Нелинейные системы на службе защиты данных

Роман Клен, *начальник отдела комплексных систем безопасности НПЦ "Интелком"*
Евгений Борисов, *старший инженер отдела комплексных систем безопасности НПЦ "Интелком"*

Защита информации сегодня является одной из быстроразвивающихся научных отраслей. В каждой организации имеются свои данные конфиденциального характера, подлежащие защите, например, сведения, составляющие государственную, банковскую или коммерческую тайну. Одним из старейших, но и самых эффективных способов защиты данных является их искажение и сокрытие от глаз злоумышленника путем шифрования.

На сегодняшний день для осуществления шифрования разработано множество средств, обладающих различными характеристиками стойкости шифра к взлому и скорости работы, при этом реализующих единый подход к решению проблемы шифрования с помощью математического преобразования одной информационной единицы в другую.

Для обеспечения защиты данных, хранимых в информационных системах, предполагается использование специальных средств шифрования, одними из которых являются так называемые "секретные диски", которые представляют собой совокупность двух составляющих: программного интерфейса-преобразователя и защищенного хранилища данных. В основе хранилища данных лежит зависимая от ОС платформа предотвращения несанкционированного доступа к выделенному хранилищу на жестком диске. В общем случае комплекс "секретного диска" должен обеспечивать максимальную защиту хранилища, а в случае хищения информации из хранилища – обеспечивать и криптографическую стойкость применяемого шифра, т.е. невозможность "прочесть" информацию даже после ее хищения.

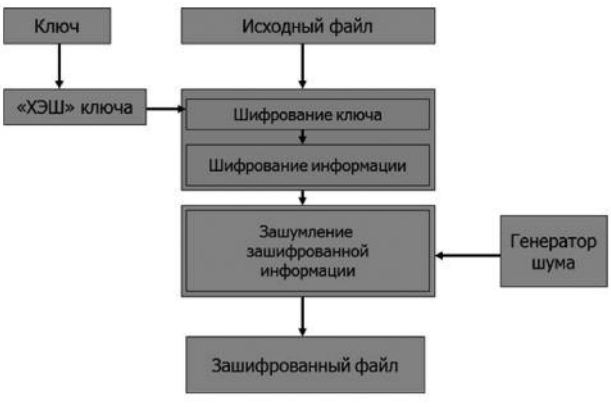

Рис. 1. Схема алгоритма шифрования файла

С точки зрения шифрования простейшим методом является обратимая логическая операция над единицей информации. В качестве единицы информации мы будем рассматривать символ – один байт. Обратимой логической операцией в данном случае является такая функция f(x, y), обладающая свойством (1): f(f(x, y), y) = x, где x – шифруемый символ, y – секретный ключ. Например, при шифровании символа "а" с ключом "б" результатом бы являлся символ "в", а при расшифровке – из символа "в" получался символ "а". Такая функция обладает очень низкой криптостойкостью, так как при использовании специальных таблиц и имея в своем распоряжении интерфейсный модуль, можно, не имея понятия о том, как работает шифратор, построить алгоритм работы и получить исходную функцию f(x, y). Также, существует еще один недостаток – при использовании одинаковых аргументов функция f(x, y) равна константе, следовательно, из последовательности "ааааа" мы получим только последовательность "ббббб", и ничто иное. Другое дело, если мы используем пару функций f(x, f(y)), когда f(y) – дискретна. В таком случае каждое следующее значение f(y) зависит от его предыдущего, yn+1 = f(yn), свойство (2). Следовательно, при использовании такого подхода, зная лишь начальное значение y_1, можно вычислить любое y_n. Таким образом, из последовательности "ааааа" мы можем получить последовательность "бвгдеж", зная лишь ключевой параметр.

Исходя из свойства (1), легко предположить применение простейшей логической операции "исключающего или", также известного как "XOR", для простейшей операции шифрования символов, при этом представляя символ в известной кодировке его числовым значением в диапазоне [0...255]. Пусть значение B является результатом операции XOR над символом A с секретным ключом K. Тогда значение A может быть также получено выполнением операции XOR над символом B с ключом K. Задача сводится к тому, чтобы использовать такую ключевую функцию $K_{n+1} = f(K)$, при которой максимально сложно "угадать" последующее значение, зная предыдущее. Таким образом, обеспечивается двойная защита данных – каждое последующее значение придется расшифровывать заново, не зная самой ключевой функции, результаты двух соседних значений будут максимально отличаться друг от друга – свойство (3), к примеру, f(1) <> f(2) ±1, а также необходимо подобрать первое значение ключевой функции для расшифровки последовательности, зная алгоритм ее работы.

Максимально удовлетворяющими перечисленным выше требованиям функциями являются так называемые "псевдослучайные" последовательности, реализованные в генераторах случайных чисел (ГСЧ). Исходя из их определения, свойств (1)–(3), алгоритм шифрования можно предста-

вить в виде:

$$B_n = An \text{ XOR } K_n, \quad (4)$$

$$K_n = RND(K_{n-1}), \quad (5)$$

Известные программные продукты, основывающиеся на обратном шифровании, используют в своей работе датчик случайных чисел процессора, а именно – инструкцию RDTSC. Такой подход имеет существенную уязвимость, т.к. работа этой инструкции основывается на времени (тиках) процессора и аппаратно зависима, т.е. высока вероятность подбора последующего значения, "угадав" нужное время "включения" работы ключевой функции. Предлагается применение принципиально нового алгоритма для создания собственного ГСЧ, основанного на простейших нелинейных функциях (рис. 1).

