## МИНОБРНАУКИ РОССИИ

## ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

## ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

## «НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ» (НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ, НГУ)

## \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Кафедра……………………………Компьютерных систем …....……………………………….

## (название кафедры)

## …………………..………...…Александр Сергеевич Лысяк ……….……..………………....

## (И., О., фамилия студента – автора работы)

## …....Анализ эффективности градиентной статистической атаки на блоковые шифры..…...

## (полное название темы магистерской диссертации)

## ……………………………………………………………………………………………………...

## МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

## по направлению высшего профессионального образования

## 230100.68 ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА

## ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

## Тема диссертации утверждена распоряжением по НГУ №\_\_\_ от «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_2012 г.

## Руководитель

## Рябко Борис Яковлевич

## (фамилия , И., О.)

## д.т.н., профессор

## (уч.степень, уч.звание)

## Новосибирск, 2012 г.

**1. Введение**

Блоковые шифры c секретным ключом находят широкое применение в системах защиты передачи и хранения информации, что делает актуальными как задачи построения надёжных блоковых шифров, так и задачи поиска эффективных криптологических атак на эти шифры (т.е. методов определения секретного ключа шифра на основе экспериментов с зашифрованными сообщениями). Многие специалисты занимаются исследованием теоретической и практической устойчивости блоковых шифров к атакам различного рода. Постоянно разрабатываются как новые шифры, так и новые атаки на них, позволяющих оценить реальную устойчивость шифров. Отметим, что для криптографии практический интерес представляют только те атаки, которые позволяют находить ключ за время меньшее, чем метод прямого перебора секретного ключа.

Шифры RC6, Blowfish, MARS, IDEA, CAST-128 – одни из претендентов на звание AES, конкурса, проводимого в 2001 году, для выбора нового криптографического стандарта США. Существуют многочисленные работы, посвящённые анализу этих шифров. Большинство из них используют результаты дифференциального и линейного криптоанализов – двух наиболее распространённых видов атак, применимых к блоковым шифрам в несколько урезанном виде: в исследованиях шифров зачастую не используются присущие им операции так называемого «забеливания» («post–whitening» и «pre–whitening»), а также уменьшено количество этапов шифрования. В последнее время появился новый тип атак, основанный на исследовании статистических свойств шифров. Например, получен результат, согласно которому выходную последовательность шифра RC6 можно отличить от случайной последовательности при наличии подходящего количества выбранного шифротекста (для *r*-го раунда необходимо бит текста). Таким образом, уже для 4-го раунда сложность будет значительной – , что делает атаку практически нереализуемой.

В данной работе предлагается новая атака, которая позволяет найти секретный ключ блокового шифра за время, существенно меньшее, чем в случае линейного и дифференциального криптоанализов, а также атаки, основанной на традиционном исследовании статистических свойств шифров. Например, для 5-го раунда RC6 временная сложность предлагаемой атаки равна , а в общем случае: . Атака основана на исследовании статистических свойств шифра RC6 при помощи разработанного Б.Я. Рябко статистического теста «стопка книг», который, по многочисленным исследованиям, является лучшим на текущий момент статистическим тестом.

В данной работе исследована эффективность статистического теста «стопка книг» на примере сравнительного анализа его с другими тестами, а также анализа им некоторых видов генераторов случайных чисел. Проведены исследования эффективности градиентной атаки на примере шифров RC6, Blowfish, MARS, IDEA, CAST-128; показаны пределы её современной практической и теоретической применимости, получены математические зависимости между эффективно взламываемыми раундами и количеством требуемых вычислительных ресурсов. Выявлены условия применимости и расширяемости атаки на произвольное количество раундов, а также особенности применения атаки и возможные её улучшения на конкретных примерах каждого из шифров. Также в данной работе показан метод подбора оптимальных параметров для статистического теста, а также их влияние на атаку; исследована временная сложность атаки и её зависимость от параметров теста и объёма шифротекста; показаны теоретические потребности в вычислительных мощностях, необходимых для осуществления атаки.

**2. Описание блокового шифрования**

У целого класса современных блоковых шифров начальный секретный ключ *K* длиной в |*K*| бит преобразуется в последовательность так называемых ключей раундов , которые используются последовательно для шифрования на разных этапах, называемых раундами шифра. В разных шифрах данные преобразования разные. Схематично процесс шифрования для такого класса шифров можно представить следующим образом. Берётся блок из входных данных фиксированной длины и подаётся на вход шифрующему преобразованию, которое преобразовывает данный входной блок в выходной. Далее, берётся следующий блок той же длины из входа и подаётся на вход шифрующему преобразованию и так далее, пока не зашифруем всю последовательность. Шифрующее преобразование представляет собой последовательность этапов шифрования, в которых весь процесс разбивается на несколько раундов, при этом на каждом следующем раунде происходит шифрование блока, полученного на предыдущем раунде.

где  – исходный блок данных, который необходимо зашифровать, *E* – операция (функция) шифрования на *i*-ом раунде,  – ключ, используемый на *i*-ом раунде, – блок данных, являющийся «выходом» *i*-ого раунда и «входом» *-*ого. В разных шифрах эти процедуры (от генерации раундовых ключей до шифрования блоков) осуществляются по-разному, причем это зависит не только от шифра, но и от значений длин блока, ключа  и числа раундов *r*, которые для многих шифров являются параметрами. Например, для шифра RC6 длина блока может принимать значения 32, 64 или 128 бит, количество раундов может быть любым целым числом, а длина ключа должна быть кратна 8 и может принимать любое значение, начиная с 8 бит. Дешифрование проводится по схеме, обратной к шифрованию:

где используются те же ключи раундов, что и при шифровании, а операция *D* – обратная функция к *E*.

Одно из основных требований, предъявляемых к современным блоковым шифрам, можно сформулировать следующим образом: любое зашифрованное сообщение должно быть «похоже» на реализацию бернуллиевского процесса, т.е. любое зашифрованное сообщение должно быть статистически неотличимо от абсолютно случайной последовательности любым статистическим тестом **(1)**. В частности, все шифры, принимавшие участие в конкурсе AES проверялись на выполнение этого условия. Для анализа выполнения этого требования будем использовать следующую общепризнанную схему. Определим *n*-битовое слово , как двоичную запись числа *i* в виде последовательности фиксированной длины *n*, где *i = 0,1, 2, …, 2n-1*, где, как и ранее, *n* – длина блока рассматриваемого шифра (т.е. состоит из *n*-битовой цепочки нулей, – из *(п*-*1)* нуля и единицы, – из *(п*-*2)* нулей и т.д.). Заданная последовательность, очевидно, имеет явные закономерности и сильно отличается от случайной. От современного блокового шифра требуется, чтобы при любом значении секретного ключа последовательность *n*-битовых слов Е(), Е(), Е(), ..., рассматриваемая как двоичная последовательность, была статистически неот­личима от случайной. Это требование, в частности, позволяет использовать блоковые шифры как генераторы псевдослучайных чисел.

**3. Описание градиентной статистической атаки**

**3.1 Общие принципы атаки.**

Описываемый  метод относится к классу атак с выбираемым шифруемым текстом (chosen ciphertexts attack). При реализации этой атаки криптоаналитик может подавать на вход шифра любой текст, анализировать полученное зашифрованное сообщение, затем, базируясь на результатах этого анализа, подавать новое сообщение и т.д. Цель атаки – нахождение (секретного) ключа, причем при этом предполагается, что криптоаналитик знает все характеристики шифра, кроме этого ключа.

