УДК 512.623 https://doi.org/10.51301/vest.su.2021.i2.09

[1]У.К. Турусбекова*, [1]С.А. Алтынбек, [2]А.С. Тургинбаева, [1]Л. Мерейхан
[1]Казахский университет экономики, финансов и международной торговли, Нур-Султан, Казахстан
[2]Евразийский национальный университет имени Л.Н. Гумилева, Нур-Султан, Казахстан
*e-mail: umut.t@mail.ru

ПОСТРОЕНИЕ ХЕШ-ФУНКЦИЙ НА ОСНОВЕ ТЕОРИИ КОНЕЧНЫХ ПОЛЕЙ С ИСПОЛЬЗОВАНИЕМ НЕПРИВОДИМЫХ МНОГОЧЛЕНОВ

Аннотация. С увеличением количества информации усугубляются проблемы, связанные с большими объемами данных, которые в дальнейшем требуют реализации процессов хранения, передачи или обработки. Работа с большими объемами файлов существенно усложняет указанные процессы, в связи с чем возникает необходимость в существовании алгоритмов, позволяющих сжимать объемы файлов до необходимого размера, приемлемого для их эффективной обработки.

Важную роль в процессе взаимодействия с файлами играют хеш-функции. Использование хеш-функций подразумевает преобразование исходных данных по определенному алгоритму в последовательность фиксированной длины. Это позволяет значительно ускорить поиск среди большого количества файлов для просмотра, изменения или удаления, для сравнения файлов, для проверки неизменяемости в тех случаях, когда данные не должны изменяться посторонними лицами. Таким образом, хеширование используется во всех областях, где возникает вопрос о хранении, передаче или обработке данных в виде файлов, а именно в криптографии, компьютерной графике, при организации данных на компьютере и в Интернете.

На практике используются функции, для которых теоретическая вероятность возникновения коллизий близка к нулю, но с появлением более мощных вычислительных инструментов поиск коллизий может оказаться не такой неописуемой задачей. По этой причине существующие алгоритмы требуют постоянного улучшения. Особую роль в этом играет проблема теории сложности, а именно алгебраическая теория чисел. В настоящее время теория чисел широко используется в криптографии; однако этого недостаточно, когда появляются эффективные алгоритмы факторизации. В связи с этим, возникает необходимость в использовании теории конечных полей. Расчеты в таких полях позволяют выполнять операции в выбранном конечном поле над его элементами, не выходя за его пределы, а также делает хеш-функции устойчивыми к восстановлению исходных данных. Поэтому в статье рассматриваются полиномиальные функции с использованием деления по модулю неприводимого многочлена в конечных полях. Использование неприводимых многочленов над конечными полями может быть более эффективным средством защиты.

Основная цель данной статьи - построение хеш-функции на основе теории конечных полей. Рассмотрены стандартные криптографические хеш-функции для алгоритмов цифровой подписи. Проанализированы методы хеширования различных типов исходных данных. Предложен метод хеширования текстовых файлов с помощью неприводимых многочленов. Результаты статьи могут быть использованы в криптографических приложениях и теории кодирования.

Ключевые слова: хеш-функция, конечное поле, неприводимый многочлен, электронная цифровая подпись, коллизия.

Введение. Экспоненциальный характер роста объема информации влечет за собой необходимость создания новых систем защиты информации и аутентификации, а также инструментов, которые обеспечивают возможность уменьшения объема памяти, необходимой для удобства хранения и передачи больших файлов. Хеш-функция - такой инструмент. Эти функции позволяют преобразовать последовательность входных данных произвольной длины в определенную последовательность фиксированной длины с помощью определенного алгоритма. Алгоритм и длина выходной последовательности должны точно определяться важностью преобразуемых данных. Начальные данные называются «сообщением» или «ключом», а результат хеш-функции - «хеш-кодом», «хешем» или

«сверткой». Результат хэш-функции также можно определить как контрольную сумму, которая однозначно проверяет целостность данных, переданных в цифровом виде. В данной статье мы исследуем методы построения криптографических хеш-функций и их использование в алгоритмах электронной цифровой подписи, а также как средство ускорения поиска в различных структурах данных.

При разработке таких систем следует учитывать уязвимость хеш-функций. Мощности набора M входных последовательностей и набора возможных значений свертки $H(M)$ могут быть найдены в любом соотношении. Как правило, набор входных данных M имеет большую размерность, чем набор всех возможных значений свертки $H(M)$, это может приводить к преобразованию разных сообщений в равные значения свертки. Такой случай называется «коллизией», и это один из важных факторов, который необходимо учитывать при построении алгоритмов хеширования. Чем ниже вероятность коллизий, тем надежнее алгоритм.

