МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики физики и информационных систем
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

02.03.03 Математическое обеспечение и администрирование информационных систем
(код и наименование направления подготовки, специальности)

Технология программирования
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Реализация и исследование криптографического алгоритма на эллиптических кривых»

Студент Д.О.Ильичев
(И.О. Фамилия) (личная подпись)

Руководитель доцент, к.ф.–м.н. ,Г.А.Тырыгина
(ученая степень, звание, И.О. Фамилия)

Консультант (ы) О.А.Головач
(И.О. Фамилия)

Тольятти 2020

Аннотация

Название бакалаврской работы - Реализация и исследование криптографического алгоритма на эллиптических кривых.

Целью бакалаврской работы является реализация криптографического алгоритма шифрования данных на основе эллиптических кривых.

Объект исследования – алгоритмы защиты информации от несанкционированного доступа.

Предметом исследования являются криптографические алгоритмы шифрования на основе эллиптических кривых.

В первой главе описывается теория эллиптических кривых, их математические свойства, их плюсы и минусы. Так же были описаны свойства их умножения, свойства сложения, групповой закон и какие кривые стоит выбирать при решении определенных задач.

Во второй главе описываются алгоритмы и протоколы на эллиптических кривых с суперсингулярными кривыми и с несуперсингулярными кривыми, а также как на точках распределяется информация и какие точки лучше для этого выбирать.

В третьей главе показана реализация криптографического алгоритма на эллиптических кривых. В качестве такого алгоритма был выбран алгоритм Диффи-Хеллмана. Была приведена его числовая и программная реализация. Было проведено исследование этого алгоритма на работу с большими значениями ключей и с нулевыми значениями.

Бакалаврская работа выполнена на 39 страницах, состоит из введения, трёх глав, включающих 38 изображений, 46 формул, заключения, списка используемой литературы, включающего 26 источников, из них 9 на иностранном языке.

Abstract

The title of the graduation thesis is «Implementation and research of cryptographic algorithm on elliptic curves»

The aim of the work is to implement a cryptographic algorithm for encrypting data based on elliptic curves.

The object of the thesis is algorithms for protecting information from unauthorized access.

The subject of the senior thesis is cryptographic encryption algorithms based on elliptic curves.

The first part of the thesis describes the theory of elliptic curves, their mathematical properties, their advantages and disadvantages. The properties of their multiplication, the properties of addition, the group law, and which curves should be chosen when solving certain problems were also described.

The second part of the thesis describes algorithms and protocols on elliptic curves with supersingular curves and with non-supersingular curves, as well as how points are distributed information and which points are better for this

The third part of the thesis shows the implementation of the cryptographic algorithm on elliptic curves. Diffie-Hellman algorithm was chosen as the algorithm. Its numerical and software implementation was given. A study of this algorithm was carried out to work with large values of keys and with zero values.

The graduation work consists of an explanatory note on 39 pages, introduction, including 38 figures, 46 formulas, the list of 26 references including 9 foreign sources.

Оглавление

Введение

Целью бакалаврской работы является реализация криптографического алгоритма шифрования данных на основе эллиптических кривых.

Объект исследования – алгоритмы защиты информации от несанкционированного доступа.

Предметом исследования являются криптографические алгоритмы шифрования на основе эллиптических кривых.

Жизнеспособность общества все больше зависит от развития информационной среды. Информация является важной частью в функционировании государственных и общественных институтов, в жизни каждого человека. В нынешних условиях сформировался новый вид трудовой деятельности, связанный с хранением, распространением и получением информации.

Информатизация ведет к созданию общего мирового информационного пространства, к унификации информационных технологий различных стран мира.

Новые технологии открывают большие возможности. Вместе с тем катастрофически возрастает цена потерь в случае нештатной ситуации или снижения надежности систем обработки и передачи информации. Можно наблюдать, что в настоящее время все более зримо проявляется зависимость экономики от надежности систем защиты информации.

В нынешних условиях защита информации становится все более актуальной и одновременно все более сложной проблемой. Это обусловлено как массовым применением методов автоматизированной обработки данных, так и широким распространением методов и средств несанкционированного доступа к информации. Поэтому важную роль в организации противодействия потенциальным угрозам занимает подход,

при котором средства защиты информации используются комплексно, каждое в соответствии со своим предназначением.

Внедрение и активное использование современных информационных технологий существенно повысили уязвимость информации, циркулирующей в современных информационно-телекоммуникационных системах.

Несанкционированное искажение, копирование, уничтожение информации в настоящее время затрагивает не только процессы, относящиеся к сфере государственного управления, но и интересы физических лиц.

С каждым днём увеличивается объем информации и увеличивается её спрос, а значит, и растёт её ценность, поэтому возрастают требования к её защите.

Актуальность темы бакалаврской работы объясняется необходимостью противостоять современным способам несанкционированного доступа к информации.

Глава 1. Информация об эллиптической криптографии

1.1 Теория об эллиптических кривых

Нил Коблиц и Виктор Миллер в 1985 году предложили использовать эллиптические кривые в криптосистемах. В 1998 году в стандартах США для решения задачи связанной с цифровой подписью использовались эллиптические кривые, а в России такой же стандарт, был принят в 2001. Важным плюсом криптосистем на эллиптических кривых является более высокая стойкость при равной трудоемкости по сравнению с обычными криптосистемами. Это из-за того, для того чтобы вычислить обратную функцию для эллиптических кривых существуют лишь алгоритмы с экспоненциальным ростом трудоемкости, в то время как для обычных систем известны лишь субэкспоненциальные методы. Криптостойкость, которая достигается в алгоритме RSA с использованием модуля в 128-байт, на эллиптических кривых используется с размером модуля в 20 байт.

1.2 Математические свойства эллиптических кривых

Эллиптические кривые – это кривые которые задаются кубическим уравнением третьей степени, с использованием переменных из поля k и точкой на бесконечной прямой [1].

Уравнение этой кривой можно записать следующим образом

$$y^2 = x^3 + ax + b, \tag{1.1}$$

где a, b – действительные числа.

Так как график кривой параллелен оси абсцисс, чтобы найти точки, являющиеся корнями, нужно решить уравнение третьей степени.

$$x^3 + ax + b = 0, \tag{1.2}$$

Здесь можно использовать формулу Кардано. Дискриминант вычисляется по формуле 1.3

$$D = \left(\frac{a}{3}\right)^3 + \left(\frac{b}{2}\right)^2 \tag{1.3}$$

При дискриминанте меньше нуля, уравнение (1.2) имеет три разных решения a, b, z; при дискриминанте равном нулю, уравнение (1.2) имеет три корня, a, b, c, два из которых одинаковые, при дискриминанте больше нуля, уравнение (1.2) имеет одно решение a и два комплексно сопряженных. Графики по результатам вычислений представлены на рисунках 1.1-1.3.

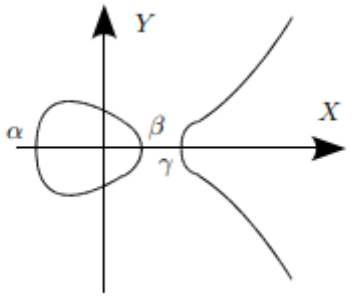

Рисунок 1.1 – Кривая с D <0

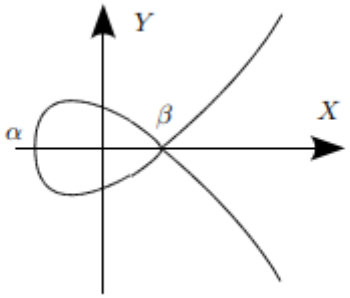

Рисунок 1.2 – Кривая с D =0

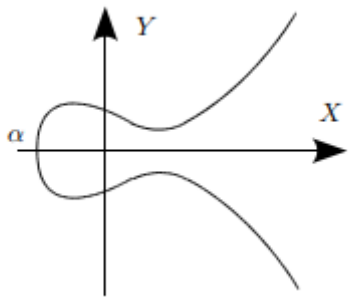

Рисунок 1.3 – Кривая с D >0

На рисунке 1.4 показаны примеры эллиптических кривых.

