**7 Лекция. Основы криптографии. Контроль целостности данных. Хеш-функции. Имитовставка. ЭЦП**

**Контроль целостности данных.**

[http://ru.wikipedia.org/wiki/Целостность\_информации](http://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BB%D0%BE%D1%81%D1%82%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%B8)

Целостности данных - при котором отсутствует любое ее изменение либо изменение осуществляется только преднамеренно субъектами, имеющими на него право.

Методы контроля целостности данных:

1. Полная копия данных
2. Контрольная сумма
3. Хеш
4. Имитовставка
5. ЭЦП

**Полная копия данных.**

Создаются полные копии данных и потом сверяются.

Преимущества:

* простота реализации
* полный контроль данных (до бита)

Недостатки:

* большой объем
* копии можно подменить
* копиями можно воспользоваться (например: если данные - пароль)

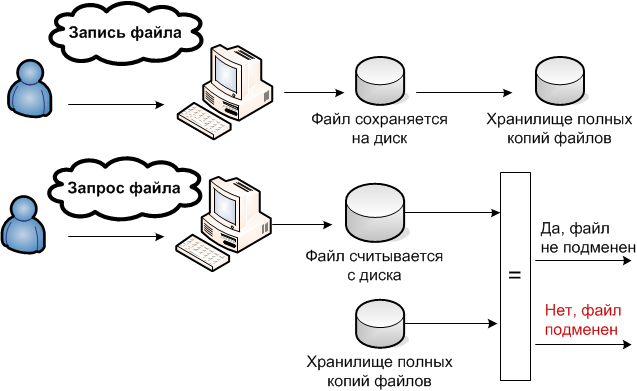


Рис. Контроль целостности с помощью полной копии данных

Применение:

1. контроль целостности файлов

**Контрольная сумма.**

[http://ru.wikipedia.org/wiki/Контрольная\_сумма](http://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D1%83%D0%BC%D0%BC%D0%B0)

Контрольная сумма - значение, рассчитанное по входным данным с помощью определённого алгоритма.

Преимущества:

* высокая скорость вычисления
* малый размер
* стандартный размер

Недостатки:

* можно подменить
* для одного значения существует множество исходных данных
* можно подобрать исходные данные к значению за приемлемое время (например: получить пароль)

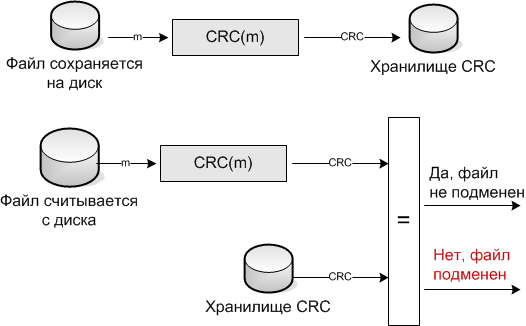


Рис. Контроль целостности с помощью контрольной суммы

Примеры контрольных сумм: CRC8, CRC16, CRC32

Пример вычисления:

исходный текст: Контроль целостности данных

|  |  |
| --- | --- |
| **crc32 (длина 32 бита)** | b4feb5a5 |

Применение:

1. контроль целостности файлов
2. контроль передаваемых данных по каналам связи
3. контроль целостности при считывании данных (например: c HDD)

**Хеш.**

[http://ru.wikipedia.org/wiki/Хеширование](http://ru.wikipedia.org/wiki/%D0%A5%D0%B5%D1%88%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)

Хеш (хэш, криптографический хеш) - значение, рассчитанное по входным данным с помощью криптографического алгоритма.

Преимущества:

* малый размер
* стандартный размер
* нельзя подобрать исходные данные к значению за приемлемое время (например: получить пароль)

Недостатки:

* низкая скорость вычисления (сопоставима с шифрованием)
* можно подменить
* для одного значения существует множество исходных данных



Рис. Основная задача хеш функций

 Вычисляют хеш шифрованием данных блочным алгоритмом в режимах CBC, но со стандартным (известным) ключом. Хешем является последний шифрованный блок.

**ГОСТ Р 34.11-94**

[http://ru.wikipedia.org/wiki/ГОСТ\_Р\_34.11-94](http://ru.wikipedia.org/wiki/%D0%93%D0%9E%D0%A1%D0%A2_%D0%A0_34.11-94)

Входное сообщение *M* разделяется на блоки *mn*,*mn*? 1,*mn*? 2,...,*m*1 по 256 бит. В случае если размер последнего блока *mn* меньше 256 бит, то к нему приписываются слева нули для достижения заданной длины блока.