Коллизии являются основными уязвимостями хэш-функций, этот факт следует учитывать при разработке различных алгоритмов хеширования в криптографических системах, а также в некоторых структурах данных, например в хеш-таблицах. Хеш-таблицы разработаны для ускорения поиска в базах данных путем хранения пары (хеш) значений «ключ» и «адрес». Возникновение коллизий неизбежно в алгоритмах, где переменная длина входных данных хешируется в значения фиксированной длины. Коллизии затрудняют выполнение операций в таблицах, потому что разные ключи будут соответствовать одному и тому же значению, что неудобно при работе с большими данными, особенно при большом количестве коллизий. Исходя из этого, необходимы методы, которые могут разрешать конфликты в хеш-таблицах.

Основными инструментами разрешения коллизий в хеш-таблицах являются методы раздельной цепочки и открытой адресации. Метод отдельной цепочки включает объединение ключей с одним и тем же адресом в связанный список, и эти списки могут быть упорядоченными или неупорядоченными [1]. Выбор типа связанных списков зависит от целей реализации, поскольку при попытке организовать ключи с одинаковыми хешами в упорядоченных списках он может выполнять операции с данными с большей скоростью. Однако использование неупорядоченных списков более целесообразно с точки зрения экономии памяти устройства. Также следует учитывать, что эффективность использования раздельной цепочки зависит от среднего количества элементов в списке. Чем он больше, тем медленнее работает метод. В отличие от этого метода, общедоступные списки не используют связанные списки. Одним из методов открытой адресации является метод линейного зондирования. Идея этого метода состоит в том, чтобы найти элемент с ключом, равным ключу поиска, и разрешить конфликты, поместив элемент в пустое место в таблице. В этом методе списки не используются, это помогает экономить память, но скорость нахождения пустого места в таблице зависит от ее разреженности [1]. Если таблица сильно загружена, может потребоваться большое количество зондирований. Если таблица заполнена полностью, это приведет к бесконечному количеству зондирований, что делает этот метод непригодным для разрешения коллизий.

Основная задача статьи - проанализировать существующие алгоритмы хеширования и построить метод, основанный на теории конечных полей.

Материал и методы. В процессе взаимодействия с файлами важную роль играют хеш-функции. В 1976 году Диффи и Хеллман впервые подчеркнули необходимость построения однонаправленной функции как составной части схемы цифровой подписи [2]. Этот год можно считать отправной точкой развития хеш-функций.

На данном этапе развития теории хеширования нет четкого определения понятия «хеш-функция», а также четкой классификации методов хеширования. По определению Д.Е. Кнут, хеш-функция преобразует любой возможный ключ K (исходные данные) в номер списка $h(K)$

в диапазоне от 1 до *m*. Это верно для структур данных, в которых хранятся значения ключа и его номера [1], [3].

Хеш-функции используются в качестве строительного блока во многих приложениях. В криптографии хеш-функции предназначены для преобразования необработанных данных в последовательность фиксированной длины для проверки неизменности данных в процессах хранения или передачи [4]. Примером таких данных могут быть ценные бумаги или пароли, которые нельзя менять и которые доступны ограниченному кругу лиц.

Некоторые хэш-функции, используемые в настоящее время, оказались уязвимыми. В работе [5] автор утверждает, что их замены должны основываться на математической теории. В работе [6] исследованы потенциальные математические принципы и структуры, которые могут обеспечить основу для криптографических хеш-функций, а также представить простую и эффективно вычисляемую хеш-функцию, основанную на неассоциативной операции с многочленами над конечным полем характеристики 2. Общий обзор хеш-функций приведен в работе [7]. В работе [8] в хронологическом порядке развития приведены основные принципы построения алгоритмов хеширования. Одним из возможных способов построения хеш-функции является использование деления по модулю неприводимого многочлена [9]. В работе [10] описан метод построения хеш-функций, основанный на вычислении остатка от деления на неприводимый многочлен. Кроме того, рассмотрена проблема поиска неприводимых многочленов. Представлены результаты использования различных неприводимых многочленов и их анализ. Проблема поиска неприводимых многочленов также рассмотрена в работе [11].

В представленной работе проанализированы методы хеширования различных типов исходных данных, а также предложен метод хеширования текстовых файлов с помощью неприводимых многочленов.

Криптографические хеш-функции в алгоритмах цифровой подписи. Хеш-функции широко используются в различных криптосистемах. Криптографическая хеш-функция должна удовлетворять ряду требований, характерных для криптографических методов защиты информации, а именно [12]:

- устойчивость к поиску первого прототипа. Это означает, что невозможно создать эффективный полиномиальный алгоритм, который в реальном времени восстанавливает сообщение M своей сверткой $H(M)$;
- устойчивость к поиску второго прототипа. Невозможно найти такое сообщение N, что $N \neq M$ и $H(M) = H(N)$, зная M и $H(M)$;
- устойчивость к столкновениям второго рода. Невозможно выбрать пару разных сообщений $N \neq M$, для которых значения хеш-функций были бы одинаковыми $H(M) = H(N)$;
- при незначительном изменении исходного сообщения результат хеш-функции должен сильно отличаться, то есть иметь лавинообразный эффект.

Эти свойства независимы и должны рассматриваться только в совокупности. Каждое свойство позволяет использовать хеш-функции в криптографии, системах аутентификации, различных структурах данных для поиска, а также в компьютерной графике и вычислительной геометрии.

