

Реализация алгоритма асимметричного шифрования RSA

Бондаренко Владислав Витальевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Козич Виталий Геннадьевич

Приамурский государственный университет имени Шолом-Алейхема

Студент

Баженов Руслан Иванович

Приамурский государственный университет имени Шолом-Алейхема

к.п.н., доцент, зав. кафедрой информационных систем, математики и методик обучения

Аннотация

Человечество вступило в новую информационную эпоху, где власть имеет тот, кто владеет информацией. В таких условиях возникает необходимость защиты своей личной информации от разного рода кибератак и шпионажа. Для этого существует множество алгоритмов шифрования, которые заметно усложняют жизнь киберпреступникам. В данной статье рассматривается реализация алгоритма асимметричного шифрования RSA на языке программирования C# в среде Visual Studio.

Ключевые слова: RSA, криптография, асимметричное шифрование, открытый ключ, секретный ключ

The implementation of the algorithm of asymmetric encryption RSA

Bondarenko Vladislav Vitalievich

Sholom-Aleichem Priamursky State University

Student

Kozich Vitaliy Gennadievich

Sholom-Aleichem Priamursky State University

Student

Bazhenov Ruslan Ivanovich

Sholom-Aleichem Priamursky State University

Candidate of pedagogical sciences, associate professor, Head of the Department of Information Systems, Mathematics and teaching methods

Abstract

Humanity has entered a new information age, where power is the one who owns the information. In such circumstances, it is necessary to protect personal information from all sorts of cyber-attacks and espionage. There are many encryption algorithms that significantly complicate life for cybercriminals. This article discusses the implementation of the algorithm of asymmetric encryption - RSA using the programming language C# in the environment of Visual Studio.

Keywords: RSA, cryptography, asymmetric encryption, public key, private key

Алгоритм RSA является алгоритмом асимметричного шифрования, что означает использование двух ключей, которые не совпадают друг с другом. Один из ключей является открытым и общедоступным и используется для шифрования сообщений. Другой, секретный ключ, надежно хранится и используется для дешифрования сообщений. Эти ключи взаимозаменяемые – можно производить шифровку и дешифровку с любым из них, но если сообщение зашифровано с помощью одного ключа, то расшифровать его можно только с помощью другого.

Весь алгоритм состоит из трех этапов: генерация ключей, шифрование, расшифрование. Для того, чтобы сгенерировать пару ключей, необходимо выбрать два простых целых числа p и q , и вычислить их модуль (произведение) $n = p * q$. Модуль представляет из себя очень большое число, которое очень сложно затем разложить на отдельные множители. Алгоритм RSA основан на сложности задачи факторизации (разложении на множители) больших чисел. Если разрядность ключа довольно высока, то, имея один из ключей, очень сложно, практически невозможно (при нынешних вычислительных мощностях) найти другой ключ. Далее вычисляется функция Эйлера $\phi(n) = (p - 1)(q - 1)$, выбирается открытая экспонента e и вычисляется секретная экспонента $d = e^{-1} \bmod \phi(n)$. Пара (e, n) составляет открытый ключ, а пара (d, n) – секретный ключ. Шифрование проводится по формуле $c = m^e \bmod n$, дешифрование – $m = c^d \bmod n$, где m – сообщение [1].

Основатель блога [raveldvlip](#) П.Двуреченский в одной из своих статей дает краткое описание алгоритма RSA и рассказывает о способах его реализации [2]. M.J.Schlabaugh является создателем приложения RSACryptoPad, исходники и примеры которого выложены на сайте CodeProject [3]. Авторы К. Нейгел, Б. Ивсен, Дж. Глинн, К. Уотсон в своей книге “C# 4.0 и платформа .NET 4 для профессионалов” подробно на конкретных примерах объясняют принципы работы приложения для шифрования и дают описание основных классов и методов, присутствующих в платформе .NETFramework, для создания приложения, реализующего систему RSA на языке программирования C# [4]. Создатели алгоритма RSA R.Rivest, A. Shamir, L.Adleman в своей статье 1978 г. рассказывают о работе алгоритма и его основных преимуществах [5]. Профессор электротехники и вычислительной техники университета Пердью Avinash Как в своей лекции, опубликованной на сайте университета, подробно разобрал все аспекты

проектирования криптосистемы RSA и привел пример, написанный на языке программирования Python [6]. Доктор В.Kaliski, являющийся одним из главных ученых и вице-президентом по исследованиям RSA Security, описал математические аспекты алгоритма RSA с открытым ключом [7]. S.Somani привел пример своего собственного приложения, написанного на языке программирования C#, с довольно простой реализацией [8].