Для нашей задачи на вход шифров подавались блоки , длина которых в двоичной записи равна длине блока данных шифра данных. Блоки подаются в последовательности , т.е. блоки следующего вида: 00…001, 00..0010, 000..0011,… **(1’)**

Введём понятие «меры случайности» последовательности данных. Мы будем использовать термины «более» и «менее» случайные по­следовательности, понимая под этим, что некоторая последовательность более случайна, чем другая, если отклонения от случайности у первой достоверно выявляются при большей длине, чем у второй, или же, что тоже самое, – величина статистики критерия для первой последовательности меньше, чем для второй (менее случайной). Допустим есть заданная последовательность , тогда применим к ней статистический тест Г, численно отображающий величину статистики критерия (что фактически равно отклонению от случайности), как , и возьмём величину, обратную к Полученное численное значение и назовём «мерой случайности» последовательности .

Если  на вход шифра подавать данные вида (1’) (что существенно, – с длиной каждого равной размеру блока шифра), то вероятность обнаружить на выходе последовательность с отклонениями от «случайности» будет значительно выше, т.к. «случайность» этой последовательности очень низкая и одновременно совпадение с размером блока уменьшает «смешиваемость» элементов последовательности и даёт менее запутанную на выходе шифра последовательность. В результате экспериментов описанных в [1] было показано, что мера случайности последовательности, полученной на выходе после *i*-го раунда, в любом блоком шифре возрастает с ростом числа раундов, что полностью соответствует требованию (1). В тоже время, в процессе дешифрования мера случайности (будем называть её просто «случайность») последовательности убывает.

Кратко опишем схему градиентной статистической атаки. Пусть есть *i*-ый раунд шифра: последовательность бит  – зашифрованная на *i* раундов последовательность, которая подаётся на вход операции дешифрования раунда *i,* и эта же последовательность – на выходе дешифратора раунда *i*: . Тогда схему дешифрования можно представить в виде: . Далее, оцениваем меру случайности последовательности При дешифровании с правильным ключом она должна сильно уменьшиться, тогда как дешифрование с неправильным ключом аналогично шифрованию, т.е. оно наоборот увеличивает меру случайности последовательности. В итоге, если Г() < Г(), то ключ раунда *Ki* подобран правильно, иначе – неправильно. Смысл атаки сводится к нахождению такого ключа *Ki*, который будет минимизировать Г(). Теоретически для этого надо перебрать все вариантов, но исходя из того, что правильный ключ даёт существенно меньшую величину Г(), нежели при неправильном ключе раунда, можно подбирать ключи до нахождения этого «существенного минимума». После нахождения ключа *Ki*, дешифруем последовательность на i-ом раунде и переходим к подбору ключа *Ki-1* и т.д.. Нахождение всех ключей раундов *Ki*аналогично нахождению секретного ключа *K*, т.к. позволяет шифровать/дешифровать любую последовательность.

**3.2 Исследование эффективности и применимости атаки.**

Эффективность данной атаки в сравнении с полным перебором исходного ключа *K* заключается в том, что мы можем находить по отдельности ключи каждого раунда. Как правило, размер ключей в раундах является постоянным и равным во всех раундах, при этом, он обычно намного меньше по длине, чем исходный ключ *K*. Используя вышеописанную схему, мы можем подбирать ключи *Ki*, что даёт операционную сложность атаки, равную **(1’’)**, где *mi* – трудоёмкость операции дешифрования одного раунда с учётом длины входной последовательности, |*Ki*| - длина раундового ключа на *i*-ом раунде, *r* – количество раундов шифра. При равных размерах раундовых ключей, можно определить верхнюю оценку, как , где *K0* – размер раундового ключа, Операционная сложность полного переборе равна . Таким образом, градиентная статистическая атака эффективна при выполнении следующего условия: **(2)** или, в ближайшем приближении: **(2’).** Формулу (2’) можно переписать в сокращённом виде: **(2’’)**. Количество раундов в (2’’) не фигурирует в силу того, что, как будет показано ниже, сложность атаки на последний раунд существенно выше (на несколько порядков) сложности атаки на предыдущие раунды шифра.

Таким образом, если существует статистический тест, который эффективно находит отклонения от случайности на -ом раунде, при рекомендованных *r* раундах, то ключ такого шифра можно найти существенно быстрее, чем методом прямого перебора. В этом состоит основная идея градиентной статистической атаки.

Исходя из схемы данного типа атаки, сразу выделяются несколько теоретических и практических ограничений применимости атаки:

1. Сильная зависимость атаки от применяемого статистического теста и его параметров. Используемый статистический тест является параметром метода. При чём в широком смысле: на разных раундах шифра можно использовать разные статистические тесты с целью уменьшения общей трудоёмкости атаки. Подходит любой тест, применимый для проверки гипотезы H0 о том, что двоичная последовательность порождается бернуллиевским источником с равными вероятностями для нуля и единицы против альтернативной гипотезы H1, являющейся отрицанием H0. Использование того или иного статистического теста сильно влияет на количество раундов шифра, выход которых заданный тест отличает от абсолютно случайной последовательности.
2. Требование стационарности операции шифрования E: зависимость только от *Ki* и *xi*. Существуют шифры, в которых функция шифрования функционально полно зависит от исходного ключа K, что делает градиентную атаку абсолютно неприменимой к таким шифрам.
3. Ограничение на размер раундовых ключей относительно исходного секретного ключа: . В некоторых шифрах размер раундового ключа либо вообще не отличается от размера исходного, либо отличается на слишком малую величину, что делает данный тип атаки бесполезным, либо не эффективным, применительно шифрам данного типа.

Для проведения практических атак в качестве статистического теста был взят тест «стопка книг», разработанный Б.Я. Рябко, и описанный в [4]. Поскольку этот тест зарекомендовал себя, как один из самых эффективных тестов для проверки криптографических генераторов (см. [4]).

**3.3 Гибридная градиентная статистическая атака и её эффективность.**

У многих современных блоковых шифров при большом числе раундов даже «очень неслучайная» последовательность после шифрования статистически неотличима от случайной при любой длине и используемом тесте. Гибридная атака является некоторым расширением стандартного варианта градиентной статистической атаки с целью повысить число эффективно взламываемых раундов (т.е. взламываемым с трудоёмкостью меньшей, чем у полного перебора) у класса шифров с хорошими статистическими данными.

Пусть, например, шифр использует r раундов. Число *d* равно номеру максимального раунда шифра, на котором используемый в атаке статистический тест выявляет отклонения от случайности, при чём d < r. Тогда градиентная атака может быть модифицирована следующим образом: для каждого набора ключей *Kd+1, ..., Kr* раундов *d+1, …, r* повто­ряем описанную выше процедуру нахождения неизвестных ключей *K1, …, Kd* раундов *1, ..., d*. Другими словами, ключи *Kd+1*, ..., *Kr* находим полным перебором, а *K1, …, Kd* – по описанному выше методу, что при некоторых соотношениях параметров может быть меньше, чем количество операций, необходимое при прямом переборе всех ключей.

Для проведения такой комбинированной атаки потребуется количество операций равно . Как не трудно понять, исход из данного соотношения, в случае, когда трудоёмкость гибридного варианта атаки совпадает с трудоёмкостью стандартного варианта.

Таким образом, применимость гибридного варианта градиентной статистической атаки определяется следующим соотношением: **(3)**. Понятно, что гибридные атаки имеют смысл, когда размер раундового ключа *|K0|* много меньше размера секретного ключа *|K|*. Исходя из формулы (3), можно заключить, что гибридный вариант атаки позволяет расширить поддающиеся стандартной градиентной атаке (*d+1*) раундов на раундов и таким образом, общее количество эффективно взламываемых раундов становится равно: , где *d* – максимальный номер раунда, на котором статистический тест выявляет отклонения от случайности.

