для $t = 2$, таблице 3 – для $t = 3$ при $\ell =$ 100 бит, $p =$ 0.46875, где $E(n_2)$, $E(n_3)$– ожидаемое количество n_2, n_3.

Таблица 2

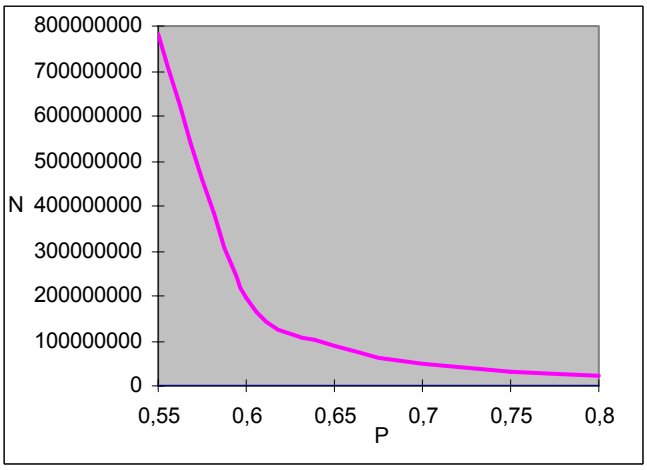
B	N, Гбайт	$E(n_2)$	Память, Гбайт
6	$2^{57} \approx 134217728$	524288	0,06
10	$2^{55,4} \approx 44171264$	912838	0,1
15	$2^{53,2} \approx 9563208$	1383604	0,16
20	$2^{50,8} \approx 1951744$	1589344	0,18
25	$2^{48,5} \approx 385813$	2097152	0,24
30	$2^{46,1} \approx 74688$	2408995	0,27
35	$2^{43,8} \approx 14266$	3178688	0,36

Таблица 3

B	N, Гбайт	$E(n_3)$	Память, Гбайт
6	$2^{41,2} \approx 2352$	135624022	16,37
10	$2^{40,1} \approx 1101$	220321874	26,74
15	$2^{38,6} \approx 387$	311582183	37,95
20	$2^{37} \approx 131$	357913941	43,66
25	$2^{35,5} \approx 46,5$	506166749	61,99
30	$2^{33,9} \approx 15,2$	581432913	71,32
35	$2^{32,3} \approx 4,92$	667891030	82,04

Как отмечается в [9], восстановление начального заполнения регистра при длине 80-100 бит при $30 \le B \le 35$, $t = 3{,}4$ посредством данного алгоритма может занять несколько недель при параллельной работе нескольких PC.

Рассмотрим поведение атаки при увеличении корреляции: пусть $P_1(x_i = z_i) = 0{,}55$, $P_2(x_i = z_i) = 0{,}6$, $P_3(x_i = z_i) = 0{,}65$, $P_4(x_i = z_i) = 0{,}7$, $P_5(x_i = z_i) = 0{,}75$, $P_6(x_i = z_i) = 0{,}8$. Соответственно вероятности ошибок будут составлять $p_1 = 0.45$, $p_2 = 0.4$, $p_3 = 0.35$, $p_4 = 0.3$, $p_5 = 0.25$, $p_6 = 0.2$. Выражение (20) определяет взаимосвязь длины перехваченной последовательности, вероятности корреляции и уровня вычислительной сложности. На рисунках 5, 6 представлена зависимость необходимой для успешного криптоанализа длины гаммы шифрующей от вероятности корреляции при $\ell = 60$, $t = 2$ и различных $B_1 = 20$ (рис.5), $B_2 = 30$ (рис.6).



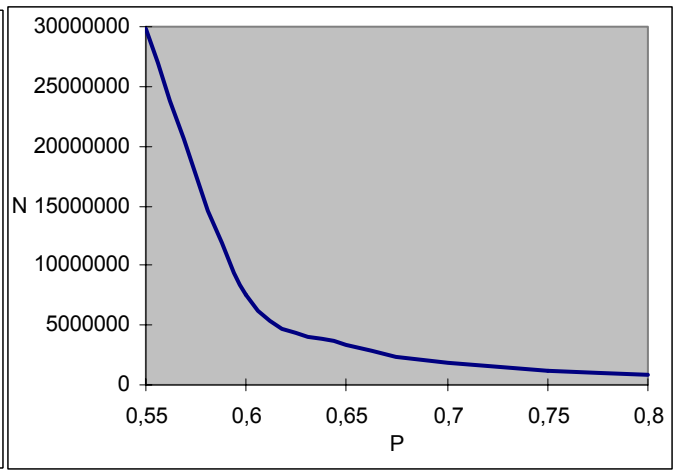


Рис.5 Рис.6

Как видно из приведенных рисунков, размер гаммы шифрующей, необходимой для проведения успешного криптоанализа, экспоненциально зависит от корреляции, присущей нелинейной функции. Таким образом, выбор нелинейной функции с низкой корреляцией является одним из необходимых условий при конструировании схем поточного шифрования.