Программа создавалась на языке программирования C# в среде Visual Studio с использованием стандартных классов для реализации криптографических алгоритмов из пространства имен System.Security.Cryptography. Приложение показывает все три этапа алгоритма и для примера производит шифрование и дешифрование текста.

Окно программы выглядит следующим образом (см. рис. 1).

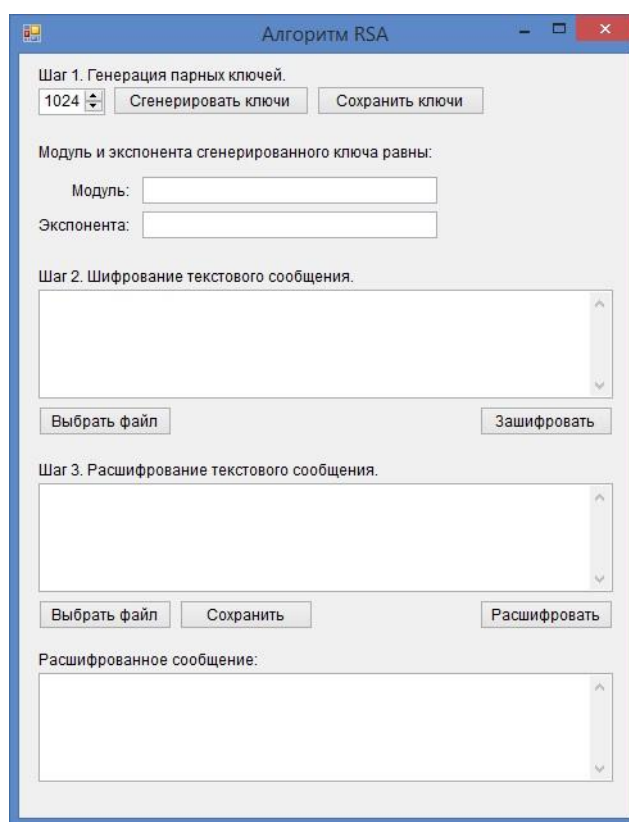


Рисунок 1 - Внешний вид программы

Первым делом для начала работы нужно сгенерировать пару ключей. Для этого на форме присутствует числовое поле, в которое необходимо ввести размер ключа в диапазоне от 384 до 2048 бит (см. рис. 2).

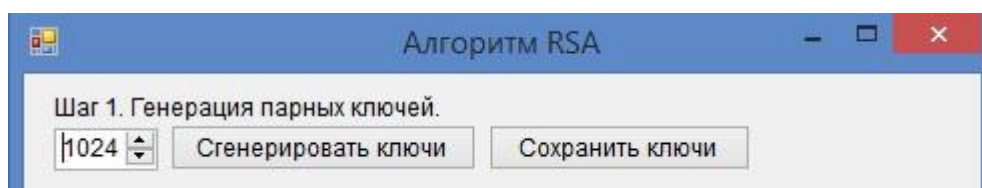


Рисунок 2 - Ввод размера ключа

Известно, чем больше размер ключа, тем выше будет криптостойкость системы. Определившись с размером, можно нажимать кнопку “Сгенерировать ключи”, после чего, чуть ниже, отобразятся значения модуля и экспоненты (см. рис. 3).

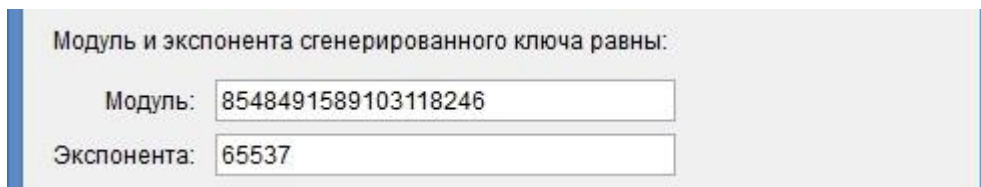


Рисунок 3 - Модуль и экспонента сгенерированного ключа

Для дальнейшей работы необходимо сохранить ключи в текстовом формате, нажав на соответствующую кнопку. Далее вводим сообщение, которое собираемся зашифровать, в текстовое поле, либо загружаем текст из файла (см. рис. 4).

