УДК 004.056.5

А.Г. Проценко, И.В. Лысенко

Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков

ИССЛЕДОВАНИЕ БЫСТРОДЕЙСТВИЯ АЛГОРИТМОВ ШИФРОВАНИЯ ДАННЫХ НА БАЗЕ ТЕХНОЛОГИИ .NET FRAMEWORK

Разработана программа, позволяющая производить анализ быстродействия алгоритмов шифрования данных, реализованных на базе технологии .NET Framework, и проведено исследование стандартных реализаций криптоалгоритмов .NET.

Ключевые слова: *алгоритмы шифрования, быстродействие, конфиденциальность, криптопреобразования.*

Введение

Постановка задачи. В высокотехнологичном информационном обществе использование средств защиты информации является жизненно необходимым, поскольку это позволяет реализовать такие свойства защищаемых данных, как конфиденциальность, целостность, аутентичность, которые обеспечиваются с помощью методов криптографии.

Большинство задач защиты информации решаются специализированными программными средствами (ПС). Наряду с коммерческими ПС защиты информации, а также ПС, встраиваемыми в некоторые приложения, для выполнения специфических задач существует возможность использовать готовые программные реализации библиотек, реализующих криптографические алгоритмы защиты данных. Такие библиотеки не только предоставляют набор уже готовых криптоалгоритмов, но и определяют удобный интерфейс для расширения и модернизации уже существующих реализаций криптоалгоритмов и написания новых. Последний подход связан с тем, что большинство современных компаний стараются разработать универсальные криптографические интерфейсы и оградить разработчика программного обеспечения от самостоятельных реализаций сложных алгоритмов. Примером тому является интерфейсы Crypto API и Cryptography New Generation (CNG) семейства операционных систем Windows компании Microsoft, а встроенные средства технологии Microsoft .NET Framework предоставляют большой набор классов для осуществления различных криптопреобразований, позволяющих организовывать собственную систему криптозащиты данных [1].

Цель данной работы – провести исследование временных характеристик алгоритмов криптопреобразования, используемых для обеспечения *конфиденциальности* данных, на основе технологии .NET Framework. Исследованию алгоритмов обеспечения *целостности* данных планируется посвятить отдельную статью.

Методика проведения исследования

Выбор криптоалгоритмов шифрования данных осуществлялся исходя из возможностей библиотек System.Security.Cryptography каркаса .NET Framework. Были включены в анализ реализации симметричных алгоритмов шифрования Rijndael, DES, 3DES, RC2 и реализацию несимметричного алгоритма шифрования RSA. Для симметричных алгоритмов был проведен анализ их быстродействия с точки зрения шифрования документов различного объёма, а также – сравнительная характеристика различных симметричных алгоритмов.

Для несимметричных алгоритмов шифрования был проведён анализ быстродействия алгоритма RSA при обработке документов различного объёма при фиксированной длине модуля и с точки зрения обработки документов фиксированного объёма при разных значениях модуля.

Разработка программного обеспечения для выполнения криптопреобразований и исследования быстродействия алгоритмов шифрования

Базовые сведения о CryptoAPI и возможностях технологии .NET по реализации крипкриптоалгоритмов

Концепция CryptoAPI подразумевает сокрытие от программиста всех тонкостей и нюансов процесса шифрования данных. Работа этого API осуществляется через так называемые криптопровайдеры (CSP от англ. Cryptographic Service Provider). По своей сути они являются отдельными приложениями (DLL или вообще отдельные сервисы), которые написаны независимо от других приложений [2]. Они подписаны цифровой подписью, так что система время от времени может проверять их подлинность.

Следует отметить гибкость такого решения. Например, при смене криптопровайдра на более современный достаточно просто сменить название используемого криптопровайдера, поскольку весь основной интерфейс стандартизован.

Вся архитектура CryptoAPI может быть разделена три основные части:

– базовые функции;

– функции шифрования и для работы с сертификатами;

– функции для работы с сообщениями.

