Лабораторная работа №4

Тема: Схема Эль-Гамаля и хэш-функция SHA-2.

Введение.

Формально, как и для любой схемы электронно-цифровой подписи (ЭЦП), для работы схемы Эль-Гамаля необходимо реализовать 3 основных алгоритма: генерацию ключей, функцию подписи и функцию проверки подписи. С учетом того, что в лабораторной не требуется реализовывать генерацию простых чисел, алгоритмы ЭЦП имеют следующий вид:

Генерация ключей (Gen)

Вход: q – простое число.
Шаги:

1. Выбрать такое четное число R, что $R < 4 (q + 1)$.
2. Вычислить число $p = qR + 1$.
3. Если $2^{qR} \neq 1 \pmod p$ или $2^R = 1 \pmod p$, то возвратиться к шагу 1, иначе p – простое.
4. Случайным образом выбрать x из Z_p и вычислить $g = x^R \pmod p$.
5. Если $g = 1$, то возвратиться к шагу 4, иначе искомое g найдено.
6. Случайным образом выбрать личный ключ d из Z_q.
7. Вычислить открытый ключ $e = g^d \pmod p$.

Выход: (p, q, g) – параметры ЭЦП; e – открытый ключ; d – личный ключ.

Функция подписи (Sign)

Вход: (p, q, g) – параметры ЭЦП; d – личный ключ; M – подписываемое сообщение (в виде строки текста произвольной длины).
Шаги:

1. Вычислить хэш-значение m от сообщения M: $m = h(M)$ ($h()$ – хэш-функция).
2. Случайным образом выбрать одноразовый личный ключ k из $Z_q / \{0\}$.
3. Вычислить $r = g^k \pmod p$.
4. Вычислить $s = k^{-1}(m – dr) \pmod q$.

Выход: (r, s) – подпись.

Функция проверки подписи (Verify)

Вход: (p, q, g) – параметры ЭЦП; e – открытый ключ; M – подписываемое сообщение (в виде строки текста произвольной длины; (r, s) – подпись.
Шаги:

1. Если r не лежит в $Z_p / \{0\}$ или s не лежит в Z_q, то вернуть *FALSE*.
2. Вычислить хэш-значение m от сообщения M: $m = h(M)$ ($h()$ – хэш-функция).
3. Если $e^r r^s = g^m \pmod p$, то вернуть *TRUE*, иначе вернуть *FALSE*.

Выход: TRUE, если подпись корректна; *FALSE,* если подпись некорректна.

Замечания:

1) Для возведения в степень рекомендуется использовать алгоритм быстрого возведения в степень. Нельзя использовать готовую реализацию этого алгоритма, если она есть в языке программирования, на котором вы выполняете лабораторную работу.
2) Найти k^{-1} можно с помощью расширенного алгоритма Евклида. Нельзя использовать готовую реализацию этого алгоритма, если она есть в языке программирования, на котором вы выполняете лабораторную работу.
3) В качестве хэш-функции используйте SHA-256. Ее реализация, написанная на C, находится в папке с условием [в файлах sha256.h и sha256.c осуществлена реализация хэш-функции SHA-256, а в файле main.c приводится пример вычисления хэш-значения]. Вы также можете найти готовую / написать свою реализацию SHA-256 (+5 баллов в случае написания своей реализации) на любом другом допустимом языке программирования.

Условие лабораторной работы:

1) Для выполнения всех заданий необходимо использовать "длинную" арифметику. Разрешается использовать любую готовую библиотеку или написать свою. При этом учитывайте, что допустимые языки программирования {C/C++, C#, Java, Python, R}.
2) Ваша программа должна принимать на вход указание, какую операцию необходимо выполнить: *Gen*, *Sign*, *Verify или Exit*. В зависимости от операции программа должна считывать (из консоли, файла и т.п.) необходимые данные, выполнять шаги выбранного алгоритма и возвращать (в консоль, файл и т.п.) результат выполнения соответствующей операции. При вызове операции *Exit* программа не считывает входных данных, а просто завершает работу.
3) Программа должна иметь проверку корректности входных данных (d лежит в Z_q для *Sign*; e лежит в Z_p для *Verify*).
4) Программа должна предлагать пользователю выполнить очередную операцию до тех пор, пока пользователь не выберет вариант выхода из программы.

*Отчет (*условия по вариантам смотрите на последней странице)*:*

1) В условиях своего варианта для заданного числа q необходимо выполнить алгоритм *Gen*.
2) Для параметров, полученных с помощью *Gen,* и сообщения вида «I, Ivan Ivanov, love CM», где «Ivan Ivanov» – ваше имя и фамилия соответственно, выполните алгоритм *Sign*.
3) Проверьте результаты подписи из 2) пункта, выполнив алгоритм *Verify.*
4) Результаты шагов 1 – 3 необходимо записать в текстовый файл, назвав его «Report.(txt, word или pdf)».

Бонусное задание.

Реализовать один из следующих методов нахождения дискретного логарифма:

1) Алгоритм Гельфонда — Шенкса;
2) Алгоритм Полига — Хеллмана;
3) ро-метод Полларда;
4) Алгоритм Адлемана;
5) Алгоритм COS.

Алгоритм должен использовать "длинную" арифметику. Успешная реализация любого из алгоритмов дает возможность НЕ выполнять 0.5 (половину) любой из 5 (пяти) основных лабораторных работ. Можно реализовать несколько алгоритмов: все бонусы суммируются (т.е., если вы реализуете 2 (два) алгоритма, то сможете НЕ выполнять 1 (одну) из 5 (пяти) основных лабораторных работ).

Порядок сдачи лабораторной работы:

1) Вам необходимо создать архив формата «.zip», название которого должно иметь вид «5_5_Ivanov.zip», где «5» – № вашей группы, а «Ivanov» – ваша фамилия латинскими буквами. В архиве должен быть исходный код вашей реализации (НЕ весь проект целиком, а только файлы с исходным кодом (.c, .cpp, .java и т.п.)) и файл «Report.(txt, word, pdf)»;

Таблица №1: Простое число q для алгоритма *Gen*

№ вар.	q
1	14099022013266194209435383627010667598306571562674759214141398399293699005933 9
2	13991793545108659509569826248818079398737601881926764004770441179265642540092 7
3	17265392903404808495067442989919361941805747377870333124750410231252433667914 3
4	21069745503233768494312119403986359118600471346357079626876568970922310829241 9
5	22552311892210046576975814841302542180722095679997507196967515632709802067301 1
6	18447319569573647197141635064265720591165938491490752652111157497401258593653 9
7	18549733733103235274678986183405423081500378350701553055548238663709596852287 9
8	20454950543416969470535961395384050794580878245871402645521693563920612968146 7
9	12360495129345426296854670004539614751079022918649208477980994891769339420400 3
10	17870629584560251426189850151267837983010093324556518465173052047174047855110 7
11	19100919954384631190943055184730968424804807128058028713839618553044439482028 7
12	16939452785205105633389383057861166747872359079474700802730497204738140783878 3
13	18338982757505633631616687845173983072256762199375324949076475898897669209499 1
14	22862002392126719373092815388674379339632445234057713898797276023641820844384 7
15	22945021850950794293124581321239337929448820972668807566389843061297746749869 9