

Применение хеш-функций для защиты паролей аутентификации

Булавин Вадим Андреевич

Воронов Данила Андреевич

г. Орёл

***Аннотация:** в данной статье рассматривается метод преобразования произвольного массива данных, состоящего из букв и цифр в строку фиксированной длины, а также оценка его возможностей для обеспечения безопасного хранения паролей аутентификации.*

***Ключевые слова:** хеширование, хеш-функция, аутентификация, коллизия, открытый текст, радужные таблицы, алгоритмы.*

Понятие аутентификации состоит в проверке подлинности, т.е. необходимости убедиться в том, что пользователь использует набор учетных данных, таких как имя пользователя и пароль, известных только ему и никто другой, не сможет получить к ним доступ. Однако существует возможность того, что злоумышленник сможет получить пароли, находящиеся в базе данных. Поэтому требуется способ, который может сохранять эти учетные данные в безопасности.

Безопасность базы данных, зависит от того, как хранится пароль. Самый простой, но и наименее безопасный формат хранения паролей – это открытый текст. Если злоумышленник получит доступ к базе данных и узнает таблицу паролей, он сможет получить доступ к каждой учетной записи пользователей.

Более надежный способ сохранить пароль – преобразовать его в данные, которые нельзя обратить в исходный пароль. Этот механизм известен как хеширование. В криптографии хеш-функция – это математический алгоритм, который отображает данные любого размера в битовую строку фиксированного размера. Хеш-функции, используемые в криптографии, имеют следующие ключевые свойства:

вычислить хеш-код легко, но трудно или невозможно повторно сгенерировать входные данные, если известно только значение хеш-функции; трудно создать начальную комбинацию, которая соответствовала бы конкретному хеш-коду.

Таким образом, в отличие от шифрования, хеширование является односторонним механизмом. Хешированные данные практически не могут быть «дехешированы».

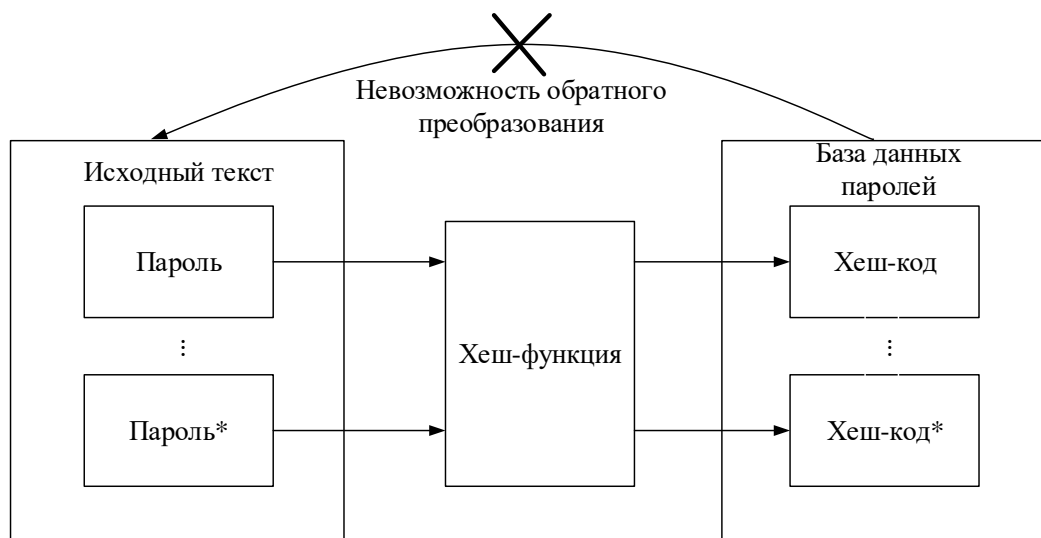


Рис. 1 Применение алгоритма хеширования для обеспечения безопасности хранения паролей в базе данных

Некоторые хеш-функции широко используются, к примеру CRC32, CityHash, FNV-1a и многие другие, но их свойства не обеспечивают требуемой безопасности, т.к. выходные данные имеют равномерное распределение. Более устойчивыми к атакам являются криптографические хеш-функции, т.к. они практически необратимы. Хеш-функции ведут себя как односторонние функции, используя математические операции, которые чрезвычайно сложно восстановить, например, оператор по модулю. Эта операция является детерминированной, поскольку одни и те же входные данные всегда дают один и те же выходные. Однако важной характеристикой операции по модулю является то, что невозможно найти исходные операнды с учетом результата. В этом смысле хеш-функции необратимы. Еще одно достоинство безопасной хеш-функции состоит в том, что ее результат трудно

предсказать. Это свойство известно как эффект лавины, и заключается в том, что при незначительном изменении входного сообщения значительно изменяется выходное. Следовательно, не существует реального способа определить, какой получится хеш-код, основываясь на значениях двух предыдущих входных сообщений.

Также есть свойство, делающее хеш-функции, подходящими для хранения паролей – это их детерминированность.

Детерминированная функция – это функция, которая при одном и том же вводе исходных данных всегда производит один и тот же вывод. Это важно для аутентификации, поскольку требуется гарантия того, что данный пароль всегда будет давать один и тот же хеш-код; в противном случае было бы невозможно последовательно проверять учетные данные пользователя с помощью этого метода.

