Лабораторная работа №7

Тема «Исследование криптографических хеш-функций»

ВЫПОЛНИЛ

Олишкевич Игорь Русланович

Группа 25-ПО

Задание: изучить криптографические хеш-функции; привести примеры, их использование.

Криптографическая хеш-функция — всякая хеш-функция, являющаяся криптостойкой, то есть удовлетворяющая ряду требований, специфичных для криптографических приложений.

Для того, чтобы хеш-функция H считалась криптографически стойкой, она должна удовлетворять трём основным требованиям, на которых основано большинство применений хеш-функций в криптографии:

* необратимость или стойкость к восстановлению прообраза: для заданного значения хеш-функции m не должен быть вычислен блок данных X, для которого {\displaystyle {H(X)=m}.}H(x) = m;
* стойкость к коллизиям первого рода или восстановлению вторых прообразов: для заданного сообщения M должно быть вычислительно невозможно подобрать другое сообщение N, для которого {\displaystyle H(N)=H(M).}H(N) = H(M);
* стойкость к коллизиям второго рода: должно быть вычислительно невозможно подобрать пару сообщений {\displaystyle (M,M'),}(M, M’) имеющих одинаковый хеш.

Данные требования не являются независимыми:

* обратимая функция нестойка к коллизиям первого и второго рода;
* функция, нестойкая к коллизиям первого рода, нестойка к коллизиям второго рода; обратное неверно.

Для криптографических хеш-функций также важно, чтобы при малейшем изменении аргумента значение функции сильно изменялось (лавинный эффект). В частности, значение хеша не должно давать утечки информации даже об отдельных битах аргумента. Это требование является залогом криптостойкости алгоритмов хеширования пользовательских паролей для получения ключей.

Применения.

Электронная подпись (ЭП). Текст, зашифрованный секретным ключом, объединяется с исходным сообщением. Тогда проверка подписи — расшифрование открытым ключом; если получившийся текст аналогичен исходному тексту — подписанный текст верен (не менялся).

Использование хеш-функции позволяет оптимизировать данный алгоритм. Производится шифрование не самого сообщения, а значения хеш-функции, взятой от сообщения. Данный метод обеспечивает следующие преимущества:

* понижение вычислительной сложности. Как правило, документ значительно больше его хеша;
* повышение криптостойкости. Криптоаналитик не может, используя открытый ключ, подобрать подпись под сообщение, а только под его хеш;
* обеспечение совместимости. Большинство алгоритмов оперирует со строками бит данных, но некоторые используют другие представления. Хеш-функцию можно использовать для преобразования произвольного входного текста в подходящий формат.

### Проверка парольной фразы. В большинстве случаев парольные фразы не хранятся на целевых объектах, хранятся лишь их хеш-значения. Хранить парольные фразы нецелесообразно, так как в случае несанкционированного доступа к файлу с фразами злоумышленник узнает все парольные фразы и сразу сможет ими воспользоваться, а при хранении хеш-значений он узнает лишь хеш-значения, которые не обратимы в исходные данные, в данном случае в парольную фразу. В ходе процедуры аутентификации вычисляется хеш-значение введённой парольной фразы, и сравнивается с сохранённым.

Примером в данном случае могут служить ОС GNU/Linux и Microsoft Windows XP. В них хранятся лишь хеш-значения парольных фраз из учётных записей пользователей.

Данная система подразумевает передачу сообщения по защищенному каналу, то есть каналу, из которого криптоаналитику невозможно перехватить сообщения или послать своё. Иначе он может перехватить парольную фразу, и использовать её для дальнейшей нелегальной аутентификации. Защищаться от подобных атак можно при помощи метода «вызов-ответ».

Пусть некий клиент, с именем *name*, производит аутентификацию по парольной фразе, *pass*, на некоем сервере. На сервере хранится значение хеш-функции*H*(*pass*, *R*2), где *R*2 — псевдослучайное, заранее выбранное число. Клиент посылает запрос (*name*, *R*1), где *R*1 — псевдослучайное, каждый раз новое число. В ответ сервер посылает значение *R*2. Клиент вычисляет значение хеш-функции *H*(*R*1, *H*(*pass*, *R*2)) и посылает его на сервер. Сервер также вычисляет значение*H*(*R*1, *H*(*pass*, *R*2)) и сверяет его с полученным. Если значения совпадают — аутентификация верна.

В такой ситуации пароль не хранится открыто на сервере и, даже перехватив все сообщения между клиентом и сервером, криптоаналитик не может восстановить пароль, а передаваемое хеш-значение каждый раз разное.