Таким образом, ноу-хау алгоритма может быть в использовании ГСЧ, основанного не на известных способах получения случайных чисел с использованием процессорного "времени", а на принципиально новых с использованием математического моделирования сложных систем с нелинейным, стохастическим поведением. Примерами подобных систем являются системы, моделирующие взаимодействие двух и более биологических видов, в которых колебания численности каждой из популяций являются псевдослучайными величинами в каждый момент времени наблюдения подобной системы.

Описанный выше пример простейшего алгоритма в силу использования ГСЧ будет иметь высокую криптостойкость, однако возникает вопрос о методе использования пароля. При моделировании рассматриваемой системы важно знать начальное значение времени наблюдений, которое будет являться исходной точкой для вычисления всех значений последовательности дискретной функции модели. Необходимо ввести математическое преобразование набора символов в число, которое можно использовать в качестве t0 – начального момента времени наблюдений, с учетом значений всех символов пароля и их знакомест. Это означает, что пароль вида "абв" не должен быть эквивалентен паролю "вба", а также паролю "АБВ" и подобным.

Так как одной из задач данной статьи ставится разработка алгоритма без использования сложных математических преобразований, предлагается следующий вариант вычисления t0. Пароль длины n символов представляется в виде набора символов таблицы ASCII. Каждому символу, исходя из таблицы, сопоставлен код в диапазоне [0...255]. Таким образом, y = ORD(a) – функция получения кода из символа. Значение i-го символа пароля получается из произведения его кода на его знакоместо. Исходя из этого, имеем:

(6)

Исходя из (4), (5), (6), имеем алгоритм шифрования. Исходный текст представляется в виде набора символов. Далее к каждому из них применяется следующая последовательность действий (рис. 2):

- вычисляется код символа по таблице ASCII;
- вычисляется значение шифркода путем использования функции "исключающее или", первый аргумент которой – код исходного символа, второй – псевдослучайное число в диапазоне от 0 до 255, заданное функцией преобразования пароля.

Одним из эффективных способов защиты данных является зашумление, т.е. включение в шифр "шумовых" символов. При этом возрастает вероятность того, что использованная при шифровке последовательность не будет получена, так как неизвестно, какие из символов в шифре являются шумом, какие – данными.

Алгоритм шифрования текста с использованием шумовых байтов прост. Весь текст представляет собой последовательность байтов данных – символов кодировки ASCII. Код символа представляется в качестве битовой маски. Технология представляет собой следующую последовательность действий:

- блок в 7 байтов представляет собой последовательность для шифрования;
- 8 байт – битовая маска зашумленного блока – содержит информацию, в каком знакоместе стоит нормальный символ, в каком – шумовой байт.

Так, например, исходная строка "12345678" может быть преобразована в "1x2x345C1" и "67xx8xxC2", при этом битовая маска символа "C1" – 10101110, "C2" – 11001000.

Стоит отметить, что техника зашумления увеличивает размер исходного текста в 2 раза.

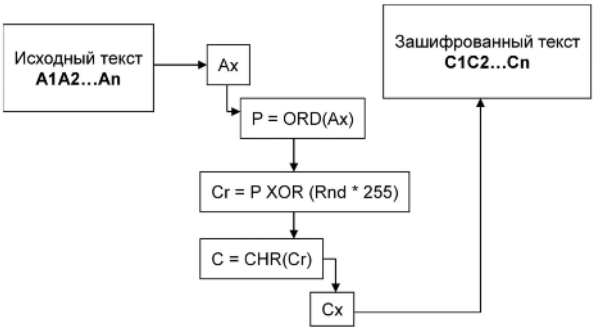

Рис. 2. Схема зашифровки информации предложенным способом

Практическая составляющая

Исходя из представленного, можно сделать вывод о том, что простейшие математические преобразования предоставляют большие возможности для их использования в различных областях, в том числе для защиты данных там, где является целесообразным применение функций с хаотическим поведением.

В заключение хотелось бы отметить, что программные продукты, использующие в своей работе подобные алгоритмы шифрования, обладают следующими ключевыми характеристиками по отношению к "классическим" шифраторам:

- высокая скорость работы (шифрования – дешифрования) ввиду использования простейших математических преобразований, которые практически не используют вычислительную мощность компьютера;
- высокая стойкость зашифрованных данных к расшифровке ввиду использования алгоритма, основанного на псевдослучайных последовательностях. Под стойкостью данных понимается требование неприемлемо большого времени для расшифровки данных с помощью современных средств вычислительной техники. Например, для расшифровки страницы текста без известного пароля (сложный пароль длиной 14 символов) понадобится около года.

Все это делает указанные алгоритмы крайне привлекательными для использования. ●

Ваше мнение и вопросы присылайте по адресу is@groteck.ru