Каждый блок сообщения подаётся на шаговую функцию для вычисления промежуточного значения хеш-функции*Hout*=f(*Hin*, *mi*)  где *Hout*, *Hin*, *mi* — блоки длины 256 бит.

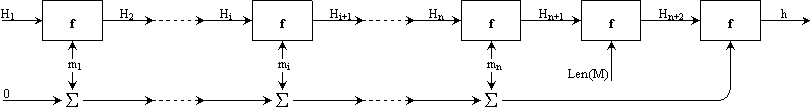


Рис. Вычисление хеш по ГОСТ Р 34.11-94 (сравните с CBC)

h — значение хеш-функции сообщения M

Len(M) - длина сообщения

Ключи для f-функции генерятся стандартным образом, что бы все пользователи могли вычислить одинаковые хеш для одних и тех же файлов.

Пример вычисления:

исходный текст: Контроль целостности данных

|  |  |
| --- | --- |
| **md2 (длина 128 бит)** | 7e347a5f3a3d8c6837d7accbed3e0b1e |
| **md4 (длина 128 бит)** | e37e491b9b4ea133df9964b25d7c7cd9 |
| **md5 (длина 128 бит)** | 8ab8a5cf989e220ff8d39be415b903d5 |
| **sha1 (длина 160 бит)** | 63cdc45bd8a857007d0be8c435e1e547653482d3 |
| **sha224 (длина 224 бит)** | 670c98e5b7ce7bd2c7efb98b6ed448e0a6f3247e3fdeb678dde61bce |
| **sha256 (длина 256 бит)** | 1a97bcc0fbcf6722a8aac7b73d12700c0022778904790ce9eee7656f12d95dc2 |
| **sha384 (длина 384 бит)** | b87b59cad0db141378d8ff04e46d00dd137113391d2a7009d640dd018680c459 310bfbdfef6b3258aeaa4b8146105698 |
| **sha512 (длина 512 бит)** | 3f7b9d8b24fb1d764026fe2b0f72ad62d65fd1c5d77a18784108a69e8ad172c2 e6583989439aea658a2111525897d43af0f6c50cf299b360c21b3e02e8706f4e |
| **ГОСТ Р 34.11-94 (длина 256 бит)** | d38e4f1bc5d03601486f4aca83fed00c82e1a36fdac27806cce4b9464af1e9f9 |

Применение:

1. контроль целостности файлов
2. контроль передаваемых данных по каналам связи
3. контроль целостности при считывании данных (например: c HDD)
4. хеши паролей
5. аутентификация (CRAM-MD5, DIGEST-MD5 и т.д.)

**Имитовставка (MAC, message authentication code — код аутентичности сообщения)**

[http://ru.wikipedia.org/wiki/Имитовставка](http://ru.wikipedia.org/wiki/%D0%98%D0%BC%D0%B8%D1%82%D0%BE%D0%B2%D1%81%D1%82%D0%B0%D0%B2%D0%BA%D0%B0)

Имитовставка -  значение, рассчитанное по входным данным с помощью криптографического алгоритма с использованием секретного элемента (ключа), известного только отправителю и получателю.

Преимущества:

* малый размер
* стандартный размер
* нельзя подобрать исходные данные к значению за приемлемое время (например: получить пароль)
* нельзя подменить без секретного элемента (ключа)

Недостатки:

* низкая скорость вычисления (сопоставима с шифрованием)
* для одного значения существует множество исходных данных
* секретный ключ известен как минимум двоим

 Вычисляют имитовставку шифрованием данных блочным алгоритмом в режимах CBC. Имитовставкой является последний шифрованный блок.

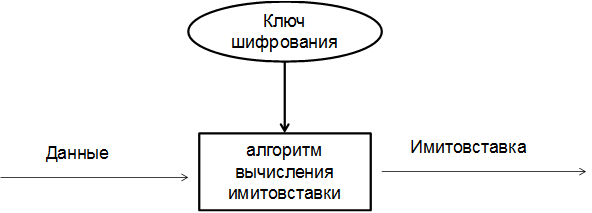
****

Рис. Вычисление имитовставки

**Имитовставка по ГОСТ 28147-89**

Длина имитовставки от 1 до 32 бит.

Открытый текст *TO* разбивается на блоки длиной 64 бита. Последний блок в случае необходимости дополняется нулями.

T_O=T_O^{(1)}\,T_O^{(2)}\,\ldots\,T_O^{(N)}

Первый блок T_O^{(1)} шифруется, что и сообщение, но с применением 16 циклов вместо 32. Результат по битам по модулю 2 складывается с вторым блоком T_O^{(2)} и так же шифруется. Результат складывается с третьим блоком... и так далее.