Анализируя корреляционные атаки, можно сделать следующие выводы:

- базовые корреляционные атаки являются малопригодными для практической реализации в силу высокой вычислительной сложности и применимы для регистров с $r \geq 60$, где r – степень образующего полинома (длина регистра);

- методы, основанные на нахождении уравнений низковесовой проверки четности, имеют слишком высокую сложность, чтобы быть практичными при увеличении длины ЛРР. Быстрые корреляционные атаки, основанные на методах декодирования с низковесовой проверкой четности, применимы только к регистрам с низковесовыми полиномами и относительно низкой степенью – согласно (4), при использовании большого количества ячеек обратных связей алгоритмы неприменимы в силу того, что уравнения проверки четности будут давать очень малую исправляющую коррекцию: увеличение p^* влечет уменьшение $(1 - p^*)$ – корректирующей способности алгоритма.

- наиболее практичными следует считать атаки, применимыми к регистрам произвольной длины. К данным атакам относятся атаки, базирующиеся на конволюционных кодах, использующих технику турбо кодов, базирующихся на восстановлении линейных полиномов, а также быстрая корреляционная атака Чепыжова, Йоханссона, Смитса;

- атаки, базирующиеся на конволюционных кодах, и атаки, использующие технику турбо кодов, при приемлемой вычислительной сложности требуют больших затрат памяти при использовании больших регистров, что накладывает ограничения на практическое использование – временная сложность фазы декодирования составляет $O(m2^B)$, т.о. на практике применяемый размер памяти составит $B = 20\text{-}30$, увеличение B влечет необходимость использования недостижимых объемов памяти;

- наиболее применимыми следует считать атаку Йонссона-Йоханссона, базирующуюся на восстановлении линейных полиномов, и быструю корреляционную атаку Чепыжова, Йоханссона, Смитса, обладающих приблизительно одинаковыми вычислительными сложностями и объемами требуемой памяти.

2. Компромисс время-память

Целью данных атак является восстановление начального состояния сдвигового регистра по фрагменту шифрующей последовательности при условии, что криптоаналитику известна схема устройства и некоторый фрагмент гаммы-шифрующей. Сложность атаки зависит от длины перехваченной гаммы-шифрующей и размера внутреннего состояния шифра. Данный вид атаки применим, если пространство состояний достаточно мало. В частности, такая атака была успешно применена к криптоалгоритму A5/1 [14-16]. В общем случае атака состоит из двух этапов:

- подготовительный этап, в котором строится большой словарь, включающий все возможные пары “состояние-выход” (одинаковой размерности n);
- основной этап, в котором делается предположение, что шифр находится в определенном фиксированном состоянии, т.е. предположение об определенном заполнении всех ячеек памяти. На основе этих входных данных генерируется выход. Далее просматривается перехваченная выходная последовательность с целью нахождения соответствия со сгенерированным выходом. Если соответствие произошло, то фиксированное состояние с большой вероятностью считается начальным заполнением регистра, в противном случае алгоритм продолжает работать.

Атака Стива Беббиджа [11]. Атака была предложена как улучшенный алгоритм исчерпывающего поиска ключа и может рассматриваться как простейшая атака “компромисс время/память”. Предложенная атака основана не на опробовании всех возможных комбинаций начального заполнения регистра, свойственной исчерпывающему поиску, а на предположении начального заполнения регистра; таким образом, сложность данной атаки может быть ниже, чем при опробовании всевозможных комбинаций.

