УДК 519.713

ПРАКТИЧЕСКОЕ ПРИМЕНЕНИЕ ХЭШ-ФУНКЦИЙ

Бондарев В. С., студ., Поздняков К. В., студ., Никонова Т. В., к.ф.-м.н., доц.

Витебский государственный технологический университет,
г. Витебск, Республика Беларусь

Реферат. *В данной работе рассматривается роль и применение хэш-функций в современной информационной среде. Хэш-функции представляют собой математические инструменты, которые принимают входные данные различного размера и преобразуют их в набор символов определенной длины, известный как хэш. Они являются важной составляющей в обеспечении безопасности данных и целостности информационных систем. В работе рассмотрены основные виды и области применения хэш-функций.*

Ключевые слова: хэш-функции, информационная безопасность, целостность данных, криптография, хранение паролей, цифровые подписи.

Хэш-функция – это математический инструмент, который принимает входные данные различного размера, например, текстовые файлы, и превращает их в набор символов определенной длины, называемый хэшем. Каждый уникальный набор входных данных будет иметь свой уникальный хэш. Главная идея хэш-функций в том, что они должны быть быстрыми в вычислении и невозможными для обратного преобразования [1].

Рассмотрим несколько примеров применения хэш-функций:

1) проверка целостности сообщений и файлов: сравнивая хэш-значения сообщений, вычисленные до и после передачи, можно определить, были ли внесены какие-либо изменения в сообщение или файл;

2) верификация пароля: проверка пароля обычно использует криптографические хэши. Хранение всех паролей пользователей в виде открытого текста может привести к массовому нарушению безопасности, если файл паролей будет скомпрометирован. Одним из способов уменьшения этой опасности является хранение в базе данных не самих паролей, а их хэшей. При выполнении хэширования исходные пароли не могут быть восстановлены из сохраненных хэш-значений, поэтому если пользователь забывает свой пароль, веб-сервисы предлагают сбросить его и придумать новый;

3) цифровая подпись: подписываемые документы имеют различный объем, поэтому зачастую в схемах электронной цифровой подписи подпись ставится не на сам документ, а на его хэш. Вычисление хэша позволяет выявить малейшие изменения в документе при проверке подписи. Хэширование не входит в состав алгоритма электронной цифровой подписи, поэтому в схеме может быть применена любая надежная хэш-функция.

Криптографическая хэш-функция – это математический алгоритм, который отображает данные произвольного размера в битовый массив фиксированного размера [1].

Результат, производимый хэш-функцией, называется «хэш-суммой» или же просто «хэшем», а входные данные часто называют «сообщением».

Рассмотрим условия «идеальной» хэш-функции:

- хэш-функция является детерминированной, то есть одно и то же сообщение приводит к одному и тому же хэш-значению;
- значение хэш-функции быстро вычисляется для любого сообщения;
- невозможно найти сообщение, которое дает заданное хэш-значение;
- невозможно найти два разных сообщения с одинаковым хэш-значением;
- небольшое изменение в сообщении изменяет хэш настолько сильно, что новое и старое значения кажутся не коррелирующими.

Рассмотрим пример воздействия хэш-функции SHA3-256 [2].

Число 256 в названии алгоритма означает, что на выходе будет находиться строка фиксированной длины 256 бит независимо от того, какие данные поступят на вход.

На рисунке ниже указано, что на выходе функции имеется 64 цифры шестнадцатеричной системы счисления. Переводя это в двоичную систему, выходит 256 бит [1].

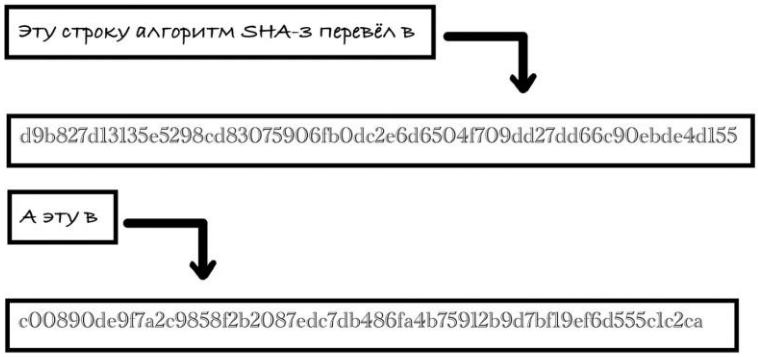

Рисунок 1 – Пример работы алгоритма SHA-3

Криптографическая хэш-функция должна уметь противостоять всем известным типам криптоаналитических атак. В теоретической криптографии уровень безопасности хэш-функции определяется с использованием следующих свойств: Pre-image resistance, Second pre-image resistance и Collision resistance [3].

Так называемое свойство Pre-image resistance заключается в том, что имея заданное значение h, должно быть сложно найти любое сообщение m такое, что h имеет значение хэша от m.

Свойство Second pre-image resistance заключается в том, что имея заданное входное значение m1, должно быть сложно найти другое входное значение m_2 такое, что хэш от m_1 равен хэшу от m_2.

Свойство Collision resistance заключается в том, что должно быть сложно найти два различных сообщения m_1 и m_2 таких, что хэши от каждого из этих сообщений будут иметь равные значения.

Такая пара сообщений m_1 и m_2 называется коллизией хэш-функции.

Далее более подробно рассмотрим каждое из перечисленных свойств.

