УДК 681.3
С.Н. Фисун, доцент, канд. техн. наук
Севастопольский национальный технический университет
А.И. Копылов
Севастопольский национальный технический университет
ул. Университетская д. 33, г. Севастополь, Украина, 99053
E-mail: fserg48@mail.ru

ИСПОЛЬЗОВАНИЕ КРИПТОГРАФИЧЕСКИХ СЕРВИСОВ .NET И JAVA ДЛЯ ЗАЩИТЫ ИНФОРМАЦИИ ОТ НЕСАНКЦИОНИРОВАННОГО ДОСТУПА

Рассматривается возможности криптографических сервисов, предоставляемых платформами .NET и Java. Данные сервисы можно рассматривать как альтернативу интерфейсу CryptoAPI корпорации Microsoft, который подробно рассматривается в статье [1]. Они позволяют использовать различные криптографические алгоритмы для шифрования данных. С использованием этих алгоритмов на языках программирования Java и C# была разработана программа шифрования и скрытия зашифрованной информации в файле.

***Ключевые слова**: защита информации, криптографические сервисы, алгоритмы шифрования.*

Необходимость использования криптографических систем на предприятиях и в финансовых институтах возрастает с каждым днем. Так, по данным отчета CSI/FBI Computer Crime and Security Survey 2009 [2], средний ущерб каждой компании, в которой в минувшем году была зафиксирована утечка конфиденциальных данных, составил $ 234,000.

Создание подобных систем требует реализации большого числа криптографических алгоритмов, функций хеширования, алгоритмов передачи ключей по открытым каналам связи, процедур идентификации и аутентификации пользователей, программ генерации и проверки электронных цифровых подписей, методов создания, хранения и уничтожения криптографических ключей [1].

Для решения таких задач корпорация Microsoft еще в 1996 г. ввела Cryptography API (CryptoAPI). Однако продолжающаяся эволюция средств защиты информации способствовала появлению новых криптографических сервисов, таких как пакеты **java.security, javax.crypto, javax.security** в Java и пространство имен System.Security.Cryptography общеязыковой исполняющей среды (Common Language Runtime, CLR) в Microsoft .NET.

В статье рассматривается применение криптографических сервисов, предоставляемых платформами Microsoft .NET и Oracle Java, для разработки программных систем шифрования/дешифрования данных. В качестве среды разработки (IDE) для Java будем использовать NetBeans 6.9.1 с пакетом Java Development Kit (JDK) 6 Update 20, а для программирования на C# – Microsoft Visual Studio 2008 с .NET Framework 3.5.

Под информационной безопасностью будем понимать состояние защищенности обрабатываемых, хранимых и передаваемых в информационной системе данных от несанкционированного ознакомления, преобразования и уничтожения, а также состояние защищенности информационных ресурсов от воздействий, направленных на нарушение их работоспособности.

Применение криптографических методов защиты обеспечивает решение основных задач информационной безопасности. Этого можно добиться путем реализации следующих криптографических методов защиты как пользовательской и служебной информации, так и информационных ресурсов в целом:

— шифрование всего информационного трафика, передающегося через открытые сети передачи данных, и отдельных сообщений;

— шифрование данных, представленных в виде файлов либо хранящихся в базе данных;

— контроль целостности программного обеспечения путем применения криптографически стойких контрольных сумм;

— применение электронно-цифровой подписи с целью обеспечения целостности и достоверности передаваемой информации.

Исходя из этого, представим базовую модель обеспечения безопасности в информационной системе в виде следующей структуры (рисунок 1):

Данная модель иллюстрирует концепцию безопасности информационной системы, с помощью которой предотвращается нежелательный доступ. Ядро модели — данные, которые требуется защитить от несанкционированного доступа. Для защиты данных используются специальные сервисы, предоставляемые Security Service Provider. В нашем случае это криптографические сервисы (шифрования, хеширования, генерации и проверки электронных цифровых подписей) предоставляемые платформами .NET и Java.

