УДК 66:681.3(075.8)

А. Д. Анкуда, магистрант (БГТУ)

ПРОЕКТИРОВАНИЕ И АНАЛИЗ ХЭШ-ФУНКЦИЙ НА БАЗЕ НЕЙРОННЫХ И ЯЧЕИСТЫХ СЕТЕЙ

В статье рассмотрены схемы алгоритмов хэширования, основывающиеся на нейронной сети или ячеистой нейроподобной сети. Приведена реализация хэш-функции на базе трехслойной хаотической нейронной сети, а также рассмотрена возможность использования ячеистых нейросетей в схеме хэширования. Проведен анализ основных криптографических свойств описанных алгоритмов, рассмотрены их достоинства и недостатки.

The schemes of hashing algorithms, based on a neural network or a cellular neural network, is proposed in this article. The implementation of the hash function, based on a three-layer chaotic neural network, is shown, as well as the possibility of using cellular neural networks in the hashing scheme. Basic cryptographic properties, advantages and disadvantages of described algorithms is analyzed.

Введение. Устойчивость некоторых хэш-функций, таких как MD5 и SHA-1, была поставлена под вопрос в результате недавних открытий [1]. Очевидно, что отказаться от них придется уже в ближайшем будущем. В связи с этим весьма перспективно выглядят нетрадиционные подходы к проектированию алгоритмов хэширования. Одним из активно исследуемых направлений в этом плане является разработка хэш-функций на основе сетевых структур – нейронных и ячеистых сетей. Нейросетевые свойства перемежения и рассеивания используются для проектирования алгоритмов шифрования, таких как потоковые или блочные шифры. Фактически нейросети обладают свойством однонаправленности, что позволяет использовать их в качестве «сжимающего блока» хэш-алгоритма.

Основная часть. Одна из возможных архитектур хэш-функции основывается на трехслойной хаотической нейронной сети. Три слоя нейронов реализуют перемежение, рассеивание и пространственное сжатие данных, а блочный режим хэширования поддерживает строки произвольной длины.

В предложенной хэш-функции [1] используется нейросеть, изображенная на рис. 1. Она состоит их трех слоев: входного, скрытого и выходного. Все они реализуют перемежение, диффузию и пространственное сжатие данных. Обозначим входы и выходы слоев соответственно $P = [P_0P_1 \ldots P_{31}]$, $C = [C_0C_1 \ldots C_7]$, $D = [D_0D_1 \ldots D_7]$ и $H = [H_0H_1 \ldots H_3]$, а нейронную сеть определим как

$$H = f_2\left(W_2D + B_2\right) = f_2\left(W_2f_1\left(W_1C + B_1\right) + B_2\right) =$$

$$= f_2\left(W_2f_1\left(W_1f_0\left(W_0P + B_0\right) + B_1\right) + B_2\right), \quad (1)$$

где f_i, W_i и B_i (i = 0, 1, 2) – передаточная функция, весовой коэффициент и смещение i-го нейрона. В качестве f_i для рассматриваемого алгоритма используется кусочно-линейное хаотическое отображение, определяемое следующим образом:

$$X\left(k+1\right) = f\left(X\left(k\right),Q\right) =$$

$$= \begin{cases} X\left(k\right)/Q, 0 \le X\left(k\right) < Q \\ X\left(k\right) - Q/0{,}5 - Q, Q \le X\left(k\right) < 0{,}5 \\ 1 - Q - X\left(k\right)/0{,}5 - Q, 0{,}5 \le X\left(k\right) < 1 - Q \\ 1 - X\left(k\right)/Q, 1 - Q \le X\left(k\right) \le 1, \end{cases} \quad (2)$$

где Q – управляющий параметр, удовлетворяющий условию $0 < Q < 0{,}5$. Здесь отображение является кусочно-линейным при $0 < Q < 0{,}5$. Такое хаотическое отображение имеет некоторые свойства, подходящие для создания шифра (например, чувствительность к начальным значениям или параметрам). Если просчитать отображение T раз (при достаточно большом T), небольшое различие в начальном значении $X(k)$ или параметре Q приведут к значительным отличиям в итерированном значении $X(k + T)$.