В группу базовых входят функции для выбора и подключения к криптопровайдеру, генерации и хранения ключей, обмена ключами. Сюда также входят возможности управления параметрами ключа, такими как: режим сцепления блоков, инициализационный вектор, а также так называемый salt value. На данный момент Microsoft поддерживает только CBC и ECB режимы сцепления блоков, хотя константы для режимов CFB и OFB уже заведены и вероятно, что в скором веремени эти режимы появятся.

В группу функций шифрования и работы с сертификатами входят все функции для хеширования данных, шифрования и расшифрования, а также функции для использования сертификатов, основной задачей которых является предоставление доступа к открытому ключу. CryptoAPI поддерживает сертификаты спецификации X.509, в которые входит информация о версии сертификата, его серийном номере, периоде действия, алгоритмах шифрования публичного ключа и др. Вообще, в CryptoAPI сертификаты представляют собой большую область, требующую отдельного изучения. Но их функции можно использовать уже при простом подписывании данных.

Под сообщениями в CryptoAPI понимаются данные в стандартизованном формате PKCS #7, разработанном RSA Laboratories. Функции для работы с ними делятся на две части: высокоуровневые функции (упрощенные) и низкоуровневые.

CryptoAPI предоставляет следующие стандартные криптопровайдеры:

– Microsoft Base Cryptographic Provider;

– Microsoft Strong Cryptographic Provider;

– Microsoft Enhanced Cryptographic Provider;

– Microsoft AES Cryptographic Provider;

– Microsoft DSS Cryptographic Provider;

– Microsoft Base DSS and Diffie-Hellman Cryptographic Provider;

– Microsoft DSS and Diffie-Hellman/Schannel Cryptographic Provider;

– Microsoft RSA/Schannel Cryptographic Provider.

Все эти CSP отличаются друг от друга своими типами, которые определяются набором параметров, включающим:

– алгоритм обмена сессионным (симметричным) ключом;

– алгоритм вычисление цифровой подписи;

– формат цифровой подписи;

– схема генерирования сессионного ключа по хешу;

– длина ключа.

На текущий момент CryptoAPI имеет 8 стандартных типов криптопровайдеров. Всех их можно разделить на две группы по алгоритму, используемому для вычисления цифровой подписи: RSA и DSS – и на три группы по алгоритму обмена сессионным ключом: RSA, DH (Diffie-Hellman) и KEY (Key Exchange Algorithm).

Windows Vista представляет новый интерфейс API криптографии, предназначенный для замены устаревшего интерфейса CryptoAPI, использовавшегося в ранних версиях Windows NT и Windows 9x [3]. Интерфейс CNG задуман в качестве долговременной замены для интерфейса CryptoAPI и предоставляет заменители для всех предлагавшихся им примитивов криптографии. CNG поддерживает все алгоритмы, предоставляемые интерфейсом CryptoAPI, но содержит также множество новых алгоритмов и является гораздо более гибким, что обеспечивает разработчикам больший контроль над способом выполнения криптографических операций и совместной работой алгоритмов при выполнении различных операций.

.NET Framework – программная технология от компании Microsoft, платформа для создания, развертывания и запуска Web-сервисов и приложений. Она предоставляет высокопроизводительную, основанную на стандартах, многоязыковую среду, которая позволяет интегрировать существующие приложения с приложениями и сервисами следующего поколения, а также решать задачи развертывания и использования интернет-приложений.

Microsoft .NET Framework включает богатый программный интерфейс CryptoAPI, предназначенный для решения широкого диапазона криптографических задач – таких как создание хешей различного типа (MD5, SHA1 и тому подобных) и реализации наиболее важных симметричных и асимметричных алгоритмов шифрования [4]. Также .NET Framework включает отдельные функции для защиты секретной информации на локальной машине или для каждого пользователя посредством полностью управляемых оболочек программного интерфейса Windows Data Protection API (DPAPI).