* 1. **Программная реализация атаки.**

Градиентная статистическая атака была реализована программно на языке С++ со следующими особенностями:

1. Возможность распараллелить подбор раундовых ключей шифра на произвольное количество потоков. Тестирование атаки и получение экспериментальных данных производилось на суперкомпьютерном кластере НГУ.
2. В качестве статистического теста был использован тест «стопка книг» с одной степенью свободы, что позволяет осуществлять статистический анализ последовательностей с линейной трудоёмкостью и малыми затратами аппаратных ресурсов. Подробнее о реализации статистического теста – в части 4.3.
3. Создана библиотека для измерения меры случайности произвольной битовой последовательности, а также для проведения атаки на заданный шифр. При том на вход требуется подавать функцию установки ключа/формирования раундовых ключей, функцию блокового шифрования, а также параметры шифра (размер блока, количество раундов). В итоге, данная библиотека представляет собой относительно универсальный API интерфейс, применимый для потенциально любых блоковых шифров, удовлетворяющих ограничениям, описанным в части 3.2.
4. **Статистический тест «стопка книг»**

**4.1 Описание эффективности.**

В [8] предложен новый статистический тест «стопка книг», эффективность которого исследовалась экспериментально. В частности, с его помощью проверялись датчики случайных чисел, приведенные в [9]. Они были исследованы автором с помощью мощного спектрального теста [10] и прошли его успешно, но тест "стопка книг" выявил отклонения от случайно­сти в последовательностях чисел, порожденных этими датчиками. Кроме того, с помощью этого теста впервые удалось найти существенные статистические недостатки у блокового шифра MARS с уменьшенным числом раундов [11], а также генераторов псевдослучайных чисел RANDU [12] и RC4 [13]. В [14] показано, что для достаточно широкого класса альтернативных гипотез тест «стопка книг» позволяет проверять выборки уже при длине порядка *,* где *S* — количество слов в алфавите, из которого сделана выборка. Размер выборки становится решающим фактором, когда S очень велико, например, 232 или 264. Именно с такими алфавитами, порой, приходится сталкиваться в современной криптографии и защите информации. Многие тесты, в част­ности критерий *Xi*-квадрат, просто неприменимы по сложности в подобных случаях, так как размер тестируемой с их помощью выборки должен быть порядка *S.*

**4.2 Описание структуры.**

Тест «стопка книг» является критерием согласия, поэтому вначале приведем описание та­кого типа критериев. Пусть имеется выборка *X = (x1, x2 ,..., хn)* из алфавита *A = {a1, a2 ,..., as}.* Рассмотрим гипотезу *Н0,* которая заключается в том, что элемен­ты выборки независимы и P(*хn =* *ai*) = p0 *= , n = 1*,...,*N*; i *= 1,..., S.*

Другими словами, элементы выборки имеют равномерное распределение, т. е. все буквы алфавита порождаются независимо и с равными вероятностями. Под критерием согласия с гипотезой H0 понимается некоторая функция от выборки , такая, что

Критерии характеризуются прежде всего ошибками первого и второго рода. Ошибка первого рода – это вероятность отвергнуть гипотезу *Н0*, если она верна. Ошибка вто­рого рода – это вероятность принять гипотезу *Н0*, если она неверна. Величина ()называется уровнем значимости критерия, () – мощностью.

Теперь опишем тест «стопка книг» с теми параметрами, которые будут использоваться в доказательстве. Перед тестированием выборки в алфавите A фиксируется произвольный порядок, который меняется после анализа каждого выборочного элемента *xn* следующим образом: буква *xn* получает номер 1; номера тех букв, которые были меньше номера этой буквы, увеличиваются на 1; у остальных букв номера не меняются. Формально эту проце­дуру можно описать так: пусть *–* это номер буквы после анализа элементов *x1, x2,... , xn-1*, тогда:

**(4)**

Такая конструкция похожа на стопку книг, если считать, что номер книги совпадает с ее положением в стопке. Книга извлекается из стопки, после чтения кладется наверх, и ее номер становится первым. Книги, которые первоначально были над ней, двигаются вниз, а остальные остаются на месте.

В отличие от многих других тестов, например критерия *Xi*-квадрат, в «стопке книг» подсчитывается не частота встречаемости букв в выборке, а частота встречаемости номеров букв при описанном упорядочивании. Перед тестированием множество всех номеров {*1, ..., S*} разбивается на две непересекающиеся части: A1 = {*1, 2,... , M*} и A2 = {*M+1, ..., S*}. Затем по выборке (*x1, x2, ..., хn*) подсчитываетсяV0 – количество номеров *,* при­надлежащих подмножеству A1, т. е. количество попаданий букв в «верхнюю часть» «стопки книг». Число (N – V0), очевидно, равно количеству попаданий в «нижнюю часть» стопки. Далее вычисляется статистика , где *P1*= |A1|/S. Если меньше критического уровня , то гипотеза Н0 принимается, иначе – отвер­гается. Величина – квантиль распределения *Xi*-квадрат уровня значимости с одной степенью свободы. Таким образом, критерий теста «стопка книг» будет выглядеть так:

Сама величина является обратной к той мере Г, о которой говорилось в части 3. Если обобщить данный тест на случай с произвольными степенями свободы (т.е.будем делить алфавит не на 2, а на произвольное количество частей), получим меру Г, равную обратной к следующей величине :

, где *r* – количество непересекающихся частей *Ai*, на которые разбивается исходное множество номеров {*1, ..., S*}; *Pi* = |*Ai*|/*S*. Известно, что распределение случайной величины асимптотически приближается к распределению с () степенью свободы при выполнении H0 (в данном случае H0 примет более общий вид: будет включать в себя проверку количества попаданий в каждую из частей стопки). Это позволяет использовать для проверки гипотезы на случайность c произвольным количеством степеней свободы квантили заданного уровня доверия: , а также ввести формально понятие «меры случайности», как обратной к значению статистики с нижним пределом практической применимости в атаке при области значений больших, чем .

**4.3 Программная реализация теста и его параметризация.**

Программная реализация теста «стопка книг» основывается на случае с одной степенью свободы. Данный вариант позволяет снизить трудоёмкость анализа и требования к расходуемым системным ресурсам. Это следует из того, что для определения части стопки, в которой находится слово, достаточно понять, находится ли оно в верхней части: если ответ «да», значит, увеличиваем счётчик верхней части; если ответ «нет», то значит, слово – в нижней части стопки, и счётчик остаётся прежним. Тест имеет два параметра, которые будем обозначать в виде пары (*w, u*) **(*5*)**, где *w* – это длина слова алфавита (в битах), а *u* – параметр размера верхней части, определяемый следующим образом: *u = log2(A1)*, где *A1* – размер верхней части (количество слов) стопки. Соответственно, размер всего алфавита равен 2*w*, а размер верхней части равен 2*u*.

Для случая с одной степенью свободы использовался следующий алгоритм:

1. На *i*-ом шаге имеем слово *Ai*фиксированной длины *w*.
2. Ищем его в верхней части стопки при помощи высокоскоростного алгоритма. Если результат поиска положителен, увеличиваем счётчик верхней части и ставим найденный элемент в голову списка, сдвигая оставшиеся на 1 вниз. Если результаты отрицательный, ставим текущий элемент на первое место и сдвигаем все элементы верхней части стопки вниз на 1, удаляя последний.
3. Вычисляем численную статистику и проверяем гипотезу в соответствии с квантилем заданного уровня доверия.