Пусть длина регистра n, тогда S - множество всех состояний генератора, равное 2^n. И пусть количество известных криптоаналитику бит гаммы шифрующей равно $2^m + n - 1$, таким образом можно составить список из $M = 2^m$ перекрывающихся n – битных под-последовательностей. Положим, что мы сгенерировали $R = 2^r$ случайных и независимых состояний генератора, и для каждого такого состояния сгенерировали соответствующую ему n - битную последовательность. Таким образом, мы имеем список из 2^m n – битных последовательностей известной гаммы шифрующей, и список из 2^r n – битных последовательностей гаммы шифрующей, сгенерированных из известных (предположенных) состояний генератора. Если последовательность из первого списка соответствует последовательности из второго списка, то с высокой вероятностью можем утверждать, что известное состояние, породившее подпоследовательность второго списка, соответствует состоянию генератора в соответствующей точке в генерации известной гаммы. Для такой пары ожидается $m + r \approx n$. Следовательно, процесс поиска соответствия между двумя списками может иметь сложность гораздо меньше, чем 2^n операций. Атака может иметь два варианта.

В первом случае при выборе $m = n/2$ потребуется порядка $2^{n/2} = \sqrt{|S|}$ известных бит гаммы шифрующей, времени порядка $n^2 2^{n/2} \approx \sqrt{|S|}$ и памяти порядка $n2^{n/2} \approx \sqrt{|S|}$.

Во втором случае при выборе $r = n/2$ потребуется порядка $2^{n/2} = \sqrt{|S|}$ известных бит гаммы шифрующей, времени порядка $n^2 2^{n/2} \approx \sqrt{|S|}$ и памяти порядка $n2^{n/2} \approx \sqrt{|S|}$. При выборе $r > n/2$, потребуется памяти порядка $> n2^{n/2}$ и предвычислительного времени порядка $> n^2 2^{n/2}$, но порядка $< 2^{n/2}$ известных бит гаммы шифрующей и времени порядка $< n^2 2^{n/2}$ – таким образом, в принципе, время для проведения атаки может быть доведено до малых значений.

Выбирая внутреннюю память генератора как $2n$, для первого случая имеем

$$\begin{aligned} O(D) &= 2^{2n/2} = 2^n = S \\ O(T) &= 4n2^{2n/2} = 4n2^n \approx S \\ O(M) &= 2n2^{2n/2} = 2n2^n \approx S \end{aligned} \qquad (21)$$

где D, T, M – длина гаммы, время атаки и объем памяти соответственно, т.е. сложность проведения атаки становится аналогичной сложности проведения полного перебора.

Атака Бирюкова-Шамира [12]. Алгоритм обобщает ранее представленные атаки данного класса и улучшает показатели взаимосвязи время/память/данные выведенным соотношением $TM^2D^2 = N^2$ для любого $D^2 \le T \le N$, $P = T = N^{2/3}$, $M = D = N^{1/3}$. В качестве показателей рассматриваются объем требуемой памяти M (количество записей), временная сложность T, время подготовительного этапа P, размер памяти N (размер внутреннего состояния генератора), количество бит рассматриваемой гаммы-шифрующей D. В таблице 4 представлены сравнительные характеристики двух представленных атак данного класса, где $A1$ – атака Беббиджа, $A2$ – атака Бирюкова-Шамира.

Таблица 4

	$\ell = 60$		$\ell = 70$		$\ell = 80$		$\ell = 90$		$\ell = 100$	
	$A1$	$A2$	$A1$	$A2$	$A1$	$A2$	$A1$	$A2$	$A1$	$A2$
Время	$2^{41.8}$	2^{40}	$2^{47.2}$	2^{46}	$2^{52.2}$	2^{54}	2^{58}	2^{60}	$2^{63.3}$	2^{66}
Память	60 Гб	2^{20}	2240 Гб	2^{23}	$2^{46.32}$	2^{27}	$2^{51.49}$	2^{30}	$2^{56.65}$	2^{33}
Гамма	0,13	2^{40}=137	4,29	2^{46}	2^{40}=137	2^{54}	4398	2^{60}	2^{17}	2^{66}

Как видно из приведенной таблицы, уже при $N = 2^{90}$ необходимый для криптоанализа размер гаммы шифрующей считается недостижимым для обеих атак (4398 Гбайт).

3. Предполагать и определять

Основная идея атаки состоит в допущении, что криптоаналитику известны гамма-шифрующая, количество сдвигов регистра между выходами схемы, а также полином обратной связи степени r, фильтр-функция f и последовательность точек съема γ_i, $i = 1, \ldots, n$. Восстановление начального заполнения регистра происходит путем восстановления некоторых фрагментов заполнения регистра на основе предположения содержимого некоторых неизвестных фрагментов схемы (содержимое ячеек регистра, элементы нелинейной функции). В общем виде атака имеет следующий вид:

- *Предполагается* содержимое некоторых ячеек регистра.
- *Определяется* полное заполнение регистра путем применения линейной рекурренты регистра на основе принятых допущений.
- Генерируется выходная последовательность. Если она эквивалентна гамме шифрующей, предположение принимается верным, иначе – переход к шагу 1.

