Лекция 3. Метод полного перебора

Цель лекции: курса.

План лекции:
Введение.
1 Метод полного перебора
2 Пример 1
3 Пример 2
4 Пример 3
Заключение
Контрольные вопросы

Ключевые слова: [выбрать самостоятельно].

Содержание лекции:

Введение

По умолчанию, если не оговаривается противное, будем считать, что задача дешифрования решается в условиях предыдущей лекции, когда известна одна или несколько пар (x, y) и известны T и T^{-1}, а неизвестен только ключ.

Несмотря на название, не все универсальные методы всегда применимы. Так как некоторые из них значительно снижают стойкость шифра, то создатели стараются делать так, чтобы универсальные методы были к их шифрам неприменимы. Вместе с тем, все множества X, Y, K конечны. Поэтому потенциально можно перебрать все их элементы и найти все решения уравнения T(x, k) = y. Следовательно, множество методов, применимых к данной шифрсистеме, непусто.

1 Метод полного перебора

Рассмотрим уравнение относительно к е K при известной паре (x, y), x ∈ X, y ∈ Y:

$$T(x, k) = y. \quad (1)$$

Пусть для простоты для ∀ пары (x, y) существует единственное k, удовлетворяющее (1). Упорядочим множество K в соответствии с заданным порядком и будем последовательно проверять ключи из K на предмет равенства в уравнении (1). Если считать проверку одного варианта ключа k ∈ K в уравнении (1) за одну операцию, то полный перебор ключей потребует |K| операций, где знаком |.| обозначается число элементов в множестве. Пусть ключ в схеме шифрования выбирается случайно и равновероятно из множества K. Тогда с вероятностью 1/|K| трудоемкость метода полного перебора равна 1. Это происходит в том случае, когда случайно выбран ключ, расположенный в нашем порядке на первом месте. Поэтому естественно в качестве оценки трудоемкости метода взять математическое ожидание (среднее) число шагов в переборе до попадания на использованный ключ. Найдем среднее число шагов в методе полного перебора, когда порядок фиксирован, а выбор ключа случаен и равновероятен.

Пусть случайная величина τ – число опробований включительно до момента обнаружения использованного ключа. При i = 1, ..., |K| случайные величины $\xi_i = 1$, если использованный ключ находится в порядке на месте i и $\xi_i = 0$ в противном случае. Тогда

$$E\tau = \sum_{i=1}^{|K|} iP(\xi_{i.} = 1) \qquad (2)$$

Если считать, что все ключи расположены в установленном порядке, то процедуру равновероятного выбора ключа можно представлять как равновероятный выбор числа i в последовательности натуральных чисел 1, …, |K| . Тогда $P(\xi_i = 1) = 1/|K|$ для любого i = 1, ..., |K| . Подставляя полученные значения в (2) получим

$$E\tau = \frac{1}{|K|}\sum_{i=1}^{|K|} i = \frac{|K|(|K|+1)}{|K|\cdot 2}$$

При больших |K| можно приблизительно считать $E\tau \approx |K|/2$.

2 Пример 1

В DES 56-битный ключ. Значит $|K| = 2^{56} \approx 10^{17}$. Средняя трудоемкость полного перебора $2^{55} \approx 0{,}5\cdot 10^{17}$.

Алгоритмы полного перебора допускают распараллеливание, что позволяет значительно ускорить нахождение ключа. Можно предложить два направления в организации параллельного вычисления ключа [1].

Во-первых, построение конвейера. Пусть алгоритм проверки равенства T(x, k) = y представим в виде детерминированной цепочки простейших действий (операций)

$$O_1, O_2, \ldots, O_N.$$

Возьмем N процессоров A_1, ..., A_N, зададим их порядок и положим, что i-ый процессор выполняет три одинаковые по времени операции: 1) прием данных от (i – 1)-го процессора; 2) выполнение операции O_i; 3) передача данных следующему (i + 1)-му процессору. Тогда конвейер из N последовательно соединенных, параллельно и синхронно работающих процессоров работает со скоростью v/3, где v – скорость выполнения одной операции процессором. Если мы со скоростью v/3 подаем на вход ключи $k \in K$, то с такой же скоростью будем получать результат на выходе. Постоянная для процессора задержка N тактов между моментом входа ключа k и ответом, является ли ключ k решением (1), мала по сравнению с |K|. Поэтому рассмотренный конвейер подтверждает допустимость предположения о том, что сложный алгоритм вычисления T(x, k) и сравнение с y, можно считать элементарной операцией (на самом деле, в нашем примере требуется 3 элементарные операции независимо от сложности алгоритма проверки ключа при $|K| >> N$).