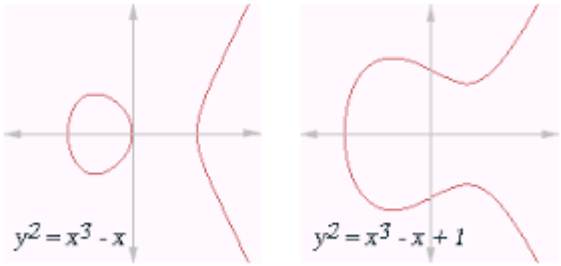

Рисунок 1.4 – Эллиптические кривые

Всего существует два вида эллиптических кривых сингулярные и несингулярные. Несингулярные кривые – это эллиптические кривые дискриминант которых не равен нулю

$$\Delta = (4a^3) + 27b^2 \neq 0\,, \tag{1.4}$$

где a и b коэффициенты уравнения (1.1).

Сингулярные кривые редко используются, т.к. есть риск значительно снизить криптостойкость алгоритмов и протоколов.

Ниже на рисунке 1.2 продемонстрирован пример несингулярной кривой

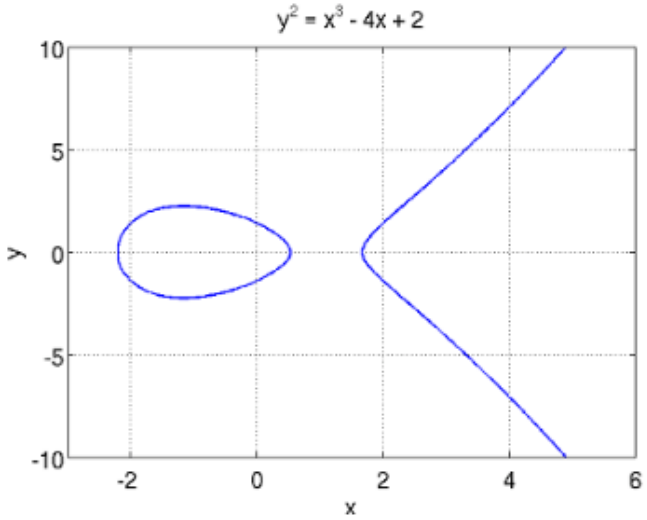

Рисунок 1.2 – несингулярная кривая

Точки на эллиптической кривой можно подвергать таким операциям как сложение. Для этого воспользуемся формулой:

$$P + Q = -R\,, \tag{1.5}$$

где P, Q, R – это точки этой кривой. График сложения точек показан на рисунке 1.3

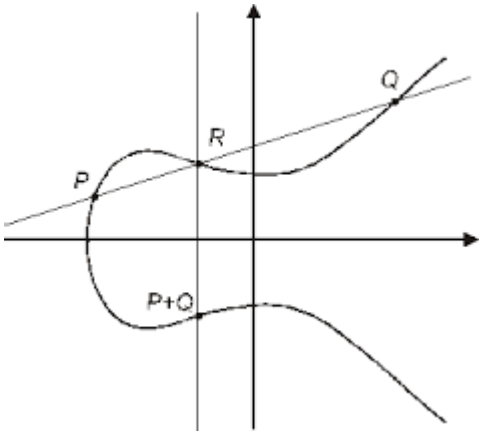

Рисунок 1.3 – Сложение точек

Предположим что $(x_P y_P)$ координаты точки P, а $(x_Q y_Q)$ координаты точки Q. Вычислим коэффициент ε

$$\alpha = \frac{y_Q - y_P}{x_Q - x_P} \tag{1.6}$$

тогда координаты точки (P+Q) равны

$$x_{(P+Q)} = \varepsilon^2 - x_P - x_Q \tag{1.7}$$

$$y_{(P+Q)} = -y_P + \varepsilon(x_P - x_R) \tag{1.8}$$

1.3 Выбор параметров для кривой

Выбираем наиболее ценные переменные для того чтобы избавиться от попыток взлома и перехвата информации, для определенных кривых. Этот выбор является наиболее хорошим в плане лучшей криптостойкости. Для использования этой стратегии применяются особенные методы, но в результате этого метода кривые отличаются от небольшой пары кривых и создают угрозы на присутствие необычных явлений, из-за которых существует вероятность изобретения алгоритма для их хакинга. Далее описывается алгоритм нужных действий для получения правильной кривой.

- Сперва нужно выбрать число P. У этой точки порядок величины зависит от количества точек на прямой. Тогда число p измеряемое в битах, считается по формуле $t = [\log p] + 1$, и она должна быть такой чтобы алгоритм нахождения логарифмов на кривой был невозможным. Переменная t =16 байт на данном этапе слишком мала, она легко вычисляется если

приложить несколько вычислительных усилий. Но размер в 20 байт невозможна для современных криптосистем.

- Следующим шагом определяются два произвольных числа a и b $\neq 0$ (mod p) и $(4a^3) + 27b^2 \neq 0\ (mod\ p)$. При определении точек можно заметить, что коэффициент b не используется. Поэтому для продуктивных вычислений лучше выбирать случайно только величину b, а величина a выбирается определенно.

- Далее находим количество точек, которые находятся на прямой. Это переменная n. Для лучших вычислений делитель n должен быть равен числу q. При n делящимся на группу подмножеств, алгоритм Диффи – Хеллмана легко справляется с вычислением логарифма на кривой с помощью этих маленьких подмножеств. При слишком большой затрате времени можно принять n=hq, где h небольшое число.

- Далее проверяем условие неравенства $(p^k - 1) mod\ q \neq 0$ для всех k на промежутке от 0 до 32. Эта проверка может предотвратить MOV атаку, а также позволяет избежать использование суперсингулярных кривых.

- Проверяем выполняется ли равенство $q \neq p$. Кривые с q = p называются аномальными и для них предложены эффективные методы вычисления логарифмов.

- Если все условия выполняются, то мы получаем пригодную для вычислений кривую. Теперь нам известны переменные p, a, b, число точек прямой n и размер подмножества точек q. Далее нужно вычислить G, которая является генератором этого подмножества. Если размер подмножества равен числу точек, то любая точка является генератором. Если размер подмножества меньше числа точек, выбирается случайная точка G', до того момента пока не получим $G = \left[{}^{n}/_{q}\right]G' \neq O$. Для вычисления случайной точки на кривой, берем случайное число x которое меньшее p, вычисляем $e = (x^3 + ax + b)\ mod\ p$ и извлекаем квадратный корень $y = \sqrt{e}\ mod\ p$.

Если есть решение этого уравнения, то вычисляем точку (x, y), если решение иррациональное, то выбираем другое число x.

1.4 Свойства сложения абелевой группы

Чтобы множество G было группой, необходимо записать сложение таким образом, чтобы оно отвечало четырем основным свойствам:

- если a и b входят в множество G, то a+b входит в это множество G. Это свойство называется замыканием;
- (a + b) + c = a + (b + c), это свойство ассоциативности;
- если существует единичный элемент 0, при котором a + 0 = 0 + a = a;
- если у каждого элемента есть противоположная величина, то есть для каждого a существует такое b, что (a + b) = 0
- и последнее свойство — это коммутативность (a + b = b + a)

Если выше перечисленные свойства выполняются, то группа называется абелевой.

1.5 Групповой закон для эллиптических кривых

Эллиптические кривые можно определить в группу:

- элементы этой группы являются точками эллиптической кривой;
- бесконечно удаленная точка 0 – это единичный элемент;
- обратная величина точки P – это точка, симметричная относительно оси x;
- Сумма трех ненулевых точек, которые лежат на одной прямой

$$P + Q + R = 0 \qquad (1.9)$$

1.6 Геометрическое сложение эллиптических кривых

Формула (1.9) позволяет нам описать геометрический способ сложения двух точек P и Q. Через две эти точки мы проводим прямую, которая пересечет кривую в некоторой точке R, при условии, что P, Q, R находятся на

одной прямой. Если мы возьмем обратную величину точки R, то это и будет сумма точек P + Q.