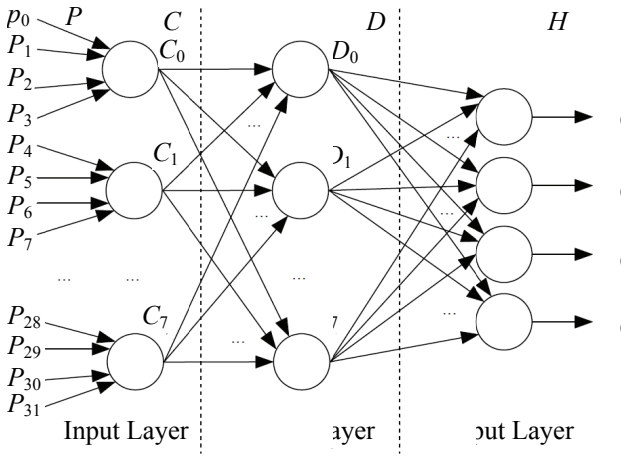

Рис. 1. Трехслойная нейросеть в качестве сжимающего блока [1]

Как правило, функция хаоса, итерированная T раз ($T \geq 50$), дает случайный выход. Основываясь на хаотическом отображении, входной слой определится следующим образом:

$$C = f^T\left(\begin{bmatrix} \sum_{i=0}^{3} w_{0,i}P_i + b_{0,0} \\ \sum_{i=4}^{7} w_{0,i}P_i + b_{0,1} \\ \cdots \\ \sum_{i=28}^{31} w_{0,i}P_i + b_{0,7} \end{bmatrix}, Q_0\right) =$$

$$= \begin{bmatrix} f^T\left(\sum_{i=0}^{3} w_{0,i}P_i + b_{0,0}\right) \\ f^T\left(\sum_{i=4}^{7} w_{0,i}P_i + b_{0,1}\right) \\ \cdots \\ f^T\left(\sum_{i=28}^{31} w_{0,i}P_i + b_{0,7}\right) \end{bmatrix} = \begin{bmatrix} C_0 \\ C_1 \\ \cdots \\ C_7 \end{bmatrix}, \quad (3)$$

где $W_0 = [w_{0,0}\ w_{0,1}\ \ldots w_{0,31}]$, B_0 имеет размерность 8×1 и T – количество итераций ($T \geq 50$). Учитывая, что вход хаотического отображения находится в промежутке [0, 1], все сложения производятся по модулю 1. Кроме того, скрытый слой и выходной слой определяются следующим образом:

$$D = f_1\left(W_1C + B_1\right) = f\left(W_1C + B_1, Q_1\right) =$$

$$= \begin{bmatrix} f\sum_{i=0}^{7} w_{1,0,i}C_i + B_{1,0}, Q_1 \\ f\sum_{i=0}^{7} w_{1,1,i}C_i + B_{1,1}, Q_1 \\ \cdots \\ f\sum_{i=0}^{7} w_{1,7,i}C_i + B_{1,7}, Q_1 \end{bmatrix} = \begin{bmatrix} D_0 \\ D_1 \\ \cdots \\ D_7 \end{bmatrix}, \quad (4)$$

$$H = f_2\left(W_2D + B_2\right) = f^T\left(W_2C + B_2, Q_2\right) =$$

$$\begin{bmatrix} f^T\left(\sum_{i=0}^{7} w_{2,0,i}D_i + B_{2,0}, Q_2\right) \\ f^T\left(\sum_{i=0}^{7} w_{2,1,i}D_i + B_{2,1}, Q_2\right) \\ \cdots \\ f^T\left(\sum_{i=0}^{7} w_{2,3,i}D_i + B_{2,3}, Q_2\right) \end{bmatrix} = \begin{bmatrix} H_0 \\ H_1 \\ \cdots \\ H_7 \end{bmatrix}. \quad (5)$$

Здесь весовая матрица W_1 имеет размерность 8×8, B_1 – размерность 8×1, W_2 – 4×8 и B_2 – 4×1. Цель скрытого слоя – диффузия изменений в C с изменениям в D. В качестве передаточной функции может быть использовано хаотическое отображение f. Для того, чтобы сохранить производительность, отображение f итерируется лишь единожды. Передаточная функция – f^T в выходном слое, где T ($T \geq 50$) – количество итераций. Повторная итерация увеличивает хаотичность связей между H и D, укрепляя криптосистему.

Блок P состоит из 32-х единиц данных, кодируется в значение хэша H, состоящее из 4-х единиц данных, смешиваясь с ключом пользователя. В свою очередь, каждая единица данных состоит из 32-х бит, которые квантуются (путем деления на 2^{32}) на дробные значения в интервале от 0 до 1. Результирующие биты хэша извлекаются из дробных единиц данных (32 бита на единицу).

