Клиентское приложение для доступа к зашифрованной информации в базе данных

Ю.А. Крыжановская
Факультет прикладной математики, информатики и механики
Воронежский государственный университет
Воронеж, Россия
e-mail: jak@mail.ru

Аннотация[1]

В работе описана реализация клиентского приложения, позволяющего осуществлять доступ к информации, хранимой в зашифрованном виде в базе данных. Реализация приложения выполнена на языке Java, также используется СУБД MySQL. Шифрование выполняется в соответствии со стандартом симметричного шифрования ГОСТ 28147-89.

1. Введение

В современном мире чаще всего базы данных являются ценными собраниями уязвимой информации. Они могут содержать личные данные, различного рода конфиденциальную информацию, например, обеспечивающую конкурентоспособность предприятия, интеллектуальную собственность и т.д. Потеря или кража таких данных может привести к потере репутации бренда, резкому снижению конкурентоспособности, штрафам и другим серьезным осложнениям. Следовательно, необходимо минимизировать возможность реализации указанных угроз, в частности, обеспечить, например, шифрование данных при хранении и передаче.

Рассматриваемое приложение для доступа к зашифрованной информации в базе предусматривает высокую степень защиты при относительно небольших системных требованиях к клиентскому устройству. Шифрование реализовано на основе «ГОСТ 28147-89 Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования» [1].

2. Решаемая задача

В рамках выполнения данной работы необходимо было разработать клиентское приложение, обеспечивающее возможность работы с зашифрованными данными, хранимыми в базе данных.

Труды третьей международной конференции "Интеллектуальные технологии обработки информации и управления", 10 - 12 ноября, Уфа, Россия, 2015

Создание приложения можно разбить на 4 основных части:

- реализация алгоритма шифрования ГОСТ 28147-89;
- реализация классов для работы с базами данных на языке программирования Java [2],
- разработака и реализация клиентского интерфейса с использованием технологии JavaFX [3];
- выполнить нагрузочное тестирование системы.

Конечная система должна иметь интуитивно понятный интерфейс, низкие системные требования и высокую скорость работы, уметь отображать изменения в базе данных в режиме реального времени и контролировать состояние подключения клиента к серверу базы данных.

2.1. Средства реализации

В качестве средств разработки были выбраны средой разработки NetBeans IDE 7.4, язык Java [2], располагающий необходимым функционалом для реализации поставленной задачи, графическая среда разработки JavaFX Scene Builder 2.0 [3], СУБД MySQL Server 5.6 [4], графический интерфейс проектирования баз данных MySQL Workbench 6.0 CE [5]. Соединение с базой данных осуществляется при помощи JDBC-драйвера, соответствующего СУБД MySql, предоставленного разработчиком СУБД..

2.2. Реализация алгоритма шифрования ГОСТ 28147-89

Для реализации шифрации базы данных был выбран стандарт ГОСТ 28147-89 в режиме гаммирования [1]. Гаммирование – это наложение (снятие) на открытые (зашифрованные) данные криптографической гаммы, то есть последовательности элементов данных, вырабатываемых с помощью некоторого криптографического алгоритма, для получения зашифрованных (открытых) данных. Для наложения гаммы при зашифровании и ее снятия при расшифровании должны использоваться взаимно обратные бинарные операции. В ГОСТ 28147-89 для

этой цели используется операция побитного сложения по модулю 2.

Гаммирование как режим шифрования имеет следующие особенности.

1. Одинаковые блоки в открытом массиве данных дадут при зашифровании различные блоки шифртекста, что позволит скрыть факт их идентичности.
2. Поскольку наложение гаммы выполняется побитно, шифрование неполного блока данных легко выполнимо как шифрование битов этого неполного блока, для чего используется соответствующие биты блока гаммы. Так, для зашифрования неполного блока в 1 бит можно использовать любой бит из блока гаммы.
3. Синхропосылка, использованная при зашифровании, каким-то образом должна быть передана для использования при расшифровании.

Для реализации криптографического алгоритма клиентской части используется язык программирования Java.

Основным классом реализации шифрования является класс dbencryption.gost89.Gost с методами encrypt (String text, String key, String message) и decrypt (String text, String key, String message).

Метод encrypt принимает на вход строку text – открытый текст, строку key – ключ шифрования, строку message – синхропосылка. На выходе метод возвращает зашифрованный текст.