Второе направление распараллеливания состоит в том, что множество натуральных чисел 1, 2, ..., |K| разбивается на непересекающиеся подмножества B_1, ..., B_R . В каждом из них может быть свой порядок. Система из R машин перебирает ключи так, что i-ая машина осуществляет перебор ключей из множества B_i, i = 1, ..., R. Система прекращает работу, если одна из машин нашла ключ.

3 Пример 2

Если одна машина опробует один ключ за 10^{-6} сек, то для того, чтобы найти ключ DES полным перебором за 24 часа на R-машинной системе при $|B_1| = \ldots = |B_R|$ надо, чтобы

[1] Амамия М., Танака Ю. Архитектура ЭВМ и искусственный интеллект. М.: Мир, 1993.

$$R = \frac{10^{17}}{2 \cdot 864 \cdot 10^{8}} = \frac{10^{9}}{2 \cdot 864} = 5{,}787 \cdot 10^{5}$$

Если нас устраивает срок 100 суток, то таких машин надо всего 5787.

В последнее время широкое распространение получили глобальные сети ЭВМ. Можно сделать вирус, который выполняет следующую задачу: опробует ключи в свободное время процессора, начиная с некоторого номера; в случае положительного решения передает ответ и самоуничтожается, а также распространяет вирус, который всем другим работающим над опробованием вирусам приказывает самоуничтожиться.

Аналогичная идея распараллеливания предложена в «китайской лотерее» [2]. В каждый телевизор или радиоприемник на внутреннем рынке Китая, вмонтирован чип, который принимает открытый и шифрованный текст и начинает опробование заданного участка ключей DES. Если ключ найден, то аппарат дает сигнал, а его владелец должен сообщить о сигнале в правительственный орган, там его ждет приз. Если правительству нужен очередной ключ, то передается в широковещательной передаче открытый и шифрованные тексты, по которым включаются чипы в телевизорах и радиоприемниках. По данным [2] на 1995 год «китайская лотерея» при использовании чипа в 1 млн. операций в сек. могла позволить вскрыть ключ DES в Китае за 280 сек, в США – за 97 сек.

Самой большой сложностью в изложенном подходе является организация деления ключевого множества. Однако можно сделать старт с любого числа и случайно выбранного ключа. Время опробования увеличится, но схема значительно упростится. Рассмотрим задачу о среднем времени опробования N процессорами (машинами) ключей из множества K при старте каждого процессора со случайной точки при каждом очередном опробовании. Пусть все машины проработали τ шагов. Тогда сделано $N\tau$ опробований. Рассмотренная задача имеет аналог в задаче распределения частиц в классической задаче о размещениях [3]. Предположим, что все машины работают синхронно. Обозначим число тактов опробования до первого обнаружения ключа через η.

Если частицы бросаются порциями по N штук (т.е. машины работают синхронно), то вероятность того, что из комплекта с номером i ни одна частица не попадет в данную ячейку равна

$$q = \left(1 - \frac{1}{|K|}\right)^{N}$$

Тогда вероятность того, что произойдет первое попадание в данную ячейку на комплекте с номером t, равна

$$P(\tau = t) = (1-q)^{t-1}q = \left(1 - \left(1 - \frac{1}{|K|}\right)^{N}\right)^{t-1}\left(1 - \frac{1}{|K|}\right)^{N}$$

Среднее геометрического распределения равно

$$E\tau = \frac{1}{1-q} = \frac{1}{1 - \left(1 - \frac{1}{|K|}\right)^{N}}$$

Так как N << |K|, то

---

[2] Schneier B. Applied Cryptography Second Edition: protocols, algorithms and source code in C. John Wiley & Sons Inc., 1996.

[3] Колчин В.Ф., Севастьянов Б.А., Чистяков В.П. Случайные размещения. М.: Наука, 1976.

$$E\tau = \frac{1}{1 - 1 + \frac{N}{|K|}} = \frac{|K|}{N}$$

4 Пример 3

При случайном старте каждого чипа при полном опробовании и при времени одного опробования на одном чипе 10^{-6} сек для определения ключа за 100 суток потребуется N = 11574 машин.

Заключение

То есть случайный старт в два раза увеличивает необходимое количество процессоров при одном и том же времени ожидания результата.

Контрольные вопросы

Смотри руководство по организации самостоятельной работы магистрантов.