Нахождение исходного пароля, на основе атаки полным перебором в значительной степени неэффективно, поскольку вычисление хеш-функций может занять продолжительный период времени. Поскольку хеш-функции детерминированы (один и тот же ввод функции всегда приводит к одному и тому же хеш-коду), то если бы несколько пользователей использовали один и тот же пароль, их хеш-код был бы идентичным. Если значительное количество людей сопоставлено с одним и тем же хеш-кодом, то это может показать, что входные данные для хеш-кода представляют собой часто используемый набор символов, и позволяет злоумышленнику значительно сузить количество входных данных, которые можно использовать для взлома с помощью грубой силы. Таким образом, применение небольших паролей, даже при обеспечении хранения их в виде хеш-кодов, делает возможным для взлома баз данных использование заранее составленных словарей.

Кроме того, с помощью атаки по «радужной таблице» злоумышленник может применить набор предварительно вычисленных цепочек хеш-кодов, чтобы найти входные данные украденных паролей, соответствующих данным кодам. «Радужная таблица» – массив данных, содержащий заранее

посчитанные хеш-функции для определенного количества распространенных паролей. Вероятность успеха атаки с применением «радужной таблицы» можно уменьшить, усилив хеширование с помощью процедуры, которая добавляет уникальные случайные данные к каждому входу в момент их сохранения. Эта практика известна как добавление «соли» при хешировании, и она производит «соленые» хеш-коды.

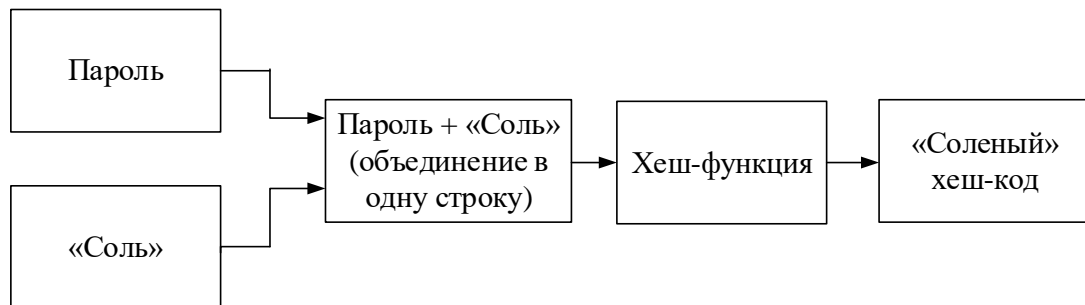


Рис. 2 Процедура усиления хеширования, путем добавления «соли»

С «солью» хеш-код зависит не только от значения пароля, но и от уникальных случайных данных. Радужная таблица строится для набора «несоленых» хеш-кодов. Если каждый предварительный образ включает уникальное значение, которое носит псевдослучайный характер, радужная таблица бесполезна. Однако возникает необходимость сохранения в тайне значения добавленной «соли». Если злоумышленнику оно станет известно, он сможет пересчитать «радужную таблицу» и получить возможность определить исходный пароль. Таким образом, система аутентификации должна предусмотреть порядок применения и хранения не только хеш-кодов паролей, но и уникальных «солей».

Поскольку хеш-функции применимы к входным данным любого размера, но вычисляют хеш-коды только фиксированной длины, можно сделать вывод, что набор всех возможных входных данных бесконечен, а набор всех возможных выходных данных конечен. В результате существуют отображения несколько разных входных данных в один и тот же хеш-код.

Следовательно, невозможно однозначно определить соответствие пароля и его хеш-кода. Этот эффект называется коллизией, и он нежелательный.

Криптографическая коллизия возникает, когда два уникальных пароля при хешировании отображаются в один и тот же хеш-код. Коллизионная атака – это попытка найти два прообраза, которые производят один и тот же хеш-код. Злоумышленник может использовать эту коллизию, чтобы обмануть системы, которые полагаются на хешированные значения, путем подделки действительного хеш-кода с использованием неверных или вредоносных данных.

Защиту от использования коллизии обеспечивает добавление «соли» к хешируемому данным.

Дополнительным требованием к криптографическим хеш-функциям, используемым для хеширования паролей, является относительная продолжительность вычислений, потому что быстро вычисляемый алгоритм может сделать атаки с использованием грубой силы более осуществимыми. Этого можно добиться за счет увеличения количества внутренних итераций или роста объема памяти для вычислений.

Таким образом, для обеспечения защиты хранящихся паролей аутентификации целесообразно использовать криптографические функции хеширования. Добавление при хешировании уникальных псевдослучайных данных («соли») уменьшает вероятности коллизий и подбора пароля злоумышленником на основе заранее подготовленных словарей.

Список литературы

1. Афанасьев, Алексей Алексеевич Аутентификация. Теория и практика обеспечения безопасного доступа к информационным ресурсам. Учебное пособие для вузов. 2017. - 270 с.
2. Kioon M. C, Wang Z. S, Shubra D.D быстрое программное шифрование и хеш-функции 2013. С. 262-277