Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Отчет по лабораторной работе №13

«Исследование криптографических алгоритмов на основе эллиптических кривых. Часть 2»

Студентка: Пунько А.А,

ФИТ 3 курс 5 группа

Преподаватель: Берников В. О.

Минск 2020

**Цель:** изучение и приобретение практических навыков разработки и использования приложений для реализации криптографических алгоритмов на основе эллиптических кривых (содержит 3 самостоятельных задания, каждое из которых рассчитано на 2 часа аудиторных занятий).

**Задачи:**

1. Закрепить теоретические знания по алгебраическому описанию и геометрическому представлению операций над эллиптическими кривыми (ЭК):

* по алгоритмам согласования ключевой информации на основе ЭК,
* алгоритмам зашифрования/расшифрования информации на основе асимметричной криптонафии и ЭК,
* алгоритмам генерации и верификации электронной цифровой подписи на основе асимметричной криптографии и ЭК,
* оценке криптостойкости систем на основе ЭК.

2. Разработать приложение для реализации указанных преподавателем методов криптопреобразования на основе ЭК.

3. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

# Теоретические сведения

Первый этап. Выбор (генерация) ЭК.

Обычно он основан на выполнении следующих условий и операций.

Входными параметрами являются: число l, число р, удовлетворяющее условию 22l-1 < р < 22l, р = 3 mod 4, 0 < a < p. Можно использовать некоторое простое число р = 22l – с, где с – небольшое натуральное число.

Выбирается число b, такое, что 0 < b < p. Таким образом, задана ЭК: Ер(а, b).

Выбираются порядок q (простое число) и генерирующая точка G, которая задается двумя координатами, например, G = (0, уG). Дополнительно к рассмотренным действиям стандарт [50] предусматривает использование вспомогательного параметра (s, seed) – произвольное 64-битное число. Здесь нижние индексе в левом столбце обозначают битовую длину числа.

Второй этап. Генерация ключевой информации.

Входными параметрами являются: р, а, b, q и G.

Генерируется тайный ключ – число d, выбранное из множества {1, 2, …, q–1}.

Вычисляется открытый ключ – точка Q: Q = dG, к открытому ключу также относятся р, а, b, q.

Отметим также, что сгенерировать ключевую информацию на основе ЭК

можно воспользовавшись известной нам библиотекой OpenSSL. Например, если воспользоваться версией OpenSSL 1.1.1L в системе Debian 9 (с помощью команды с двумя разными псевдонимами (выделены жирным): openssl ecparam –name secp192k1 –genkey –out secp192k1 , openssl ecparam –name secp521r1 –genkey –out secp521r1, то получим тайные ключи соответственно следующего содержания:

MFwCAQEEGLDsGwgZq/Kq4suR74ftkipbKMRmoWDtlqAHBgUrgQQAH6E0AzIABPfKAz FU+QKsh+I7a6K5taNUe3TZAdLMp92RpYoT0PIrmGD3QVRcqAmqZSba6kanKg==

MIHcAgEBBEIAvv7P//lWx3QQis5Hb25eN/UY5isVJk+s56ZDSTleUcrqj2mNH4Y3 xWLXGMtpmDJRiHalCv3MDt/T5h67daHaViagBwYFK4EEACOhgYkDgYYABABgOPla 5ygHB/j79g0R2N12/tv4YlIj6ZA+t2FhtvEMPvj9QHMg5sN45yjGKmLlIwEMP2YW xjPj3YL0Z0uLO9BBYwBUGVCPEWKylC8x5qGL1ypG6shCPTUcXQxLuFMmKv+AaDH2 4TCdBvl9nYANhlxZKv96Pb/lari3OKZkmO5zgVWKCw==

Отметим еще раз, что ЭК в криптографических приложениях обычно используется на этапе генерации либо согласования ключевой информации. Таким образом, можно отметить 3 направления использования ЭК в криптографии:

* в алгоритмах согласования (передача) ключевой информации (на основе идеи Диффи-Хеллмана),
* в алгоритмах асимметричного шифрования/дешифрования сообщений,
* в алгоритмах генерации/верификации ЭЦП.

