КРИПТОГРАФИЧЕСКИЙ АЛГОРИТМ RSA И АЛГОРИТМЫ СОЗДАНИЯ ЦИФРОВОЙ ПОДПИСИ НА ЕГО ОСНОВЕ

Герасимёнок Н.А., Готченя Д.Г., Лазук И.С.

Белорусский государственный университет информатики и радиоэлектроники
г. Минск, Республика Беларусь

Стройникова Е.Д. – ст. преподаватель

В современных реалиях мы не то что каждый день, а чуть ли не каждый час используем Интернет, а, следовательно, передаем огромный поток данных. А там, где есть передача данных, сразу возникает необходимость в защите этих данных, следовательно, и в шифровании. Остановимся на рассмотрении алгоритма шифрования RSA. В ходе данной работы была реализована консольная программа на языке программирования C++ (в среде Visual Studio Code).

Алгоритм RSA

Первый этап – генерация ключей:

1) Выбираем два больших случайных простых числа, приблизительно заданного размера (например 2048 бит каждое).

2) Находим их произведение, именуемое модулем, которое обычно обозначается n, где

$$n = p \cdot q.$$

3) Вычисляем мультипликативную арифметическую функцию Эйлера

$$\varphi(n) = (q - 1) \cdot (p - 1).$$

4) После этого для открытого ключа необходимо выбрать целое число e $(1 < e < \varphi(n))$, взаимно простое со значением функции $\varphi(n)$.

5) Для генерации закрытого ключа необходимо вычислить число d, оно, в свою очередь, должно удовлетворять сравнению

$$d \cdot e \equiv 1 \ (mod \ \varphi(n)),$$

d называется секретной экспонентой.

После проведенных действий получаем пару ключей:

– открытый ключ $\{e, n\}$, который публикуется в открытом доступе;

– закрытый ключ $\{d, n\}$, который остается у отправителя и держится в секрете.

Второй этап – работа протокола:

Выбираем целое число m для шифрования, чтобы его зашифровать, воспользуемся открытым ключом:

$$c = E(m) \equiv m^e \ (mod\ n).$$

Полученное c и есть зашифрованное число. Для того чтобы из c обратно получить исходное число m, необходимо использовать закрытый ключ:

$$m = D(c) \equiv c^d \ (mod\ n).$$

Алгоритмы цифровой подписи

Также криптосистема RSA позволяет реализовать аутентификацию, создав цифровую подпись. Данная подпись подтвержает целостность отправляемой информации и может быть использована вместо стандартной подписи от руки.

Алгоритм с открытым текстом

Первый этап – генерация цифровой подписи:

1) Возьмем открытый текст m.

2) Для создания подписи, обозначаемой s , необходимо воспользоваться сгенерированным ранее закрытым ключом:

$$s = S_A(m) \equiv m^d \ (mod\ n).$$

3) Передаем пару $\{s, m\}$, состоящую из подписи и открытого текста.

Второй этап – проверка неизменности сообщения с помощью электронной подписи:

1) Используя полученную подпись и открытый ключ, сгенерируем прообраз текста m, обычно обозначаемый m':

$$m' = P_A(s) \equiv s^e \ (mod\ n).$$

2) Сравниваем прообраз m' и сам текст m, тем самым выясняя подлинность подписи и целостность текста.

Алгоритм с закрытым текстом

Данный способ предоставляет бóльшую защиту, однако требует дополнительных вычислений.

Алгоритм:

1) Отправитель шифрует исходный текст алгоритмом RSA с помощью открытого ключа получателя.

2) Далее отправитель должен сгенерировать цифровую подпись с помощью своего закрытого ключа.

3) Шифруем полученную подпись открытым ключом получателя.

4) Отправляем зашифрованный текст и подпись.

5) Получатель расшифровывает текст и подпись с помощью своего закрытого ключа.

6) С помощью расшифрованной подписи и открытого ключа получателя создаем прообраз исходного текста.

7) Сравниваем прообраз и исходный текст.

Вывод

Алгоритм RSA является классическим примером шифров с открытым ключом, он надежно выполняет свою работу. С его помощью можно как шифровать текст, так и создавать цифровую подпись, что делает его довольно универсальным. В основе алгоритма RSA лежит использование односторонних функций, что на данный момент обеспечивает хорошую защиту. Однако с каждым годом вычислительные мощности стремительно растут, открываются новые математические методы разложений больших чисел на простые множители, что в перспективе ослабляет данный способ шифрования. При наличии достаточной выборки текста, зашифрованного чистым алгоритмом RSA, используя частотный анализ, возможно «обойти» данный шифр. Также скорость шифрования

оставляет желать лучшего, поэтому в наше время RSA используется в связке с другими, более быстрыми, алгоритмами шифрования.

Модифицированный алгоритм RSA

На самом деле, можно довольно легко модифицировать алгоритм шифрования RSA, чтобы он смог «сопротивляться» частотному анализу.

Алгоритм:

1) После перевода символов в числа добавляем в начало еще одно случайное число.
2) Начиная уже со второго числа, переопределяем его значение следующим образом: берем существующее значение и складываем его с предыдущим, а далее берем остаток от деления на n.
3) Новый полученный набор чисел шифруем открытым ключом и передаем зашифрованное сообщение.
4) Для расшифровки используем закрытый ключ, осталось получить исходные числа.
5) Чтобы «распутать», идем с начала, минуя первое число, и рассматриваемому числу присваиваем остаток от деления на n от разности самого числа с предшествующим ему. Первое число не используем, т. к. соответствующего ему символа в исходном сообщении не было.
6) Далее из чисел восстанавливаем текст, действуя по стандартному алгоритму.

Данным набором действий устраняется однозначное соответствие между зашифрованным числом и буквой в исходном сообщении. Также добавленное в начало случайное число окончательно убирает эту связь.

Ссылка на программную реализацию:
https://github.com/LaRtik/RSAsignature.

Список использованных источников:

1. Фергюсон, Н. *Практическая криптография /* Н. Фергюсон, Б. Шнайер*. ; пер. с англ. – М. [и др.] : Диалектика, 2005. – 421 с.*

*2. Преимущества и недостатки алгоритма шифрования RSA [*Электронный ресурс]. *– 2017–2021. – Режим доступа :* https://studwood.ru/1685074/informatika/preimuschestva_nedostatki_algoritma_shifrovaniya.

*3. Использование криптосистемы RSA в настоящее время [*Электронный ресурс]. *– 2007. – Режим доступа :* http://www.cyberguru.ru/algorithms/cryptography/rsa.html?start=10.

*4. RSA – Википедия [*Электронный ресурс]. *– 2021. – Режим доступа :* https://ru.wikipedia.org/wiki/RSA#%D0%98%D1%81%D1%82%D0%BE%D1%80%D0%B8%D1%8F.