Метод decrypt принимает на вход строку text – шифрованный текст, строку key – ключ шифрования, строку message – синхропосылка. На выходе метод возвращает расшифрованный (открытый) текст.

Реализация самого алгоритма шифрования инкапсулирована в классе dbencryption.gost89.GostImpl. В качестве таблицы замен использована таблица замен, используемая Центральным Банком РФ.

Основные методы класса GostImpl:

- getInstance() – возвращает экземпляр класса GostImpl;
- encrypt(String text, String key, String message) – основной метод зашифрования, принимающий на вход открытый текст, ключ, синхропосылку и возвращающий зашифрованный текст;
- decrypt(String text, String key, String message) - основной метод расшифрования, принимающий на вход шифрованный текст, ключ, синхропосылку и возвращающий открытый текст;
- extendKey(String key) – принимает на вход ключ, введённый пользователем и преобразовывает его в 32 байтный вид. В качестве расширения используется половина MD5 хэш-функции;
- extendMessage(String message) - принимает на вход синхропосылку, введённую пользователем и преобразовывает её в 32 байтный вид. В качестве расширения используется половина MD5 хэш-функции;
- gammaFunctionForEncryption(byte[] text, byte[][] keys, byte[] message) – гамма функция, используемая при зашифровании;
- gammaFunctionForDecryption(byte[] text, byte[][] keys, byte[] message) – гамма функция, используемая при расшифровании;
- stringKeyToByte(String key) – принимает ключ в формате строки и возвращает ключ в формате массива байт;
- mainEncryptCycle(String message, byte[][] keys) – основной цикл зашифрования;
- mainDecryptCycle(String message, byte[][] keys) – основной цикл расшифрования;
- mainStep(byte[] message, byte[] key) – основной шаг преобразования;
- sumMod2to32(byte[] n0, byte[] key) – функция (S0+C1)mod 232, описанная в 2.1 Шаг 2 данной работы;
- subBytes(String s1) – замена байт по таблице sTable;
- shiftBits(String s1) – сдвиг бит влево;
- byteTo8BitsFormat(byte b) – преобразовывает байт в 8 битный формат;
- bitsTo32Format(String bits) – дополнение набора бит до 32 путём добавления нулевых бит слева;
- binarToInt(String binar) – перевод числа из двоичного вида в целое десятичное число.

2.3. Реализация классов для работы с базами данных на Java

Соединение с базой данных в языке программирования Java осуществляется при помощи стандарта JDBC, реализованного в виде пакета java.sql, входящего в состав Java SE. JDBC драйверы для работы с конкретными типами СУБД предоставляются официальными разработчиками СУБД. В данной работе использована СУБД MySql и соответствующий JDBC драйвер.

В СУБД MySQL существуют стандартные функции для защиты информации, однако, все они реализуют зарубежные алгоритмы шифрования, поэтому было принято решение выполнить реализацию шифрования по российскому стандарту. Кроме того, все стандартные функции можно выполнить только

на уровне выполнения запросов, т.е. нет единой точки входа для работы с информацией в шифрованной БД.

Для упрощённой работы с БД были написаны 3 вспомогательных класса в пакете dbencryption.db:

1. DBConnection – класс, который реализует подключение к базе данных. Он имеет следующую структуру:
   - String dbName – имя базы данных;
   - String connectionName – имя соединения;
   - String hostname – адрес сервера;
   - String port – порт;
   - String username – имя пользователя;
   - String password – пароль;
   - методы для установки и получения вышеуказанных полей.
2. TableData – класс, описывающий структуру одной строки таблицы и является оболочкой класса java.util.HashMap, в которой ключом является имя колонки, а значением – значение колонки таблицы.
3. DBHelper – основной класс, позволяющий производить все необходимые манипуляции с базой данных. Он имеет следующие методы:
   - getInstance() – возвращает экземпляр класса DBHelper;
   - getConnection() – метод, возвращающий подключение к БД типа java.sql.Connection;
   - disconnect() – отключает соединение и освобождает ресурсы;
   - connect(DBConnection dbConn) – создаёт подключение к БД, используя dbConn;
   - executeQuery(String query) – отправляет SQL запрос на сервер и возвращает набор значений типа java.sql.ResultSet;
   - getPrimaryColumns(String dbName, String tableName) – возвращает список полей, которые составляют первичный ключ в базе с именем dbName и таблицы с именем tableName;
   - updateLine(String tableName, ArrayList<MutableTextField> line) – отправляет UPDATE запрос на сервер для обновления значений таблицы tableName по условию записи line;
   - insertLine(String tableName, ArrayList<MutableTextField> line) - отправляет INSERT запрос на сервер для добавления строки в таблицу tableName со значениями line;
   - deleteLine(String tableName, TableData data) – удаляет строку из таблицы tableName по условию записи data.