Данная схема соответствует равенству (4) и рисунку, приведённому ниже.

Особенности программной реализации данного алгоритма состоят в следующем:

1. Проверка принадлежности заданного элемента верхней части осуществляется за время, равное О(1), т.е. за константное, не зависимое ни от каких параметров теста и потенциально – от параметров всей атаки, что являются большим плюсом.

Такое время удалось достичь за счёт использования алгоритма поиска с хэш-индексацией, который описан ниже.

1. Распараллелить данный алгоритм в силу конструктивных особенностей самого теста «стопка книг» невозможно (в отличие, к примеру, от частотного теста ), но можно распараллелить саму атаку (подбирать параллельно несколько ключей), что и было сделано при реализации атаки на суперкомпьютере.
2. В памяти хранится только верхняя часть стопки, оптимальный размер которой в соответствии с исследованиями, приведёнными в [14], обычно равен , т.е. *u* = , что понижает хранимый объём данных с 2*w* до 2*w/2* и к примеру, при размере слова 32 бита уменьшает объём хранимой информации в 216 раз, что приводит к существенной – по сравнению с другими тестами – экономии аппаратных ресурсов.

Непосредственно верхняя часть стопки хранится в памяти в виде двусвязного динамического списка, благодаря чему операции перестановки текущего элемента на первое место со сдвигом всех последующих осуществляется всего за 2 операции.

Алгоритм поиска элемента в стопке заключается в следующем. Как было сказано выше, верхняя часть стопки хранится в виде двусвязного списка. Требуется проверить наличие искомого элемента в заданном списке. К списку добавлен механизм хэш-таблиц, заключающийся в следующем. Eсть ограниченное множество допустимых значений хэш-функции H, принимающей на вход слово из алфавита теста, при том это множество состоит из упорядоченного ряда натуральных чисел фиксированной мощности равной |H| (параметр хэш-таблицы). Число называется коэффициентом заполнения хэш-таблицы. Также есть массив указателей, хранящий адреса разных элементов списка. При том элементы с одинаковым хэш-номером объединим в один хэш-класс. Помимо стандартных связей, отображающих положение всех элементов в стопке, элементы хэш-класса имеют связи между собой: хэш-массив содержит адрес первого элемента хэш-класса (он может быть любым), а тот указывает на следующий элемент того же класса и так далее. В итоге, все элементы хэш-класса объединяются в ещё один – второстепенный – список. В итоге, встречая элемент *а*, определяется его хэш-номер, по нему – через хэш-таблицу – адрес первого элемента его хэш-класса, далее идём по списку хэш-класса и смотрим наличие заданного элемента в списке. Таким образом, операционная сложность поиска сводится к количеству операций, равному коэффициенту заполнения хэш-таблицы. Понятно, что чем выше коэффициент заполнения, тем меньше пространственная трудоёмкость, но тем больше временная. Тем не менее, временная трудоёмкость операций поиска, добавления, перемещения элементов в любом случае остаётся равной *О*(1).

**5. Анализ генераторов псевдослучайных последовательностей с использованием статистического теста «стопка книг»**

Генераторы псевдослучайных чисел (ГПСЧ) играют важную роль в современной криптографии: они используются в хэш-функциях, ассиметричных криптоалгоритмах, системах аутентификации. К таким генераторам предъявляются два важных требования: генерируемая последовательность не должна статистически отличаться от абсолютно случайной за приемлемое (полиномиальное) время вычислений и знания типа генератора и какой-то произвольной части последовательности не должно давать возможность предсказать следующий её элемент. Как было сказано в части 2, под эти требования хорошо подходят любые блоковые шифры, но их реализация и трудоёмкость, порой, достаточно велики, поэтому в настоящее время всё ещё используются традиционные виды ГПСЧ. Останавливаться на втором требовании в данной части не будем – ограничимся исследованием статистических свойств ГПСЧ. Одним из наиболее известных и используемых в настоящее время традиционных ГПСЧ является линейный конгруэнтный генератор (ЛКГ). ЛКГ основывается на следующем соотношении: , где *xi* – очередной элемент генерируемой последовательности фиксированной битовой длины; *x0, a, b, m* – параметры метода.

Тестом «стопка книг» был проведён анализ ЛКГ в разных его вариациях. За базу был взят вариант ЛКГ, реализованный в языке С. В качестве результирующей последовательности берётся 16 старших бит из 32-битного числа (параметр m = 232), что делает его применимым практически в чистом исходном виде.

Рассмотрим следующие типы ЛКГ:

1 тип. Стандартный ЛКГ (rand()) с последовательным шагом по номерам итерации, из генерируемых 16 бит урезаются R бит.

2 тип. Стандартный ЛКГ (rand()) с выбором начального *Х0* в каждой итерации по принципу: , где *i* – номер итерации; *a*, *b* – оптимизируемые константы-параметры метода.

3 тип. Стандартный ЛКГ, реализованный «вручную» с параметром m = 232.

Количество старших бит, записываемых в выходную последовательность на основе исходной 16-битной, определяется числом R, как 2R бит (для всех типов).

Таблица 1.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Генератор, тип | Параметры (*w, u*) |  | *Q0.50* | *Q0.95* | Режим R | Длина ПСП, бит |
| ЛКГ (1) | (16, 8) | 21,29 | 100 | 100 | 16 | 218 |
| ЛКГ (1) | (16, 8) | 18,66 | 100 | 100 | 8 | 219 |
| ЛКГ (1) | (24, 16) | 18,66 | 64 | 23 | 1 | 225 |
| ЛКГ (2) | (16, 8) | 110288 | 100 | 100 | 16 | 218 |
| ЛКГ (2) | (8, 4) | 95,95 | 100 | 98 | 8 | 210 |
| ЛКГ (2) | (16, 8) | 167,42 | 93 | 89 | 1 | 216 |
| ЛКГ (3) | (8, 4) | 1780,22 | 92 | 81 | 16 | 211 |
| ЛКГ (3) | (8, 4) | 1313,86 | 100 | 100 | 8 | 210 |
| ЛКГ (3) | (8, 4) | 826,67 | 100 | 100 | 1 | 29 |

По выше представленным результатам видно, что наиболее удачной оказалась версия ЛКГ первого типа при записи в выходную последовательность только одного старшего бита из всего 16-битного генерируемого числа (который сам по себе является урезанием 32-битного слова). Также хорошо видна закономерность о необходимом размере входной последовательности: чем больше длина ПСП, тем выше выявляемые отклонения от случайности. Соответственно, если один тест находит отклонения от случайности на фиксированной ПСП при меньшей длине последовательности, чем другой, то первый тест будем называть более эффективным, чем второй: трудоёмкость его анализа будет ниже, а вероятность обнаружить отклонения от случайности, как и отражение реальных статистических свойств последовательности, выше.

Приведём данные тестирования ЛКГ тестом «стопка книг» из [4], не останавливаясь на подробном описании каждого из типов ЛКГ.