Таким образом, открытый текст P, содержащий 1024 бита, преобразуется в 4 компонента хэша H по 128 бит каждый. Генератор ключей используется для генерации подключей: W_0, B_0, Q_0, W_1, B_1, Q_1, W_2, B_2 и Q_2, состоящих из 151-го бита. Ключ $K = k_0k_1\ \ldots k_{127}$ разделяется на 4 подключа: $K_0 = k_0k_1\ \ldots k_{31}$, $K_1 = k_{32}k_{33}\ \ldots k_{63}$, $K_2 = k_{64}k_{65}\ \ldots k_{95}$ и $K_3 = k_{96}k_{97}\ \ldots k_{127}$. Генерация подключей происходит следующим образом:

$$\begin{aligned} X_0(k) &= f^{T+k}\left(K_0,\ K_1\right), \\ X_1(k) &= f^{T+k}\left(K_2,\ K_3\right), \\ K_S(k) &= \left(X_0(k) + X_1(k)\right) \bmod 1. \end{aligned} \quad (6)$$

Здесь $K_S(k)$ ($k = 0, 1, \ldots, 150$) – k-й подключ. Операция mod определяется следующим образом:

$$a \bmod 1 = \begin{cases} a, 0 \leq a < 1, \\ a - 1, 1 \leq a < 2. \end{cases} \quad (7)$$

Блочный хэш преобразует 1024 бита в 128 бит. С другой стороны, многоблочный хэш предназначен для кодирования открытого текста с нефиксированной длиной в 128 бит. Вначале открытый текст M дополняется до длины 1024 путем добавления в конец сообщения M одного единичного бита и нескольких нулевых. Затем дополненное сообщение делится на n блоков $M_0, M_1, \ldots, M_{n-1}$. Далее эти блоки кодируются в многоблочном режиме, иначе говоря, M_i-е значение хэша блока H_{M_i} модулируется его ключом $K_{M_{i-1}}$. Окончательным значением хэша является

$$\begin{aligned} H_M = K_{M_{n-2}} \oplus H_{M_{n-1}} &= \left(K_{M_{n-3}} \oplus H_{M_{n-1}}\right) \oplus \\ \oplus H_{M_{n-1}} &= \ldots = \left(K \oplus H_{M_0}\right) \oplus \\ \oplus H_{M_1} &\oplus \ldots \oplus H_{M_{n-1}}, \end{aligned} \quad (8)$$

где "⊕" обозначает операцию побитового исключающего "ИЛИ".

В процессе исследования алгоритма были раскрыты некоторые его недостатки. Так как нейронная сеть способна работать только с числами с плавающей запятой (типы данных float и double) в диапазоне [0, 1], возникает необходимость в предварительном преобразовании

входного значения к последовательности вещественных чисел. Универсальный алгоритм состоит в том, чтобы задать исходный алфавит и уже на его основе произвести преобразование. Однако если сообщение состоит из символов достаточно большого алфавита, или же исходный алфавит неизвестен, то такой подход становится малоэффективным. Поэтому в предложенной реализации применено следующее преобразование: (1) сконвертировать входную строку в массив значений типа byte на основе кодировки, которая достоверно включает все символы, использованные в сообщении; (2) преобразовать каждое значение в массиве к типу с плавающей запятой путем поэлементного деления значения на 2^8–1. Аналогичное преобразование необходимо произвести и для выхода нейросети.

В предложенной реализации применена следующая схема сжатия: (1) весовые коэффициенты сети конфигурируются в соответствии с заданным ключом пользователя; (2) преобразованный в последовательность вещественных чисел блок подается в нейронную сеть и вычисляется ее выход – хэш переданного блока; (3) полученное значение хэша используется в качестве ключа пользователя: с его помощью сеть реконфигурируется и алгоритм возвращается к шагу 2; (4) выход нейросети для последнего блока представляет собой хэш всего сообщения.

Приведенная схема сжатия делает алгоритм достаточно чувствительным к передаваемому сообщению и гарантирует достаточную сложность обратного восстановления сообщения по хэшу. Однако процесс реконфигурации сети (изменения весовых коэффициентов при помощи функции хаоса в соответствии с ключом пользователя) требует больших временных затрат, что приводит к необходимости поиска альтернативной функции сжатия.

