УДК 004.65:004.056

13. ОБЗОР АЛГОРИТМОВ ФОРМИРОВАНИЯ ХЕШ ФУНКЦИЙ

Яковлев Н.Е., Ковалевский Я.А. студенты гр.378101, Раптунович О. М., магистрант группы 376741

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

Ефремов А.А. – канд. экон. наук, доцент каф. ЭИ

В данной статье рассматривается роль и применение хеш-функций в различных областях, включая информационную безопасность, хранение данных и структуры данных. Обсуждаются история хеширования, ключевые свойства хеш-функций, методы обеспечения безопасности паролей и цифровых подписей, а также применение хеш-таблиц. Подробно рассматриваются различные виды хеш-функций их преимущества и недостатки. Данная статья предоставляет понятное объяснение принципов работы хеш-функций и их важность в современных информационных технологиях.

Ключевые слова. Хеширование, хеш, хеш-таблицы, ключ, алгоритм, коллизия.

Хеширование давно стало неотъемлемой частью информационных систем. Впервые информацию захешировал Христиан Гюйгенс в 1650-х, а хеширование в современном виде было описано Арнольдом Думи в 1956 году.

Хеш-функции играют важную роль в области информационной безопасности и хранения данных. Они представляют собой алгоритмы, которые принимают на вход данные произвольной длины, называемые ключом и преобразуют их в фиксированный набор битов, называемый хешем.

Математически в общем виде хеш-функция определяется формулой:

$$h = H(M) \tag{1}$$

где h – это хеш, M – это ключ, H – это хеш-функция.

Одним из ключевых свойств хорошей хеш-функции является равномерное распределение значений хеша, что обеспечивает равномерное распределение хешей по всему диапазону возможных значений. Это свойство важно для предотвращения ситуаций, когда одни и те же хеши генерируются для разных входных данных, что может привести к коллизиям и потере данных.

Важной характеристикой хеш-функций является их необратимость, то есть отсутствие возможности восстановления исходных данных из хеша. Это обеспечивает безопасность данных, поскольку нельзя восстановить конфиденциальную информацию, даже если известен хеш. Для криптографически стойких хеш-функций это свойство является обязательным и гарантирует невозможность восстановления исходных данных без знания входных данных.

Так же ещё одно свойство хеш-функций - уникальность. Идеальная хеш-функция выдает стопроцентно уникальный результат для каждого возможного набора данных. В реальности такое невозможно, и иногда случаются коллизии — одинаковые хеши для разных сведений. Но существующие хеш-функции достаточно сложны, поэтому вероятность коллизии сводится к минимуму.

Еще одним важным аспектом хеш-функций является их устойчивость к коллизиям. Коллизия возникает, когда разные входные данные приводят к одному и тому же значению хеша. Хорошая хеш-функция должна обладать высокой устойчивостью к коллизиям, что обеспечивает надежность и безопасность при хранении и обработке данных. Для этого используются различные методы и алгоритмы, направленные на минимизацию вероятности возникновения коллизий при использовании хеш-функций в различных приложениях и системах.

Хеш-функций существует множество видов. Но среди них нужно выделить основные:

- Хеш-функции вычисления контрольной суммы

- Криптографические хеш-функции
- Хеш-функции для хеш-таблиц

Хеш-функции для вычисления контрольной суммы играют важную роль в обеспечении целостности данных во множестве приложений, включая передачу данных через сети, хранение данных на дисках и другие сценарии, где важно убедиться в том, что данные не были искажены или повреждены в процессе передачи или хранения.

При передаче информации через ненадежные каналы, такие как сети передачи данных или беспроводные соединения, данные могут быть подвержены воздействию шумов, помех или даже злонамеренных атак.

Хеш-функции, такие как алгоритмы циклического избыточного кодирования (далее – CRC), становятся необходимым инструментом для обнаружения и исправления ошибок в таких ситуациях.