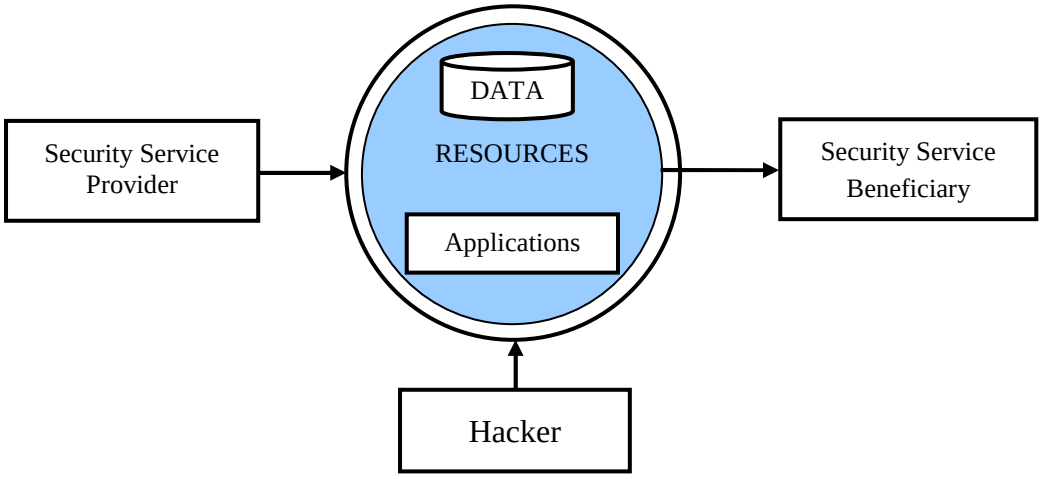

Рисунок 1 — Базовая модель безопасности

Под шифрованием будем понимать процесс преобразования открытых данных в зашифрованные по определенным правилам с применением ключей.

Основное назначение любого шифра — обеспечение возможности передачи сообщения по незащищенным каналам (не обязательно сетевым) с защитой этого сообщения от прочтения посторонними лицами.

Пусть есть две функции E(M1,K1) и D(M2,K2), зависящие от сообщений M1 и M2 и ключей K1 и K2, такие, что D(E(M,K1),K2)=M для некоторой пары ключей K1 и K2 и любого M, тогда E(.) и D(.) — функции шифрования и дешифрования, соответственно. Результат C=E(M,K1) называется криптограммой (или шифрограммой).

К функциям E(.) и D(.) предъявляются следующие требования:

— они должны быть легко вычислимы для любых M, если известны K1 и K2;

— вычисление D(M,?) — тяжелая задача при неизвестном ключе K2 (т.е., не зная ключа дешифрования, мы не можем вычислить исходное сообщение по криптограмме);

— вычисление ключей K1 и K2 — тяжелая задача при наличии некоторого набора пар {M,E(M,K1)} (имея набор криптограмм и исходных сообщений, мы не можем вычислить ключи);

— вычисление K2 (в случае K2 отличного от K1) при известном K1 — также должно быть трудной задачей (это требование относится к асимметричным шифрам. Для цифровой подписи K2 и K1 меняются ролями).

Надежность шифра определяется именно по его соответствию вышеперечисленным требованиям.

Если K1 = K2, то шифр называется симметричным, в противном случае — асимметричным. Примеры известных симметричных алгоритмов шифрования — DES, 3DES, AES, Blowfish, ГОСТ, асимметричных — RSA, ECC.

Шифры также бывают блочными и потоковыми. *Блочный шифр* работает с сообщениями фиксированного размера (например, 64 бита), а *поточный* — шифрует весь поток данных посимвольно (например, побайтно). Известные блочные шифры – DES, IDEA, Blowfish, потоковые — RC4. Блочность шифра не означает невозможность шифрования им сообщений, превышающих по длине размер блока.

Основное предназначение криптографических хешей — контроль подлинности данных путем вычисления от них некоторой функции H(.), дающей результат фиксированной (и обычно небольшой) длины. Функция H(.) должна удовлетворять следующим требованиям:

— для любых сообщений m, h = H(m) легко вычисляема;

— задача нахождения такого u (отличного от m), чтобы H(u) = h, должна являться трудной, при неизвестном m;

— задача нахождения такого u, что H(u) = H(m) является трудной при известном m.