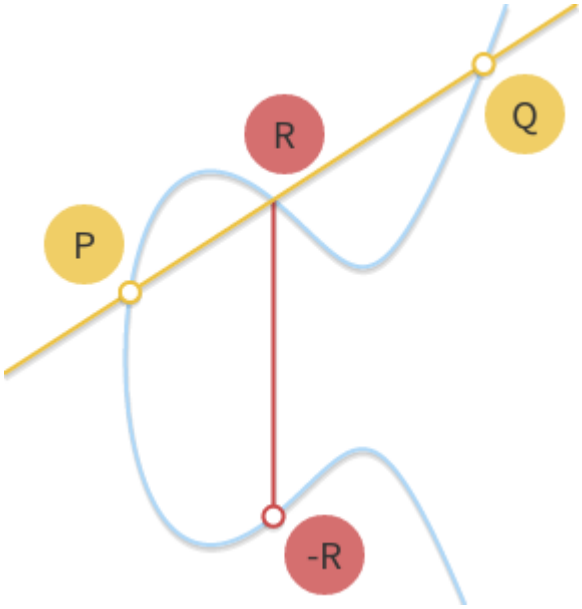

Рисунок 1.4 – График суммы точек P + Q

В геометрическом сложении есть несколько особенностей:

- если P или Q равны нулю, то мы не сможем провести через них прямую, т.к. 0 не находится на плоскости xy. Но из-за того, что мы обозначили 0 как единичный элемент, то для любой P и Q будет выполняться условие

(P + 0 = P) (0 + Q = Q).

- если P = -Q, то прямая, которую мы построим через эти две точки будет вертикальна и не будет пересекать кривую в точке. Если P является обратной величиной Q, то P + Q = 0.
- если P = Q, тогда через точку проходит бесконечное количество прямых. Представим, что есть некая точка Q’ которая не равна P. Далее нужно сделать так чтобы точка Q’ стремилась к точке P. Благодаря этому прямая проходящая через P и Q’ будет касательной к кривой.

1.7 Эллиптическая криптография

Эллиптическая криптография – наука изучающая эллиптические кривые над проектными плоскостями [1].

Круг использования эллиптических кривых довольно большой. Они используются в алгоритмах цифровой подписи, асимметричного шифрования и обмена ключами.

Эллиптические кривые часто применяются в алгоритмах шифрования данных:

алгоритм обмена ключами. Два абонента делятся секретным ключом, который будет одинаковым для обоих. Используют они канал для передачи сообщений, который не защищен от прослушивания, но защищён от фальсификации. Ключ, который они получают, они используют для обмена сообщениями;

алгоритм цифровой подписи используется для авторизации пользователей, отправки электронных версий документов и других данных. Если в электронной версии документа стоит подтвержденная цифровая подпись, значит она имеет юридическую силу.

алгоритм с открытым ключом. Это алгоритм зашифровывания информации. В этом алгоритме доступный ключ отправляется по открытому каналу и применяют его для проверки электронной подписи и шифрования сообщения. Для расшифровки сообщения и создания электронной подписи используется закрытый ключ.

1.8 Плюсы и минусы эллиптической криптографии

Основные плюсы:

размер ключа меньше чем в асимметричной криптографии [2];

эллиптические алгоритмы работают быстрее классических.

благодаря тому, что длина ключа относительно небольшая, а скорость работы алгоритма асимметричной криптографии на эллиптических кривых достаточно высокая, эти алгоритмы могут использоваться в смарт-картах и других устройствах которые имеют ограниченные вычислительные ресурсы [2];

дискретное логарифмирование на эллиптических кривых не имеют субэкспоненциальных алгоритмов для решения [2];

Основные минусы эллиптической криптографии:

эллиптическая криптография будет терять свою актуальность при появлении алгоритмов с использованием логарифмирования.

в эллиптической криптографии нужно учитывать на какой кривой происходят вычисления и какой ключ используется. Если их не учитывать появляется вероятность угрозы алгоритмов путем появления уязвимостей, а также вероятность ошибки [2].

В данной главе были продемонстрированы математические свойства эллиптических кривых, приведены наиболее популярные системы, в которых используется эллиптическая криптография, выявлены ее плюсы и минусы.

Благодаря вышеперечисленной информации следует вывод, что полностью перейти на эллиптическую криптографию возможно, но этот переход не является необходимым.

Глава 2. Эллиптические кривые в криптографии

2.1 Алгоритмы на эллиптических кривых

2.1.1 Эллиптические кривые над полем GF(2^n)

В основном криптография использует кривые над нечетным полем, но часто используются кривые над полем GF(2^n). Ниже на рисунке 2.1 показан алгоритм сложения и удвоения эллиптических кривых

ВХОД: Коэффициент a эллиптической кривой
точки $P_1 = (x_1, y_1)$ (или $P_1 = \mathcal{O}$) и $P_2 = (x_2, y_2)$ (или $P_2 = \mathcal{O}$).
ВЫХОД: $P = P_1 + P_2$.

Если $P_1 = \mathcal{O}$, то вернуть $P = P_2$,
если $P_2 = \mathcal{O}$, то вернуть $P = P_1$,
если $P_2 = -P_1$, то вернуть $P = \mathcal{O}$,
если $x_1 \neq x_2$, то
  вычислить $\lambda = \frac{y_1+y_2}{x_1+x_2}$, $x_3 = \lambda^2 - x_1 - x_2$,
  вернуть $P = (x_3, -y_1 + \lambda(x_1 - x_3))$,
иначе принять $x = x_1$, $y = y_1$,
  вычислить $\lambda = \frac{3x^2+a}{2y}$, $x_3 = \lambda^2 - 2x$,
  вернуть $(x_3, -y + \lambda(x - x_3))$.

Рисунок 2.1 - Алгоритм сложения и удвоения точек на кривых

Суперсингулярные кривые. Основной отличительной особенностью этих кривых в сравнении с несуперсингулярными, это легкость вычисления порядка кривой. Чтобы использовать эти кривые не обязательно использовать сложный алгоритм вычисления порядка кривой.

Суперсингулярную кривую можно описать следующим уравнением

$$Y^2 + a_3 Y = X^3 + a_4 X + a_6, \quad a_i \in \mathrm{GF}(2^n) \qquad a_3 \neq 0$$

(2.1)

С помощью замены переменных $X = a_3 X$, $Y = a_3 Y$ уравнение можно привести к виду

$$Y^2 + Y = a'_2 X^3 + a'_4 X + a'_6, \quad a'_i \in \mathrm{GF}(2^n) \qquad a'_2 \neq 0$$

(2.2)

Если из этого уравнения извлекается корень третьей степени, то заменив переменную $X = (a'_2)^{-1/3} X$ уравнение переписывается следующим образом

$$Y^2 + Y = X^3 + a_4X + a_6,\ a_i \in \mathrm{GF}(2^n)$$

(2.3)

Несуперсингулярные кривые. Уравнение таких кривых представляется в следующем виде

$$Y^2 + XY = X^3 + a_2X^2 + a_6$$

(2.4)

С заменой переменных $Y = Y + kX$, где $k^2 + k + a_2 = a'_2$ приводит эту кривую к виду

$$Y^2 + XY = X^3 + a'_2X^2 + a_6$$

(2.5)

2.1.2 Скалярное умножение на суперсингулярных кривых

Скалярное умножение точки - это основа в арифметике эллиптических кривых. Лучше всего использовать балансные системы счисления.

Алгоритмической особенностью этих кривых является более быстрая работа с удвоением точки чем с умножением. Оценка сложности складывается за счет операций сложения точек. В аддитивных цепочках можно ускорить вычисления, когда точка P не задается заранее. Если же P известна заранее придется использовать другие алгоритмы, однако в них использование суперсингулярных кривых не принесет особой пользы.

Проективные координаты. При увеличении количества умножений в четыре с половиной раза можно получить более быстрые вычисления, не смотря на то что время преобразования будет в пять раз больше. После этого можно переходить к проективным координатам.