На данный момент существование необратимой хеш-функции остается недоказанным фактом и является нерешенной проблемой в компьютерных науках. Обычно поиск обратного значения является сложной вычислительной задачей.

Криптографические хеш-функции широко используются в алгоритмах электронной цифровой подписи. Использование хэш-функций не обязательно, но это удобный инструмент для преобразования необработанных данных в последовательность фиксированной длины, что значительно снижает вычислительную сложность применения цифровой подписи к такой последовательности.

Электронная цифровая подпись служит средством аутентификации, а не средством защиты информации. К цифровым подписям предъявляются следующие требования [13]:

1. Гарантия целостности информации в процессах ее передачи и хранения.
2. Предоставление доступа к информации исключительно ее законным пользователям.
3. Абсолютная аутентификация информации в соответствии с выбранными формами ее представления.
4. Невозможность отказа от авторства.
5. Гарантия прослеживаемости информации за счет последовательного анализа динамики формирования информационных сообщений в полном объеме.

Вышеуказанные требования позволяют использовать инструмент электронной цифровой подписи в экономических обязательствах и в других сферах, а также в системах, требующих однозначной идентификации участников. Такие документы могут содержать несколько электронных подписей, определяющих согласие всех вовлеченных в процесс подписания участников.

Схема алгоритма создания и проверки электронной подписи следующая [12]:

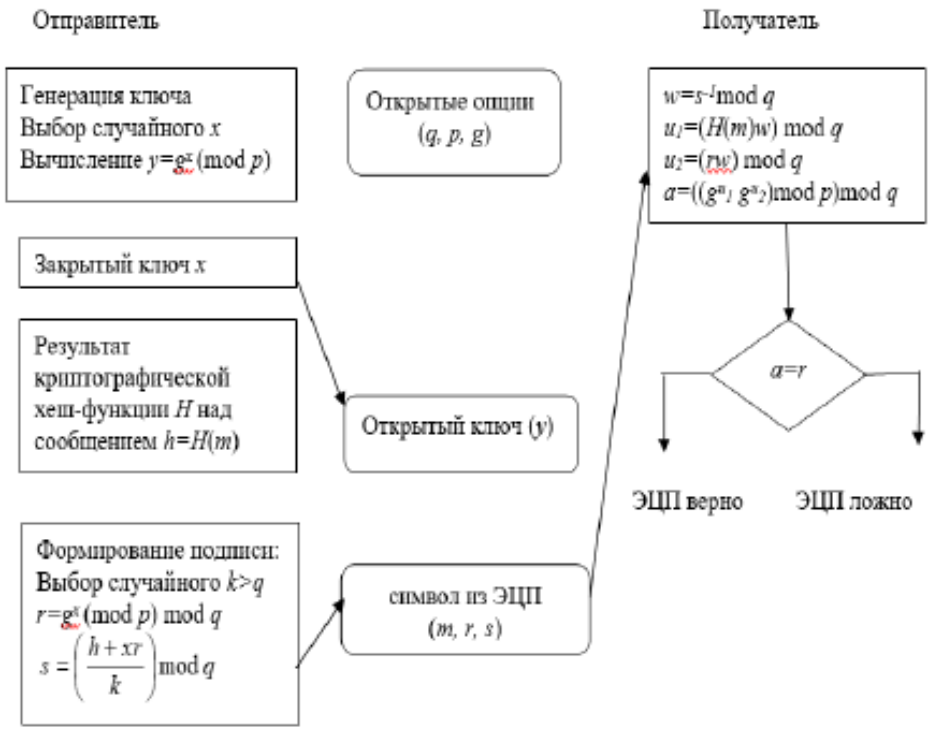

Рисунок 1. Схема алгоритма создания и проверки электронной подписи.

Из этой схемы видно, что формирование электронной подписи подписывает не сам документ или сообщение, а его свертку, то есть результат применения криптографической хеш-функции. Такая функция должна соответствовать требованиям для криптографических хеш-функций. Нет необходимости каждый раз работать с большим объемом данных, а только с их сверткой.

При разработке алгоритмов криптографических хеш-функций для систем цифровой подписи большое внимание уделяется последнему свойству, так как оно гарантирует целостность информации. Получатель должен получить сообщение в точности так, как отправитель хотел отправить, исключая возможность перехвата и изменения сообщения. В случае случайного или умышленного изменения документа электронная подпись будет считаться недействительной, поскольку рассчитывается по специальному алгоритму на

основе исходного документа и соответствует только ему. Как следствие, подделка документов в большинстве случаев становится нецелесообразной.

В алгоритмах электронной цифровой подписи хэш-функция является отдельным компонентом и может выбираться в зависимости от индивидуальных требований системы, а также может изменяться в процессе ее улучшения или для обеспечения динамического изменения системы защиты, когда она подвергается воздействию. к несанкционированному доступу.