Большинство популярных хеш-функций генерируют хеш длиной 128 бит и более. Примерами наиболее распространенных хеш-функций являются MD5 и SHA. Значения хеш-функций часто используются в системах электронной цифровой подписи для генерации дайджеста сообщения, который затем и подписывается тем или иным алгоритмом. Электронная цифровая подпись (ЭЦП) позволяет установить некую отметку, указывающую на принадлежность электронного сообщения конкретному автору.

Реализовать алгоритмы шифрования на языках C# и Java можно с использованием 2-х подходов:

1) разработка, отладка и тестирование программы по опубликованным криптографическим алгоритмам;

2) разработка программы с использованием криптографических сервисов, предоставляемых платформами .NET и Java.

Второй подход является более оптимальным, т.к. позволяет значительно сократить время разработки программы.

Для сравнения производительности предложенных подходов были дополнительно разработаны программные системы с использованием опубликованных криптографических алгоритмов (AES, DES, 3DES).

Тестирование производительности выполнялась на компьютере с процессором Intel Pentium Dual Core E2180, с объемом оперативной памяти 3 Гб. В качестве ОС для тестирования были выбраны операционные системы корпорации Microsoft — Windows XP Professional (с Service Pack 3) и Windows Seven Ultimate Edition.

Результаты тестирования сведены в таблицу 1. Скорости алгоритмов сгруппированы следующим образом. Первая группа ($I \in (0\text{ мс}; 100\text{ мс}]$) обладает самой высокой скоростью выполнения, далее следуют вторая ($II \in (100\text{ мс}; 200\text{ мс}]$) и третья ($III \in (200\text{ мс}; \infty)$) группы.

Таблица 1 — Выполнение шифрования и дешифрования на различных платформах

Платформа / Алгоритм	Windows Seven				Windows XP			
	C#	**Java**	**C# (API)**	**Java (API)**	**C#**	**Java**	**C# (API)**	**Java (API)**
AES128	II	II	I	II	I	II	I	I
AES192	II	III	II	II	II	II	I	II
AES256	III	III	II	III	III	III	II	II
DES	II	II	II	II	II	II	II	II
3DES	III	III	III	III	III	III	II	III

Алгоритм AES обеспечивает последовательно высокое выполнение для шифрования, дешифрования, хотя выполнение и понижается для 192- и 256-битных ключей.

Алгоритм DES также обеспечивает достаточно высокую скорость шифрования, дешифрования за счет простой структуры алгоритма и малой длинны ключа.

Что касается алгоритма Triple DES, то по скорости шифрования/дешифрования он является аутсайдером.

Из результатов тестирования видно, что оптимальной, с точки зрения производительности, программной платформой является платформа .NET и язык программирования C#.

Уровень производительности языка Java уступает платформе .NET, однако его можно рассматривать как инструмент для кроссплатформенной разработки.

Кроме того, из результатов тестирования видно, что подход с использованием криптографических сервисов, предоставляемых платформами .NET и Java является более производительным. Также данный подход позволяет значительно сократить время разработки ПО.

Криптографические сервисы в платформы Java представлены пакетами **java.security, javax.crypto** и **javax.security**. Общую структуру системы безопасности Java можно представить в следующем виде (рисунок 2):