Подстановкой $X = \frac{X}{Z}\ Y = \frac{Y}{Z}$ в уравнения эллиптических кривых Ep1, Ep2, Ep3 получим однородные уравнения относительно X, Y

$$Ep_1\colon\ Y^2Z + YY^2 = X^3$$

(2.6)

$$Ep_2\colon\ Y^2Z + YZ^2 = X^3 + XZ^2$$

(2.7)

$$Ep_3\colon\ Y^2Z + YZ^2 = X^3 + XZ^2 + Z^3 \qquad (2.8)$$

Проективные координаты (X, Y, Y) и (X, Y, Z) эквивалентны при условии что существует такое число t множества K при котором выполняется условие $X' = tX, Y' = tY, Z' = tZ$. Класс равного соотношения обозначим (X: Y: Z). Все классы этих трех координат и является проективными координатами.

Эллиптическая кривая теперь является множеством проективных точек на плоскости. Точка которая лежит на прямой будет единственной нулевой точкой с координатами x = 0, y = 1, z = 0.

Пусть $P = (X_1 : Y_1 : 1) \in E_i$ и $Q = (X_2 : Y_2 : Z_2) \in E_i$. Представим, что $P, Q \neq Q$ и $P \neq Q$. Для точки $R = P + Q, R = (X'_3 : Y'_3 : 1)$ необходимо применить формулу сложения, после применения которых, учитывая что $X'_2 = X_2/Z_2,\ Y'_2 = Y_2/Z_2$ получим

$$X'_3 = \frac{a^2}{b^2} + X_1 + \frac{X_2}{Zz_2},\ Y'_3 = 1 + Y_1 + \frac{a}{b}\left(\frac{a^2}{b^2} + \frac{X_2}{Zz_2}\right)$$

(2.9)

Где $a = Y_1Z_2 + Y_2,\ b = X_1Z_2 + X_2$. Из этого следует

$$Z_3 = b^3Z_2,\ X_3 = X'_3Z_3,\ Y_3 = Y'_3Z_3,$$

находим $R = (X_3 : Y_3 : Z_3)$, где

$$X_3 = a^2bZ_2 + b^4,\ Y_3 = (1 + Y_1)Z_3 + a^3Z_2 + ab^2X_2,\ Z_3 = b^3Z_2.$$

(2.10)

2.1.3 Скалярное умножение на несуперсингулярных кривых

Заменив алгоритмы для сложения и удвоения точек можно использовать предыдущие алгоритмы на суперсингулярных кривых для несуперсингулярных.

Метод Монтгомери для несуперсингулярных кривых. Криптографы Лопес и Дахаб создатели этого метода. Для рассмотрения используется кривая $Y^2 + XY = X^3 + aX^2 + b$. Вычисления производятся после каждого шага $Q_2 = Q_2 + Q_1,\ Q_1 = 2Q_1$ где $Q_i = Q_j + P, i \neq j$. По формулам сложения

и удвоения точек используемые для несуперсингулярных кривых, находим обновленные координаты точек $Q_1 = (x_1 y_1)$ и $Q_2 = (x_2 y_2)$

$$x_1^2 + \frac{b}{x_1^2}. \tag{2.11}$$

$$\left(\frac{y_1+y_2}{x_1+x_2}\right)^2 + \frac{y_1+y_2}{x_1+x_2} + x_1 + x_2 + a \tag{2.12}$$

При x_1=0 результатом удвоения точек получается бесконечно удаленная точка.

Метод Монтгомери в проективных координатах. Этот метод использует координаты точек P_i представляя их X_i/Z_i. Вычисление точки 2Pi происходит по формуле

$$\frac{X}{Z} = \frac{X_i^4 + bZ_i^4}{Z_i^2 X_i^2} = \left(\frac{X_i}{Z_i}\right)^2 + b\left(\frac{Z_i}{X_i}\right)^2 = x_i^2 + \frac{b}{x_i^2} \tag{2.13}$$

где $X = X_i^4 + bZ_i^4, \quad Z = Z_i^2 X_i^2.$ Нахождение координат точки $P_1 + P_2$ происходит по формуле.

$$\frac{X_3}{Z_3} = x + \frac{X_1 Z_2 X_2 Z_1}{(X_1 Z_2 + X_2 Z_1)^2} = x + \frac{\frac{X_1 X_2}{Z_1 Z_2}}{\left(\frac{X_1}{Z_1} + \frac{X_2}{Z_2}\right)^2} = x + \frac{x_1 x_2}{(x_1 + x_2)^2}$$

(2.14)

где $X_3 = xZ_3 + (X_1 Z_2)(X_2 Z_1), \quad Z_3 = (X_1 Z_2 + X_2 Z_1)^2$

Метод Лопеса-Дохаба использования проективных координат. Чтобы уменьшить количество операций для проективных координат нашлось решение приравнивать координатным переменным (X:Y:Z) аффинную точку $\left(\frac{X}{Z}, \frac{Y}{Z^2}\right)$. Тогда в проективных координатах кривая $Y^2 + XY = X^3 + aX^2 + b$ представляется в виде

$$Y^2 + XYZ = X^3 Z + aX^2 Z^2 + bZ^4 \tag{2.15}$$

Смешанное проективно-аффинское сложение точек выполняется с помощью девяти умножений, девяти сложений и пяти возведений в квадрат.

Как и в суперсингулярных кривых, этот метод можно использовать с любой вариацией метода аддитивных цепочек при которых строятся линейные цепочки. P сразу вычисляется в аффинных координатах, потому что в конце придется переводить в афиинные координты, затрачивая по два деления на точку.

При k содержащей u(k) удвоений и s(k) сложений, трудность вычислений равна $L(k) = 4u(k) + 8s(k)$, а в алгоритме Коблица (если b=1) $L(k) = 3u(k) + 8s(k)$.

2.2 Протоколы на эллиптических кривых

2.2.1 Выбор точки и размещение данных

Криптография использует точки эллиптической кривой для особенных точек. Криптографическая задача решает выбирать случайную точку или брать точку на координатах которой находятся данные. Значение x можно получить как часть бинарного вектора, значение y координаты определяется с помощью уравнения кривой. Это связано с тем что нужно решить квадратное уравнение.

Сперва нужно выбрать точку на эллиптической кривой. Для суперсингулярных кривых алгоритм показан на рисунке 2.2.

ВХОД: Уравнение $Y^2 = f(X)$ суперсингулярной эллиптической кривой над $GF(2^n)$.
ВЫХОД: Точка (x, y), удовлетворяющая уравнению кривой.
0. Инициализация:
в соответствии с (2.2), (2.3) вычислить матрицу T
1. Выбрать случайно ненулевой элемент x поля $GF(2^n)$.
2. Вычислить $\sigma = f(x)$.
3. Вычислить след $\mathrm{tr}(\sigma)$ элемента σ: $t = \mathrm{tr}(\sigma) = \sum_{i=0}^{n-1} \sigma^{2^i}$.
4. Если $t = 1$, то перейти к п. 1.
5. Решить уравнение (2.1).
6. Вернуть (x, y).

Рисунок 2.2 – Алгоритм суперсингулярных кривых

Алгоритм для несуперсингулярной кривой показан на рисунке 2.3

ВХОД: Уравнение $Y^2 + XY = f(x)$, где $f(x)$ — значение правой части уравнения при $X = x$.
ВЫХОД: Точка (x, y)
0. Инициализациия: вычислить матрицу $T = T_1 + T_2$.
1. Выбрать случайно ненулевой элемент x поля $GF(2^n)$.
2. Вычислить $\sigma = f(x) \cdot x^{-2}$.
3. Вычислить след элемента σ: $t = \mathrm{tr}(\sigma) = \sum_{i=0}^{n-1} \sigma^{2^i}$.
4. Если $t = 1$, то перейти к п. 1.
5. Решить уравнение $y \cdot T = \sigma$. Если решения нет, перейти к п.1.
6. Вычислить $y = yx$.
7. Вернуть (x, y).

Рисунок 2.2 - Алгоритм несуперсингулярных кривых

2.2.2 Распределение ключей

Для примера используем несколько протоколов. Криптостойкость данных алгоритмов складывается за счет сложности вычисления дискретного логарифма и алгоритма Диффи-Хеллмана. Проблемы вычисления логарифма появляется когда известна группа циклов, известна $y = g^x$ определенного элемента и необходимо определить показатель x.