Такие хеш-функции, как MD5, RIPEMD-160 и SHA-1 широко используются в алгоритмах цифровой подписи. Каждый из них является обобщением алгоритма MD4 и носит итерационный характер [11].

Цифровая подпись должна обладать определенной стабильностью, иначе ее полезность может вызвать сомнения. Алгоритм MD5 - один из многих алгоритмов, гарантирующих необходимый уровень защиты. Этот алгоритм более устойчив к атакам с коллизиями, чем MD4, но он не является неуязвимым, поскольку было показано, что можно создать несколько разных сообщений с одним и тем же хеш-значением. В SHA-1 улучшен лавинный эффект. Более длинное хеш-значение можно получить с помощью алгоритма SHA-1, а его устойчивость к аналогичным атакам делает этот алгоритм более безопасным, чем MD5, но уступает по производительности. Функция RIPEMD-160 по производительности сравнима с SHA-1. В таблице 1 приводится сравнение производительности вышеуказанных алгоритмов [14].

Таблица 1. **Таблица сравнения производительности хеш-алгоритмов**

Алгоритм	Количество циклов	Скорость Мбит/с	Относительная производительность
MD4	241	191.2	1.00
MD5	337	136.7	0.72
SHA-1	837	55.1	0.29
RIPEMD-160	1013	45.5	0.24

Упомянутые выше алгоритмы относятся к классу специализированных хеш-функций. Также существуют алгоритмы на основе блочных шифров (MDC-2, MDC-4) и модульной арифметики (MASH-1, MASH-2). Такие алгоритмы имеют большую вычислительную сложность и низкую скорость. Алгоритмы, основанные на модульной арифметике, не получили широкого распространения в криптографии из-за особых требований к программному и аппаратному обеспечению.

Специализированные методы просты в реализации, поэтому широко используются не только в электронных цифровых подписях, но и в других системах защиты информации и аутентификации. Следует отметить, что такие методы более уязвимы и могут быть атакованы методами криптоанализа.

При использовании криптографических хеш-функций в алгоритме электронной цифровой подписи важна разрядность выходной последовательности, так как она должна соответствовать стандартам, предлагаемым для документов разного уровня значимости. Таким образом, оптимальный размер выходной последовательности составляет 160 - 256 бит. Следует учитывать, что эти цифры будут корректироваться в большей степени с ростом возможностей техники.

Хеш-функции для разных типов исходных данных. Методы, которые будут обсуждаться в этом подразделе, предназначены для преобразования массивов исходных целочисленных данных. Такие методы применимы в случаях, когда необходимо отобразить один числовой набор в другой или если исходные данные другого типа предварительно преобразуются в целые числа с помощью определенных алгоритмов.

Один из самых известных методов - метод деления [1]. Основная операция - деление по модулю, и хэш вычисляется как остаток от деления на количество всех возможных значений хеш-функции:

$$H(k)=k(\text{mod } M)$$

где k - ключ (входные данные), M - количество всех возможных хеш-кодов. Чтобы избежать большого количества коллизий, лучше выбрать простое число в качестве значения M и избежать выбора чисел по степени с основанием 2. Эта функция называется модульной. Этот метод эффективен, потому что хеш-распределение равномерно в промежутке [0; M-1]. Такой метод имеет высокую вычислительную сложность для больших значений M, что делает его непригодным для использования.

Другой метод, основанный на делении, - это нахождение хэша с использованием коэффициентов многочлена, которые получаются из остатка от деления исходных данных K, представленных в виде многочлена $K(x)$ на предварительно выбранный многочлен $P(x)$ по модулю 2 [15]:

$$K(x)\text{mod } P(x)=h_{m-1}\, x^{m-1}+\ldots+h_1\, x+h_0$$
$$H(x)= h_{m-1}\ldots\, h_1 h_0$$

где h_i - остатки от деления многочлена, $i=\overline{0,\ m-1}$.

Как и в предыдущем случае, правильный выбор многочлена $P(x)$ минимизирует вероятность коллизий между почти идентичными входными данными. В этом случае в качестве многочлена целесообразно использовать неприводимые многочлены. Они аналогичны простым числам в теории конечных полей. В данной статье будет проанализирована теория неприводимых многочленов. Методы, основанные на модульной арифметике, имеют большую вычислительную сложность, особенно при больших исходных данных.

Метод умножения используется в дополнение к методам, основанным на делении. Для вычисления хэш-кода методом умножения необходимо использовать хеш-функцию [14]:

$$H(k)=\lfloor M * (K * A) \rfloor$$

где A - рациональное число в диапазоне [0, 1], M - число всех возможных хеш-кодов, {} - операция взятия дробной части, ( ) - скобки приоритета, $\lfloor\ \rfloor$ - это операция взятия целой части числа.

Желательно выбрать значение M как степень двойки, потому что это переход к цифрам в компьютере и его можно быстро вычислить. Для равномерного распределения хеш-кодов важно выбрать соответствующее значение A. Чаще всего выбирается A≈0,6180339887, этот выбор основан на свойствах золотого сечения [16].