Сложность атаки зависит от конкретной реализации схемы и пропорциональна количеству предположений. Так, количество возможных предположений равно 2^x, где x – количество предположений, приведших к успеху.

4. Инверсионные атаки

Целью данных атак является восстановление начального состояния сдвигового регистра по фрагменту шифрующей последовательности при условии, что известны полином обратной связи степени r, фильтр-функция f и последовательность точек съема γ_i, $i = 1, \ldots, n$. Рассматривается размер памяти M, где $M = \gamma_n - \gamma_1$, и $M \le r$-1 . Имеется в виду, что шифр обращаем (инвертируем), если известно начальное состояние памяти [13]. Пусть выходная последовательность $(y(t))_{t=0}^{\infty}$ генератора задается как

$$y(t) = f(x(t - \gamma_1), \ldots, x(t- \gamma_n)), \qquad t \ge 0, \tag{22}$$

нелинейная функция $f(z_1, z_2, \ldots, z_n)$ сбалансирована для каждого значения $(z_2, \ldots, z_n)$, то есть

$$f(z_1, z_2, \ldots, z_n) = z_1 + g(z_2, \ldots, z_n). \tag{23}$$

Тогда соотношение (22) представимо как

$$x(t) = y(t) + g(x(t - \gamma_2), \ldots, x(t- \gamma_n)) , \qquad t \ge 0. \tag{24}$$

В самом общем виде алгоритм можно описать следующим образом. Предполагаются (не проверенные ранее) M бит $(x(t))_{t=-M}^{-1}$ неизвестного начального состояния памяти, после чего, согласно (13), генерируется отрезок $(x(t))_{t=0}^{r-M-1}$ входной последовательности по известному отрезку гаммы $(y(t))_{t=0}^{r-M-1}$. Далее, используя линейную рекурренту ЛРР, генерируется последовательность $(x(t))_{t=r-M}^{N-1}$ по первым r битам $(x(t))_{t=0}^{r-M-1}$. На основании $y(t)$, вычисляется $(\hat{y}(t))_{t=r-M}^{N-1}$ по $(x(t))_{t=r-2M}^{N-1}$ и сравнивается с наблюдаемым $(y(t))_{t=r-M}^{N-1}$. Если они совпадают, принимается предложенное начальное состояние памяти и процедура считается законченной, в противном случае предполагаются следующие не проверенные ранее M бит $(x(t))_{t=-M}^{-1}$ неизвестного начального состояния памяти с последующем повторением алгоритма.

Атака привлекательна тем, что ее вычислительная сложность имеет порядок $O(2^M)$ и не зависит от размера памяти ЛРР – r. Ввиду этого для противостояния атакам данного вида рекомендуется выбирать M достаточно большим, предпочтительно близким к максимальному значению r-1, а также обращать внимание на правильность подбора n точек съема – множество точек съема должно удовлетворять полному множеству положительных разностей с выбором

$$n \le \sqrt{2r}\,. \tag{25}$$

где r – длина регистра[13].

5. Ключевая загрузка, инициализация/реинициализация

Как показывает анализ рассматриваемых в открытой литературе процедур ключевой загрузки, инициализации и реинициализации, в силу использования различными схемами шифрования уникальных для каждой разработанной схемы процедур, не существует некоего единого способа оценки стойкости данных процедур. Желательной является конструкция, исключающая загрузку ключа посредством линейных операций и нулевое заполнение регистра; каждый бит инициализированного регистра должен являться нелинейной функцией от каждого бита используемого ключа. Использование линейных операций делает возможным применение криптоанализа посредством решения системы линейных уравнений. Поскольку единого алгоритма загрузки не существует, каждый раз, в зависимости от используемой структуры схемы, процедуы ключевой загрузки будут различны.

