МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт искусственного интелекта
Базовая кафедра №252 – информационной безопасности

ПРАКТИЧЕСКАЯ РАБОТА

Тема практической работы: «Алгоритм шифрования RC 5 XOR»

Студент группы ККСО-05-21	Копытин А.А.	_____ (*подпись*)
Руководитель практической работы	старший преподаватель Плешаков А.С.	_____ (*подпись*)
Работа представлена к защите	<___>_____2022 г.	
Допущен к защите	<___>_____2022 г.	

Москва - 2022

Содержание

1. Введение

Блочный шифр — разновидность симметричного шифра, оперирующего группами бит фиксированной длины — блоками, характерный размер которых меняется в пределах 64–256 бит. Если исходный текст (или его остаток) меньше размера блока, перед шифрованием его дополняют. Фактически, блочный шифр представляет собой подстановку на алфавите блоков, которая, как следствие, может быть моно- или поли алфавитной. Блочный шифр является важной компонентой многих криптографических протоколов и широко используется для защиты данных, передаваемых по сети.

К достоинствам блочных шифров относят сходство процедур шифрования и расшифрования, которые, как правило, отличаются лишь порядком действий. Это упрощает создание устройств шифрования, так как позволяет использовать одни и те же блоки в цепях шифрования и расшифрования. Гибкость блочных шифров позволяет использовать их для построения других криптографических примитивов: генератора псевдослучайной последовательности, поточного шифра, имитовставки и криптографических хешей.

2. RC 5

RC5 (*Ron's Code 5* или *Rivest's Cipher 5* — это блочный шифр, разработанный в 1994 году
Роном Ривестом из компании RSA Security Inc. с переменным количеством раундов, длин ой блока и длиной ключа. Это расширяет сферу использования и упрощает переход на бол ее сильный вариант алгоритма.

Существует несколько различных вариантов алгоритма, в которых преобразования в «пол-раундах» классического RC5 несколько изменены. В классическом алгоритме используются три примитивных операции и их инверсии:

сложение по модулю 2^w

побитовое исключающее или (XOR)

операции циклического сдвига на переменное число бит (x <<< y)

Основным нововведением является использование операции сдвига на переменное число бит, не использовавшиеся в более ранних алгоритмах шифрования. Эти операции одинаково быстро выполняются на большинстве процессоров, но в то же время значительно усложняют дифференциальный и линейный криптоанализ алгоритма.

Шифрование по алгоритму RC5 состоит из двух этапов. Процедура расширения ключа и непосредственно шифрование. Для расшифрования выполняется сначала процедура расширения ключа, а затем операции, обратные процедуре шифрования. Все операции сложения и вычитания выполняются по модулю 2^w.

2.1 Параметры

Т.к. алгоритм RC5 имеет переменные параметры, то для спецификации алгоритма с конкретными параметрами принято обозначение RC5-W/R/b, где

- W — половина длины блока в битах, возможные значения 16, 32 и 64. Для эффективной реализации величину W рекомендуют брать равным машинному слову. Например, для 32-битных платформ оптимальным будет выбор W=32, что соответствует размеру блока 64 бита.
- R — число раундов, возможные значения от 0 до 255. Увеличение числа раундов обеспечивает увеличение уровня безопасности шифра. Так, при R=0 информация шифроваться не будет. Также алгоритм RC5 использует таблицу расширенных ключей размера слов, которая получается из ключа, заданного пользователем.
- b — длина ключа в байтах, возможные значения от 0 до 255.

2.2 Расширение ключа

Перед непосредственно шифрование или расшифрованием данных сп выполняется процедура расширения ключа. Процедура генерации ключа состоит из четырёх этапов:

Генерация констант

Разбиение ключа на слова

Построение таблицы расширенных ключей

Перемешивание

Генерация констант

Для заданного параметра W генерируются две псевдослучайные величины используя две математические константы: e (экспонента) и f (Золотое сечение).

$$Q_w \leftarrow Odd((f-1)*2^w)$$
$$P_w \leftarrow Odd((e-1)*2^w)$$

где Odd() — это округление до ближайшего нечетного целого.