Рассмотрим случай использования ЭК в алгоритме Диффи-Хеллмана. Предположим, что Eр – это ЭК над Fр, а Q – заранее определенная и согласованная сторонами А и В точка на E. Отправитель A выбирает тайное случайное число kA, вычисляет точку РА = kA\*Q и отправляет ее получателю B. B действует аналогично: он случайным образом выбирает число kB, вычисляет случайное число kA, вычисляет точку РВ = kВ\*Q и отправляет результат стороне A. Общий ключ P = kA\*kB\*Q. Отправитель A вычисляет P путем умножения числа РВ, полученного от получателя B, на его секретное число kA. Похожим образом действует другая строна.

Рассмотрим это на примере алгоритма Эль-Гамаля. Вспомним, что зашифрованное сообщение М или каждый зашифрованный блок (mi) этого сообщения состоят из двух чисел. Вспомним лабораторную работу № 8, где блок шифртекста (ci) в соответствии мы обозначали двумя символами аi и bi и вычисляли как аi = gk mod p, bi = (yk•mi) mod p.

Поскольку символы а и b мы зарезервировали в текущей работе для обозначения параметров ЭК, то блок шифртекста сейчас будем обозначать соответственно символами Сi1 и Ci2. При использовании ЭК зашифрование предполагает представление сообщения в виде точки Р (или представления каждого блока сообщения в виде разных точек Рi) ЭК с известной точкой G и известным Q. Соответственно шифртекст – это две точки на той же ЭК: С1 и C2 или Сi1 и Ci2. Предположим, что шифруемое сообщение М – это точка Р на ЭК. Сторона А выбирает некоторое случайное число k и далее выполняет вычисления с использованием открытого ключа стороны В: С1 = kG, С2 = P + kQ.

Получатель для расшифрования сообщения вычисляет: P = С2 – dC1. Знак «–»означает сложение с инверсией: инверсией по отношению к точке (х, у) является точка (х, –у) на ЭК.

# Практическая часть

В данной части лабораторной работы было необходимо создать приложение, реализующиее зашифрование и расшифрование текста на основе ЭК. Стартовое окно приложение представлено на рисунке 1.

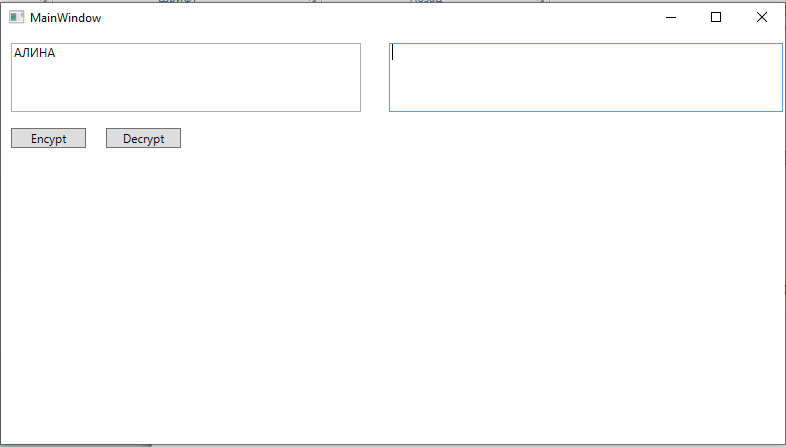


Рисунок 1 – Стартовое окно приложения

Результат зашифрования представлен на рисунке 2.

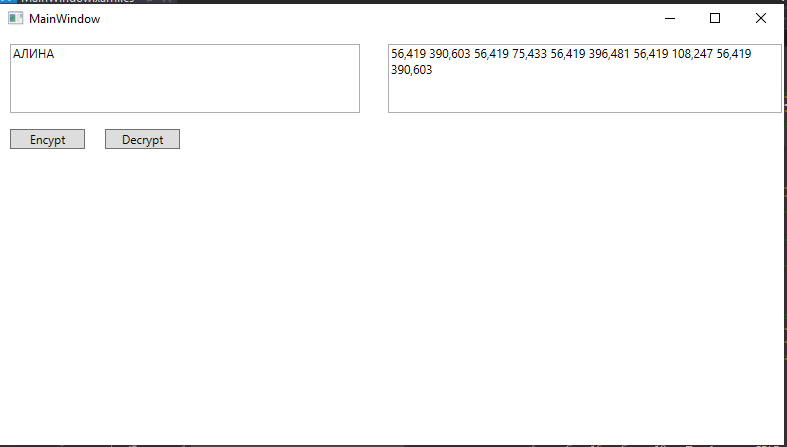


Рисунок 2 – Результат зашифрования текста

Алгоритм, реализующие зашифрование текста, представлен в листинге 1.