Метод свертки также используется как хэш-функция. Входные данные делятся на несколько частей, после чего выполняются операции сложения или неидентификации. Если значения двух цифр равны (0 или 1), то операция отсутствия идентичности дает 0, в противном случае -1 [16].

Существует много других хэш-функций, объем которых зависит от набора хешированных ключей. Невозможно однозначно определить, какую из функций выбрать, без учета характера входных значений. Когда выбирается хеш-функция, важно вычислять ее эффективно, поскольку поиск объекта за одну попытку не будет более эффективным, если на эту попытку затрачено больше времени, чем на несколько попыток с альтернативным

методом. Необходимо учитывать тот факт, что сжатие информации в системах защиты является неотъемлемой частью системы и должно выполняться в разумные сроки при обеспечении надлежащей надежности.

Эти методы широко используются при сжатии документов, которые будут подписаны или переданы по сети. Если исходные данные не являются числом, то перед применением хэш-функций, описанных выше, их необходимо преобразовать в целые числа. Например, для символьной строки как двоичного числа можно интерпретировать внутреннее двоичное представление кода для каждого символа.

Недостатком таких методов является то, что для большинства компьютеров двоичные представления всех букв или цифр очень похожи друг на друга. Чтобы избежать таких ситуаций, были созданы метод слияния и метод взвешивания для строковых входных данных.

В методе слияния с порядковым номером в ANSI-последовательности каждой буквы используется создание целого числа [16]. Когда есть некоторое целочисленное представление строки символов, то для того, чтобы убедить ее в приемлемом размере, можно использовать метод свертки или метод середины квадрата, который был дан ранее в работе.

Метод взвешивания использует значение позиции каждого символа, чтобы избежать коллизий при использовании анаграмм (перестановка букв определенного слова или фразы, которая приводит к другому слову или фразе) в качестве ключей [16]. В статье будет предложен метод хеш-функций, основанный на теории конечных полей.

Результаты и обсуждения. Построение хеш-функций, основанное на теории конечных полей. Конечные поля широко используются в криптосистемах. Большое количество криптографических протоколов и криптосистем основано на теории конечных полей. Алгоритмы, основанные на эллиптических кривых, которые являются одним из ключевых объектов исследования в современной криптографии, также используют конечные поля.

Конечное поле - это конечное множество, на котором определены произвольные операции, называемые сложением, умножением, вычитанием и делением (кроме деления на 0). Конечное поле обычно обозначается как F_q, где q - порядок поля, который всегда является степенью простого числа, его называют характеристикой поля [17]. Наиболее распространенным полем является $F_p=Z_p$, состоящее из всех остатков $\{0,1,\ldots,p\text{-}1\}$ от деления по модулю простого числа p. Операции в таких полях соответствуют правилам модульной арифметики.

Конечные поля удовлетворяют ряду следующих свойств [18]:

1) Характеристика конечного поля F отлична от нуля и является простым числом.

2) Для любых двух элементов a и b конечного поля с характеристикой p выполняется равенство:

$$(a+b)^p = a^p + b^p .$$

3) Любое конечное поле содержит p^k элементов, где p - характеристика поля, k - некоторое натуральное число.

4) Для заданных p и k существует единственное поле из p^k элементов с точностью до изоморфизма, которое обозначается F_{p^k}.

5) Мультипликативная группа $F^*_{p^k}$ ненулевых элементов поля F_{p^k} целиком порождается степенями некоторого элемента поля (она циклическая). Изучение мультипликативных групп имеет прикладное значение в криптографии, а также в задачи возведения в степень или извлечения корней [17].

Поле F_{p^k} может быть построено как фактор-кольцо $F_p[x]/f(x)$, где $f(x)$ - неприводимый многочлен над полем. Чтобы построить поле из элементов, достаточно найти неприводимый многочлен степени k над полем F_p и такой многочлен существует всегда [18].

Многочлен $f(x) \in F[x]$ неприводим над полем F или в кольце $F[x]$, если он имеет положительную степень и равенство $f(x) = g(x)h(x)$, где $g(x), h(x) \in F[x]$ выполняется, только если $g(x)$ или $h(x)$ – постоянный многочлен (степень многочлена ≤ 0) [18].

Можно построить методы хеширования на основе теории неприводимых многочленов. Результат этого метода следующий. Информационная последовательность вводится в алгоритм, который разбивается на структурные элементы - символы. Каждый символ в последовательности ANSI имеет порядковый номер, представленный как число. Простое число p выбирается в зависимости от количества возможных символов k в тексте, подаваемом на вход, оно удовлетворяет условию $p \geq k$. Далее, для преобразования этих символов целесообразно использовать неприводимый на поле характеристики p многочлен. Хеш-значение каждого символа можно получить, подставив его порядковое значение (в последовательности ANSI) в качестве аргумента неприводимого многочлена, а затем разделив его на простое число p по модулю. Чтобы получить хеш-значение всего текста, необходимо просуммировать хеш-значения каждого символа.