Классы криптографии в .NET делятся на три слоя. Первый слой – набор абстрактных классов; эти классы позволяют реализовать шифрование данных, обеспечивая таким образом их конфиденциальность. Сюда входят следующие:

– AssymetricalAlgorithm. Класс представляет ассиметричное шифрование, использующее пару ключей «открытый–секретный». Данные, зашифрованные одним ключом, могут быть расшифрованы только другим;

– SymmetricalAlgorithm. Класс представляет симметричное шифрование, использующее разделённое секретное значение. Данные, зашифрованные с ключом, могут быть расшифрованы только тем же самым ключом;

– HashAlgorithm. Этот класс представляет генерацию и верификацию хешей, которые известны также, как однонаправленные алгоритмы криптопреобразования.

Второй уровень включает классы, представляющие специфические алгоритмы шифрования. Они наследуются от базовых абстрактных классов и также являются абстрактными.

Классы третьего уровня представляют набор реализаций шифрования. Каждый из этих классов наследуется от класса алгоритма второго уровня. Это значит, что такой алгоритм шифрования, как DES,может иметь множество классов-реализаций. Хотя некоторые шифрующие классы .NET Framework реализованы целиком в управляемом коде, большинство всё же представляют тонкие оболочки вокруг библиотеки CryptoAPI. По сути, управляемые классы выполняют всю свою работу в мире .NET под надзором CLR, в то время как неуправляемые классы используют вызовы из неуправляемой библиотеки CryptoAPI.

На рис. 1 показаны классы криптографии .NET Framework. Более подробно структура классов криптографии .NET описана в работе [2].

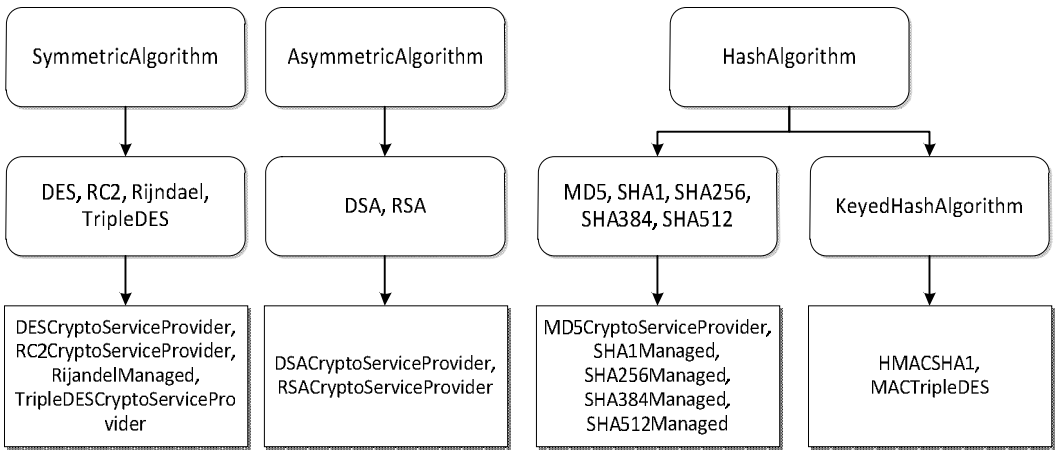

Рис. 1. Иерархия классов криптографии .NET Framework, базирующаяся на стандартном CryptoAPI для ОС Windows XP

CNG API предоставляет множество функций, которые выполняют основные криптографические операции, такие как создание хэшей или шифрование и дешифрование данных. Общая структура примитивов криптографии, которыми оперирует CNG, представлена на рис. 2. CNG реализует множество криптографических алгоритмов. Каждый алгоритм или класс алгоритмов предоставляет свои примитивные API. На одной машине может быть установлено несколько реализаций одного и того же алгоритма, однако, возможно использование только одной реализации по умолчанию в определённый момент времени.