Таблица 2.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Генератор, тип | Параметр *w* | *Q0.50* | *Q0.95* | Режим R | Длина ПСП, бит |
| ЛКГ (B) | 24 | 52 | 4 | 8 | 1680000 |
| ЛКГ (C) | 24 | 70 | 17 | 1 | 7920000 |
| ЛКГ (D) | 16 | 74 | 16 | 1 | 160000 |
| ЛКГ (E) | 12 | 99 | 97 | 1 | 12000 |
| ЛКГ (F) | 8 | 83 | 80 | 1 | 240 |

Как видно из таблицы 2, тест «стопка книг» находит гарантированные отклонения от случайности за приемлемое время для всех типов ЛКГ как из таблицы 1, так и из таблицы 2. Тем не менее, в работах Д.Кнута [10] варианты ЛКГ B и C успешно прошли испытания на различных статистических тестах, рекомендованных национальным институтом стандартов США NIST [8]. Кроме того, тест «стопка книг» сравнивалась с 15 лучшими статистическими тестами, рекомендованными NIST, явно превосходя их по эффективности [8, 13].

**6. Анализ градиентной статистической атаки на шифр RC6**

**6.1 Описание структуры шифра.**

RC6 – симметричный блочный криптографический алгоритм, производный от алгоритма RC5. Вариант шифра RC6, заявленный на конкурс AES, поддерживает блоки длиной 128 бит и ключи длиной 128, 192 и 256 бит, но сам алгоритм, как и RC5, может быть сконфигурирован для поддержки более широкого диапазона длин как блоков, так и ключей (от 0 до 2040 бит). Является финалистом AES и используется во многих современных пакетных криптосистемах (например, PGP и GPG), поэтому его криптоанализ представляет практический интерес.

Для спецификации алгоритма с конкретными параметрами, принято обозначение RC6-w/r/b, где

* w – длина блока в битах.
* r – число раундов.
* b – длина ключа в битах. Возможные значения: 0..255 байт.

В данной работе рассматривается самый распространённый вариант шифра RC6-128/r/128-256, когда на вход подаются 4 блока по 32 бита, а ключ может принимать 3 длина: 128, 192 и 256 бит. Перед описанием схемы шифрования введём необхо­димые обозначения.

: младшие *n* бит в числе X;

: операция XOR;

: циклический сдвиг а на b бит влево;

: циклический сдвиг а на b бит вправо;

*Si* : *i*-ый раундовый ключ;

R : число раундов шифрования;

(*Ai, Bi, Ci, Di*) : вход *i*-го раунда: четыре 32-битных блока (128 бит);

:;

:;

: конкатенация *х* и *у*.

На начальном этапе секретный 128- (192, 256)-битный ключ преобразуется в массив размера 32-битных раундовых ключа по следующему алгоритму.

Вход (функции генерации раундовых ключей):

* Константа ~Q_{32} = 9E3779B9
* Константа ~P_{32} = B7E15163
* b-байтный ключ, заданный пользователем, предварительно преобразованный в массив из *c* слов .
* *r* – количество раундов.

Алгоритм 1 (функции генерации раундовых ключей):

1. *S0* = *P32*
2. for *i* = 1 to 2*r* + 3 do

*Si* = *Si-1* + *Q32*

1. A = B = *i* = *j* = 0
2. v =
3. for s = 1 to v do:

A = *Si* = (*Si* + A + B) <<< 3

B = *Lj* = (*Lj* + A + B) <<< (A + B)

*i* = (*i* + 1) mod (2*r* + 4)

*j* = (*j* + 1) mod c

С помощью полученного в результате массива раундовых ключей происходит шифрование данных по следующему алгоритму.

Алгоритм 2 (RC6, шифрование):

1. for *i* = 1 to *r* do:

Пункты алгоритма 1, 3 называются операциями «забеливания».

**6.2 Атака на RC6**

В качестве статистического теста для исследования всех раундов шифра RC6 был взят тест «стопка книг» с одной степенью свободы. Параметры теста будем обозначать в соответствии с (5). Все практические результаты будем проводить для квантилей двух уровней доверия: 0,95 и 0,99.

В ходе работы была проверена устойчивость шифра RC6 по отношению к градиентной атаке.  Были рассмотрены 128-битный и 64-битный режимы RC6 cо 128-битным ключом (в ч. 6.2, 6.3 – без операций «забеливания», в ч. 6.4 – модификация для RC6 c операциями «забеливания»). Как видно из алгоритма в ч. 6.1, шифр RC6 устроен таким образом, что он зашифровывает 128-битное число в два этапа. Сначала первые 32 бита шифруются с помощью первого ключа так называемого полураунда, потом следующие 32 бита, используя другой ключ полураунда, оставшиеся 64 бита на данном раунде не меняются (они шифруются на следующем раунде). Исходя этого, можно проводить атаку не на весь раунд, а на каждый полураунд в отдельности, что значительно сокращает трудоёмкость атаки.

В таблице 3 приведены результаты тестирования шифра c помощью «стопки книг». Было проведено 100 тестов со 100 различными случайно подобранными ключами.

Таблица 3.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Раунд | Размер выборки | Параметры теста |  |  |  |
| 3 |  | (24, 10) | 100 | 0 | 6.05\*107 |
| 4 |  | (24, 10) | 100 | 0 | 14775.6 |
| 5 |  | (32, 22) | 16 | 14 | 1.72 |
| 5 | 229 | (32, 22) | 87 | 81 | 157.14 |

Показано, что отклонения от случайности в 16% случаев могут быть зафиксированы на 5 раунде. Это говорит о том, что градиентная атака может эффективно применяться  на 6-ой раунд шифра при размере выборки . Но об эффективности применения атаки на все 100% можно говорить лишь в случае 5 раундов, т.к. на 5 раунде фиксация отклонений от случайности происходит лишь в 16% случаев, а средняя величина *Xi2* ощутимо меньше квантиля уровня доверия 0,95. Так же по тестам на раунде 5 хорошо видно, что 14% ключей дают значения *Xi2* меньше квантиля уровня доверия 0,05, что говорит о высоком уровне случайности выходной последовательности. Фактически можно предположить, что существует определённый класс «слабых» ключей, для которых данная атака может эффективно применяться на 6 раундов, а есть «сильные» ключи, для которых атака эффективна лишь на 5 раундов.

Было проведено исследование зависимости эффективности рассматриваемой атаки от размера выборки и от параметров теста. Результаты исследования представлены в нижеследующей таблице.

Таблица 4.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № теста | Раунд | Размер выборки | Параметры теста |  |  |
| 1 | 4 |  | (8, 7) | 50 | 133.4 |
| 2 | 4 |  | (8, 6) | 50 | 210.9 |
| 3 | 4 |  | (16, 15) | 100 | 1265.0 |
| **4** | **4** |  | **(16, 10)** | **100** | **18484.9** |
| 5 | 4 |  | (8, 7) | 100 | 1653.5 |
| 6 | 4 |  | (8, 6) | 78 | 1204.7 |
| 7 | 4 |  | (16, 15) | 100 | 21142.5 |
| **8** | **4** |  | **(16, 8)** | **100** | **118515.0** |
| 9 | 4 |  | (24, 20) | 65 | 170.4 |
| 10 | 4 |  | (24, 10) | 100 | 14775.7 |
| 11 | 4 |  | (32, 24) | 70 | 36.3 |
| 12 | 4 |  | (32, 16) | 11 | 1.02 |
| **13** | **4** |  | **(16, 8)** | **100** | **123552** |
| **14** | **4** |  | **(24, 10)** | **100** | **83179.5** |
| 15 | 4 |  | (32, 18) | 11 | 1.68 |
| 16 | 4 |  | (32, 24) | 68 | 51.2 |

Из данной таблицы видно, что на каждом отдельном файле заданного размера есть свой оптимальный набор параметров. Этот набор существенно зависит от размера файла. Чем больше размер, тем больше должна быть длина слова в тесте. Сверху длина слова ограничена условием применимости критерия *Xi2*: (S – размер выборки в блоках), но даже при выполнении этого условия, но при слишком большом значении *w* тест даёт плохие результаты. Точно так же и при слишком маленьком значении *w* или слишком маленькой длине слова (*u*), тест плохо распознаёт случайность последовательности. Исходя из представленных данных, оптимальная длина слова *w* равна в среднем: **(6)**, где L – длина входной последовательности в битах (L = ). В таблице жирным шрифтом выделены оптимальные параметры для каждого из размеров последовательности. Оптимальный размер верхней части варьируется и точно его определить нельзя, но в среднем оптимум *u* находится, как **(6’)**. Кроме того, задача подбора пары оптимальных параметров для теста является очень существенной, что ярко видно на примере 14 и 15 тестов, а так же 8 и 12 тестов. При оптимально и правильно подобранных под конкретный размер выборки (не превышающей ) параметрах теста шифр RC6 может быть взломан до 5-6 раундов, а при неправильно подобранных параметрах это число снижается до 3-4 раундов, а в некоторых случаях даже ниже.

