Криптография, Лекция № 9

10 ноября 2014 г.

1 Протоколы электронной подписи

Эта задача похожа на задачу идентификации. Здесь сторона должна доказать, что она действительно передавала некоторое сообщение. Как достигается надежность обычной подписи? Вообще, не очень надежными основаниями, просто как-то так получается, что у всех людей разные подписи. По хорошему, электронная подпись должна зависеть от текста документа. Было понятие ГПСЧ, и генератор был хорошим, если его действительно нельзя отличить от последовательности случайных чисел. Сперва создавался протокол со случайными числами, которые заменялись потом на псевдо случайные, и если протокол был надежным а генератор хорошим, то и новый протокол хороший. Помимо псевдо случайных чисел бывают и другие абстракции: случайные функции, надежный канал.

S - подписывающий, V - проверяющий, m - подписываемое сообщение. S изготавливает подпись s и отправляет V. V принимет или не принимает подпись, возвращая 0 или 1. Два варианта:

1. Схема с закрытым ключом:
   Имеется генератор ключей K, и ключ d известен и S и V

2. Схема с открытым ключом:
   d известен S, e известен V

Требования:

1. Корректность:

$$\forall m \; Pr\{V(d, m, S(d, m)) = 1\} \simeq 1$$

2. Надёжность (в самом сильном смысле, против атаки с выбором сообщения):
   Противник (последовательность схем полиномиального размера):
   выбирает m_1, получает $s_1 = S(d, m_1)$
   ...
   выбирает m_k, получает $s_k = S(d, m_k)$
   Выбирает $m_{k+1}, sk + 1$
   Атака успешна, если $V(d, m_{k+1}, s_{k+1}) = 1$ и $m_{k+1} \notin \{m_1, \ldots, m_k\}$.

1.1 Конструкция с закрытым ключом

Идеальный случай, это когда закрытый ключ является оракульным доступом к случайной функции f. Тогда подпись будет значением этой функции на сообщении: $s = f(m)$. У проверяющего оракульный доступ к той же самой f. Почему это надежно? Потому, что если случайная функция хорошая, то значения в k точках ничего не схажут противнику о значении в $k+1$ точке.

В реяльном случае закрытый ключ - номер псевдо-случайной функции. Здесь уже $s = f_d(m)$. Корректность тривиальна. Для проверки надежности нужно сравнить с предыдущим случаем, если в реальном случае атака будет успешной, то из этой атаки можно изготовить отличитель для псевдослучайной функции.

Remark 1.
При помощи протокола электронной подписи можно устроить протокол идентификации. Сперва сервер посылает сообщение, клиент его подписывает. Сервер проверяет ее с помощью ключа. Да и вообще, если существует надежный протокол электронной подписи, то существует односторонняя функция.

1.2 Конструкция с открытым ключом

1.2.1 Одноразовая подпись 1 бита

Есть односторонняя функция f. (x_0, x_1) - закрытый ключ, а $(y_0, y_1) = (f(x_0), f(x_1))$ - закрытый ключ. $S(d, \sigma) = x_\sigma$, $V(e, \sigma, s) = 1$, если $f(s) = y_\sigma$. Корректность очевидна. Надежность: (x_0, x_1) почти наверное различные, и противнику нужно зная значение в одной точке восстановить значение в другой. Понятно, что это почти то же самое, что и обратить. Здесь важно, что ключ выбирается случайно (с помощью генаратора).

1.2.2 Одноразовая подпись 1 сообщения фиксированной длины

Здесь можно сделать всё почти также.

$$d = (x_0^1, x_1^1, \ldots, x_0^k, x_1^k)$$

$$y = (y_0^1, y_1^1, \ldots, y_0^k, y_1^k)$$

$$S(d, \sigma_1, \ldots, \sigma_k) = (x_{\sigma_1}^1, \ldots, x_{\sigma_k}^k)$$

Одноразовая, так как из подписей под $00\ldots0$ и $11\ldots1$ можно получить любую подпись соответсвующей комбинацией. Надежность выполнена примерно по тем же соображениям.

Хочется избавиться от фиксированной длины, и если нужно отправить много сообщений, схлопывать их в одно. В модели, когда верификатор хранит некоторую информацию из предыдущих сессий. Это уже так просто сделать не получается и водятся хэш-функции.