2.4. Реализация клиентского интерфейса

Клиентская часть представляет собой приложение с графическим интерфейсом, поддерживающее обработку пользовательского ввода данных и пользовательских действий.

Интерфейс клиентской части был разработан с помощью программы JavaFX Scene Builder. Было создано графическое окно MainPage.xml и обработчик событий dbencryption.MainPageController.java.

Класс MainPageController имеет следующие поля:

- public static final String fileName = System.getProperty("user.home")+File.separator+". DBEncription"+File.separator+"Connections.txt" – имя файла, в котором находится сохранённый список соединений;
- public static List<DBConnection> savedConnections – список сохранённых соединений;
- private DBConnection currConnection – текущее соединение;
- private static Stage NewWindow – новое окно, которое открывается при некоторых действиях;
- private String encrPassword – пароль для зашифрования/расшифрования;
- private String encrMessage – синхропосылка для зашифрования/расшифрования;
- private final List<String> columnNames – имена полей текущей таблицы/запроса;
- private final List<TableData> tableLines – все записи текущей таблицы или запроса;
- private final DBHelper dbHelper – экземпляр класса DBHelper для работы с БД;
- private String tableName – имя текущей таблицы;

В классе MainPageController есть следующие обработчики пользовательского ввода:

- showConnections(ActionEvent event) – открывает список возможных новых подключений;
- newMySQLConnection – открывает форму для создания нового подключения к СУБД MySQL;
- deleteConnection – позволяет удалить сохранённое подключение из списка;
- connectAction – подключается к выбранному сохранённому подключению и загружает список таблиц; при ошибке подключения появится

соответствующее сообщение об ошибке, при успешном подключении пользователь вводит пароль и синхропосылку для шифрования\расшифровки как показано на Рисунке 1:

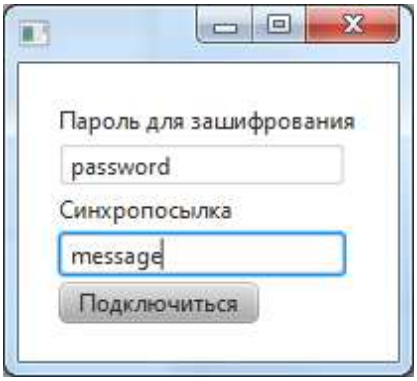

Рис. 1. Ввод пароля и синхропосылки для шифрования\расшифровки

- disconnectAction – удаляет текущее соединение и освобождает занятые ресурсы;
- executeQuery – обрабатывает пользовательский SELECT запрос и выводит результат в виде таблицы; при неверном запросе мы получим соответствующее сообщение об ошибке;
- viewTable – позволяет вывести первые 5000 записей выбранной таблицы;
- editRecord – позволяет редактировать выбранную запись;
- deleteRecord – метод, позволяющий удалить выбранную запись;
- addRecord – метод, позволяющий добавить запись, выбрав соответствующий пункт контекстного меню;
- initialize – метод-точка входа в программу., инициализирует главную страницу.

Вспомогательные классы расположены в пакете dbencryption.utils. Класс CommonUtils содержит методы, возвращающие запрос на выбор полей, составляющих первичный ключ по заданному имени БД и списку таблиц, и на получение списка таблиц для заданного имени БД. Класс MutableTextField extends TextField позволяет расширить функционал стандартного класса TextField путём добавления возможности проверки было поле изменено или нет.

После запуска приложения пользователь может создать новое подключение к базе данных, подключиться к базе, используя существующее сохранённое подключение, удалить существующее сохранённое подключение. Список сохранённых подключений загружается каждый раз при запуске программы и хранится в документах пользователя в папке «.DBEncription» в файле «Connections.txt». Также имеется возможность редактирования сохранённого подключения.

Далее после выбора подключения пользователю предлагается ввести пароль и синхропосылку для шифрования/расшифрования данных в базе. Плюсом подобного подхода является то, что пароль может быть уникальным в рамках пользователя или группы пользователей, т.е. после подключения пользователь будет видеть только данные, зашифрованные введённой парой пароль-синхропосылка.