Алгоритмы CRC оперируют на уровне битов и применяются для вычисления контрольной суммы для блоков данных. Эта контрольная сумма затем встраивается в сообщение и отправляется вместе с ним. При получении сообщения получатель также вычисляет контрольную сумму для полученных данных и сравнивает ее с контрольной суммой, полученной в сообщении. Если контрольные суммы совпадают, это свидетельствует о том, что данные были переданы без искажений. В противном случае, если контрольные суммы не совпадают, это указывает на наличие ошибок в данных, и получатель может запросить повторную передачу.

Важно отметить, что хеш-функции CRC имеют различные варианты, каждый из которых предоставляет разные уровни надежности и эффективности в зависимости от требований конкретного применения. Например, CRC-32 широко используется в сетевых протоколах, таких как Ethernet и ZIP-архивации, в то время как CRC-16 может быть предпочтителен для более легких приложений с ограниченными ресурсами.

Таким образом, использование алгоритмов CRC для вычисления контрольной суммы является важным шагом для обеспечения целостности данных в различных приложениях, где надежность передачи данных играет критическую роль.

Криптографические хеш функции в основном применяются для безопасного хранения паролей и для цифровых подписей.

Применение хеш функций для безопасного хранения паролей заключается в том, что пороли в базе данных хранятся не в исходном виде, а в захешированном, поэтому если злоумышленник получит доступ к базе данных, то он не сможет использовать эти пароли.

Для цифровой подписи хеш-функции используются с целью подтверждения того, что данные не были изменены. Для этих целей данные хешируются и отправляются одновременно в исходном виде и в виде хеша, а после получения исходный вид хешируется и сравнивается с полученным хешем.

Разработка алгоритмов формирования криптографических хеш функций является очень важной задачей в информационной безопасности. Главной целью при разработке новых алгоритмов является защита от взлома. Например, при хранении пароля в базе данных, злоумышленник может попытаться вычислить ключ, который при передаче в хеш функцию будет преобразовываться в хеш, совпадающий с захешированным паролем. Все современные хеш функции имеют защиту от такого рода атак. При попытке подобрать ключ, который хеш-функция преобразует в такой же хеш, как и заданный, может уйти очень большое количество времени. Многие хеш-функции были взломаны, так как были найдены методы нахождения коллизий исходных хешей. Для современных алгоритмов такие методы не были найдены, поэтому эти методы широко используются в реальных проектах. Не исключено, что в будущем эти алгоритмы станут не безопасны, так как для них могут быть найдены коллизии, и разработчикам данных методов необходимо будет разработать более безопасные методы.

Для того, чтобы выбрать нужную криптографическую хеш функцию необходимо учитывать различные факторы. Различные виды криптографических хеш функций, вместе с их различиями представлены в таблице 1.

Таблица 1 – Виды криптографических хеш-функций

Алгоритм/Семейство алгоритмов	Преимущества	Недостатки	Год взломан	Использование
MD5	Быстрый, широко используется	Недостаточная стойкость к коллизиям	2004 год (были обнаружены коллизии)	Использовался для хеширования паролей
Семейство SHA-1	Быстрый, широко используется	Недостаточная стойкость к коллизиям.	2017 год (были обнаружены коллизии)	Использовался для хеширования паролей, создания цифровые подписи

Семейство SHA-2	Высокая стойкость к коллизиям, широко используется	Длинные хеши, требует больше вычислительных ресурсов	Не взламывался	Хеширование паролей, цифровые подписи, блокчейн
Семейство SHA-3	Высокая стойкость к коллизиям, быстрый	Относительно новый, менее распространен	Не взламывался	Хеширование паролей, цифровые подписи, блокчейн
HMAC	Поддерживает аутентификацию с использованием ключей	Может быть сложным в использовании	Не взламывался	Аутентификация сообщений, аутентификация запросов к API
Bcrypt	Медленный, устойчив к атакам перебора паролей	Может быть медленным для больших нагрузок	Не взламывался	Хеширование паролей
Scrypt	Медленный, устойчив к атакам перебора паролей	Может быть медленным для больших нагрузок	Не взламывался	Хеширование паролей, доказательство работы

Исходя из данных, представленных в таблице 1 можно сделать вывод, что выбора криптографической хеш-функции во многом зависит от области использования и от потребности в скорости вычисления хеша.