Для w = 16, 32, 64 получатся следующие константы:

$$P_{16}=1011011111100001_2=B7E1_{16}$$
$$Q_{16}=1001111000110111_2\_\{2\}=9E37_{16}$$
$$P_{32}=10110111111000010101000101100011_2=B7E15163_{16}$$
$$Q_{32}=10011110001101110111100110111001_2=9E3779B9_{16}$$
$$P_{64}=B7E151628AED2A6B_{16}$$
$$Q_{64}=9E3779B97F4A7C15_{16}$$

Разбиение ключа на слова

На этом этапе происходит копирование ключа $K_0...K_{b-1}$ в массив слов $L_0...L_{c-1}$, где b/u, где u= W/8, то есть, количество байт в слове.

Если b не кратен W/8, то L дополняется нулевыми битами до ближайшего большего размера c, кратного W/8.

В случае если b=c=0, то мы устанавливаем значение c=1, а $L_0=0$

Построение таблицы расширенных ключей

На этом этапе происходит построение таблицы расширенных ключей $S_0...S_{2*(R+1)-1}$, которое выполняется следующим образом:

$$S_0=P_w$$
$$S_{i+1}=Si+Q_w$$

Перемешивание

Циклически N раз выполняются следующие действия:

$$G=S_i= (S_i+G+H) <<< 3$$
$$H=L_i=(L_i+G+H) <<< (G+H)$$
$$i=(i+1) \bmod (2(R+1))$$
$$j=(j+1) \bmod c,$$

причем G,H,i,j — временные переменные, начальные значения которых равны 0.

Количество итераций цикла N — это максимальное из двух значений 3 * c и (3 * 2 * (R+1)).

2.3 Анализ параметризации

В этом разделе мы обсудим обширную параметризацию, которую предоставляет RC5.

Прежде всего, следует отметить, что не предполагается, что RC5 будет безопасным для всех возможных значений параметров. Например, R = 0 практически не обеспечивает шифрования, а R = 1 легко взломать. И выбор b = 0 явно не дает никакой безопасности.

С другой стороны, выбор максимально допустимых значений параметров был бы излишним для большинства приложений.

Мы допускаем диапазон значений параметров, чтобы пользователи могли выбрать алгоритм шифрования, безопасность и скорость которого оптимизированы для их приложения, предоставляя при этом эволюционный путь для корректировки своих параметров по мере необходимости в будущем.

Выбор R влияет как на скорость шифрования, так и на безопасность. Для некоторых приложений высокая скорость может быть наиболее критичным требованием — требуется наилучшая безопасность, достижимая в пределах заданного времени шифрования. Выбор небольшого значения R (скажем, R = 6) может обеспечить некоторую внутреннюю безопасность, хотя и скромную, в рамках заданного ограничения скорости.

В других приложениях, таких как управление ключами, основной задачей является безопасность, а скорость не имеет большого значения. Выбор R = 32 раундов может подойти для таких приложений. Поскольку RC5 является новой конструкцией, необходимы дальнейшие исследования для определения безопасности, обеспечиваемой различными значениями R. Пользователи RC5 могут захотеть скорректировать значения R, которые они используют, на основе результатов таких исследований.

Точно так же размер слова W также влияет на скорость и безопасность. Например, выбор значения u, превышающего размер регистра ЦП, может снизить скорость шифрования. Размер слова W = 16 в первую очередь предназначен для исследователей, которые хотят изучить свойства безопасности естественного «уменьшенного» RC5. По мере того, как 64-битные процессоры становятся обычным явлением, можно переходить на RC5-64 как естественное расширение RC5-32. Также может быть удобно указать W = 64 (или больше), если RC5 будет использоваться в качестве основы для хеш-функции, чтобы иметь 128-битные (или больше) блоки ввода/вывода.

Может показаться необычным и рискованным указывать алгоритм шифрования, допускающий небезопасный выбор параметров. У нас есть два ответа на эту критику:

- Фиксированный набор параметров может быть не менее опасен, поскольку параметры нельзя увеличить при необходимости.
- Ожидается, что разработчики предоставят реализации, обеспечивающие выбор достаточно больших параметров. Хотя в принципе небезопасный выбор можно использовать, на практике он будет запрещен.

Не ожидается, что типичная реализация RC5 будет работать с любым блоком управления RC5. Скорее, это может работать только для определенных фиксированных значений параметров или параметров в определенном диапазоне. Параметры W, R и b в полученном или переданном управляющем блоке RC5 затем используются просто для проверки типа — значения, отличные от поддерживаемых реализацией, будут запрещены. Таким образом, гибкость RC5 используется на этапе проектирования системы, когда параметры выбираются, а не во время выполнения, когда неподходящие параметры могут

быть выбраны неосторожным пользователем.