Этот метод эффективен для преобразования и сжатия исходных текстовых данных в указанный диапазон значений. Результатом этого преобразования является целочисленное значение свертки, что упрощает его использование в дальнейшем.

Использование неприводимых многочленов в качестве хеш-функции минимизирует вероятность коллизий. Это основано на том факте, что неприводимый многочлен не имеет корней в данном поле, и когда значение аргумента подставляется, функция не будет равна нулю. Можно выбрать те же числа, которые будут корнями этого многочлена и дадут нулевое значение, используя приводимый многочлен над заданным полем. Это означает, что некоторые символы можно заменить или полностью удалить, это не изменит окончательное значение хеш-функции. Таким образом, в таких алгоритмах целесообразно применять в точности неприводимые многочлены.

Однако такие методы дороги с точки зрения времени работы алгоритма и его вычислительной сложности для больших объемов входных данных. Кроме того, отдельной проблемой является выбор неприводимого многочлена над полем, который зависит от размерности исходных данных. Чем больше размерность исходных данных, тем сложнее задача нахождения неприводимого многочлена над полем данной характеристики простого числа.

Отдельный вопрос - это поиск простого числа из набора простых чисел. Основная проблема заключается в построении алгоритма факторизации простых множителей для определения простоты данного числа. К такому числу должны предъявляться особые требования, и поиск числа с заданными свойствами и размерностью, удовлетворяющего криптографическим системам, является другой сложной задачей.

Заключение. Как уже отмечалось ранее, хеш-функции могут использоваться в системах защиты информации и аутентификации, а также при формировании оптимальных структур данных. В ходе работы были проанализированы разрешения коллизий в таблицах данных. Рассмотрены методы хеширования целочисленных и строковых исходных данных с определением их достоинств и недостатков.

Также был предложен метод построения хеш-функции, основанный на теории конечных полей, с использованием неприводимого полинома над выбранным конечным полем характеристики простых чисел. Доказано, что этот метод устойчив к столкновениям. Но в то же время у него есть недостаток в смысле вычислительной сложности из-за работы с

большими массивами входных данных. Также должен быть построен эффективный метод нахождения простого числа заданной размерности.

Благодарности. Работа выполнена при поддержке грантового финансирования научно-технических программ и проектов Министерством науки и образования Республики Казахстан (грант «Интеллектуальная система поддержки и контроля дистанционных образовательных технологий», 2020-2022 годы, № AP08856687).

ЛИТЕРАТУРА

[1] Sedgewick R. Fundamental algorithms on C++.- Kyiv: Publishing house "DiaSoft".- 2001.- 688 p.

[2] Whitfield Diffiee, Martin E. Hellman. New directions in cryptography // IEEE Trans. on Information Theory, Vol. IT-22, No. 6.- 1976. - P. 644-654.

[3] Graham R., Knuth D. Concrete mathematics. A foundation for computer scienc .– 1998. - 703 p.

[4] Шнайер Б. Прикладная криптография: протоколы, алгоритмы, исходные тексты на Си / пер. с англ.; под ред. Н. Дубновой. - Изд. 2-е.- М.: Диалектика, 2003. – 610 с.

[5] Landau S. Find Me a Hash // Notices Amer. Math. Soc. – 2006. - V.53. – P. 330-332.

[6] Shpilrain V. Hashing with Polynomials // Information Security and Cryptology – ICISC 2006. - LNCS 4296. - Springer, 2006. - P. 22-28. DOI: https://doi.org/10.1007/11927587_4

[7] Menezes A., P. van Oorschot, S Vanstone. Handbook of Applied Cryptography, CRC Press, 1997.

[8] Авезова Я.Э. Современные подходы к построению хеш-функций на примере финалистов конкурса SHA-3 // Вопросы кибербезопасности.- 2015.- №3(11).- с.60-67.

[9] Хомич Э. А. Неприводимые многочлены над конечными полями и связь с криптографией // Academic Publicistics.- 2017.-№3.- с. 19-22.

[10] Турусбекова У.К., Тургинбаева А.С. Хеширование на основе многочленов// Вестник КазНУ. Серия математика, механика, информатика.- 2020.-№3(107).-с.74-83.

[11] Abdymanapov S., Turusbekova U., Turginbayeva A., Altynbek S. Research of irreducible normal polynomials special type over a field of characteristic 2 // IAENG International Journal of Applied Mathematics. - 2020.-V.50(4). – P. 777-782.

[12] Smart. N. Cryptography. - Moscow: Techno sphere. - 2005.- 528 p.

[13] Vostrov G., Bezrukova Yu. Modeling of dynamic data ptotection processes based on a discrete logarithm // ELTECS.-2017.

[14] https://ru.wikipedia.org/wiki/RIPEMD-160

[15] https://ru.wikipedia.org/wiki/Хеширование.

[16] https://studopedia.ru/2_80095_funktsiiheshirovaniya.html

[17] Лидл Р., Нидеррайтер Х. Конечные поля. - в 2 т. / пер. с англ.; под ред. В.И. Нечаева. - М.: Мир.- 1988.- Т.1.- 430 с.