I=E'_k(T_O^{(N)} \oplus E'_k(T_O^{(N-1)} \oplus E'_k(\,\ldots\,E'_k(T_O^{(2)} \oplus E'_k(T_O^{(1)}))\,\ldots\,))

Первые 32 бита получившегося блока составляют имитовставку. Спецификация шифра предусматривает использование в качестве имитовставки и меньшее количество бит по желанию, но не большее.



Рис. Проблема имитовставки

Получатель должен знать ключ, и этот ключ позволяет ему генерировать сообщения с тем же значением имитовставки, что и у присланного сообщения, таким образом, имитовставка на основе симметричного шифра не дает знания — отправитель или получатель сформировал эту имитовставку.

Отсюда следует, что имитовставка на основе симметричного шифра не может заменять собой **электронную подпись!**

Применение:

1. контроль целостности файлов
2. контроль передаваемых данных по каналам связи
3. аутентификации источника данных (не во всех случаях)

Обычные  хэш-алгоритмы использовать для вычисления имитовставки нельзя (MD5 и т.д.) т.к. отсутствует секретный ключ. Поэтому создан **HMAC**.

**HMAC (Hash-based Message Authentication Code)** - механизм включения секретного ключа в существующие хэш-алгоритмы.

<http://ru.wikipedia.org/wiki/HMAC>

**ЭЦП.**

[http://ru.wikipedia.org/wiki/ЭЦП](http://ru.wikipedia.org/wiki/%D0%AD%D0%A6%D0%9F)

Электронная цифровая подпись - зашифрованное значение вычисленного хеша по входным данным.

Преимущества:

* малый размер
* стандартный размер
* нельзя подобрать исходные данные к значению за приемлемое время (например: получить пароль)
* нельзя подменить без секретного элемента (ключа)
* секретный ключ известен одному

Недостатки:

* низкая скорость вычисления (сопоставима с шифрованием)
* для одного значения существует множество исходных данных

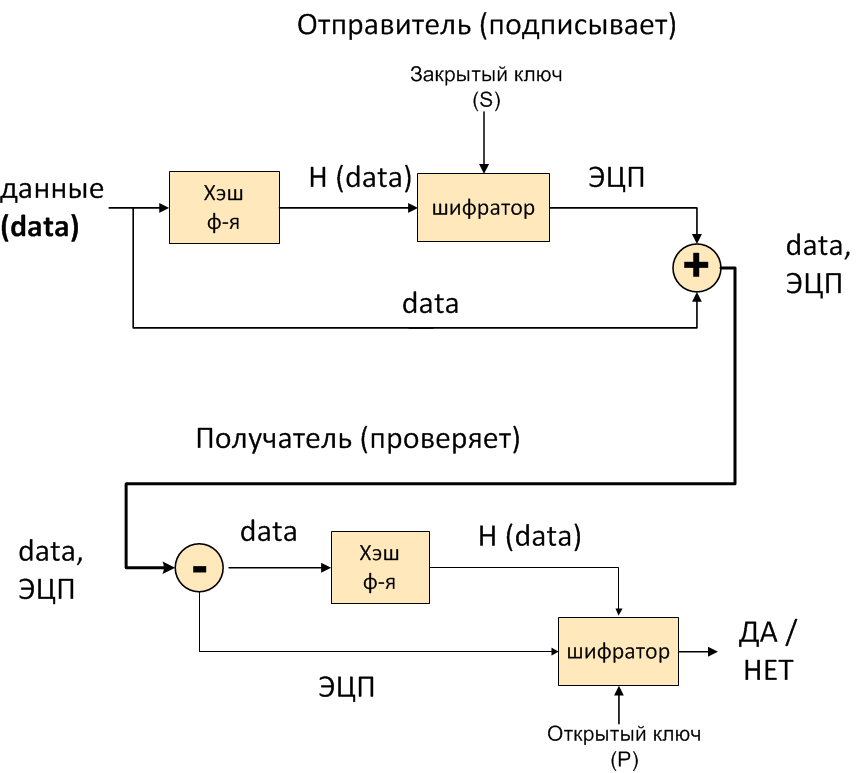


Рис. Создание и проверка ЭЦП

Алгоритм:

1. вычисляется хеш
2. шифруется хеш

Применение:

1. контроль целостности файлов
2. контроль передаваемых данных по каналам связи
3. аутентификации источника данных (кто создал подпись)