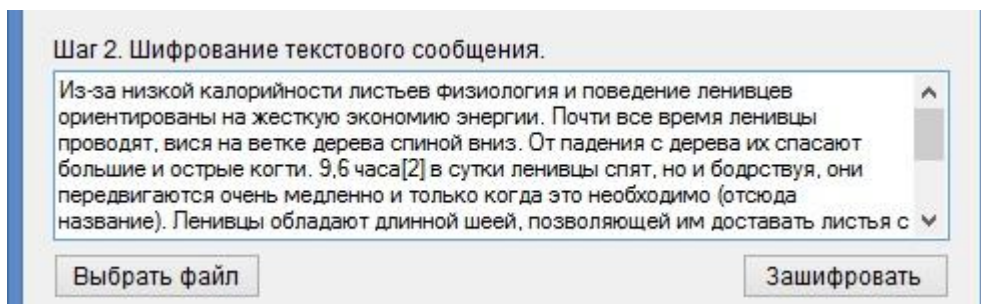


Рисунок 4 - Ввод шифруемого сообщения

Нажимаем кнопку “Зашифровать”, после чего в диалоговом окне выбираем открытый ключ, который был сохранен ранее. Если указан правильный ключ, то сообщение успешно зашифруется и выведется в другое текстовое поле, располагающееся ниже (см. рис. 5).

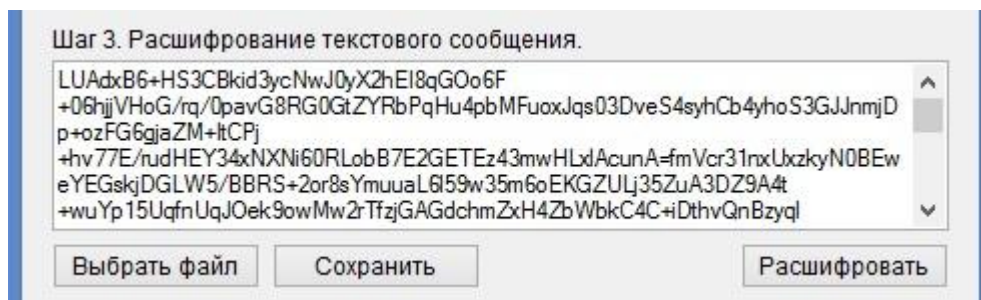


Рисунок 5 - Вывод зашифрованного сообщения

Если же будет выбран неправильный ключ или выбранный файл не будет являться ключом, то программа обработает исключение и выведет сообщение (см. рис. 6).

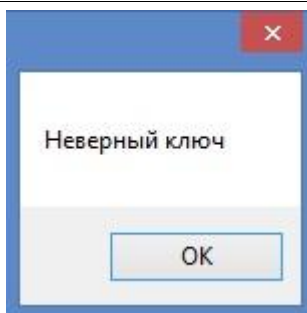


Рисунок 6 - Сообщение о неправильном ключе

Зашифрованное сообщение можно сохранить в текстовый файл для его последующего отправления получателю. После нажатия кнопки “Расшифровать” откроется диалоговое окно для открытия файла с секретным ключом, после успешной загрузки, которого расшифрованное сообщение выведется в соответствующее текстовое поле (см. рис. 7).

