# лабораторная работа №3 “Атака «Дня Рождения»”

## Цель работы

Целью работы является знакомство с классическими атаками на шифрование с использованием алгоритма RSA.

## Основные сведения

**Парадокс дня рождения**

Парадокс дней рождения — утверждение, что если дана группа из 23 или более человек, то вероятность того, что хотя бы у двух из них дни рождения (число и месяц) совпадут, превышает 50 %. Для группы из 60 или более человек вероятность совпадения дней рождения хотя бы у двух её членов составляет более 99 %, хотя 100 % она достигает, когда в группе не менее 366 человек (с учётом високосных лет — 367). Такое утверждение может показаться противоречащим здравому смыслу, так как вероятность одному родиться в определённый день года довольно мала, а вероятность того, что двое родились в конкретный день — ещё меньше, но является верным в соответствии с теорией вероятностей. Таким образом, оно не является парадоксом в строгом научном смысле — логического противоречия в нём нет, а парадокс заключается лишь в различиях между интуитивным восприятием ситуации человеком и результатами математического расчёта.

**Атака "Дня рождения"**

Применяя так называемый "парадокс дня рождения" к слабым хэш-функциям формулируется следующая задача:

Предположим, количество выходных значений хэш-функции равно . Вычислим минимальное значение , такое что для конкретного значения и значений вероятность того, что хотя бы для одного выполнялось равенство была бы больше .

Для одного вероятность того, что , равна . Т.е. соответственно , , для 64, 128, 160 и т.д. -битных алгоритмов. Таким образом, вероятность того, что , равна .

Если создать значений, то вероятность того, что ни для одного из них не будет совпадений, равна произведению вероятностей, соответствующих одному значению, т.е. . Следовательно, вероятность, по крайней мере, одного совпадения равна . Разложив в биноминальный ряд Ньютона и, затем, аппроксимируя его придём к упрощённой формуле , с искомой вероятностью имеем .

Таким образом для *m*-битового хэш-кода достаточно выбрать сообщений, чтобы вероятность совпадения хэш-кодов была заведомо больше .

На примере слабой хэш-функции в MySQL (64-битный хеш) всё множество её значений вариантов. Чтоб найти совпадение с исходным паролем с вероятностью потребуется подать на вход случайных паролей.

В контексте применимости вычислений на основе технологии CUDA, примем среднюю скорость вычислений одного хеша 8 000 000 000 000 п/c. на GF8600GT получим время 1152921сек.~320часов.~13 дней. Это означает что можно не искать исходный пароль, последовательно подавая на вход пароли, генерируемые простым брутфорсом (последовательно) а подавать случайное значение пароля из широкого диапазона > чем число возможных хэшей и в 50% через 13 дней непрерывного брута мы получаем хэш, который совпадёт с хэшем от исходного пароля.

**Поиск коллизий**

Теперь рассмотрим следующую задачу: обозначим вероятность того, что в множестве из элементов, каждый из которых может принимать значений, есть хотя бы два с одинаковыми значениями. Вычислим минимальное значение , чтобы .

Число различных способов выбора элементов таким образом, чтобы при этом не было дублей, равно .

Всего возможных способов выбора элементов равно .

Вероятность того, что дублей нет, равна

Вероятность того, что есть дубли, соответственно равна

С учетом <http://www.intuit.ru/department/security/networksec/8/2.html> имеем: если хэш-код имеет длину бит, т.е. принимает значений, то .

Подобный результат называется "парадоксом дня рождения", потому что в соответствии с приведенными выше рассуждениями для того, чтобы вероятность совпадения дней рождения у двух человек была больше , в группе должно быть всего 23 человека. Этот результат кажется удивительным, возможно, потому, что для каждого отдельного человека в группе вероятность того, что с его днем рождения совпадет день рождения кого-то другого в группе, достаточно мала.

На примере слабой хэш-функции в MySQL , подставим значение в формулу и получаем . Следовательно, столько потребуется подать случайных паролей на вход, чтобы с вероятностью выше получить совпадение хэшей от любой одной пары паролей, т.е получить с вероятностью ситуацию называемой коллизия. С учетом приведенной скорости перебора с использованием CUDA получаем, что подавая на вход случайные пароли первую коллизию (а значит пару паролей, от которых хэш одинаков) мы найдём с вероятностью 50% уже на .! Для сравнения посчитаем тоже самое для более стойкого алгоритма md5 (128-битный) учитывая более медленную скорость генерации хэшей от паролей получим следующее:

, требуемых случайных паролей, скорость перебора 500 000 000 пасс/сек. время нахождения первой коллизии с вероятностью 36893488147,5 сек ~10248191 часов ~427008 дней ~1169 лет. Таким образом сроки локальной атаки md5 по методу атаки дней рождения даже с поддержкой CUDA достаточно большой срок чтобы судить о её неосуществимости. Однако в случае организации распределенных вычислений этот срок можно значительно сократить.

**Как использовать атаку "Парадокс дня рождения"**

Обычно требуется восстановить по известному хэшу изначальный пароль, а не найти любую пару паролей, у которых один и тот же хэш, поэтому в лоб применение атаки "Дня рождения" обычно не используется. Но, тем не менее, возможны различного рода атаки, основанные на "парадоксе дня рождения". Возможна следующая стратегия:

1. Атакующий создает вариантов случайных паролей, далее подготавливает такое же количество паролей, каждое из которых является поддельным и предназначено для замены исходного пароля.

2. Два набора паролей сравниваются в поисках пары, имеющих одинаковый хэш-код. Вероятность успеха в соответствии с "парадоксом дня рождения" больше, чем . Если соответствующая пара не найдена, то создаются дополнительные исходные и поддельные пароли до тех пор, пока не будет найдена пара.

3. Атакующий предлагает отправителю один вариант пароля для генерации хэша (оттиска, подписи). Эта же подпись (хэш) может быть затем присоединена к поддельному варианту для передачи получателю. Так как оба варианта имеют один и тот же хэш-код, будет создана одинаковая подпись. Атакующий будет уверен в успехе, даже не зная ключа шифрования (первого варианта сообщения, пароля и т.д.).

## Задание

Последовательно изучить принципы атаки с использованием «парадокса дней рождения». Выбрать слабую хеш-функцию ( бита), сгенерировать для неё набор искомых значений, провести атаку дней рождения в прямой форме. Сгенерировать таблицы для атаки поиском коллизий, провести поиск коллизий. Оценить производительность методов атаки, построить сравнительные графики зависимости скорости работы от размера хеша.

## Порядок выполнения

1. Изучить принципы атаки с использованием «парадокса дней рождения».
2. Выбрать слабую хеш-функцию, сгенерировать для неё набор искомых значений, разработать и реализовать алгоритм проводящий атаку дней рождения в прямой форме.
3. Для этой же хеш-функции сгенерировать таблицы для атаки поиском коллизий, разработать и реализовать алгоритм проводящий поиск коллизий.
4. Практически оценить производительность методов атаки, построить сравнительные графики зависимости скорости работы от размера ключа.

## Содержание отчета

1. Цель работы.
2. Задание.
3. Краткая теория.
4. Выдержки из текста программы, реализующие критичные для работы алгоритмов атаки участки.
5. Примеры выбора хеш-функций и результаты работы соответствующих этим функциям методов атаки.
6. Графики производительности.
7. Выводы по работе.