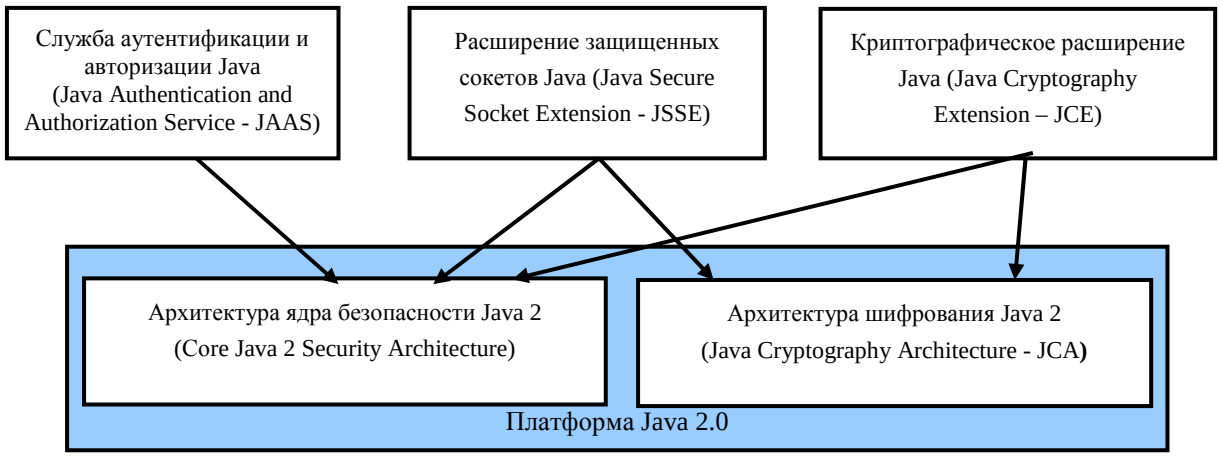

Рисунок 2 — Стандартные компоненты системы безопасности Java 2

Архитектура шифрования Java 2 (JCA) обеспечивает основные сервисы необходимые для шифрования на платформе Java. Вообще говоря, JCA является только интерфейсом, различные реализации которого могут быть использованы. В стандартной платформе применяется базовая реализация этого интерфейса. В состав JCA входит:

— основные классы и интерфейсы, формирующие набор сервис-провайдеров JCA и API криптографических операций;

— набор классов и интерфейсов управления сертификатами;

— набор классов для управления ключами.

Криптографический сервис-провайдер (Cryptographic Service Provider – CSP) представляет собой класс, реализующий одну или несколько криптографических функций, которые подчиняются интерфейсам, определенным в JCA.

В стандартном пакете поставки идет CSP, реализующей следующие алгоритмы и механизмы: MD5 и SHA-1, DSA, хранилище ключей JKS, производитель сертификатов по стандарту X.509 и алгоритм генерации псевдо случайных чисел SHA1PRNG (стандарт IEEE).

Американские законы запрещают экспорт некоторых видов криптографического ПО за пределы США и Канады или разрешают экспорт с урезанными ключами. Стандартные классы из JCA не попадают под это ограничение, поэтому они поставляются в составе платформы Java 2, а JCE поставляется отдельно. Наиболее известная открытая реализация совместимая с JCE это пакет Cryptix JCE, который включает в себя:

— шифры (Blowfish, CAST5, DES, IDEA, MARS, RC2, RC4, RC6, Rijndael, Serpent, SKIPJACK, Square, TripleDES, Twofish);

— протоколы обмена ключами (Диффи-Хелмана);

— методы шифрования (CBC, ECB, OFB);

— хэш-функции (MD2, MD4, MD5, RIPEMD-128, RIPEMD-160, SHA-0, SHA-1, Tiger);

— MAC-коды (HMAC-MD2, HMAC-MD4, HMAC-MD5, HMAC-RIPEMD-128, HMAC-RIPEMD-160, HMAC-SHA-0, HMAC-SHA-1, HMAC-Tiger);

— подписи (RawDSA, RSA);

— асимметричные шифры (ElGamal, RSA).

Расширение защищенных сокетов (JSSE) является стандартным интерфейсом Java для работы с SSL (Secure Socket Layer). SSL является связным протоколом для обеспечения безопасной связи на основе стека протоколов TCP/IP, хотя может быть реализован поверх любого надежного транспортного протокола. SSL поддерживает шифрование данных, поддержку целостности данных, возможность аутентификации, не анулируемости как клиента, так и сервера.

Служба аутентификации и авторизации Java (JAAS) обеспечивает стандартный способ ограничения доступа к ресурсам на основе аутентификации пользователей. Основные понятия данной службы приведены в [4].

В качестве примера рассмотрим реализацию алгоритма шифрования DES на языке Java с использованием Java Cryptography Extension (JCE).