Рассмотрим зависимость времени анализа выборки от её размера и параметров теста. В нижеследующей таблице приведены данные для 4 раунда теста и 100 случайно выбранных ключей.

Таблица 5.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № теста | Размер выборки | Параметры теста |  | Время |
| 1 |  | (16, 8) | 11752.6 | 0.14 с |
| 2 |  | (16, 8) | 191344.2 | 1.6 с |
| 3 |  | (16, 8) | 1364622.9 | 24.5 с |
| 4 |  | (24, 23) | 693.4 | 7.2 с |
| 5 |  | (24, 16) | 6008.0 | 5.7 с |
| 6 |  | (24, 12) | 24056.4 | 5.1 с |

В таблице 5 хорошо видна строго линейная зависимость размера выборки от времени анализа файла тестом. Кроме того, данные результаты говорят о том, что с ростом размера выборки растёт величина , при чём растёт она так же, как и время, линейно от размера файла, что существенно. Таким образом, можно сделать предположение о возможности повышения числа взламываемых данным статистическим тестом раундов (число раундов, в которых тест распознаёт отклонения от случайности) с ростом размера файлов. Вполне понятно, что с ростом размера файлов так же будет расти и оптимальный размер слова, что требует дополнительного и существенного расхода памяти. Тесты 4-6 в таблице 5 подтверждают идею оптимального объёма верхней части, как корень квадратный из объёма всего алфавита теста: в случае верхней части, равной по объёму корню квадратному из *u*, значение в 34.7 раза больше. Кроме того, чем меньше верхняя часть, тем быстрее происходит анализ выборки. Соответственно, кроме правильного подбора параметров теста, для взлома шифра требуется также определить оптимальный размер выборки, которая нужна для взлома заданного количества раундов, так как при слишком большой выборке время анализа выборки, а соответственно, и каждого подбираемого ключа, становится слишком большим.

В работе [2] было проведено исследование данной атаки в 64-битном режиме шифра RC6. Полученные результаты представлены в таблице 6.

Таблица 6.

|  |  |  |  |
| --- | --- | --- | --- |
| Раунд | Число тестов | http://www.contrterror.tsure.ru/www/magazine7/05-18-Menyashev-Monarev-Ryabko-Fionov.files/image024.gif | Объем выборки |
| 3 | 100 | 100 | http://www.contrterror.tsure.ru/www/magazine7/05-18-Menyashev-Monarev-Ryabko-Fionov.files/image026.gif |
| 4 | 100 | 100 | http://www.contrterror.tsure.ru/www/magazine7/05-18-Menyashev-Monarev-Ryabko-Fionov.files/image026.gif |
| 5 | 100 | 100 | http://www.contrterror.tsure.ru/www/magazine7/05-18-Menyashev-Monarev-Ryabko-Fionov.files/image022.gif |
| 6 | 20 | 16 | http://www.contrterror.tsure.ru/www/magazine7/05-18-Menyashev-Monarev-Ryabko-Fionov.files/image028.gif |

Из приведённых данных хорошо видно, что в этом режиме количество раундов, в которых статистический тест «стопка книг» выявляет отклонения от случайности в среднем выше на 1, т.е. в 64-битном варианте RC6 число взламываемых раундов равно 6-7. Таким образом, становится ясно, что эффективность атаки существенно зависит от размера блока в шифре: чем он больше, тем менее эффективна атака.

**6.3 Программная реализация и оценка эффективности атаки**

До сих пор в рассмотрение попадала работа шифра RC6 на выборках длиной до . Из таблицы 2 явно видно, что эффективность работы теста, а соответственно, и всей атаки существенно зависит от объёма выборки: при объёме выборки   
 число взламываемых раундов равно уже 4 (для 4 раунда = 25). Исходя из этого, можно сделать предположение о том, что для повышения числа взламываемых раундов на 1, надо увеличивать выборку в 28 раз **(7)**. Тем не менее, возникает законный вопрос: можно ли взламывать градиентной статистической атакой 128-битный шифр RC6 на большем, чем 5-6, числе раундов? Ответ на данный вопрос содержится в исследованиях американского института стандартов NIST. С применением менее эффективных, чем «стопка книг» статистических тестов, они получили следующие данные, касательно взлома RC6 при помощи градиентной статистической атаки.

Таблица 7.

|  |  |  |  |
| --- | --- | --- | --- |
| Раунд | Размер ключа | Объём выборки | Расход памяти |
| 12 | 128 | 294 | 242 |
| 14 | 128 | 2118 | 2112 |
| 14 | 192 | 2110 | 242 |
| 14 | 256 | 2108 | 274 |
| 15 | 256 | 2119 | 2138 |

Данные таблицы 7 можно интерпретировать следующим образом. Число взламываемых раундов 128-битного шифра RC6 на домашнем компьютере составляет 5-6. В общем случае, можно взламывать потенциально весь шифр, т.е. все 20 раундов, но для этого потребуется на данный момент технически недостижимый объём памяти. Несмотря на то, что в исследовании, результаты которого представлены в таблице 7, принимал участие статистический тест, по своим показателям уступающим «стопке книг», тем не менее исходя из данных, полученных на «стопке книг», можно сделать вывод о том, что на высоких раундах разница со стандартными статистическими тестами не очень существенная. Тем не менее, для точного ответа на данный вопрос требуется провести исследование рассматриваемой атаки на суперкомпьютере, после чего сравнить результаты с другими тестами.

Ответим теперь на вопрос: какое количество раундов можно взломать на современном этапе развития вычислительной техники, а также, какова интерполяционная зависимость числа раундов от потребностей в памяти, если учесть, что данные таблицы 7 соответствуют тесту «стопка книг»?

Будем называть шифр с *n* раундами эффективно взламываемым градиентной статистической атакой, если с применением этой атаки количество операций по взлому шифра будет существенно меньше полного перебора, т.е. если для неё выполнено условие (2), (2’) или (2’’).

Так как раунды шифра RC6 можно разбить на полураунды, то реализуя на нём рассматриваемую атаку, можно подбирать ключи для каждого из полураундов в отдельности. В 128-битном варианте шифра RC6 каждый ключ полураунда имеет длину 32 бита, т.е. для подбора ключа к раунду *r* требуется в худшем случае операций, где *mr* – размер выборки, подаваемой на вход дешифратора *r*-го раунда. При этом, стоит отметить, что подбирая ключи к полураундам, следует анализировать не всю выходную последовательность, а только чётные/нечётные блоки, так как за операцию одного полураунда меняется только половина последовательности (чётные/нечётные блоки) и анализировать остальные смысла нет.