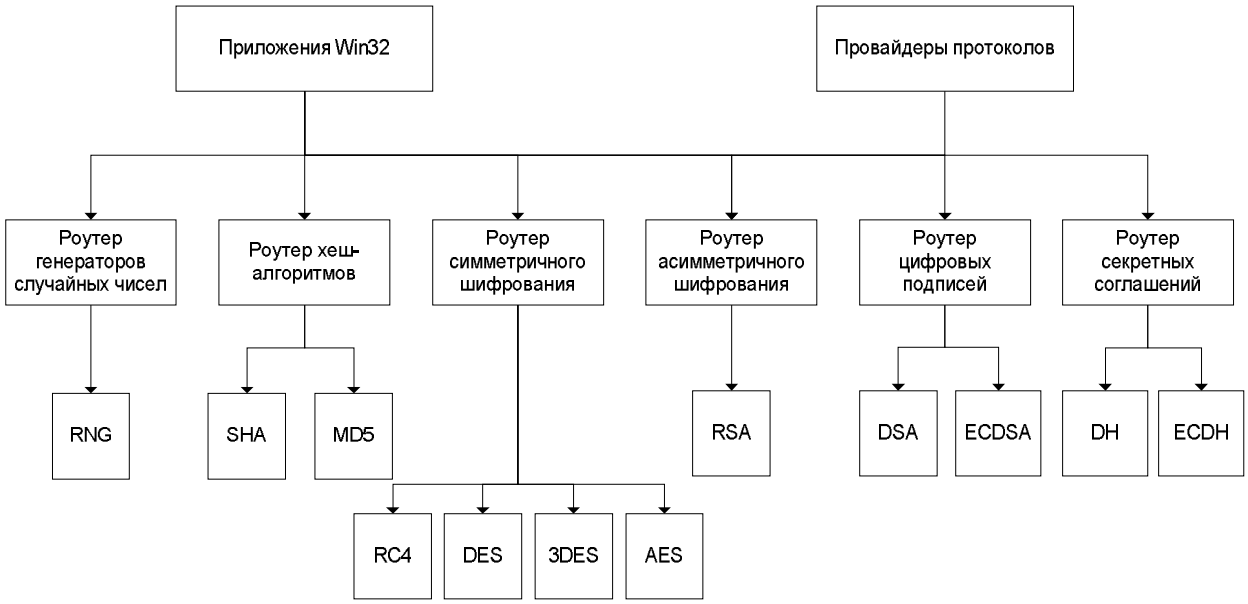

Рис. 2. Иерархия примитивов криптографии .NET Framework CNG для ОС Windows Vista и старше

Каждая реализация алгоритма криптографии в CNG представляется примитивным маршрутизатором. Приложения, использующие CNG, поддерживают связь с маршрутизатором через файлы библиотек – Bcrypt.dll в пользовательском режиме, или Ksecdd.sys в режиме ядра. Маршрутизатор предоставляет процедуры управления всеми примитивами алгоритмов. Основная задача маршрутизатора – отследить вызов определённой реализации алгоритма и передать вызов коду этой реализации.

Изложение и анализ результатов

В качестве характеристики времени, затрачиваемого на процесс шифрования, здесь и далее использовалась временная единица тик (tick). Microsoft в своём каркасе определяет тик, как наименьшую единицу измерения времени: 1 тик приравнивается к 100 наносекундам.

При выполнении одноразового анализа, на графиках видны резкие пики, обусловленные в основном распределением процессорного времени на системные нужды (ресурсы ОС и фоновых процессов). Так как криптопреобразования – ресурсоёмкая задача, доступность ресурсов системы, таких, как процессорное время и оперативная память, влияет на результат анализа. Для сглаживания результирующих данных производилось усреднение их значений по результатам десяти замеров.

В каркасе .NET Framework симметричный криптоалгоритм Rijndael представлен managed-реализацией RigndaelManaged. Из названия типа согласно классификации Microsoft следует, что реализация данного криптоалгоритма выполнена полностью средствами .NET Framework.

В исследовании рассматривались реализации криптоалгоритмов с размерами ключей 128, 192, 256 битов. Как видно из рис. 3, с ростом размерности ключа растёт и время, затрачиваемое на обработку документа. Причём с ростом объёма документа разница во времени между реализациями, использующими разные ключи, увеличивается незначительно.

