Ментальный покер

Рассмотрим задачу честной карточной игры по сети. Основной задачей в такой игре будет честное распределение карт (раздача) по игрокам без возможности мошенничества со стороны любого из них. Задача будет поставлена следующим образом.

Пусть есть колода K из $|K|$ карт (для удобства будем считать, что все карты натуральные попарно различные числа, такое сопоставление можно легко ввести для любой реальной колоды из 52 карт, если, например, сопоставить картам числа в диапазоне от 2 до 53). В игре участвуют $n \leq |K|$ игроков. Необходимо раздать каждому игроку по $m_i, 1 \leq i \leq n, \sum_{i=1}^{n} m_i \leq |K|$ карт таким образом, чтобы никто из игроков не мог знать карты другого. Более того, даже если предположить предварительный сговор между любым подмножеством игроков, они не должны иметь возможности повлиять на честность раздачи карт.

До начала игры и раздачи карт формируется большое простое число P (для большей надёжности это должно быть число Софи Жермен, то есть $P = 2Q + 1$, где P, Q - простые числа). Каждый игрок формирует два секретных ключа $1 < C_i, D_i < P - 1, 1 \leq i \leq n$, такие, что $C_i D_i \ mod\ (P - 1) = 1$. Кроме того формируется колода $K = \{k_1, k_2, \dots, k_{|K|} | k_i \in \{2, P - 1\}, k_i \neq k_j, i \neq j\}$. После получения исходных параметров можно приступить к раздаче карт, которая выполняется по следующему алгоритму:

1. Шифрование колоды:
   Каждый игрок по очереди шифрует все карты колоды при помощи своего ключа C и перемешивает их, после чего передаёт колоду следующему игроку. Опишем указанные действия для i-го игрока:
   $K_{1..i} = shuffle\left(K_{1..i-1}^{C_i}\ mod\ P\right)$
   Здесь функция $shuffle()$ случайным образом перемешивает карты в колоде (переставляет элементы массива), а $K_{1..i-1}$ – это колода, зашифрованная и перемешанная по очереди всеми игроками от 1 до $i - 1$. Так как колода K является множеством, то распишем подробнее:
   $K^C mod\ P = \{k_1^C mod\ P, \dots, k_{|K|}^C mod\ P\}$
2. После первого шага получается колода, зашифрованная и перемешанная всеми игроками по очереди. Теперь любой игрок (например, тот, который шифровал последним) может раздать всем остальным игрокам карты в любом порядке. Так как на каждом шаге

осуществлялось случайное перемешивание, то карты находятся уже в произвольном порядке и можно раздавать их подряд, в последовательном порядке, начиная с первой карты.

3. Теперь у каждого игрока на руках находятся $m_i, 1 \leq i \leq n$ карт, зашифрованных всеми игроками. Для того, чтобы узнать свои карты i-й игрок должен передать свои карты по очереди каждому игроку, чтобы те их расшифровали при помощи своих ключей D. Собственный ключ игрок использует в последнюю очередь. Опишем процесс для одной карты k i-го игрока при помощи псевдокода:

```
FOR j=1..n DO
  IF j ≠ i THEN
   k = k^Dj mod P
  FI
OD
```

$k = k^{D_i} mod\ P$

Для того, чтобы обезопасить протокол от попытки подмены карты каким-либо игроком в процессе игры, все пункты алгоритма записываются в лог, а после окончания игры каждый игрок публикует свои ключи C, D. Таким образом, любой игрок, имея записи процесса раздачи и все ключи, может проверить, что все данные корректны и ни на каком этапе не произошло подмены. Естественно, это лишь один из вариантов защиты, другие можно найти в научной литературе. Тем не менее, этот метод не нужно применять при выполнении лабораторной работы, так как он относится к протоколу лишь косвенно.

Алгоритм слепой подписи

Данный алгоритм применяется во многих криптографических протоколах, его использование для системы «электронные деньги» рассмотрено в работе [1], в данном же разделе мы рассмотрим его на примере протокола «анонимное голосование».

Опишем задачу обеспечения анонимного голосования по сети следующим образом. Пусть есть некоторый сервер, который выдаёт участникам голосования подписанные бюллетени, каждому участнику выдаётся только один бюллетень. Необходимо организовать голосование таким образом, чтобы ни сервер, ни кто-либо из участников не узнали как проголосовал конкретный участник, при этом необходимо обеспечить легитимность голосования, то есть, чтобы каждый участник мог получить только один бюллетень, который невозможно подделать и/или использовать повторно.

Опишем алгоритм голосования. Пусть есть сервер, который выдаёт бюллетени. В начале голосования сервер генерирует общие данные согласно системе RSA. Формируются два секретных больших неравных простых числа $P, Q, P \neq Q$. Вычисляется известное всем участникам число $N = PQ$. Кроме того сервер формирует секретный ключ C и открытый ключ D, связанные соотношением $CD \ mod \ \phi(N) = 1$, где $\phi(N) = (P-1)(Q-1)$. Очевидно, что в данном случае оба ключа должны быть числами, взаимнопростыми с $\phi(N)$. Отметим, что размеры чисел P, Q составляют 1024 бит, а числа N соответственно 2048 бит. Таким образом, всем участникам голосования известны числа D, N и только серверу известно число C. Теперь, когда участник, например, Алиса, хочет проголосовать, этот процесс будет представлять из себя следующее:

1) *Алиса формирует некоторое случайное число rnd длиной 512 бит. Далее формируется число v, которое хранит в закодированном виде результат голосования Алисы. Например, если голосование состояло только из одного вопроса с ответом Да или Нет, то закодировать результат можно как 1 или 0. Кроме того, в число добавляется некоторая служебная информация (данные о голосовании, адрес сервера и т.д.). В итоге получим некоторое число n по формуле*

$$n = rnd|v,$$

где $|$ означает конкатенацию. Необходимо учесть, что это число не должно превышать 1024 бита, таким образом под служебную информацию и результат голосования выделено 512 бит, что вполне достаточно для хранения всей информации.