Наконец, отметим, что RC5 может использоваться в некоторых приложениях, не требующих криптографической защиты.

2.4 Шифрование

Перед первым раундом выполняются операции наложения расширенного ключа на шифруемые данные:

$$A = (A + S_0) \bmod 2^w$$
$$B = (B + S_1) \bmod 2^w$$

В каждом раунде выполняются следующие действия:

$$A = ((A \oplus B) <<< B) + S_{2i}$$
$$B = ((B \oplus A) <<< A) + S_{2i+1},$$

где i —- номер текущего раунда начиная с первого, S_n —- фрагмент расширенного ключа, <<< n —- операция циклического сдвига на n битов влево.

Алгоритм поразительно прост — в нем используются только операции сложения по модулю 2 и по модулю 2^w, а также сдвиги на переменное число битов. Последняя из операций представляется автором алгоритма как революционное решение, не использованное в более ранних алгоритмах шифрования (до алгоритма RC5 такие использовались только в алгоритме Madryga, не получившем широкого распространения), — сдвиг на переменное число битов является весьма просто реализуемой операцией, которая, однако, существенно усложняет дифференциальный и линейный криптоанализ алгоритма. Простота алгоритма может рассматриваться как его важное достоинство — простой алгоритм легче реализовать и легче анализировать на предмет возможных уязвимостей.

2.5 Расшифрование

Для расшифрования данных используются обратные операции, то есть для i = R, R-1,...,1 выполняются следующие раунды:

$$B = ((B - S_{2i+1}) >>> A) \oplus A$$
$$A = ((A - S_{2i}) >>> B) \oplus B,$$

где >>> —- операция циклического сдвига на n битов вправо.

После выполнения всех раундов, исходное сообщение находится из выражения:

$$B = (B - S_{2i}) \bmod 2^w$$
$$A = (A - S_0) \bmod 2^w$$

3 Цели RC5

C5 был разработан с учетом следующих целей:

- RC5 должен быть симметричным блочным шифром. Тот самый секретный криптографический ключ используется для шифрования и для дешифрования. Открытый текст и зашифрованный текст - битовые последовательности фиксированной длины (блоки).
- RC5 должен подходить для аппаратного или программного обеспечения. Это

означает, что в RC5 должны использовать только примитивные вычислительные операции, обычно встречающиеся в типичных микропроцессорах.

- RC5 должен быть быстрым. Это подразумевает, что RC5 ориентирован на слова: базовыми вычислительными операциями должны быть операторы, работающие на полных словах.
- RC5 должен быть адаптирован к процессорам с различной длиной слова. Например, когда станут доступны 64-битные процессоры, RC5 сможет использовать их большую длину слова. Следовательно, количество битов в слове является параметром RC5; различные варианты выбора этого параметра приводят к различным алгоритмам RC5.
- RC5 должен быть итеративным по структуре, с переменным количеством раундов. Пользователь может явно выбирать между более высокой скоростью и более высокой безопасностью. Количество раундов R является вторым параметром RC5.
- RC5 должен иметь криптографический ключ переменной длины. Пользователь может выбрать уровень безопасности, подходящий для его приложения, или в соответствии с внешними соображениями, такими как экспортные ограничения. Таким образом, длина ключа b (в байтах) равна третий параметр RC5
- RC5 должен быть простым. Его должно быть легко реализовать. Что еще более важно, простая структура, возможно, более предпочтительна для анализа и оценки, так что криптографическую стойкость RC5 можно определить быстрее.
- RC5 должен иметь низкие требования к памяти, чтобы его можно было легко реализовать на смарт-картах или других устройствах с ограниченным объемом памяти. (И последнее, но не менее важное!) RC5 должен обеспечивать высокую безопасность при выборе подходящих значений параметров.
- RC5 должен подчеркивать использование ротации, зависящей от данных, и поощрять оценку криптографической стойкости, которую может обеспечить ротация, зависящая от данных.
-

4 Криптостойкость

RSA потратила много времени на анализ его работы с 64-битным блоком. Так в период с 1995 по 1998 г. они опубликовали ряд отчётов, в которых подробно проанализировали криптостойкость алгоритма RC5. Оценка для линейного криптоанализа показывает, что алгоритм безопасен после 6 раундов. Дифференциальный криптоанализ требует 2^{24} выбранных открытых текстов для алгоритма с 5 раундами, 2^{45} для 10 раундов, 2^{53} для 12 раундов и 2^{68} для 15 раундов. А так как существует всего лишь 2^{64} возможных различных открытых текстов, то дифференциальный криптоанализ невозможен для алгоритма в 15 и более раундов. Так что рекомендуется использовать 18-20 раундов, или по крайней мере не меньше 15 вместо тех 12 раундов которые рекомендовал сам Ривест.