Для этих целей в языке Java предусмотрен класс javax.crypto.Cipher, который реализует базовые функции популярных криптографических алгоритмов шифрования. Для создания экземпляра такого класса используется статистический метод Cipher.getInstance, который в качестве строкового параметра получает имя криптографического алгоритма шифрования, например, Cipher chr = Cipher.getInstance ("DES").

Для инициализация экземпляра класса и указания режима работы используется функция init:

— режим шифрования chr.init(Cipher.ENCRYPT_MODE, key);

— режим дешифрования chr.init(Cipher.DECRYPT_MODE, key).

Где параметр key – это 56 битный ключ алгоритма DES. Данный параметр имеет тип javax.crypto.SecretKey и может быть создан с помощью класса javax.crypto.KeyGenerator. Шифрование или дешифрование выполняет функция doFinal класса Cipher.

В платформе Microsoft .NET криптографические сервисы представлены пространством имен System.Security.Cryptography общеязыковой исполняющей среды (Common Language Runtime, CLR).

На самом высоком уровне пространство имен Cryptography можно разбить на четыре основные части:

— алгоритмы шифрования;

— вспомогательные классы;

— сертификаты X.509;

— цифровые подписи.

Главное предназначение этого пространства — предоставлять классы с алгоритмами таких операций, как шифрование и создание хэшей. Эти алгоритмы реализуются на основе расширяемого шаблона (pattern), включающего два уровня наследования. На вершине иерархии располагается

абстрактный базовый класс, имя которого соответствует типу алгоритма (Например: AsymmetricAlgorithm или HashAlgorithm). От такого класса наследует абстрактный класс второго уровня, предоставляющий открытый интерфейс для использования данного алгоритма. Например, SHA1 (Secure Hash Algorithm) представляет собой производный от HashAlgorithm класс и содержит методы и свойства, специфичные для алгоритма SHA1. Наконец, сама реализация алгоритма является производной от класса второго уровня; именно ее экземпляр создается и используется клиентским приложением.

К именам неуправляемых реализаций обычно добавляется суффикс «CryptoServiceProvider» (Например: SHAlCryptoServiceProvider), указывающий на то, что данная реализация на самом деле предоставляется провайдером криптографического сервиса (Cryptographic Service Provider, CSP), который установлен на уровне операционной системы и действует как оболочка CryptoAPI. В имена управляемых реализаций включается суффикс «Managed» (Например: SHAlManaged). Такие реализации не опираются на CryptoAPI и содержат исключительно управляемый код. Кроме того, стоит отметить, что при создании экземпляра одного из конкретных классов исходные конструкторы всегда записывают в параметры объекта разумные и безопасные значения (если это возможно). Так, алгоритмы асимметричного шифрования, опирающиеся на криптографию с открытым ключом, генерируют случайную пару ключей, а алгоритмы симметричного шифрования — случайный ключ и вектор инициализации (initialization vector, IV).

Второй основной набор классов в пространстве имен System.Security.Cryptography включает как классы, реально применяемые в процессе шифрования и расшифровки данных, так и разнообразные вспомогательные классы. Это пространство имен содержит, например, абстрактный класс RandomNumberGenerator, от которого наследуют классы RNGCryptoServiceProvider, ToBase64Transform и FromBase64Transform (используются при соответствующих преобразованиях данных).

Пространство имен Cryptography не только предоставляет алгоритмы шифрования, но и содержит дочернее пространство имен X509Certificates. Последнее объединяет всего три класса, предназначенных для операций с сертификатами Authenticode X.509 v.3. Класс X509Certificate предоставляет статические методы CreateFromCertFile и CreateFromSignedFile для создания экземпляра сертификата.

В пространстве имен Cryptography также присутствует дочернее пространство имен XML, используемое системой защиты .NET Framework для цифровой подписи XML. Эта спецификация описывает синтаксис, а также правила создания и представления цифровых подписей, которые могут быть применены к любому информационному наполнению.