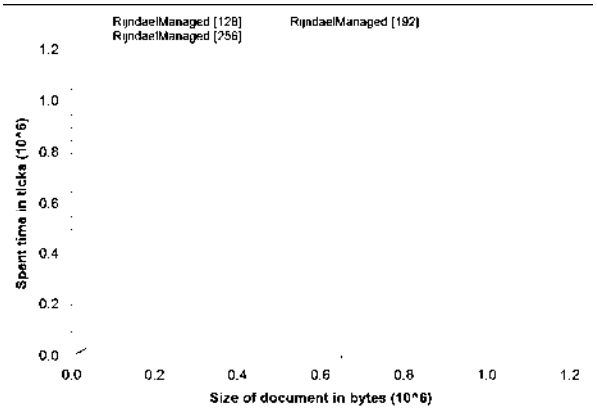

Рис. 3. Результаты анализа реализации криптоалгоритма RijandelManaged для документов разного объёма

Симметричный алгоритм RC2 представлен типом RC2CryptoServiceProvider. Данный тип является оболочкой для реализации алгоритма из библиотеки CryptoAPI. В каркасе .NET Framework для алгоритма RC2 допустимы длины ключа в диапазоне от 40 до 128 битов с шагом 8 битов. Для анализа использовались экземпляры класса провайдера с ключами 40, 64, 96 и 128 битов. На рис. 4 можно видеть, что длина ключа не влияет на производительность алгоритма. Все реализации показали примерно равные результаты. Как уже замечалось ранее, небольшие расхождения можно объяснить непостоянным распределением процессорного времени.

Симметричные алгоритмы DES и 3DES (Triple DES) представлены типами DESCryptoServiceProvider и TripleDESCryptoServiceProvider соответственно. Данные типы также являются оболочками и делегируют ответственность реализациям алгоритмов библиотеки CryptoAPI. Реализация алгоритма DES поддерживает длину ключа 64 бита, тогда как реализация 3DES позволяет задавать ключи длиной 128 бит (используются 2 разных ключа) и 192 бита (используются 3 разных ключа).

Как видно из рис. 5, реализация алгоритма 3DES выполняет шифрование данных приблизительно в 2,5–3 раза медленнее реализации алгоритма DES для документа того же объёма.

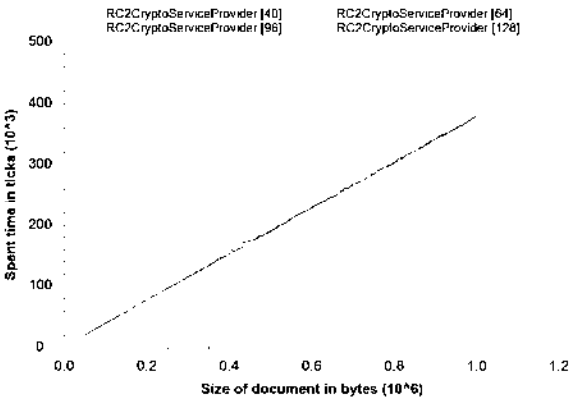

Рис. 4. Результаты анализа реализации криптоалгоритма RC2CryptoServiceProvider для документов разного объёма

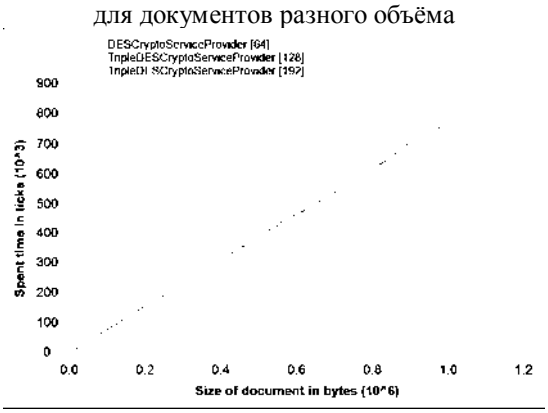

Рис. 5. Результаты анализа реализаций криптоалгоритмов DES и 3DES для документов разного объёма

Это объясняется тем, что по своей сути 3DES вызывает реализацию DES трижды. Следует также заметить, что время работы алгоритма 3DES не зависит от длины ключа.