Для групп точек эллиптической кривой проблема с использованием дискретного логарифма является определением числа k по известной точке P и точке Q, которая равна произведению этих чисел. Для эллиптических кривых не придумано полиномиальных алгоритмов решения, но существует решение для суперсингулярных кривых, который переводит эту проблему к DLP.

Классической проблемой Диффи-Хеллмана является использование его алгоритма в мультипликативной группе поля. Суть этой проблемы в вычислении a^{xy} по элементам a, a^xи a^y.

Протокол Диффи-Хеллмана. Вместо ключа классической криптосистемы можно взять случайную точку (x, y),

Рассмотрим пример. E – это эллиптическая кривая и P – это точка этой кривой. Абонент A выбирает случайное число k_A, вычисляет координаты точки $k_A P$ и отправляет абоненту B. Абонент B делает тоже самое и отправляет абоненту A. Точка $P = k_B k_A P$ является общим ключом. Абонент

А вычисляет эту точку, умножая свой ключ на ключ полученный от абонента В, а В вычисляет, умножая свой ключ на полученный ключ абонента А. Благодаря тому что группа точек является абелевой, Результат не зависит от того в каком порядке буду происходить вычисления, тогда абоненты получат одинаковую точку (x,y) и могут использовать координату x как ключ одинаковой криптосистемы. Проблемой для сторонних людей будет в вычислении секретной точки, так как они не будут знать секретные ключи абонентов. В этом и заключается проблема Диффи-Хеллмана.

Протокол Месси-Омуры. Это протокол, позволяющий передавать информацию по открытым каналам связи без передачи информации.

Мультипликативный вариант. Протокол был описан к мультипликативной группе Z_p^*, где p это простое число. Абонент А прикрепляет к ящику свой ключ и отсылает его абоненту В. Абонент В получает посылку и прикрепляет к ящику свой ключ и отсылает его абоненту А. После этого абонент А открепляет свой ключ от ящика и снова отсылает его абоненту В. Абонент В, получив посылку открепляет свой ключ от ящика. Иногда вместо механических замков можно использовать электронные. Для этого необходимо выбрать большое простое число p. Абоненты А и В выбирают случайные числа e_A и e_Bи вычисляют числа d_A и d_B, обратные по модулю $\varphi(p) = p - 1$ (φ-функция Эйлера)

$$e_A * d_A \equiv 1\big(mod\varphi(p)\big), e_B * d_B \equiv 1\big(mod\varphi(p)\big)$$

(2.16)

Числа $(e_A d_A)$, $(e_B d_B)$ это секретные ключи. Для m при 0<m<p справедливо равенство $m^{e_A * d_A} = m \, mod \, p$, так как $m^{e_A * d_A} = m^{j*\varphi(p)+1} = m^{j*\varphi(p)} * m = m$. Первый множитель равен 1 по теореме Эйлера.

Аддитивный вариант. На эллиптической кривой находятся постоянные переменные, переменные перемножаются с точками кривой, по модулю ее порядка.

Применив алгоритм преобразования вычислим

$$d \equiv e^{-1}(mod\ N). \quad (2.17)$$

Благодаря теореме сравнимости по модулю получаем $e * d = jN + 1$. Для всех точек этой кривой с порядком N выполняется свойство $(e * d)P = P$.

Можно вычислить Q=eP и R=dQ, используя e и d из формулы (2.17) и точку эллиптической кривой.

В алгоритме, который использует алгебру эллиптических кривых, параметрами для системы используют уравнение кривой и поле, над которым она построена. По этому протоколу абонент А выбирает число e_A и вычисляет по формуле (2.15) число d_A. Абонент В выбирает свое число e_Bи вычисляет число d_B.

Абонент А зашифровывает информацию в точку М и умножает на свое число e_A и получает точку $P_1 = e_A M$. Далее он отсылает ее В. Абонент В производит вычисления точки $P_2 = e_B P_1$. Далее А расшифровывает свой замок и вычисляет точку $P_3 = d_A P_2$ и пересылает ее абоненту В. Абоненту В нужно снять шифровку с сообщения. Умножая точку А на свой ключ и найти $M = d_B P_3$. Сообщение m может выступать в качестве ключа симметричной криптосистемы.

Протокол Менезеса-Кью-Венстоуна (MQV – протокол). Протоколы, которые рассматривались ранее имеют проблему, что посторонний человек сможет перехватить информацией при перехвате сообщение. Например, в протоколе Диффи-Хеллмана злоумышленник, назовем его абонентом С, сможет завладеть информацией, которую абоненту В посылает абонент А и заполучить закрытый ключ абонента В. Абонент В и абонент С владеют одинаковыми закрытыми ключами. Далее С сделает тоже самое для абонента А, после чего А и С имеют общий закрытый ключ. При осторожных действиях третьего абонента А и В не будут знать, что у них есть посредник. Для того чтобы этого не случилось необходима авторизация кратковременных ключей, для этого используются закрытые ключи. В алгоритме одноразовые ключи привязываются к многоразовым из-за этого

злоумышленник не может перехватить информацию или сообщение. При использовании одноразового ключа становится невозможно использовать многоразовые ключи при передаче сообщения.

$$kP = (k \; mod \; N)P \tag{2.18}$$

Константу k можно вычислять в модульной арифметике кольца Z_N так и Z.

Поэтому существует два варианта решения этой задачи.

Первый вариант реализует модульную арифметику целых чисел. Второй вариант реализует циклические свойства подгруппы точек эллиптической кривой.

Если реализовывается модульная арифметика, то с числами можно выполнить некоторые математические действия, например, сложение или умножение, а если реализовывать арифметику эллиптической кривой, то числа можно умножать на точки эллиптической кривой.

Оба варианта предопределяют что абоненты знают точку P кривой порядка N, с которой происходят все вычисления.

Открытые ключи абонента B известны абоненту A

$$Q_B = d_B P = (a_B b_B), \quad R_B = k_B P = (\, x_B y_B) \tag{2.19}$$

Открытые ключи абонента A известны абоненту B

$$Q_A = d_A P = (a_A b_A), \quad R_A = k_A P = (x_A y_A) \tag{2.20}$$

2.2.3 Криптосистемы Эль-Гамаля

Протокол Диффи-Хеллмана можно рассмотреть в криптосистеме Эль - Гамаля на эллиптической кривой.

Рассмотрим пример, при котором пользователь один хочет отправить сообщение пользователю два. Поэтому строим криптосистему Эль-Гамаля с математическими свойствами эллиптических кривых. Параметрами являются эллиптическая кривая и точка высшего порядка.

Пользователь A выбирает никому неизвестное число $k_A \in Z_N^*$, вычисляет свой общедоступный ключ (E,P,Y), где $Y = k_A P$.

Абонент B для передачи секретного сообщения m

- получает копию открытого ключа (E, N, P, Y);
- прикладывает сообщение в точку $M \in E(F)$;
- задает случайное число $r \in Z_N^*$;
- вычисляет временный ключ $\Delta = rY$;
- вычисляет криптограмму;
- пересылает криптосистему абоненту A.

Для того чтобы расшифровать сообщение пользователь A, применяет свой закрытый ключ ,вычисляя $k_A C_1$, далее то что получилось он прибавляет к C_2, и получает точку M

$$-k_A rP + M + \Delta = M + k_A rP - k_A rP = M$$

(2.21)

Аналитик теперь знает чем является общедоступный ключ (E, P, Y) и криптосистема ($C_1 C_2$), чтобы найти точку M необходимо вычислить точку $k_A rP$. Теперь необходимо перейти к решению алгоритма Диффи-Хеллмана, или проще говоря решить логарифмическую задачу. Использовать тот же рандомизатор не получится. При условии, что у аналитика получилось дешифровать криптосистему или вычислить значение точки M, аналитик может узнать и другие сообщения или информацию, которая была зашифрована с этим рандомизатором.