Рассмотрим процесс определения корреляции вводимого ключа с инициализированным ключом (“утечки” вводимого ключа) с точки зрения теории информации. Пусть имеем ключ длиной n, и обозначим как K_{in}^n и K_{out}^n вводимый и инициализированный ключи соответственно. И пусть k обозначает количество бит инициализированного ключа, введенных нелинейными преобразованиями, $0 \le k < n$. Тогда, если $H(K_{out}^n)$ обозначает неопределенность относительно вводимого ключа K_{in}^n и $H(K_{out}^n \mid K_{in}^{n-k})$ – условную неопределенность при знании n-k бит вводимого ключа, взаимная информация между K_{out}^n и K_{in}^{n-k} составляет

$$I(K_{out}^n\ ;\ K_{in}^k) = H(K_{out}^n) - H(K_{out}^n \mid K_{in}^{n-k}) \tag{26}$$

Очевидно, что $I(K_{out}^n\ ;\ K_{in}^{n-k}) = 0$ при $k = n$. Таким образом, обеспечивая нелинейные преобразования для каждого бита вводимого ключа, мы увеличиваем неопределенность криптоаналитика относительно каждого бита инициализированного ключа.

На основе выше изложенного можно сделать следующие выводы:

- эффективность применения корреляционных атак зависит, во-первых, от выбора выходной функции f, и, во-вторых, от параметров ЛРР;
- для противостояния корреляционным атакам в целом минимальная длина регистра должна быть не менее 100 бит, образующий полином не должен быть прореженным. Выбором плотного образующего полинома высокой степени исключается возможность применения атак данного класса, основанных на низковесовых проверках четности, - согласно (4)

$$p^* = \frac{p \cdot s^{m-h} \cdot (1-s)^h}{p \cdot s^{m-h} \cdot (1-s)^h + (1-p)(1-s)^{m-h} s^h},$$

увеличение количества точек съема обратных связей ЛРР уменьшает корректирующую способность алгоритма;

- среди корреляционных атак наиболее применимыми следует считать атаку Йонссона-Йоханссона, базирующуюся на восстановлении линейных полиномов, и быструю корреляционную атаку Чепыжова, Йоханссона, Смитса, обладающих приблизительно одинаковыми вычислительными сложностями и объемами требуемой памяти. Данные атаки применимы к регистрам с произвольным весом. На сегодняшний день посредством данных атак в течении нескольких часов может быть восстановлено начальное состояние регистров с $\ell = 70$ бит, предполагается возможность восстановления за несколько недель регистров длиной 80-100 бит;

- для противостояния атакам Йонссона-Йоханссона и Чепыжова, Йоханссона, Смитса необходимо осуществлять выбор нелинейной функции с $P(x_i = z_i)$ как можно ближе к 0,5, а также согласно (20)

$$N \approx 1/4 \cdot (2B \cdot t! \cdot \ln 2)^{1/t} \cdot \varepsilon^{-2} \cdot 2^{(l-B)/t}, \; N >> n_t$$

при заданных верхних граничных значениях B, t определять максимальную длину гаммы N_{max} шифрующей, после генерации которой производить перезагрузку схемы;

- для противостояния атаке “компромисс время-память” размер внутреннего состояния генератора должен быть по меньшей мере в 2 раза больше размера вводимого ключа - в таком случае сложность атаки будет равной или больше сложности полного перебора (21):

$$O(D) = 2^{2n/2} = 2^n = S$$
$$O(T) = 4n2^{2n/2} = 4n2^n \approx S$$
$$O(M) = 2n2^{2n/2} = 2n2^n \approx S$$

где D, T, M – длина гаммы, время атаки и объем памяти соответственно;

- для противостояния атаке “предполагай и определяй”, “разделяй и вскрывай” используемые схемы должны обладать значительным объемом внутреннего состояния, таким образом, чтобы

$$2^k < 2^x,$$

где k – длина ключа, x – количество предположений, приведших к успеху; кроме того, множество точек съема для обратных связей ЛРР и множество точек съема для НЛФ должны быть подобраны таким образом, чтобы не взаимодействовать взаимоослабляющим образом – для увеличения неопределенности криптоаналитика при производимых предположениях одна и та же ячейка ЛРР не может использоваться как для обратных связей ЛРР , так и НЛФ;

- для противостояния инверсионной атаке рекомендуется выбирать M достаточно большим, предпочтительно близким к максимальному значению r-1, точки съема нелинейной функции должны удовлетворять множеству положительных разностей с целью уменьшения просачивания информации о внутреннем состоянии регистра, количество точек съема n определяется как

$$n \le \sqrt{2r};$$

- ключевая загрузка должна исключать линейные операции загрузки ключа, используя алгоритмы с нелинейными преобразованиями; желательна реализация, при которой каждый бит начального состояния регистра является функцией от нелинейного преобразования всех бит ключа, в таком случае

$$I(K_{out}^n \; ; \; K_{in}^{n-k}) = 0,$$

т.е. знание вводимого ключа исключает знание инициализированного ключа (при гарантированной стойкости нелинейных преобразований).