На рис. 6 представлен результат сравнительного анализа всех симметричных алгоритмов, рассматривавшихся выше. Как видно из рисунка, наилучшее время криптопреобразования для всех размеров документов показал алгоритм DES. Однако, как известно, данный алгоритм обладает наименьшей криптостойкостью по сравнению с остальными рассматриваемыми симметричными криптоалгоритмами.

Алгоритму DES по времени криптопреобразования незначительно уступает алгоритм RC2, причём для разных объёмов документов разница во времени криптопреобразований увеличивается незначительно. Весомым преимуществом данного алгоритма является возможность в достаточно широком диапазоне варьировать длинами ключей, изменяя тем самым стойкость алгоритма. Алгоритм 3DES, как уже было сказано ранее, выполняет криптопреобразования в 2,5–3 раза медленнее, чем DES. Как преимущество, стойкость данного криптоалгоритма выше, чем у DES.

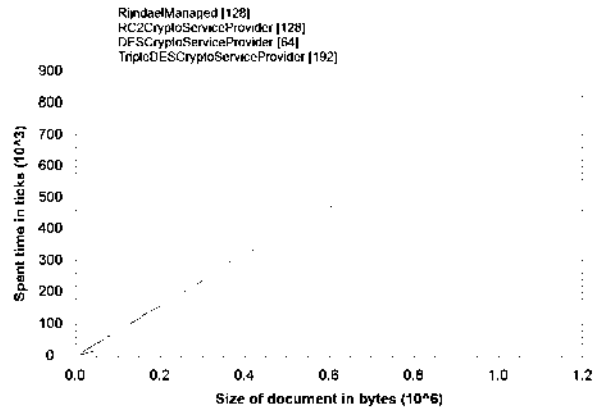

Рис. 6. Результаты анализа различных реализаций симметричных криптоалгоритмов для документов разного объёма

Алгоритм Rijndael, являющийся де-юре и де-факто стандартом блочного симметричного шифрования, показывает худшие результаты по времени криптопреобразований. Это можно объяснить тем, что Rijndael – единственный из представленных реализаций симметричных алгоритмов, реализованный полностью в managed-коде. С ростом объёма обрабатываемого документа разница во времени, затрачиваемом на криптопреобразования, только увеличивается. Время работы реализации алгоритма 3DES отличается незначительно.

В каркасе .NET Framework несимметричный алгоритм RSA представлен типом RSACryptoServiceProvider. Данный тип также является оболочкой для реализации в библиотеке CryptoAPI. В отличие от рассмотренных выше симметричных алгоритмов, реализация RSA не предусматривает криптопреобразований на документах большого объёма. Размер преобразуемого блока зависит от длины используемого ключа. Реализация RSA поддерживает ключи размером от 384 до 16384 битов. Данный класс провайдера позволяет выполнять не только операции шифрования и дешифрования, но и проверку цифровой подписи.

Результаты анализа реализаций, использующих ключи 512, 768, 1024, 2048 битов, для разных объёмов документов представлены на рис. 7.

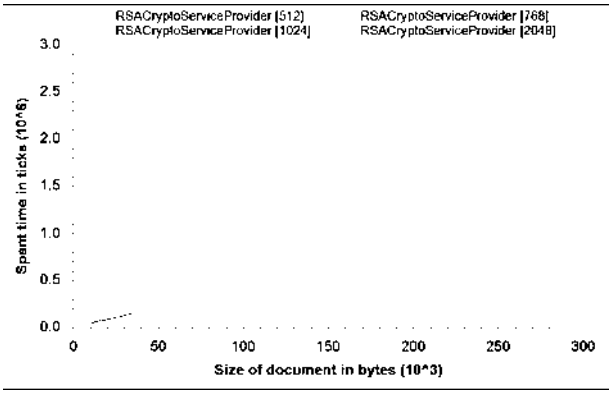

Рис. 7. Результаты анализа реализации криптоалгоритма RSACryptoServiceProvider для документов разного объёма

Реализация RSA, использующая ключ 2048 битов, работает значительно дольше, чем реализации, использующие менший по длине ключ. Причём разница во времени увеличивается с ростом объёма документа. Для реализаций с размером ключа, менее 1024 битов, разница увеличивается незначительно.