2.2.4 Протоколы цифровой подписи

Электронная цифровая подпись. Подпись в электронном документе – это некий цифровой код, который в зависимости от информации в сообщении, использованного ключа и задающегося рандомизатора меняется. Обозначим M, K, R, S – это группа сообщений, которые можно использовать, группа ключей для определенного шага, рандомизаторов и электронных подписей.

Ее можно записать в следующем виде

$$Sign(M, K, R): M \text{ x } K \text{ x } R \rightarrow S$$

(2.22)

При такой записи она является электронной подписью пары точек (m, h(m)), эти пары точек являются сообщением, которое готовится быть подписанным. Множество M_δ пар этих точек имеет некоторые криптографические свойства.

- Группа точек H имеет не такую большую мощностью, в сравнении с множеством M_δ: $|H| \ll |M \text{ x } H| = |M_\delta|$.
- Все элементы h группы точек H имеет огромное количество прообразов

$$|H| \ll |Im\, m^{-1}(h)| - |\{m: (m, h) \in M_\delta\}| \quad (2.23)$$

- Если задать первую координату, то можно узнать элемент множества. Чтобы это сделать нужно узнать, чему равно h(m) первой координаты по вычислению числа хеш – функции. Вычислять первую координату по второй считается невозможным так как хеш-функция имеет свойство односторонности.
- При условии что задается элемент $(m, h(m)) \in M_\delta$, считается практически недосягаемым выбрать такой элемент $\big(m', h(m')\big) = \big(m', h(m)\big) \in M_\delta, m' \neq m$, т.е. элемент отличается от того, который задали только первой координатой.
- Считается невозможным взять две группы чисел (m, h) и (m', h), у которых координаты второго элемента совпадали бы.

Электронная подпись сообщения h(m) с ключом подписи k осуществляет проверку возможности использования ключа проверки k'.

Проверяется расчеты предиката проверки $P(S, K')$, в нем K' - это группы ключей для проверки.

Проверочный предикат для подписи с учетом возврата сообщения можно описать следующим образом

$$P(s, k') = \{(m', h'(m)) \in M_\delta\}$$

(2.24)

В нем m’ – это информация для проведения подлинности подписи, а h’(m) хеш-функция информации. Отображение Sign (M, K, R) имеет несколько свойств, которые обеспечивают подлинность подписи.

- Отображение в одну сторону Sign(h(m), K, R) считается невозможным в плане вычисления ключа. Значение отображения s = Sign (h(m), k, r), такое что P (s, k’).
- Для цифровой подписи с известным заранее сообщением производим вычисления s = Sign(h(m), k, r), P(s, k’), считается невозможным фальсифицировать сообщение m’ чтобы оно имело такое же цифровое значение. Одним словом, подделать подпись не получится.
- Считается невозможным подобрать два сообщения m и m’ с одинаковой электронной подписью т.е предикат проверки с заданным ключом будет удовлетворено k.
- При отсутствии информации о цифровой подписи нельзя узнать сообщение и ее значение под этим сообщением

Признаки, описанные выше подкрепляются стойкой хеш-фукнцией, в которой имеются раннее упомянутые признаки и биективные преобразования.

2.2.5 Электронная подпись Эль-Гамаля с возвратом сообщения – схема Nyberg-Rueppel.

Этот алгоритм построен на проверке можно ли изъять «отпечаток» какого-то сообщения с помощью проверки подписи s, которую абонент получает при подписке документа используя ключ подписи k, рандомизатор r и главная задача проверить условие

$$h'(m) = h(m')$$

(2.25)

где m’ это сообщение, а h’(m) – это хеш-значение сообщения, которое берется из подписи на определенном ключе.

Рассмотрим пример. Y(F) сгруппированные точки кривой Y, P – стандартная точка общедоступного ключа, N – порядок этой точки, k – закрытый ключ того кто подписывает документ. Общедоступным ключом пользователя будет точка

$$Q = kP \tag{2.26}$$

e=h(m) – это хеш-функция h в сообщении.

Последовательность вычислений электронной подписи такой.

- Берется случайная точка r, 0 <r <N, оно должно быть таким, чтобы компонента точки не была равна 0

$$R = rP = (x, y) \tag{2.27}$$

- Используем x координату точки R как целое число и вычисляем

$$c = (x + e) mod\ N \tag{2.28}$$

$$d = (r - kc) mod\ N \tag{2.29}$$

Если c равно нулю или d равно нулю, то мы возвращаемся к шагу с формулой 2.25.

В этом случае точки c и d являются подписью документа m, такого что h(m)=e.

Для проверки не фальсифицирована ли хеш-функция h(m) выполняется следующее.

- Проверяется что $1 < \mathrm{c} < \mathrm{N} - 1, 1 < \mathrm{d} < \mathrm{N} - 1$
- Вычисляем

$$R' = dP + cQ \tag{2.30}$$

- Используя x координату точки R’ в виде двоичной записи числа, вычисляем

$$e' = (c - x') mod\ n \tag{2.31}$$

- При совпадении значения точки e’ со значением хеш-функции последнее удостоверяется.

Глава 3. Реализация и тестирование алгоритма

3.1 Протокол Дифии-Хеллмана

Передача информации с помощью открытых каналов была большой проблемой. Для этой проблемы нашлось решение после того как появился алгоритм Диффи-Хеллмана. Этот алгоритм дал возможность пересылать сообщения без отправки каких-либо полезных сведений для расшифровки. Сам алгоритм позволяет пользователям обмениваться ключами без опасности перехвата информации.

Криптографию с открытыми ключами предложили использовать Уитфилд Диффи и Мартин Хеллман.

Они привнесли в криптографию понятие что можно использовать ключи шифрования и расшифрования — исключая возможность утечки информации закрытого ключа с помощью открытого ключа. Впервые этот алгоритм был представлен на Национальной компьютерной конференции 1976 года.

Ниже мы опишем его числовую реализацию.

3.2 Числовая реализация

Возьмем в пример простую ситуацию. Абоненту А нужно отправить сообщение абоненту В и злоумышленник пытается его перехватить. Логичным решением будет шифрование, но даже если способ шифрования известен злоумышленнику, то без ключа он его не расшифрует. Однако для расшифрования абоненту А необходим ключ абонента В, который он передаст по сети, в это время злоумышленник может перехватить его и расшифровать сообщение. Эту проблему решает протокол Диффи-Хеллмана.

Диффи-Хеллман работает по принципу неполного обмена ключом шифрования по сети. У каждой стороны есть открытый ключ (который может видеть каждый, включая злоумышенника) и закрытый ключ (его может видеть только пользователь компьютера). На рисунке 3.1 показана схема личных и открытых ключей.

Рисунок 3.1 – Схема ключей и абонентов

Предположим, что абонент А не знает ничего кроме открытого ключа абонента В. Нужно создать частичный ключ шифрования используя 3 известных параметра. Открытый и закрытый ключ абонента А и открытый ключ абонента В.

$$A_{partial} = A_{pub}^{A_{pri}} mod\ B_{pub} \quad (3.1)$$

$$B_{partial} = A_{pub}^{B_{pri}} mod\ B_{pub} \quad (3.2)$$

Полученный частичный ключ мы отправляем абоненту В, абонент В отправляет свой частичный ключ абоненту А. Злоумышленник перехватывает отправленные частичные ключи и будет пытаться с помощью него и открытых ключей получить закрытые. Особенность вычисления по модулю в том, что функция заставляет значение циклически изменяться. Если к примеру, полученное число 151 значение будет между 151-1 и 0. Существует бесконечно множество чисел, по модулю которые равны частичному ключу абонента А или В, что делает подборку чисел практически невозможной. Далее идет генерация полного ключа. После того как абоненты обменялись частичными ключами вычисляем полные ключи

$$A_{full} = B_{partial}^{A_{pri}} mod\ B_{pub} \quad (3.3)$$

$$A_{full} = A_{partial}^{B_{pri}} mod\ B_{pub} \quad (3.4)$$

Полученные полные ключи должны совпасть по значению. Заключается это в следующем соотношении:

$$(g^a mod\ p)^b mod\ p = (g^b mod\ p)^a mod\ p = g^{ab} mod\ p \quad (3.5)$$

В нем a и b это закрытые ключи, а g и p открытые ключи. Абонентам удалось обменяться друг с другом по сети достаточным количеством информации, чтобы сгенерировать общий ключ шифрования, не ставя под угрозу свои закрытые ключи. После этого абонент B передает зашифрованное сообщение и абонент A расшифровывает его.

