ОБЗОР МЕТОДОВ ИДЕНТИФИКАЦИИ ФЛЕШ-НАКОПИТЕЛЕЙ С ИСПОЛЬЗОВАНИЕМ ФИЗИЧЕСКИ НЕКЛОНИРУЕМЫХ ФУНКЦИЙ

Белясова А. В.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Заливако С. С. – кандидат техн. наук

В данной работе рассматриваются некоторые из существующих методов идентификации, реализованных на основе физически неклонируемых функций, для устройств на основе NAND флеш-памяти. Приводится экспериментальная оценка основных характеристик методов, таких как производительность, стабильность и уникальность. Все методы основаны на вычислении выходных значений ФНФ и применении алгоритма генерации идентификатора для устройства. Предложены возможные направления работы с целью улучшения производительности методов.

В сфере производства товаров актуальной проблемой является их нелегальное копирование. За последние 10 лет количество товаров-подделок увеличилось в 2 раза. Производители несут серьезные убытки, что характерно в том числе для рынка электроники [1]. Чтобы защитить свои товары, производители прибегают к таким методам, как аппаратное шифрование, хеширование, внедрение водяных знаков и другие. Однако многие из этих методов требуют значительных аппаратурных затрат, которые сопоставимы с затратами на само устройство. На данный момент одним из менее требовательных к аппаратуре методов являются физически неклонируемые функции (ФНФ) [1]. В данной работе рассматриваются ФНФ, использующие уникальность пороговых напряжений для различных состояний элементов NAND флеш-памяти [2] и ошибки, возникающие в страницах, смежных с записываемыми.

Первый метод реализации ФНФ называется методом частичного стирания (partial erasure). Его алгоритм в следующем. Стиранию подвергается выбранный блок, затем выбранная страница записывается “нулями”. Выполняется фиксированное число итераций частичного стирания выбранного блока [3]. Пороговые напряжения разные для элементов памяти, поэтому только некоторые из них поменяют состояние на ‘1’. Этим элементам присваивается номер итерации стирания. После последней итерации всем нестертым элементам памяти будет присвоено число, равное номеру следующей итерации. Значения, присвоенные каждому из элементов, являются выходными значения ФНФ.

Второй метод – метод частичной записи (partial programming). Алгоритм аналогичен методу частичного стирания, за исключением того, что выполняются итерации частичного программирования, а не стирания. Также вначале, после стирания блока, запись в выбранную страницу не происходит. Операция частичного программирования имеет фиксированный период времени выполнения.

Третий метод называется методом внесения ошибок в результате записи (program disturbance). Алгоритм производит стирание выбранного блока и фиксированное число записей в выбранную страницу. На каждой итерации проверяется состояние битов соседней страницы. В результате каждому биту присваивается число, равное номеру итерации, на которой он изменил состояние, или номеру итерации, следующей за последней, если бит так и не был запрограммирован. Главным недостатком данного метода является его применимость только на 2D NAND (при плоской организации флеш-памяти). На более современном 3D NAND помехи в соседних страницах при чтении/записи встречаются существенно реже.

После получения выходных значений ФНФ необходимо применить алгоритм генерации идентификаторов. Часто для извлечения идентификаторов используется метод нечеткого экстрактора, которые использует криптографическую хеш-функцию и коды коррекции ошибок (ECC) для повышения стабильности, что требует дополнительных аппаратурных затрат. В работе [3] описываются более экономичные методы генерации идентификатора, не требующие дополнительных затрат, такие как bit-map и position-map. Каждый метод генерирует идентификаторы длины X из выходных значений ФНФ для последовательности ячеек (элементов памяти) длины Y, где $Y >= X * 2$.

В bit-map методе ячейки разбиваются на пары и вычисляется разность значений ФНФ в каждой

паре. Выбирается *X* пар с максимальной разностью. В position-map методе ячейки сортируются по значению ФНФ. Ячейки с минимальным и максимальным значениями образуют пару и исключаются из рассмотрения. Таким образом выбираются *X* последовательных пар. В паре ячейки сортируются по адресу в памяти. На основе выбранных пар генерируется идентификатор. Если первая ячейка в паре имеет большее значение ФНФ, чем вторая, то соответствующий бит идентификатора равен единице, иначе нулю.

Результаты экспериментов в работе [3] показали, что производительность для метода частичного стирания на разных устройствах составила от 12,21 до 14,67 Кбит/с, для метода частичного программирования – от 14,45 до 22,38 Кбит/с, для метода внесения ошибок в результате записи – от 7,35 до 9,72 Кбит/с. Это позволило генерировать 128-битные идентификаторы за время от 93,83 мс до 2,48 с.