На рис. 8 представлены результаты анализа при фиксированном объёме документа (10^5 байт) и переменной длине ключа. Время криптопреобразований растёт линейно в зависимости от длины ключа, однако для алгоритмов, использующих ключи длиной свыше 3800 битов, наблюдается резкий рост времени криптопреобразования.

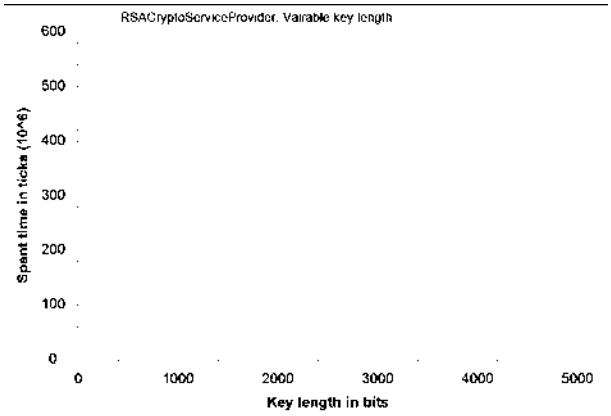

Рис. 8. Результаты анализа реализации криптоалгоритма RSACryptoServiceProvider для документа фиксированного объёма и переменной длины ключа

Также, как видно из рис. 8, увеличение длины модуля в 2 раза с 512 до 1024 битов приводит к увеличению времени криптопреобразования документа объёмом 10^5 байт с $8{,}05\cdot10^{-1}$ сек. до 1,64 сек (в 2 раза), в то время как увеличение длины модуля в 2 раза с 1024 до 2048 бит приводит к увеличению времени криптопреобразования до 4,77 сек. (в 3 раза), а с 2048 до 4096 битов – увеличивает время криптопреобразований до 46,23 сек. (почти в 10 раз).

Таким образом, можно видеть подтверждение того факта, что использование RSA для шифрования документов большого объёма нецелесообразно. В то же время, RSA, как известно, используется многими приложениями для постановки цифровой подписи (где длина шифруемого дайджеста сообщения, как правило, не превосходит нескольких сот битов) и шифрования сеансовых ключей симметричных криптоалгоритмов.

В табл. 1 приведены результаты сравнительного анализа симметричных алгоритмов и несимметричного алгоритма RSA. В каждой ячейке таблицы записано среднее время в секундах, которое требуется криптоалгоритму, указанному в строке таблицы, преобразовать документ размером, определённым в столбце таблицы. Как видно из таблицы, алгоритм RSA с длиной модуля в 512 битов работает в среднем на два порядка дольше, чем реализации симметричных алгоритмов в среднем. При этом увеличение длины модуля в 2 и 4 раза (1024 и 2048 битов соответственно) приводит к увеличению времени криптопреобразования примерно в 2 и в 4 раза соответственно.