[18] Crandall R. E., Pomerance C. B. Prime Numbers: cryptographic and computational aspects, Transl. from English / Ed. and with a preface by V. Chubarikova.-Moscow: URSS: Book House "LIBROKOM", -2011.- 664 p.

REFERENCES

[1] Sedgewick R. Fundamental algorithms on C++.- Kyiv: Publishing house "DiaSoft".- 2001.- 688 p.

[2] Whitfield Diffiee, Martin E. Hellman. New directions in cryptography // IEEE Trans. on Information Theory, Vol. IT-22, No. 6.- 1976. - P. 644-654.

[3] Graham R., Knuth D. Concrete mathematics. A foundation for computer scienc .– 1998. - 703 p.

[4] Shnaier B. Prikladnaya kriptografiya: protokoly, algoritmy, iskhodnye teksty na Ci / per. s angl.; pod red. N. Dubnovoi. - Izd. 2-e.- M.: Dialektika, 2003. – 610 s.

[5] Landau S. Find Me a Hash // Notices Amer. Math. Soc. – 2006. - V.53. – P. 330-332.

[6] Shpilrain V. Hashing with Polynomials // Information Security and Cryptology – ICISC 2006. - LNCS 4296. - Springer, 2006. - P. 22-28. DOI: https://doi.org/10.1007/11927587_4

[7] Menezes A., P. van Oorschot, S Vanstone. Handbook of Applied Cryptography, CRC Press, 1997.

[8] Avezova Ya.E. Sovremennye podkhody k postroeniyu khesh-funktsii na primere finalistov konkursa SHA-3 // Voprosy kiberbezopasnosti.- 2015.- №3(11).- s.60-67.

[9] Khomich E. A. Neprivodimye mnogochleny nad konechnymi polyami i svyaz' s kriptografiei // Academic Publicistics.- 2017.-№3.- s. 19-22.

[10] Turusbekova U.K., Turginbaeva A.S. Kheshirovanie na osnove mnogochlenov// Vestnik KazNU. Seriya matematika, mekhanika, informatika.- 2020.-№3(107).-s.74-83.

[11] Abdymanapov S., Turusbekova U., Turginbayeva A., Altynbek S. Research of irreducible normal polynomials special type over a field of characteristic 2 // IAENG International Journal of Applied Mathematics. - 2020.-V.50(4). – P. 777-782.

[12] Smart. N. Cryptography. - Moscow: Techno sphere. - 2005.- 528 p.

[13] Vostrov G., Bezrukova Yu. Modeling of dynamic data ptotection processes based on a discrete logarithm // ELTECS.-2017.

[14] https://ru.wikipedia.org/wiki/RIPEMD-160

[15] https://ru.wikipedia.org/wiki/Хеширование.

[16] https://studopedia.ru/2_80095_funktsiiheshirovaniya.html

[17] Lidl R., Niderraiter Kh. Konechnye polya. - v 2 t. / per. s angl.; pod red. V.I. Nechaeva. - M.: Mir.- 1988.- T.1.- 430 s.

[18] Crandall R. E., Pomerance C. B. Prime Numbers: cryptographic and computational aspects, Transl. from English / Ed. and with a preface by V. Chubarikova.-Moscow: URSS: Book House "LIBROKOM", -2011.- 664 p.

[1]Ү.Қ. Тұрысбекова*, [1]С.А. Алтынбек, [2]А.С. Тургинбаева, [1]Л. Мерейхан

[1]Қазақ экономика, қаржы және халықаралық сауда университеті, Нұр-Сұлтан, Қазақстан

[2]Л.Н. Гумилев атындағы Еуразиялық ұлттық университеті, Нұр-Сұлтан, Қазақстан

*e-mail: umut.t@mail.ru

КЕЛТІРІЛМЕЙТІН КӨПМҮШЕЛІКТЕРДІ ҚОЛДАНУ АРҚЫЛЫ АҚЫРЛЫ ӨРІСТЕР ТЕОРИЯСЫ НЕГІЗІНДЕ ХЕШ-ФУНКЦИЯЛАРДЫ ҚҰРУ

Андатпа. Ақпарат көлемінің ұлғаюымен, үлкен көлемдегі мәліметтермен байланысты проблемалар шиеленіседі, олар ілгеріде сақтау, жіберу немесе өңдеу процестерін жүзеге асыруды талап етеді. Файлдардың үлкен көлемімен жұмыс жасау бұл процестерді едәуір қиындатады, сондықтан оларды тиімді өңдеу үшін қолайлы көлемде файлдардың көлемін сығуға мүмкіндік беретін алгоритмдердің болуы қажет.