Метод частичного программирования оказался быстрее метода частичного стирания, так как период частичного стирания был выбран на порядок больше периода частичного программирования для всех исследованных чипов NAND при практически одинаковых значениях количества итераций методов (например, 245 и 250 соответственно). Метод внесения ошибок в результате записи оказался самым медленным, из-за того что количество его итераций было на порядок больше количества итераций предыдущих методов (3000-4000) и для его реализации использовалась полная операция программирования, длительность которой больше длительностей операций частичного стирания и частичного программирования [3].

Для оценки характеристики уникальности описанных ФНФ было вычислено расстояние Хэмминга между 10000 парами 128-битных идентификаторов. Данная случайная величина должна была иметь биноминальное распределение с N=128 и p=0,5 [3]. Матожидание величины в таком случае равно $N/2$, что значит, что различие идентификаторов должно быть 50%-ным в среднем. Эксперимент дал следующие результаты: 49,93% для метода частичной записи, 49,95% для метода частичного стирания и 46,86% для метода внесения ошибок в результате записи. Все методы дали хороший результат: отклонение от желаемых 50% составило не более 5% для каждого метода.

В работе [3] были произведены оценки стабильности посредством вычисления *отклонения*. *Отклонение* вычислялось как доля несовпавших битов. Было проанализировано, как зависит стабильность от количества выбранных для генерации битов. Пусть Y - количество генерируемых битов, X - количество выбранных для генерации битов, Y/X - их отношение (его максимальное значение - 0,5 [3]). Результаты экспериментов показали, что при уменьшении соотношения Y/X, уменьшается *отклонение*. Для bit-map метода уменьшение от 0,5 до 0,055 привело к уменьшению *отклонения* от 14,42% до 0,0001% при любых изменениях факторов окружающей среды. Для position-map метода уменьшение от 0,5 до 0,474 привело к уменьшению *отклонения* от 0,023% до 0,0001%.

Эталонным *отклонением* в работе [3] считалось отклонение 0,0001%, или 10^{-6}. Это значит, что в одном из 1000000 случаев генерации битов произойдет ошибка. Таким образом, при генерации 128-битного идентификатора ошибка произойдет примерно один раз из 1000000/128=7812,5 попыток генерации. При этом вероятность ошибки в идентификаторе- около 0,000128. Если это недостаточно надежно, можно вычислить идентификатор повторно. При этом вероятность двойной ошибки значительно ниже ($0{,}000128^2$).

Рассмотренные методы идентификации устройств могут быть успешно использованы для защиты авторских прав. Для их реализации не нужна дополнительная аппаратура, что позволяет использовать эти методы для устройств с ограниченными ресурсами (например, устройства Интернета вещей). Методы генерируют уникальные и стабильные идентификаторы, что позволит им также найти применение в генерации закрытых ключей для алгоритмов шифрования, тем самым устраняя необходимость в хранении секретных ключей непосредственно в памяти.

Главный недостаток рассмотренных методов - низкая производительность. Основная причина больших задержек в работе данных методов - многочисленное программирование/стирание выбранных страниц/блоков. Одним из последующих направлений деятельности может стать разработка метода, который предварительно вычисляет выходные значения ФНФ и компактно сохраняет данные ФНФ в память (NAND). Затем данные считываются и предоставляются алгоритму генерации идентификатора. Данный подход позволит вынести все самые длительные операции на этап предварительной обработки, что предположительно даст прирост в производительности при генерации идентификатора. Узким местом данного подхода является считывание данных ФНФ из NAND. Время чтения страницы данных размером 4 килобайта из NAND составляет от 25 до 100 мкс [4]. При максимальном количестве данных, необходимых каждому из рассмотренных методов для генерации идентификатора, одной страницы размером 4 килобайта предположительно хватит для их хранения. Таким образом, одного чтения будет достаточно для генерации идентификатора, что может позволить разработать метод значительно быстрее, чем 1-2 секунды.

Список использованных источников:

1. Физически неклонируемые функции: защита электроники от нелегального копирования [Электронный ресурс]. — Режим доступа: https://habr.com/ru/post/343386/. — Дата доступа: 23.03.2020.

2. Understanding Flash: Floating Gates and Wear [Electronic resource]. — Mode of access: https://flashdba.com/2015/01/09/understanding-flash-floating-gates-and-wear/. — Date of access: 25.03.2020.

3. Extracting Robust Keys from NAND Flash Physical Unclonable Functions / J. Shijie, X. Luning, L. Jingqiang, J. Yafei // CCA Secure PKE with Auxiliary Input Security and Leakage Resiliency , 2015. — P. 437-454.

4. Coding for SSDs – Part 2: Architecture of an SSD and Benchmarking [Electronic resource]. — Mode of access: http://codecapsule.com/2014/02/12/coding-for-ssds-part-2-architecture-of-an-ssd-and-benchmarking/. — Date of access: 26.03.2020.