3.3 Программная реализация

Перейдем к программной реализации алгоритма. Алгоритм Диффи-Хеллмана реализован на языке Python, в среде разработки VSC (Visual Studio Code).

Для начала нам нужно создать конструктор. Для этого используем метод __init__().

test.py

> Users > Дима > Desktop > test.py

```
1   class Diffi_Hellman():
2       def __init__(self, public_key_a, public_key_b, private_key):
3           self.public_key_a = public_key_a
4           self.public_key_b = public_key_b
5           self.private_key = private_key
6           self.full_key = None
7
```

Рисунок 3.2 – конструктор для создания ключей

В этом конструкторе создаются переменные для открытых и закрытых ключей. Так как пользователь A будет знать лишь свой открытый, закрытый ключ и открытый ключ пользователя B, нам не нужно создавать дополнительную переменную для закрытого ключа. В этот метод мы передаем два открытых ключа и закрытый ключ для каждого пользователя. Так же имеется переменная для полного ключа, которую в дальнейшем мы

будем вычислять с помощью открытых и частичных ключей, поэтому на данном этапе она остается пустой.

Далее нам нужна функция для генерации частичного ключа для обоих пользователей.

```
def part_key_generation(self):
    part_key = self.public_key_a ** self.private_key
    part_key = part_key % self.public_key_b
    return part_key
```

Рисунок 3.3 – функция генерации частичного ключа

В этой функции мы вычисляем частичный ключ с помощью открытого, закрытого ключа одного пользователя и открытого ключа другого пользователя. Для вычисления используем формулы 3.1, 3.2.

Затем создаем функции для вычисления полного ключа. Для этого мы используем наши частичные ключи

```
def full_key_generation(self, part_key_u):
    full_key = part_key_u ** self.private_key
    full_key = full_key % self.public_key_b
    self.full_key = full_key
    return full_key
```

Рисунок 3.4 – функция генерации полного ключа

В эту функцию мы передаем переменную part_key_u, это переменная, в которую мы передаем значение частичного ключа для каждого пользователя. Для пользователя А это частичный ключ В, а для пользователя В это частичный ключ А. Для вычисления используем формулы 3.3, 3.4.

Теперь создаем функции для шифрования и расшифровывания сообщения.

```
def encrypt_message(self, message):
    encrypted_message = ""
    key = self.full_key
    for c in message:
        encrypted_message += chr(ord(c) + key)
        #print(encrypted_message)
    return encrypted_message
```

Рисунок 3.5 – функция шифрования сообщения

В этой функции каждый символ считывается и преобразовывается в целое число, после чего к этому значению прибавляется значение полного ключа и это число снова преобразуется снова в символ. Это происходит для каждого символа в нашем сообщении. Функция chr используется для преобразования значения integer в значение character. Функция ord() это функция обратная функции chr, она преобразует значение character в значение integer.

Далее создаем функцию для расшифровки сообщения. В нем мы так же используем функции chr и ord() для преобразования сообщения в числовое значение и обратно в словесное значение.

```
def decrypt_message(self, encrypted_message):
    decrypted_message = ""
    key = self.full_key
    for c in encrypted_message:
        decrypted_message += chr(ord(c) - key)

    #print(decrypted_message)
    return decrypted_message
```

Рисунок 3.6 – функция расшифровки сообщения

Так как в функции зашифровывания сообщения мы прибавляли значение ключа к значению сообщения, то в этой функции мы наоборот вычитаем это значение из значения сообщения.

Переходим к части ввода информации.

```
message = input('Введите Ваше сообщение: ')
a_public = int(input('Введите a_public: '))
a_private = int(input('Введите a_private: '))
b_public = int(input('Введите b_public: '))
b_private = int(input('Введите b_private: '))
```

Рисунок 3.7 – ввод информации

Здесь мы задаем сообщение, которое мы хотим передать, вводим открытый ключ пользователя a, закрытый ключ пользователя a, открытый ключ пользователя b и закрытый ключ пользователя b.

```
UserA = Diffi_Hellman(a_public, b_public, a_private)
UserB = Diffi_Hellman(a_public, b_public, b_private)
```

Рисунок 3.8 – вызов функции для передачи ключей

Здесь мы вызываем функцию Diffi_Hellman() и передаем туда значения ключей. Так как пользователям известны лишь 3 ключа (свой открытый, открытый собеседника и свой закрытый) мы передаем 3 значения.

Далее вычисляем частичные ключи для пользователя a и пользователя b.

```
a_part = UserA.part_key_generation()
b_part = UserB.part_key_generation()
print(a_part, b_part)
```

Рисунок 3.9 – вычисление частичных ключей

Для пользователя a вызываем функцию вычисления частичного ключа, тоже самое делаем и для пользователя b. И выводим их на экран.

Следующим шагом вычисляем полные ключи, для вычисления которых нам требуются частичные.

```
a_full = UserA.full_key_generation(a_part)
b_full = UserB.full_key_generation(b_part)
print(a_full, b_full)
```

Рисунок 3.10 – вычисление полных ключей

Для пользователя a вызываем функцию вычисления полного ключа, тоже самое делаем и для пользователя b. При вызове этих функций мы передаем в них значение частичных ключей пользователей. И выводим их на экран.

Последним шагом будет зашифровывание сообщения, его расшифровка и вывод обоих сообщений на экран.

```
b_encrypted = UserB.encrypt_message(message)
a_decrypted = UserA.decrypt_message(b_encrypted)

print(b_encrypted, a_decrypted)
```

Рисунок 3.11 – шифрование и дешифрование сообщения

Здесь пользователь b вызывает функцию зашифровки сообщения и передает туда сообщение, введенное пользователем. Пользователь a вызывает функцию дешифровки сообщения и передает в нее зашифрованное сообщение пользователя b. После всех вычислений мы выводим зашифрованное и расшифрованное сообщение.

3.4 Результаты

В качестве входных данных возьмем значения ключей из примера числовой реализации. В ней открытый ключ пользователя a 197, его закрытый ключ 199. У пользователя b открытый ключ 151, закрытый 157. В качестве сообщения возьмем «Hello, how u doing». Ниже на рисунке 3.12 продемонстрирован результат работы программы с такими входными данными.

```
C:\Users\Дима\Desktop>python test.py
Введите Ваше сообщение: Hello,how u doing
Введите a_public: 197
Введите a_private: 199
Введите b_public: 151
Введите b_private: 157
147 66
75 75
?º··?w??AkAk????? Hello,how u doing
```

Рисунок 3.12 – результат работы программы

Как видно на рисунке результаты получились такие же, как и в числовой реализации. Открытый ключ пользователя a равен 147, а у b 66. Закрытый же ключ у них одинаков и равен 75 и в самом низу показано зашифрованное сообщение и расшифрованное.

Для исследования проверим как будет работать алгоритм если задать достаточно большое число в качестве ключей или такое число, при котором остаток от деления будет равен нулю.

Для начала проверим работу алгоритма при задании такого ключа при котором остаток от деления будет равен нулю.

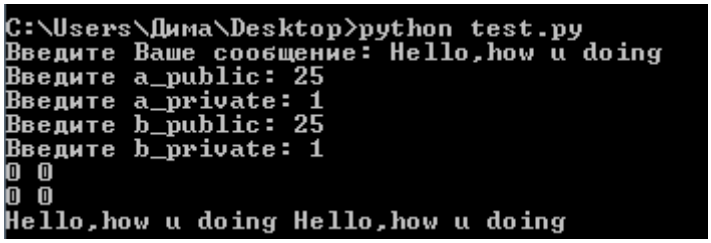

Рисунок 3.13 – получение нулевых ключей

Как видно на рисунке 3.13 если остаток от деления равен нулю, то частичные и полные ключи будут равны нулю, при этом сообщение не будет зашифровано. Т.е. пользователям нельзя задавать значения, при которых остаток от деления будет равен нулю.