Collision resistance: как уже упоминалось ранее, коллизия происходит, когда разные входные данные производят одинаковый хэш. Таким образом, хэш-функция считается устойчивой к коллизиям до того момента, пока не будет обнаружена пара сообщений, выдающая одинаковый выход. Стоит отметить, что коллизии всегда будут существовать для любой хэш-функции по той причине, что возможные входы бесконечны, а количество выходов конечно. Хэш-функция считается устойчивой к коллизиям, когда вероятность обнаружения коллизии настолько мала, что для этого потребуются миллионы лет вычислений. Несмотря на то, что хэш-функций без коллизий не существует, некоторые из них достаточно надежны и считаются устойчивыми к коллизиям.

Pre-image resistance: это свойство называют сопротивлением прообразу. Хэш-функция считается защищенной от нахождения прообраза, если существует очень низкая вероятность того, что злоумышленник найдет сообщение, которое сгенерировало заданный хэш. Это свойство является важным для защиты данных, поскольку хэш сообщения может доказать его подлинность без необходимости раскрытия информации.

Second pre-image resistance: данное свойство называют сопротивлением второму прообразу. Атака по нахождению второго прообраза происходит, когда злоумышленник находит определенный вход, который генерирует тот же хэш, что и другой вход, который ему уже известен. Другими словами, злоумышленник, зная, что h равен значению хэшу от m_1 пытается найти m_2 такое, что хэш от m_2 равен h.

Национальный институт стандартов и технологий (NIST) в течение 2007–2012 провел конкурс на новую криптографическую хэш-функцию, предназначенную для замены SHA-1 и SHA-2, и им стал алгоритм Keccak [4].

Хэш-функции семейства Keccak построены на основе конструкции криптографической губки, в которой данные сначала «впитываются» в губку, а затем результат «отжимается» из губки.

Любая губчатая функция Keccak использует одну из семи перестановок Keccak-f которая обозначается Keccak-f [b] , где $b \in \{25, 50, 100, 200, 400, 800, 1600\}$[3].

Keccak-f перестановки представляют собой итерационные конструкции, состоящие из последовательности почти одинаковых раундов. Число раундов n_r зависит от ширины

перестановки и задаётся как $n_r = 12 + 2l$, где $2l = b / 25$.

В качестве стандарта SHA-3 была выбрана перестановка Keccak-f [1600], для неё количество раундов $n_r = 24$.

Хэш-функции играют ключевую роль в развитии сферы безопасности данных, обеспечивая целостность информации, защиту от подделок и обеспечение конфиденциальности.

Несмотря на свою широкую применимость, важно осознавать ограничения и уязвимости хэш-функций, такие как коллизии и возможность атак методом подбора. Это подчеркивает необходимость использования дополнительных мер защиты данных, включая соль и итеративные методы хэширования.

Дальнейшее развитие и исследование в области хэш-функций представляет собой перспективное направление, включая улучшение стойкости алгоритмов, разработку новых методов защиты и адаптацию к изменяющимся угрозам информационной безопасности [5].

В целом, хэш-функции остаются неотъемлемой частью современной криптографии и информационной безопасности, обеспечивая надежность и целостность данных в различных сценариях и поддерживая доверие пользователей к цифровым системам и сервисам.

Список использованных источников

1. A Beginner's Guide to Cryptographic Hash Functions [Электронный ресурс]. – Режим доступа: https://medium.com/asecuritysite-when-bob-met-alice/a-beginners-guide-to-cryptographic-hash-functions-2b18fcd2d2c0. – Дата доступа: 06.04.2024.
2. What are Cryptographic Hash Functions [Электронный ресурс]. – Режим доступа: https://www.crypto101.io/cryptographic-hash-functions. – Дата доступа: 04.04.2024.
3. An introduction to hashing functions for data mining [Электронный ресурс]. – Режим доступа: https://bpostance.github.io/posts/introduction-to-hashing/. – Дата доступа: 07.04.2024.
4. Understanding Cryptographic Hash Functions [Электронный ресурс]. – Режим доступа: https://towardsdatascience.com/understanding-cryptographic-hash-functions-36f56dfa2a7a. – Дата доступа: 03.04.2024.
5. Hash functions [Электронный ресурс]. – Режим доступа: https://m-mokhtari.gitbook.io/cryptography/hash-functions. – Дата доступа: 07.04.2024.

УДК 519.876.5

ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ ТЕХНОЛОГИЧЕСКИХ ПРОЦЕССОВ

Козик Д. С., студ., Дмитриев А. П., к.т.н., доц.

Витебский государственный технологический университет,
г. Витебск, Республика Беларусь

Реферат. *В статье рассмотрены теоретические аспекты имитационного моделирования технологических процессов, в частности его методы и инструменты. Имитационные модели и принципы статистического имитационного моделирования, виртуально воссоздавая и изучая сложные производственные системы, минимизируют риски и затраты, связанные с физическими экспериментами.*

Ключевые слова: имитационное моделирование, технологические процессы, проектирование, анализ.

В современном мире, где технологии развиваются с невероятной скоростью, возникает необходимость в эффективных методах для планирования, анализа и оптимизации технологических процессов. Имитационное моделирование является одним из ключевых инструментов, позволяющих достигать этих целей, так как предоставляет возможность виртуально воссоздать и изучить сложные системы, минимизируя риски и снижая затраты, связанные реальным выполнением работ.

Имитационное моделирование – частный случай математического моделирования.