private void Button\_Click(object sender, RoutedEventArgs e)

{

Init();

string text = TextToEncrypt.Text;

TextToDecrypt.Text = "";

(int x, int y) cort;

int k = 3;

int xG = 0;

int yG = 1;

string[] numbersQ = Multiply(d, xG, yG).Split(',');

int xQ = int.Parse(numbersQ[0]);

int yQ = int.Parse(numbersQ[1]);

foreach (char s in text)

{

hash.TryGetValue(s, out cort);

string numbersC1 = Multiply(k, xG, yG);

string[] numbersExpr1 = Multiply(k, xQ, yQ).Split(',');

string numbersC2 = SumTwoPoints(cort.x, int.Parse(numbersExpr1[0]), cort.y, int.Parse(numbersExpr1[1]));

TextToDecrypt.Text += numbersC1 + ' ' + numbersC2 + ' ';

}

TextToDecrypt.Text = TextToDecrypt.Text.Remove(TextToDecrypt.Text.Length - 1);

}

Листинг 1 – Алгоритм зашифрования текста

Результат расшифрования представлен на рисунке 3.

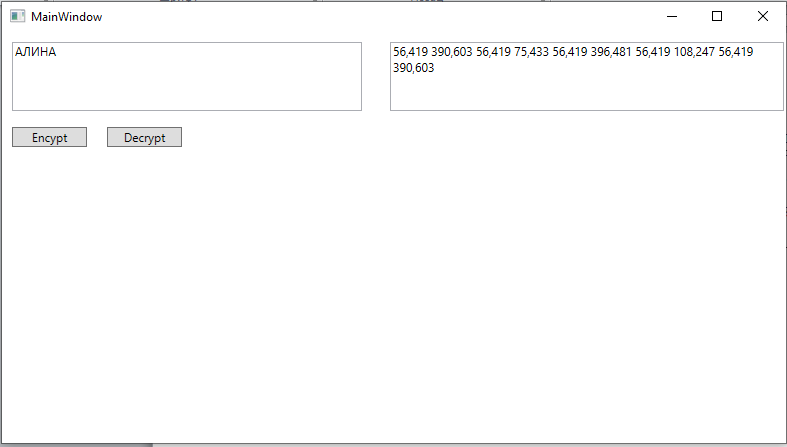


Рисунок 3 – Результат расшифрования текста

Алгоритм, реализующие расшифрование текста, представлен в листинге 2.

string[] text = TextToDecrypt.Text.Split(' ');

TextToEncrypt.Text = "";

for (int i = 0; i < text.Length; i += 2)

{

string[] numbers1 = text[i].Split(',');

string[] numbers2 = text[i + 1].Split(',');

string[] numbersC1 = Multiply(d, int.Parse(numbers1[0]), int.Parse(numbers1[1])).Split(',');

string[] result = SumTwoPoints(int.Parse(numbers2[0]), int.Parse(numbersC1[0]), int.Parse(numbers2[1]), mod(-int.Parse(numbersC1[1]), p)).Split(',');

(int x, int y) cort = (int.Parse(result[0]), int.Parse(result[1]));

char s;

obrhash.TryGetValue(cort, out s);

TextToEncrypt.Text += s;

}

Листинг 2 – Алгоритм расшифрования текста

# Вывод

В данной части лабораторной работы был изучен способ использования эллиптических кривых в криптографии для генерации ключевой информации, рассмотрены способы использования этой информации в различных алгоритмах и создано приложение, реализующие зашифрование и расшифрование текста с использованием ЭК и алгоритма Эль-Гамаля.