Файлдармен өзара әрекеттесу процесінде хеш - функциялары маңызды рөл атқарады. Хеш-функцияларды қолдану белгілі бір алгоритмге сәйкес бастапқы деректерді тұрақты ұзындықтағы тізбекке түрлендіруді көздейді. Бұл көптеген файлдар ішінен оларды қарау, өзгерту немесе жою, сонымен қатар мәліметтерді бөгде адамдар өзгертуге болмайтын жағдайларда олардың өзгертілмегендігін тексеру үшін файлдарды салыстыруды іздеуді жеделдетуге мүмкіндік береді. Осылайша, хештеу деректерді файл түрінде сақтау, жіберу немесе өңдеу мәселесі туындайтын барлық салаларда қолданылады, дәлірек, криптографияда, компьютерлік графикада, компьютер мен Интернетте мәліметтерді ұйымдастыру барысында.

Практикада соқтығысудың теориялық ықтималдығы нөлге жақын функциялар қолданылады, бірақ қуатты есептеу құралдарының пайда болуымен соқтығысуды табу шешілмейтін мәселе болмауы мүмкін. Осы себепті қолданыстағы алгоритмдер үнемі жетілдіруді қажет етеді. Мұнда күрделілік теориясы, атап айтқанда, алгебралық сандар теориясы ерекше рөл атқарады. Сандар теориясы қазіргі кезде криптографияда кеңінен қолданылады; алайда тиімді факторизация алгоритмдері пайда болған кезде бұл жеткіліксіз. Осыған байланысты ақырлы өрістер теориясын қолдану қажеттілігі туындайды. Мұндай өрістердегі есептеулер оның элементтері бойынша таңдалған ақырлы өрісте амалдарды оның шектерінен шықпай орындауға мүмкіндік береді, сонымен қатар хеш - функцияларын бастапқы деректерді қалпына келтіруге төзімді етеді. Сондықтан мақалада ақырлы өрістерде келтірілмейтін көпмүшелікті модуль бойынша бөлуді қолданатын полиномиалды функциялар қарастырылады. Келтірілмейтін көпмүшеліктерді ақырлы өрістерде қолдану тиімді қорғаныс құралы бола алады.

Ұсынылған мақаланың негізгі мақсаты - ақырлы өрістер теориясына негізделген хеш - функциясын құру. Электронды-цифрлық қолтаңба алгоритмдері үшін стандартты криптографиялық хеш - функциялары қарастырылған. Әр түрлі бастапқы деректерді хештеу әдістері талданды.

Мәтіндік файлдарды келтірілмейтін көпмүшеліктер көмегімен хештеу әдісі ұсынылған. Мақаланың нәтижелерін криптографиялық қосымшалар мен кодтау теориясында қолдануға болады.

Негізгі сөздер: хеш - функция, ақырлы өріс, келтірілмейтін көпмүшелік, электронды-цифрлық қолтаңба, соқтығысу.

[1]U.K. Turusbekova*, [1]S.A. Altynbek, [2]A.S. Turginbayeva, [1]L. Mereikhan
[1]Kazakh University of Economics, Finance and International Trade, Nur-Sultan, Kazakhstan
[2]L.N. Gumilyov Eurasian National University, Nur-Sultan, Kazakhstan
*e-mail: umut.t@mail.ru

CONSTRUCTION OF HASH FUNCTIONS BASED ON THEORY OF FINITE FIELDS WITH THE USE OF IRREDUCIBLE POLYNOMIALS

Abstract. With an increase in the amount of information, the problems associated with large amounts of data are aggravated, which in the future require the implementation of storage, transfer or processing processes. Working with large volumes of files significantly complicates these processes, and therefore there is a need for the existence of algorithms that allow compressing the volumes of files to the required size, acceptable for their efficient processing.

Hash functions play an important role in the process of interacting with files. The use of hash functions implies the transformation of the original data according to a certain algorithm into a sequence of fixed length. This allows you to significantly speed up the search among a large number of files to view, modify or delete, to compare files, to verify immutability in cases where the data should not be changed by unauthorized persons. Thus, hashing is used in all areas where the question of storing, transmitting or processing data in the form of files arises, namely in cryptography, computer graphics, when organizing data on a computer and on the Internet.

In practice, functions are used for which the theoretical probability of collisions is close to zero, but with the advent of more powerful computational tools, collision detection may not be such an indescribable task. For this reason, existing algorithms require continuous improvement. A special role in this is played by the problem of complexity theory, namely, algebraic number theory. Number theory is now widely used in cryptography; however, this is not enough when efficient factorization algorithms appear. In this regard, there is a need to use the theory of finite fields. Calculations in such fields allow you to perform operations in the selected finite field on its elements without going beyond its limits, and also makes hash functions resistant to the restoration of the original data. Therefore, the article discusses polynomial functions using modulo division of an irreducible polynomial in finite fields. Using irreducible polynomials over finite fields can be a more effective means of protection.

The main purpose of this paper is to construct a hash function based on the theory of finite fields. Standard cryptographic hash functions for digital signature algorithms are considered. Methods of hashing of various types of initial data are analyzed. A method for hashing text files using irreducible polynomials is proposed. The results of the article can be used in cryptographic applications and coding theory.

Keywords: hash function, finite field, irreducible polynomial, electronic digital signature, collision.