Таблица 1

Результаты сравнительного анализа

Крипто-алгоритм	Объём документа, кбайт		
	20000	50000	100000
DES 64 bit	$1{,}10\cdot10^{-3}$	$2{,}03\cdot10^{-3}$	$3{,}80\cdot10^{-3}$
3DES 192 bit	$1{,}85\cdot10^{-3}$	$4{,}24\cdot10^{-3}$	$8{,}50\cdot10^{-3}$
RC2 128 bit	$1{,}35\cdot10^{-3}$	$2{,}60\cdot10^{-3}$	$4{,}72\cdot10^{-3}$
Rijndael 128 bit	$2{,}12\cdot10^{-3}$	$4{,}46\cdot10^{-3}$	$8{,}54\cdot10^{-3}$
RSA 512 bit	$1{,}25\cdot10^{-1}$	$3{,}08\cdot10^{-1}$	$6{,}14\cdot10^{-1}$
RSA 1024 bit	$2{,}51\cdot10^{-1}$	$6{,}31\cdot10^{-1}$	1,26
RSA 2048 bit	$6{,}51\cdot10^{-1}$	1,80	3,61

Стоит заметить, как было указано выше, дальнейшее увеличение размера ключа может привести к скачкообразному увеличению времени преобразований. Время работы алгоритма DES в среднем ниже в 2,5 раза, чем у алгоритма 3DES с максимально возможной длиной ключа 192 бита. Также можно отметить, что для всех рассмотренных алгоритмов шифрования время работы растёт пропорционально размеру документа.

Заключение

1. В ходе данной работы было разработано программное средство для исследования быстродействия алгоритмов шифрования данных на базе технологии Microsoft .NET Framework.

2. Наибольшим быстродействием обладает реализация алгоритма DES, однако из всех рассматриваемых криптоалгоритмов данный имеет наименьшую криптостойкость.

3. Быстродействие реализаций алгоритмов RC2 и 3DES не зависит от длины ключа.

4. Реализация алгоритма Rijndael работает значительно медленнее других рассмотренных реализаций симметричных криптоалгоритмов. Данный факт можно объяснить тем, что из всех рассмотренных алгоритмов Rijndael – единственный полностью реализованный на managed-коде класс.

5. Реализация алгоритма RSA показывает время работы, превышающее на 1 – 2 порядка время работы реализаций симметричных криптоалгоритмов. Данная реализация предназначена для работы с небольшими объёмами данных при небольших размерах ключей.

6. Разработанное программное средство может быть использовано при проведении лабораторных занятий для изучения алгоритмов шифрования данных по дисциплинам направления «Информационная безопасность».

Список литературы

1. Авдошин С.М. Криптотехнологии Microsoft / С.М. Авдошин, А.А. Савельева // Приложение к журналу «Информационные технологии». – 2008. – №9. – С. 23-30.

2. Нортроп Т. Разработка защищенных приложений на VisualBasic .NET и VisualC#.NET: Учебный курс Microsoft / Т. Нортроп. – М.: Русская Редакция, 2007. – 688 с.

3. Использование криптографии с помощью API CNG в Windows Vista. [Электронный ресурс]. – Режим доступа к ресурсу: http://msdn.microsoft.com/ru/magazine/cc163389.aspx.

4. Шнайер Б. Прикладная криптография: протоколы, алгоритмы, исходные тексты на языке Си / Б. Шнайер. – М.: Триумф, 2002. – 820 с.

Поступила в редколлегию 29.03.2011

Рецензент: д-р техн. наук, проф. В.С. Харченко, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков.

ДОСЛІДЖЕННЯ ШВИДКОДІЇ АЛГОРИТМІВ ШИФРУВАННЯ ДАНИХ НА БАЗІ ТЕХНОЛОГІЇ .NET FRAMEWORK

О.Г. Проценко, І.В. Лисенко

Розроблено програму, що дозволяє здійснювати аналіз швидкодії алгоритмів шифрування даних, що реалізовані на базі технології .NET Framework, і досліджені стандартні реалізації криптоалгоритмів .NET.

Ключові слова: *алгоритми шифрування, швидкодія, конфіденційність, криптоперетворення.*

TESTING OF THE PERFORMANCE OF DATA ENCRYPTION ALGORITHMS BASED ON .NET FRAMEWORK TECHNOLOGY

O.G. Protsenko, I.V. Lysenko

The application developed which allows to study performance of data encryption algorithms based on .NET Framework technology and performance of standard implementations of .NET algorithms is investigated.

Keywords: *encryption algorithm, performance, privacy, crypto converting.*