RSA Security Challenge

Для стимуляции изучения и применения шифра RC5 RSA Security 28 января 1997 года предложила взломать серию сообщений, зашифрованных алгоритмом RC5 с разными параметрами, назначив за взлом каждого сообщения приз в 10 000$. Шифр с самыми слабыми параметрами RC5-32/12/5 был взломан в течение нескольких часов. Тем не менее, последний осуществлённый взлом шифра RC5-32/12/8 потребовал уже

5 лет вычислений в рамках проекта распределённых вычислений RC5-64 (здесь 64=b8, длина ключа в битах) под руководством distributed.net. По-прежнему неприступными пока остаются RC5-32/12/b для b от 9 до 16. distributed.net запустил проект RC5-72 для взлома RC5-32/12/9, в котором по состоянию на октября 2013 года удалось перебрать около 3% ключей.

Атака по времени выполнения

На платформах, где операция циклического сдвига на переменное число битов выполняется за различное число тактов процессора, возможна атака по времени исполнения на алгоритм RC5. Два варианта подобной атаки были сформулированы криптоаналитиками Говардом Хейзом и Хеленой Хандшух (англ. Helena Handschuh). Они установили, что ключ может быть вычислен после выполнения около 220 операций шифрования с высокоточными замерами времени исполнения и затем от 228 до 240 пробных операций шифрования. Самый простой метод борьбы с подобными атаками — принудительное выполнение сдвигов за постоянное число тактов (например, за время выполнения самого медленного сдвига).

Наиболее реальным методом взлома алгоритма RC5 (не считая варианты с небольшим количеством раундов и с коротким ключом) является полный перебор возможных вариантов ключа шифрования. Что означает, что у алгоритма RC5 практически отсутствуют недостатки с точки зрения его стойкости.

5 Варианты алгоритма

Т.к. одним из свойств RC5 является его простота в реализации и анализе, вполне логич но, что многие криптологи захотели усовершенствовать классический алгоритм. Общая ст руктура алгоритма оставалось без изменений, менялись только действия выполняемые над каждым блоком в процессе непосредственно шифрования. Так появилось несколько разли чных вариантов этого алгоритма.

6 RC5XOR

В этом алгоритме сложение с ключом раунда по модулю заменено операцией XOR:

Этот алгоритм оказался уязвим к дифференциальному и линейному криптоанализу. Бир юкову и Кушилевицу удалось найти атаку методом дифференциального криптоанализа дл я алгоритма RC5XOR-32/12/16, используя 228 выбранных открытых текстов.

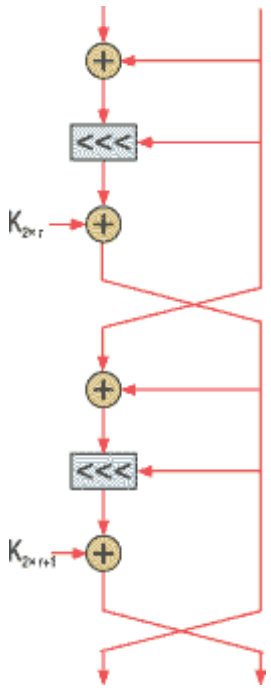

Рис.1

Алгоритм RC5XOR, в котором сложение с ключом раунда по модулю 2 заменено операцией XOR (рис.1):

A = ((A (+) B) <<< B) E K2*r mod 2w **.

Здесь и далее в качестве примера приведено только преобразование для вычисления левого субблока; правый вычисляется в следующей половине раунда аналогичным образом.

Данный алгоритм оказался менее стоек, чем RC5, как к линейному, так и к дифференциальному криптоанализу. В частности, Бирюков и Кушилевиц предложили атаку методом дифференциального криптоанализа, вскрывающую алгоритм RC5XOR-32/12/16 на основе 228 выбранных открытых текстов.