Далее проверим работу алгоритма при больших значениях ключей.

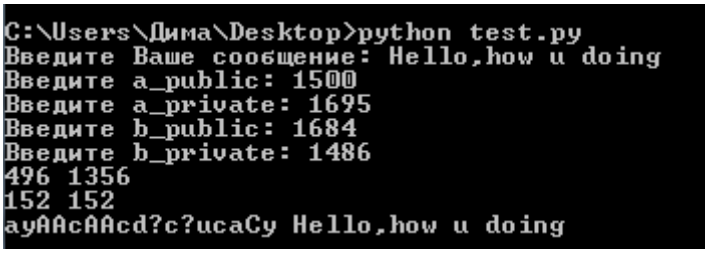

Рисунок 3.14 – работа алгоритма при больших значениях ключей

Даже не смотря на большие размеры ключей алгоритм работает исправно без ошибок. Однако существует проблема вычислений таких больших значений при высчитывании ключей человеком, а не компьютером. Связано это с возведением в степень числа при котором получается слишком большое значение, из-за которого любой калькулятор будет давать ответ ∞ (бесконечность).

Заключение

Выпускная квалификационная работа посвящена реализации и анализа криптографического алгоритма на эллиптических кривых.

Были выполнены следующие цели:

Описаны эллиптические кривые их теория, математические свойства, недостатки и преимущества;

Описаны алгоритмы и протоколы на эллиптических кривых.

В качестве алгоритма для реализации был выбран протокол Диффи-Хеллмана.

В ходе выполнения работы был рассмотрен криптографический алгоритм Диффи-Хеллмана и была продемонстрирована реализация в виде программного кода, который показывает работу алгоритма.

Была описана теория алгоритма, приведена числовая и программная реализации. Алгоритм так же был протестирован на наличие ошибок, которых в ходе работы не было выявлено. В качестве исследования алгоритма был проведен тест на большие и нулевые значения.

Протокол обмена ключами Диффи-Хеллмана является достаточно надежным методом обмена ключами через незащищенные каналы передачи информации. Он используется в системах, обеспечивающих безопасность передаваемых данных и протоколах, использующихся сегодня. За долгие годы использования протокола обмена ключами Диффи-Хеллмана было разработано множество атак, но и сам протокол не стоял на месте и множество исследователей изменяли и модифицировали его. Это доказывает, что безопасный обмен секретными ключами является важным аспектом защиты информации. Что делает протокол обмена ключами Диффи-Хеллмана востребованным и в наше время.

В результате мы получили работающий исправно алгоритм с его теорией, числовой и программной реализацией, а также его тестирование и исследование.

Список используемой литературы

1. Жданов, О. Н. Применение эллиптических кривых в криптографии: учеб. пособие / О. Н. Жданов, Т. А. Чалкин. — Красноярск: СибГАУ, 2011 — 65 с.
2. Рябко, Б. Я. Криптографические методы защиты информации: учеб. пособие / Б. Я. Рябко, А. Н. Фионов. — 2-е изд., стер. — М.: Горячая линия – Телеком, 2012. — 229 с.
3. Бабаш, А. В. Информационная безопасность и защита информации: учебное пособие/ А. В. Бабаш, П.Н. Башлы, Е. К. Баранова. — М.: РИОР, 2013. — 222 с.
4. Гомес, Ж. Математики, шпионы и хакеры. Кодирование и криптография /Ж. Гомес. — М.: Де Агостини, 2014. —144 с.
5. Грушо, А. А. Теоретические основы компьютерной безопасности: учеб. пособие для вузов / А. А. Грушо. — М.: Академия, 2009. — 272 с.
6. Жуков, А. Е. Системы блочного шифрования: учеб. пособие по курсу «Криптографические методы защиты информации» / А.Е. Жуков — М.: Издательство МГТУ им. Н.Э. Баумана, 2013. — 79с.
7. Загинайлов, Ю. Н. Основы информационной безопасности: курс визуальны лекциий: направление подготовки "Информационная безопасность", специальность "Экономическая безопасность"/ Ю.Н. Загинайлов. — М.: Директ-Медиа, 2015 -105 с.
8. Иванов, М. А. Криптографические методы защиты и информации в компьютерных системах и сетях: учеб. пособие / М. А. Иванов, И. В. Чугунков. — М.: МИФИ, 2012 — 400 с.
9. Фороузан, Б. А. Математика криптографии и теория шифрования: учеб. пособие / Б. А. Фороузан. — М.: ИНТУИТ, 2016 — 510 с.
10. Ященко, В. В. Введение в криптографию / В. В. Ященко, Н. П. Варновский, Ю. В. Нестеренко — 4-е изд., доп. М.: МЦНМО, 2012. — 348 с.

11. Нестерова, Л. Ю., Карпенкова Н. В. Создание криптографии с помощью модулярной математики // Молодой ученый. — 2014. — №21.1. — с. 237-240.
12. Литвинская, О. С. Основы теории передачи информации. Учебное пособие / О.С. Литвинская, Н.И. Чернышев. - М.: КноРус, 2015. - 168 с.
13. Бабаш, А. В. История криптографии. Часть I / А.В. Бабаш, Г.П. Шанкин. - М.: Гелиос АРВ, 2016. - 240 с.
14. Шаньгин, В. Ф. Информационная безопасность и защита информации / В.Ф. Шаньгин. - Москва: Огни, 2016. - 551 с.
15. Шнайер, Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си / Б. Шнайер. - М.: Триумф, 2012. - 518 с.
16. Шумский, А.А. Системный анализ в защите информации / А.А. Шумский. - Москва: СПб. [и др.] : Питер, 2013. - 224 с.
17. Литвинская, О. С. Основы теории передачи информации. Учебное пособие / О.С. Литвинская, Н.И. Чернышев. - М.: КноРус, 2015. - 168 с.
18. Gilson, D. Blockchain.info issues refunds to Bitcoin theft victims / D. Gilson, 2013. [Электронный ресурс]: http://www.coindesk.com/blockchain-info-issuesrefunds-to-bitcoin-theft-victims/
19. Chandrasekhara, K.R. “Elliptic Curve based authenticated session Key establishment protocol for High Security Applications in Constrained Network environment” / K.R. Chandrasekhara, M.P. Pillai1 and Sebastian, 2010. [Электронный ресурс]: http://www.arxiv.org/pdf/1202.1895
20. Griffiths, I. Programming Java: Building Windows 8, Web, and Desktop Applications for the .NET 4.5 Framework / I. Griffiths, 2012. – [Электронный ресурс]: https://giants.ict.griffith.edu.au./JPL/
21. Herlihy, M. The Art of Multiprocessor Programming. / M. Herlihy, N. Shavit. — Revised 1st Edition. — Morgan Kaufmann, 2012. — 537 p.
22. Ingalls, J. M. Interior Ballistics / J.M. Ingalls. — Nabu Press. — 2014. — 258 p.

23. Michaelis, K. Randomly failed! The state of randomness in current Java implementations / K. Michaelis, C. Meyer, and J. Schwenk, In E. Dawson, editor // CT-RSA. — volume 7779 of LNCS. – 2013. – pp. 129 —144.
24. Georgiev, M. The most dangerous code in the world: Validating SSL certificates in non-browser software / M. Georgiev, S. Iyengar, S. Jana, R. Anubhai, D. Boneh, and V. Shmatikov, In T. Yu, G. Danezis, and V. D. Gligor, editors // ACM Conference on Computer and Communications Security. — 2012. — pp. 38 – 49.
25. Pelland, P. Moving to Microsoft Visual Studio 2010 / P. Poland, K. Haines – Microsoft Press, 2011. — 336 p.
26. Nils, G. “Comparing Elliptic Curve Cryptography and RSA on 8-bit CPus”/ Nils Gura, Arun Patel, Arvinderpal Wander, Hans Eberle // 6th International Workshop Cambridge, MA, USA. — 2004. — 119 p.