Исходя из предположения (4), а так же данных таблицы 7, можно сделать вывод о том, что объём выборки, при которой статистический тест «стопка книг» может успешно применяться при реализации градиентной атаки, действительно равен **(8)**. Число R при этом может быть потенциально любым. Отсюда легко сделать оценку для числа операций: . Следует заметить, что вышеприведённая оценка справедлива лишь для взлома самого старшего раунда шифра, тогда как атака должна применяться на каждом раунде, начиная со старшего к младшему. Тем не менее, эту оценку можно применять и для всей атаки, а не только для старшего раунда. Поясним почему. Исходя из формулы (1’’) видно, что число операций линейно зависит от размера выборки mi. Одновременно, из формулы (5) следует, что на каждом следующем – более младшем – раунде объём выборки для взлома уменьшится в раз. Таким образом, уже второй – следующий – член суммы ряда в (1’’) на ~3 порядка меньше, т.е. его вклад в сумму слишком мал. Расход памяти в байтах будет равен **(8’)**, где *(u,w)* – параметры теста. Такая оценка памяти подтвердилась экспериментально и является точной. Из (6), (6’) и (8) следует, что и из (5) получаем:

. По этим формулам, а так же данным из таблиц 4 и 7 составим нижеследующую таблицу 8.

Таблица 8.

|  |  |  |  |
| --- | --- | --- | --- |
| Раунды | Количество операций | Расход памяти (ОЗУ) | Расход памяти (ПЗУ) |
| 5 |  |  |  |
| 7 |  |  |  |
| 9 |  |  |  |
| 12 |  |  |  |
| 14 |  |  |  |
| 15 |  |  |  |
| 20 |  |  |  |

**6.4 Модификация атаки для варианта RC6 с операциями «забеливания»**

Методы градиентной атаки, предложенные выше, рассматривались в применении к шифру RC6 без операций пред- и пост- забеливания, то есть фактически – в отношении неполного варианта шифра. Рассмотрим модифицированную версию атаки, которую можно применять к полному варианту шифра RC6 [3].

Процесс шифрования блока (*Ai, Bi, Ci, Di*), описанного ч. 6.1, можно представить следующим образом (приведём пример для нечётного случая): , **(9)**. Отсюда видно, что раундовые ключи в функции шифрования *G* не участвуют, а только прибавляются в виде обыкновенных слагаемых к конечному результату. При этом, преобразование *G* является обратимым и не требует для обращения никаких ключей. Таким образом, если подать на вход *i*-го полураунда последовательность, к которой применено обратное преобразование, т.е. преобразование вида: , то на выходе (*i+1*)-го полураунда получим последовательность стандартного входа в *i*-ый раунд плюс подключи (*i+1*)-го полураунда., которые сами по себе оказывают слабое влияние на статистические свойства последовательности. Применим данную схему для первого полураунда шифра RC6, а на вход будем подавать последовательность вида:

; **(10)**, где – последовательность вида (1’), а *S0*, *S1* – ключи пред-забеливания. Тогда на выходе 1 раунда получим последовательность (в этом легко убедиться подстановкой данного выражения в алгоритм 2), что ведёт к тому, что на выходе *r*-го раунда мы получим последовательность статистически равную последовательности на выходе -го раунда.

Модифицированная версия атаки сводится к тому, что на вход мы подаём последовательность вида не (1’), а вида (10), что ведёт к фактическому поступлению на вход второго раунда последовательности вида (1’) и сокращению раундов шифрования на единицу. При этом, если в последовательности (10) правильно подобраны подключи (*S0*, *S1*), то результат должен давать статистику по раунду; если же они подобраны неверно, то результат должен давать статистику (значение ), равную *r* раундам шифра. После подбора подключей (*S0*, *S1*), подбираем по тому же принципу подключи (*S2*, *S3*) и так далее. Таким образом, мы можем подбирать подключи, начиная с начала, а не с конца, как это было в традиционном варианте атаки.

Понятно, что если подбирать ключи сразу парой, то сложность перебора составит , т.к. все подключи имеют длину в 32 бита, поэтому можно модифицировать текущий вариант так, чтобы подключи подбирались раздельно. Для этого просто будем подбирать вначале первый подключ: смотреть, на каком из них значение максимально, затем – второй по тому же принципу. Как показывают полученные результаты (таблицы 9 и 10), данный подход имеет практическую эффективность.

Итак, модифицированная атака на RC6 со сложностью перебора :

* 1. Для каждого *Si* вычисляем последовательность, вида (8) для *i*-го раунда при константном *Si+1* и прибавляем к ней последовательность, полученную после шифрования на *i* раундов, подключи которых уже найдены.
  2. Подаём полученную последовательность на вход шифру и вычисляем значение статистики .
  3. Выбираем *Si*, для которого значение максимально (является существенно большим, чем для класса неправильных подключей).
  4. Фиксируем правильный *Si* и возвращаемся на шаг 1 для подбора *Si+1*.
  5. После подбора обоих подключей переходим на раунд (*i*+1).

Приведём результаты экспериментов. В таблице 9 приведены статистические результаты модифицированной версии атаки с использованием правильных и неправильных подключей последнего раунда.

При этом брались пары правильных подключей (*S0*, *S1*) и для каждой пары проверялись 100 неправильных пар подключей. В первой колонке представлены номера раундов.

Таблица 9.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Размер выборки | Значение (прав. ключи) | Максимальное (неправ. ключи) | *E*()  (неправ. ключи) | > Q0.99 (неправ. ключи) |
| 1 | 214 | 49.64 | 12.76 | 2.63 | 9 |
| 2 | 216 | 218.07 | 12.39 | 2.89 | 9 |
| 3 | 218 | 940.99 | 18.20 | 3.15 | 28 |
| 4 | 220 | 3621.84 | 20.69 | 3.06 | 40 |
| 5 | 222 | 15360.21 | 25.42 | 3.37 | 68 |

Основные выводы:

* Разница между случайностью при правильно подобранных и неправильно подобранных ключах раунда растёт линейно от размера выборки.
* При достаточном размере выборки значимость уровня доверия квантиля отпадает (ищем максимальное ).

Далее – в таблице 10 – приведены результаты модифицированной атаки на 5-ый раунд шифра. При этом было взято 5 подключей (*S0*, *S1*), в которых *S0* был правильным (*S1* – произвольный), и для каждой пары проверено 100 неправильных пар подключей.

Таблица 10.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Размер выборки | Минимальное  (прав. ключи) | Максимальное (неправ. ключи) | *E*()  (неправ. ключи) |
| 1 | 220 | 27930 | 3.51 | 1.51 |
| 2 | 220 | 2127 | 80.93 | 3.38 |
| 3 | 220 | 1891 | 17.06 | 2.31 |
| 4 | 220 | 2776 | 16.87 | 1.64 |
| 5 | 220 | 1821 | 6.72 | 2.36 |

Ключи пост-забеливания можно найти после подбора всех раундовых ключей и ключей пред-забеливания простым раздельным перебором до получения исходной последовательности вида (1’) на минимальном количестве блоков шифра (напр. 22).