Ячеистая нейронная (нейроподобная) сеть – это парадигма параллельного программирования наподобие нейросетей. Ячеистые нейронные сети являются массивами идентичных динамических систем, ячеек, которые связаны только локально. Любая ячейка соединена только со своими соседними ячейками, т. е. смежная ячейка прямо взаимодействует с каждой. На несоседние ячейки оказывается косвенное взаимодействие из-за распространяющегося эффекта динамики в сети.

Основное преимущество ЯНС – облегченная параллелизация за счет более простой архитектуры. При таком подходе отсутствует необходимость в хранении сложной структуры нейросети в памяти. Кроме того, хэширование на основе нейросетей является неоднородным – какие-то из полученных сверток сообщений более подвержены атакам нахождения прообраза, чем другие. В случае с ячеистыми сетями такой недостаток пропадает.

В работе Янга [2] представлена новая параллельная хаотическая хеш-функция, основанная на ячеистой нейронной сети с четырьмя измерениями (4-D CNN). Сообщение расширяется путем итерирования хаотического логистического отображения (chaotic logistic map), после чего разделяется на блоки, каждый из которых имеет длину 512 бит. Все блоки обрабатываются в параллельном режиме, что является одной из существенных характеристик представленного алгоритма. Каждый 512-битный блок разделяется на четыре 128-битных суб-блока, каждый суб-блок разделяется на четыре 32-битных значения, после чего эти четыре значения смешиваются с четырьмя новыми значениями, порожденными хаотическим отображением кота Арнольда. Четверка полученных таким образом значений побитно обрабатывается операцией исключающего ИЛИ с четырьмя начальными значениями или же ранее порожденной четверкой значений, после чего они подаются на входы ячеистой нейронной сети.

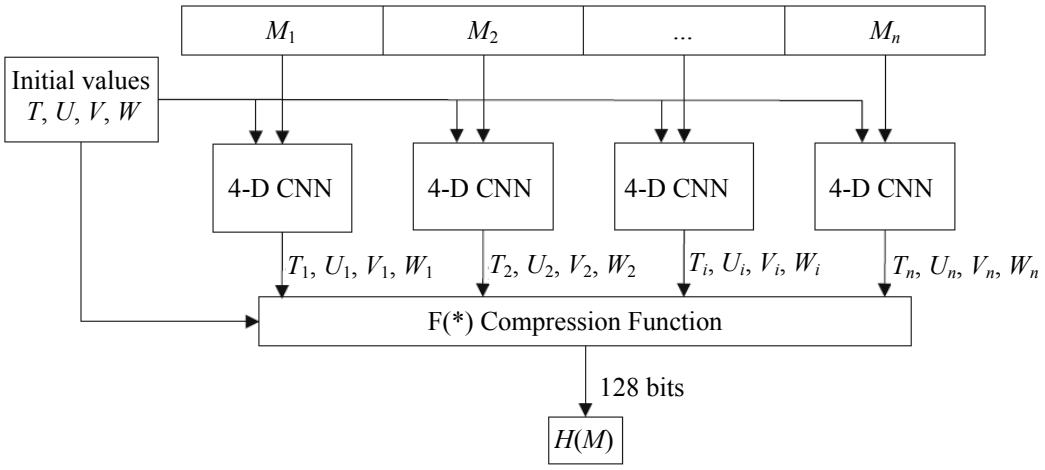

Рис. 2. Схема параллельного алгоритма хеширования Янга [2]

Сгенерированные значения всех блоков передаются в функцию сжатия (compression function), вырабатывающую окончательное 128-битное значение хеша. Схема алгоритма приведена на рис. 2.

Заключение. Рассмотренные хэш-функции на базе сетевых структур обладают криптографической стойкостью и могут быть использованы, например, в системах цифровой подписи. С другой стороны, такие хэш-функции сравнительно медленны, и вопрос их оптимизации остается открытым.

Литература

1. Li, Yantao. A novel Hash algorithm construction based on chaotic neural network / Yantao Li, Di Xiao, Shaojiang Deng // Neural Comput&Applic, Springer-Verlag London Limited. – 2010. – P. 133–141.

2. Parallel chaotic Hash function construction based on cellular neural network / Yantao Li [et al.] // Neural Comput & Applic. – 2011. – P. 1563–1573.

Поступила 11.03.2013