В основе работы такой структуры данных как хеш-таблица лежит хеш-функция. Хеш-таблицы — это одна из ключевых структур данных, применяемых в компьютерных науках и инженерии для эффективного хранения и поиска информации. Они оперируют на основе принципа хеширования, который заключается в использовании хеш-функций для преобразования ключей или значений в индексы массива.

Применение хеш-функций в хеш-таблицах обеспечивает мгновенный доступ к данным. При добавлении элемента данных в хеш-таблицу его ключ обрабатывается хеш-функцией, определяющей индекс массива, где будет располагаться элемент. При поиске элемента ключ также проходит хеширование, после чего программа направляется непосредственно к соответствующему индексу массива, обеспечивая сверхбыстрый доступ к данным. Благодаря этой высокой скорости операции поиска хеш-таблицы широко применяются в различных областях, включая базы данных, кэширование, сетевые маршрутизаторы и многие другие. Важно отметить, что они также эффективно решают проблему коллизий, применяя методы управления коллизиями, такие как метод цепочек или открытая адресация.

Хеш функции, используемые в хеш-таблицах, должны быть эффективными с точки зрения равномерного распределения значений, чтобы минимизировать коллизии.

Самые популярные виды хеш функций в хеш таблицах:

1. Хеш-функция деления. Данная хеш-функция это – один из наиболее простых методов хеширования. Он вычисляет остаток от деления ключа на размер хеш-таблицы. Такая хеш-функция обеспечивает высокую скорость вычисления хеша, что делает данную функцию подходящей для использования в хеш-таблицах, так как при работе с хеш-таблицами важна скорость доступа к данным.
2. Хеш-функция умножения. Этот метод использует умножение ключа на дробное число и дальнейшее извлечение дробной части результата. Главной частью алгоритма формирования данной хеш функции является выбора константы умножения. Это должно быть положительное число, обычно близкое к значению золотого сечения. Однако это и является главным недостатком данной хеш-функции. Результат данной хеш-функции сильно зависит от выбора константы для умножения, неправильный выбор данной константы может привести к неравномерному распределению значений хеша и многократному повышению числа коллизий.
3. Универсальное хеширование. Этот алгоритм хеширования заключается в использовании случайной хеш-функции из определённого семейства хеш-функций. Сначала выбирается семейство хеш-функций, из которого при вставке ключа в хеш-таблицу случайным образом выбирается одна из хеш-функций данного семейства. Преимущество данного алгоритма заключается в снижении вероятности возникновения коллизий до минимального значения. Этот подход использует семейство хеш-функций, из которого выбирается случайная функция для каждого использования.
4. Хеш-функция FNV – семейство различных хеш-функций. Хеш-функции FNV представляют из себя простые в реализации хеш-функции, которые обеспечивают равномерное распределение хешей по всему диапазону возможных значений, что означает, что даже небольшие изменения

ключа приводят к существенному изменению результата хеш-функции. Также эти хеш-функции обладают очень низкой вероятностью возникновения коллизии.

Список использованных источников:

1. Функции хеширования. Понятия, требования, классификация, свойства и применение / И.Д. Горбенко, И.А. Штанько // Радиоэлектроника и информатика, 1998. – С.64-69.

2. Анализ современных методов хеширования / Г.Х. Шамухамедов [и др.] // Science Time. 2015. – С.590-593.

3. Исследование реализаций алгоритмов контрольной суммы CRC / С.В. Клименко [и др.] // Известия Петербургского университета путей сообщения. 2018. – С.471-476.

4. Хеш-функция, что это такое? [Электронный ресурс]. – Режим доступа: https://habr.com/ru/articles/534596/

5. Универсальное хеширование [Электронный ресурс]. – Режим доступа: https://proproprogs.ru/structure_data/std-hash-funkcii-universalnoe-heshirovanie

6. Универсальное хеширование [Электронный ресурс]. – Режим доступа: https://proproprogs.ru/structure_data/std-hash-funkcii-universalnoe-heshirovanie

.