2) *Алиса формирует некоторое случайное число r взаимнопростое с N. Очевидно, что вероятность случайным образом сгенерировать число, для которого не выполняется это условие крайне мала, она сопоставима с вероятностью случайным образом обнаружить делители числа N.*
3) *Алиса вычисляет криптографическую хэш-функцию от числа n. Необходимость использования этой функции мы обоснуем ниже.*
$$h = SHA3(n), h < N$$
4) *Теперь Алисе необходимо вычислить число $\bar{h} = h\, r^d (mod\ N)$. Полученное число она отправляет по защищённому верифицированному каналу связи на сервер.*
5) *Сервер помечает, что выдал бюллетень Алисе (это необходимо, чтобы Алиса не могла проголосовать дважды), и вычисляет число $\bar{s} = \bar{h}^C mod\ N$. Полученное значение отсылается обратно Алисе.*
6) *Алиса вычисляет подпись своего бюллетеня по формуле*
$$s = \bar{s}\, r^{-1} mod\ N,$$
где r^{-1} - это инверсия числа r по модулю N.
7) *Подписанный бюллетень $< n, s >$ отправляется на сервер голосования по анонимному каналу связи. Сервер проверяет корректность бюллетеня, проверив равенство $SHA3(n) = s^d mod\ N$, и, в случае корректности, учитывает голос и заносит информацию о бюллетене в открытую базу данных. Открытой она должна быть для того, чтобы любой участник мог убедиться в честности прошедшего голосования.*

Во-первых, докажем, что подпись s действительно является корректной подписью для бюллетеня n.

$$s = \bar{s}\, r^{-1} mod\ N = \bar{h}^C\ r^{-1}\ mod\ N = (h\, r^D)^C\ r^{-1}\ mod\ N$$
$$= h^C r^{DC} r^{-1}\ mod\ N$$

Здесь $r^{DC}\ mod\ N = r^1$, так как по следствию из теоремы Эйлера $r^{DC} mod\ N = r^{DC\ mod\ \phi(N)}\ mod\ N$, а $DC\ mod\ \phi(N) = 1$. Таким образом получаем

$$h^C r^{DC} r^{-1}\ mod\ N = h^C r^1 r^{-1}\ mod\ N = h^C\ mod\ N$$

Далее покажем, почему необходимо использовать криптографическую хэш-функцию, а не какую-либо произвольную $f(x) < N$. Допустим, мы использовали некоторую функцию $f(x) < N$. Предположим, что эта функция обладает мультипликативностью, т.е. $f(ab) = f(a)f(b)$. Тогда

существует возможность, имея в наличии два разных бюллетеня, получить корректный третий, используя следующие формулы:

$$n_3 = n_1 n_2$$

$$s_3 = s_1 s_2$$

Полученный таким образом бюллетень пройдет проверку на корректность, хоть и получен «нелегально». Чтобы избежать этого, необходимо выбрать такую функцию $f(x)$, которая не будет обладать ни мультипликативностью, ни аддитивностью, ни какими-либо другими свойствами, позволяющими использовать её для подделки бюллетеня, и нам известно, что криптографические хэш-функции обладают всеми необходимыми свойствами. Поэтому целесообразнее всего использовать именно их. В протоколе мы выбрали SHA3 как наиболее безопасную и защищённую от «взлома» на данный момент функцию.

Слепой подписью этот алгоритм называется потому, что сервер в момент подписывания не знает и не может узнать реального значения n и даже его значения хэш-функции, что также немаловажно, ведь криптографическая хэш-функция обладает таким свойством, что в процессе голосования крайне маловероятна ситуация, когда два разных бюллетеня будут иметь одинаковое значение хэш-функции. Это означает, что сервер мог бы, запомнив значение хэша бюллетеня конкретного человека, однозначно идентифицировать его в процессе передачи по анонимному каналу связи и связать выбранное решение с конкретным человеком, лишив тем самым процесс анонимности.

Отдельно отметим использованный в протоколе термин «анонимный канал связи», подходов для организации подобного канала существует большое число и это тема для отдельного научного исследования, поэтому, в рамках протокола, мы не будем затрагивать эту тему. Примером такого канала передачи данных является VPN, хоть он и обладает рядом недостатков с точки зрения защищённости.

В рамках выполнения задания на лабораторную работу не выдвигается требований организовывать анонимный канал связи и какую-либо работу по сети, достаточно реализовать в рамках локальной машины любой из протоколов, использующих алгоритм «слепой» подписи.

Заключение

В результате освоения данного курса обучающийся получает общие сведения об основах криптографических методах защиты информации. В ходе выполнения лабораторных работ приобретаются навыки практического использования наиболее используемых криптографических протоколов и формируются навыки построения аналогичных протоколов для более конкретных задач. В дальнейшем полученные навыки позволяют понимать более сложные методы защиты информации, сформированные на основе эллиптических кривых и применяющиеся в современных информационных система. Знание устройства подобных методов помогает, в том числе, лучше встраивать их в существующие информационные системы и обнаруживать уязвимости в уже реализованных решениях.