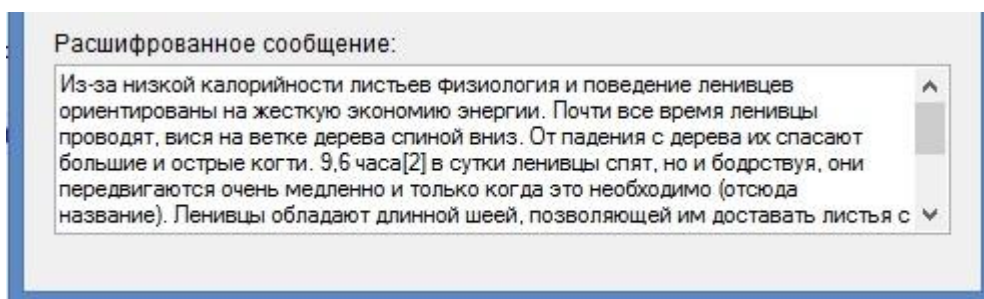


Рисунок 7 - Вывод расшифрованного сообщения

Стоит заметить, что длина текстового сообщения не может превышать длину ключа, поэтому в данном проекте была реализована разбивка текста на блоки, каждый из которых шифруется, а затем они, обратно склеиваются, что позволяет набирать текст неограниченного размера.

Для лучшего понимания как использовать данную систему, можно представить ситуацию, когда два человека, например, Иван и Мария хотят обменяться друг с другом текстовыми сообщениями. Иван запускает программу, генерирует пару ключей, которые затем сохраняет, и отправляет открытый ключ Марии. Мария с помощью программы и полученного открытого ключа шифрует какое-нибудь сообщение и отправляет его Ивану. Получив зашифрованное сообщение, Иван дешифрует его с помощью своего секретного ключа.

Максимальный размер когда-либо взломанного ключа на данный момент составляет 768 бит. Этот факт подтверждает криптостойкость алгоритма RSA. Единственный минус – низкая скорость шифрования в сравнении с симметричными алгоритмами шифрования. Для того чтобы устранить этот недостаток, многие разработчики создают гибридные криптосистемы, где данные шифруются с помощью симметричного ключа, а ключ в свою очередь уже шифруется с помощью алгоритма RSA. Криптосистема RSA используется во многих приложениях для шифрования и создания цифровых подписей таких как: PGP, TLS/SSL, IPSEC/IKE, S/MIME.

В результате работы было разработано приложение для шифрования текстовых данных с помощью алгоритма асимметричного шифрования RSA на языке программирования C# в среде Visual Studio. Программа позволяет шифровать данные с размером ключа до 2048 бит, что обеспечивает надежную защиту от злоумышленников. Данное программное обеспечение может стать основой для оборудования криптозащитой многих компаний, фирм, банков, государственных учреждений, где особенно важно сохранять конфиденциальность информации, а также для создания электронной подписи.

Библиографический список

1. Баженов Р.И. Информационная безопасность и защита информации: практикум. Биробиджан: Изд-во ГОУВПО «Дальневосточная государственная социально-гуманитарная академия», 2011. 140 с.
2. RSA – алгоритм шифрования с открытым ключом. Описание и реализация алгоритма RSA. // Собрание авторских статей (блог paveldvlip) URL: <http://www.paveldvlip.ru/algorithms/rsa.html> (дата обращения: 28.12.2015).
3. Public Key RSA Encryption in C# .NET // CodeProject. URL: <http://www.codeproject.com/Articles/10877/Public-Key-RSA-Encryption-in-C-NET> (дата обращения: 28.12.2015).
4. C# 4.0 и платформа .NET 4 для профессионалов / К. Нейгел, Б. Иввен, Дж. Глинн, К. Уотсон. Под ред. Ю.Н. Артеменко. СПб.: Диалектика, 2011. 1440 с.
5. Rivest R., Shamir A., Adleman L. A method for obtaining digital signatures and public-key cryptosystems // Communications of the ACM. 1978. Volume 21 Issue 2.
6. Public-Key Cryptography and the RSA Algorithm // College of Engineering - Purdue University URL: <https://engineering.purdue.edu/kak/compsec/NewLectures/Lecture12.pdf> (дата обращения: 28.12.2015).
7. The Mathematics of the RSA Public-Key Cryptosystem // Mathematics Awareness. URL: <http://www.mathaware.org/mam/06/Kaliski.pdf> (дата обращения: 28.12.2015).
8. RSA Algorithm with C# // C# Corner. URL: <http://www.c-sharpcorner.com/UploadFile/75a48f/rsa-algorithm-with-C-Sharp2/> (дата обращения: 28.12.2015).