**6.5 Модификация атаки с уменьшением длины раундового ключа**

Из вида функции шифрования RC6 (9) видно, что подключи *Si* и *Si+1* участвуют только в явном виде, как слагаемые, изменяющие часть всего блока. Соответственно, если большая часть ключа полураунда (а не весь ключ целиком) раунда подобрана правильно, итоговая последовательность должна стать менее случайной, но её мера случайности должна быть больше, чем при полностью правильно подобранном ключе полураунда. Этот факт позволяет разработать следующую модификацию атаки: вначале подбираем ключ полураунда, перебирая только первые (32 – *p*) бит, где *p* – количество урезанных бит, начиная с конца, – параметр метода, находим часть ключа с максимальным значением , далее подбираем оставшиеся *p* бит, максимизирующие . Если же все промежуточные значения оказываются меньше исходного (или не превышают квантиль заданной значимости), то подбираем подключ по традиционному алгоритму. Сложность метода тогда будем считать, как мат.ожидание от традиционной трудоёмкости: **(11)**, где *p* – вероятность успеха модифицированноuj варианта атаки.

В таблице 11 представлены результаты атаки заданным методом. Было произведено шифрование последовательности размера 225 элементов (т.е. 229 бит для длины слова = 16 бит) на 5 раундов и далее, для 100 различных ключей было произведено 100 попыток найти первый ключ полураунда. В таблице представлен результат до дешифрования (*E*() 1), промежуточный (после подбора части подключа: *E*() 2) и конечный: после подбора всего подключа. Также приведено количество успешных попыток подбора подключа таким способом. Использованные параметры статистического теста: (24, 16).

Таблица 11.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Количество урезанных бит | *E*()  1 | *E*()  2 | *E*()  3 | Монотонных последовательностей |
| 2 из 32 | 10,35 | 283339 | 479478 | 88 |
| 4 из 32 | 10,35 | 172447 | 479478 | 79 |

Исходя из (11), получаем трудоёмкость модифицированного варианта:

;

.

В общем итоге, видно, что вариант с 4 урезанными битами – лучший и снижает сложность атаки на любой раунд шифра в 4 раза. В частности, сложность атаки на 5 раунд шифра может быть уменьшена с 248 до 246.

**6.6 Итоги.**

Подведём основные итоги анализа эффективности градиентной статистической атаки на шифр RC6:

* Найти ключи пред- и пост- забеливания можно с трудоёмкостью = T(последнего раунда) в терминах *О*-символики, т.е. с трудоёмкостью традиционной градиентной атаки: , хотя точная трудоёмкость атаки возрастает из-за лишних операций обращения последовательности.
* Наличие операций пред- и пост- забеливания не усложняет атаку по сравнению с традиционным вариантом, если использовать модификацию, описанную в ч.6.4.
* Модификации атаки, описанные в ч.6.4 и ч.6.5 можно совместить, что снизит трудоёмкость традиционного варианта атаки в 4 раза, позволив при том находить ключи забеливания.
* Полученные результаты можно интерпретировать следующим образом. При доступных на текущий момент вычислительных мощностях градиентная статистическая атака с применением теста «стопка книг» может быть применена к 9-раундовому шифру RC6, для этого потребуется: 32 Гб ОЗУ, 32 Тб ПЗУ, а также операций перебора. Тогда как в случае атаки полным перебором понадобится операций. Начиная с 13 раунда, применять градиентную статистическую атаку имеет смысл лишь в случае использования 192-/256-битных ключей – это следует из (2’’). Для 19-20 раундов – лишь в случае 256-битных ключей. Также можно интерпретировать эти результаты по-другому: при 128-битном ключе и доступных ресурсах 32 Гб ОЗУ, 32 Тб ПЗУ, количество эффективно взламываемых градиентной атакой раундов равно 10 (87+32 = 119 < 128). При 192-битном ключе число эффективно взламываемых раундов увеличивается до 12 (87 + 32\*3 = 183 < 192). А в случае 256-битного ключа это число равно 14 (87 + 5\*32 = 247 < 256).

**7. Заключение**

В результате проделанной работы были получены множественные результаты по атакам и анализу криптографических свойств пяти различных блоковых шифров: RC6, MARS, IDEA, CAST-128, Blowfish; исследованы свойства линейного конгруэнтного генератора, а также особенностей применения статистического теста «стопка книг». В процессе исследовательской работы были разработаны новые модификации градиентной статистической атаки и инновационные методы её применения в случае конкретных видов шифров. Также были выявлены и практически обоснованы теоретические зависимости между такими величинами, как эффективность атаки на примере конкретных шифров, временная и аппаратная сложность, параметры применения атаки.

Итоговые результаты по атакам на шифры приведены в ниже следующей таблице.

|  |  |  |  |
| --- | --- | --- | --- |
| Название шифра | Максимальный раунд | Лучшие опубликованные результаты | Трудоёмкость |
| RC6 | 5/6 | 5 | 246/253 |
| MARS | 21/22 | 16 | 256/2120 |
| CAST-128 | 3-4 | – | 263 |
| IDEA | 2 | 6 | 2108 |
| Blowfish (стат. отклон.) | 3 | 3 | 225 |

Как видно из данных таблицы, по шифрам RC6, MARS и CAST-128 были получены результаты, превосходящие все ранее известные. Также результаты анализа тестом «стопка книг» линейного конгруэнтного генератора показали полную уязвимость его статистических свойств во всех модификациях (что также превосходит ранее известные результаты).

**Литература**

1. Knudsen L., Meier W. Correlations in RC6 with a reduced number of rounds // FSE 2000. LNCS 1978(2000). Springer–Verlag. P. 94–108.
2. А.А. Меняшев, В.А. Монарев, Б.Я. Рябко, А.Н. Фионов: «Применение градиентной статистической атаки к блоковым шифрам RC5, RC6 И SAFER» // Научно-практический журнал «Информационное противодействие терроризму», 2006, №7.
3. В.А. Монарёв, А.М. Лубкин: «Эффективная атака на блоковый шифр RC6» // Вестник СибГУТИ, 2010, №4.
4. Б.Я. Рябко, А.Н. Фионов: «Криптографические методы защиты информации», 2006, Москва.
5. Б.Я. Pябко, В.А. Монарев, Ю.И. Шокин. Новый тип атак на блоковые шифры // «Проблемы передачи информации», т. 41, н.4., 2005, с.181– 128.
6. Б. Шнайер: «Прикладная криптография» // N.-Y. Wiley, 1996.
7. B.Ya. Ryabko, V.A. Monarev. Using information theory approach to randomness testing // Journal of Statistical Planning and Inference, 2005, -Vol. 133, № 1, -PP. 95-110.
8. Рябко Б.Я., Пестунов А.И. «Стопка книг» как новый статистический тест для случайных чисел // Пробл. передачи информации. 2004. Т. 40, вып. 1. С. 73-78.
9. L'EcuYER P. Tables of linear congruential generators of different sizes and good lattice structure // Math. of Comp. 1999. Vol. 68. P. 249-260.
10. Кнут Д.Э. Искусство программирования. Т. 2: Получисленные алгоритмы. М.: Изд. дом "Вильямс", 2000.
11. Pestunov A. Statistical Analysis of the MARS Block Cipher // Cryptology ePrint Archive. Report 2006/217. 2006. <http://eprint.iacr.org/2006/217>.
12. Ryabko B., Monarev V. Using information theory approach for randomness testing // J. of Statistical Planning and Reference. 2005. Vol. 133, N 1. P. 95-110.
13. Doroshenko S., Ryabko B. The experimental distinguishing attack on RC4 // Cryptology ePrint Archive. Report 2006/070. 2006.<http://eprint.iacr.org/2006/070>**.**
14. А.И. Пестунов. Теоретическое исследование свойств статистического теста «стопка книг». Ж. Вычислительные технологии №6, Т.11, 2006 г.