В заключении рассмотрим созданную авторами программу шифрования и сокрытия зашифрованной информации в файле. Программа написана на языках C# и Java с использованием технологий соответствующих платформ. Для работы программы необходимо сначала задать алгоритм шифрования, а затем указать ключ шифрования. В качестве алгоритма шифрования может быть выбран один из следующих алгоритмов: DES, 3DES, AES с ключом 128, 192 или 256 бит. Если ключ задан не верно, то программа выводит соответствующее сообщение и пользователь не сможет зашифровать информацию до тех пор, пока не задаст корректный ключ. Для загрузки текстовой информации (для шифрования) в программу необходимо в диалоговом режиме определить имя файла, содержащего шифруемую информацию. В результате шифрования программа выводит зашифрованную информацию в текстовое поле интерфейса программы. Кроме того программа вычисляет время, в течении которого выполнялось шифрование информации. Это позволяет оценить производительность программы и выбранного алгоритма шифрования на рассмотренных платформах.

Использование криптографических сервисов, предоставляемых платформами .NET и Java уменьшает затраты времени на создание программных комплексов, предназначенных для защиты информации от несанкционированного доступа в информационных системах, обеспечивает повышение их надежности и быстродействия. На языках Java и C# были написаны программы, выполняющие шифрование и дешифрацию информации с использованием симметричных алгоритмов шифрования DES, 3DES и AES. Данные программы могут быть применены в задачах защиты информации.

В дальнейшем на основе разработанных приложений планируется выполнить сравнение производительности симметричных алгоритмов шифрования в ОС Linux. Объект исследования — среда разработки ПО на языке C# в ОС Linux — Mono. Предполагается что скорость шифрования / дешифрации в ПО, написанном на языке C#, значительно уменьшится, в то время как язык Java обеспечит высокий уровень производительности за счет полной совместимости с ОС Linux.

Библиографический список использованной литературы

1. Фисун С.Н. Использование криптодрайверов для защиты информации от несанкционированного доступа / С.Н. Фисун, О.И. Куржеевская // Зб. наук. пр. Севастопольського військово-морського Ордена Червоної Зірки інститута ім. П.С. Нахімова. — Севастополь, 2008. — Вип. 2 (15). — С. 29–33.

2. Richardson R. CSI/FBI Computer Crime and Security Survey 2009 // R. Richardson. — Computer Security Institute Publications, 2009. — 14 p.

3. Панасенко С.П. Алгоритмы шифрования. Специальный справочник / С.П. Панасенко. — СПб.: БХП-Петербург, 2009. — 76 с.

4. Петров А.А. Компьютерная безопасность. Криптографические методы защиты. / А.А. Петров. — М.: ДМК, 2000. — 448 с.

5. Хорстманн К. Java 2. Библиотека профессионала, том II. Тонкости программирования / К. Хорстманн, Г. Корнелл. — 7-е изд. — М.: Издательский дом «Вильямс», 2007. — 1168 с.

Поступила в редакцию 18.01.2011 г.

Фісун С.М., Копилов А.І. Використання криптографічних сервісів .NET та Java для захисту інформації від несанкціонованого доступу в інформаційній системі

Розглядається можливості криптографічних сервісів, що надаються платформами .NET та Java. Ці сервіси можна розглядати як альтернативу інтерфейсу CryptoAPI корпорації Microsoft, який докладно розглядається в статті [1]. Вони дозволяють використовувати різні криптографічні алгоритми для шифрування даних. З використанням цих алгоритмів на мовах програмування Java та C# була розроблена програма шифрування і приховування зашифрованої інформації у файлі.

Ключові слова: захист інформації, криптографічні сервіси, алгоритми шифрування.

Fisun S.N., Kopylov A.I. The use of cryptography .NET and Java to protect information from unauthorized access in information systems

We consider the possibility of cryptographic services provided by the platforms .NET and Java. These services can be considered as an alternative interface CryptoAPI from Microsoft, which is discussed in [1]. They allow the use of various cryptographic algorithms to encrypt/decrypt data. Using these algorithms, programming languages Java and C # we developed software to encrypt data and hide encrypted information in the file.

Keywords: information security, cryptographic services, encryption algorithms.