На основе сформулированных выводов с целью противостояния криптоаналитическим атакам при построении схем поточного шифрования целесообразно руководствоваться следующими требованиями:

- для генерации регистром последовательности максимальной длины и высокой линейной сложности данный регистр должен быть основан на примитивном полиноме обратной связи, кроме того, длина ℓ ЛРР и алгебраический порядок d нелинейной функции должны быть достаточно велики, чтобы величина $\binom{\ell}{d}$ была значительно больше, чем длина гаммы шифрующей; при использовании нескольких регистров их длины должны быть взаимно-простыми [13];
- для достижения максимальной линейной сложности степень образующего полинома должна быть простым числом;
- минимальная длина регистра должна быть не менее $\ell = 100$ бит, образующий полином иметь приблизительно около $\ell/2$ ненулевых коэффициентов обратной связи;
- содержимое первой и последней ячеек регистра должны являться входными данными к нелинейной функции, множество всех точек съема должно удовлетворять множеству положительных разностей, множество точек съема для обратных связей ЛРР и множество точек съема для НЛФ должны исключать совместное использование одних и тех же ячеек;
- количество точек съема для НЛФ определяется исходя из выражения (25);
- при построении нелинейных функций данные функции должны удовлетворять критериям стойкости: быть сбалансированными, обладать высокой нелинейностью, удовлетворять критерию распространения (обладать корреляционным иммунитетом), иметь высокую алгебраическую степень.
- при длине ключа k бит внутреннее состояние генератора (внутренняя память) должно быть не менее $2k$ бит;
- генерируемая гамма шифрующая не должна превышать значения N_{max}, определенного согласно (20);
- каждый бит начального состояния регистра должен являться функцией от нелинейного преобразования всех бит ключа.

Литература

1. T.Siegenthaller. Decrypting a class of stream cipher using ciphertext only. *IEEE Trans. Comput*.,vol. C-34, pp.81-85, Jan. 1985.
2. T.Siegenthaller. Cryptanalysis Representation of Nonlinearly Filtered ML-Sequences. *Advances in Cryptology: Proc. Eurocrypt'85*, pp. 103-110, Springer-Verlag, 1986.
3. W.Meier, O.Staffelbach. Fast correlation attacks on stream ciphers. *Journal of Cryptology*, vol. I, no. 3, pp.159-176, 1989.
4. R.Forre. A Fast Correlation Attack on Nonlinearly Feedforward Filtered Shift-Register Sequences. *Advances in Cryptology: Proc. Eurocrypt'89*, pp. 586-595, Springer-Verlag, 1990.
5. Michalevich, J.Golic
6. V.Chepyzhov, B.Smeets. On a Fast Correlation Attacks on Certain Stream Ciphers. *Advances in Cryptology: Proc. Eurocrypt'91*, pp. 176-185, Springer-Verlag, 1991.
7. F.Jonsson, T.Johansson. Improved Fast Correlation Attacks on Stream Ciphers via Convolutional Codes. *http: //www.it.lth.se./thomas*/, 1999.
8. F.Jonsson, T.Johansson. Fast Correlation Attacks Based on Turbo Code Techniques. *http: //www.it.lth.se./thomas*/, 1999.
9. F.Jonsson, T.Johansson. Fast Correlation Attacks Through Reconstruction of Linear Polinomials. *http: //www.it.lth.se./thomas*/, 2000.

10. V.Chepyzhov, T.Johansson, B.Smeets. A simple algorithm for fast correlation attacks on stream ciphers. *http: //www.it.lth.se./thomas*/, 2000.
11. S.Babbage. A Space/Time Trade-Off in Exhaustive Search Attacks on Stream Ciphers. Vodafone Ltd, Newbury, UK. 09.04.1996.
12. A.Biryukov, A.Shamir. Cryptanalytic Time/Memory/Data tradeoffs for Stream Ciphers. 2000.
13. J.D.Golic. On the Security of Nonlinear Filter Generators.
14. J.D.Golic. Cryptanalysis of Alleged A5 Stream Cipher. *Advances in Cryptology: Proc. Eurocrypt'97*, pp. 239-255, Springer-Verlag, 1997.
15. E.Biham, O.Dunkelman. Cryptanalysis of the A5/1 GSM Stream Cipher. NES/DOC/TEC/WP3/005/a.
16. A.Biryikov, A.Shamir. Real Time Cryptanalysis of the Alleged A5/1 on a PC. 09.09.1999.