Далее после успешного подключения загружается список таблиц для данной базы данных. Пользователь может выполнять запросы, вводя их в соответствующее поле, просмотреть первые 5000 записей выбранной таблицы.

Пользователь может выбрать любую запись и, выбрав из контекстного меню пункт «Редактировать запись», отредактировать выбранную запись.

Также есть возможность добавить записи в таблицу, нажав правой кнопкой мыши на название таблицы в списке таблиц и выбрав соответствующее контекстное меню. После выбора меню «Добавить запись» откроется новое окно, в котором можно ввести все необходимые поля записи. Есть возможность удаления записи, выбрав соответствующее контекстное меню, нажав на строку записи.

По завершении работы пользователь может выбрать в списке сохранённых подключений пункт контекстного меню «Отключиться». При этом делается коммит всех открытых транзакций и совершается отключение от базы данных. Или можно просто закрыть окно приложения, при этом совершится аналогичное действие.

2.5. Нагрузочное тестирование системы

Для проверки скорости работы использовался компьютер с операционной системой Windows 7, 4Гб оперативной памяти, и 4-х ядерным процессором с частотой 2.2 гигагерца. На этой же машине установлен сервер СУБД MySQL.

Загрузка программы после запуска занимает 0.05 секунды. Подключение к выбранной БД занимает 0.13 с. Далее была проверена скорость расшифрования на примере выборки записей из одной таблицы.

Ниже представлена таблица зависимости времени отображения информации от количества выбранных записей (Таблица 1).

Таблица 1.

Количество записей	Время отображения (сек)
10	0.114
100	0.225
500	1.194
1000	2.324

2000	3.143
5000	4.089

Сравним скорость отображения записей, используя те же запросы, используя программу MySQL Workbench [5], разработанную компанией Oracle и предоставляющую возможность доступа к БД без поддержки расшифровывания данных. Результаты приводятся ниже (Таблица 2).

Таблица 2.

Количество записей	Время отображения (сек)
10	0.015
100	0.092
500	0.163
1000	0.301
2000	0.527
5000	1.184

Как видно из таблиц временной зависимости, задержка отображения информации пользователю в разработанном приложении является незначительной и выражается только во времени расшифровки самих данных. Можно сделать вывод, что приложение является нагрузоустойчивым и может быть использовано для обеспечения быстрого и безопасного доступа к зашифрованной информации.

3. Заключение

В ходе выполнения работы была получены следующие результаты:

- разработана программа, которая предоставляет возможность доступа к зашифрованной информации в базе данных;
- реализован алгоритм шифрования ГОСТ 28147-89 на языке программирования Java.;
- реализована система для работы с СУБД MySQL с применением технологии JDBC драйверов;
- реализован графический интерфейс с использованием технологии JavaFX.;
- проведены тесты нагрузоустойчивости, которые показали, что программа быстро справляется с большими объёмами данных.

В результате проделанной работы получена готовая программа, позволяющая манипулировать шифрованными данными в базе данных. Конечная программа удовлетворяет всем предъявленным требованиям, располагает всеми необходимыми возможностями для администрирования шифрованных баз данных и интуитивно понятным интерфейсом.

Данную работу можно применить для обеспечения безопасного хранения и использования информации на сервере БД. Так как использован российский стандарт шифрования ГОСТ 28147-89, программу можно использовать, в том числе и для обеспечения безопасного хранения информации в государственных учреждениях, на предприятиях, в учебных заведениях.

Список используемых источников

1. ГОСТ 28147-89. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования – М. : Госстандарт России: Изд-во стандартов, 1989. – 28 с.
2. Документация Java. «Подробнее о технологии Java» [текст]. – (URL: http://www.java.com/ru/about/) (дата обращения 29.09.2015).
3. «JavaFX Scene Builder» [электронный ресурс]. – (URL: http://www.oracle.com/technetwork/java/javase/downloads/javafxscenebuilder-info-2157684.html) (дата обращения 29.09.2015).
4. «MySQL» [электронный ресурс]. – (URL: http://dev.mysql.com/downloads/mysql) (дата обращения 29.09.2015).
5. «MySQL Workbench 6.0» [электронный ресурс]. – (URL: http://dev.mysql.com/downloads/workbench/6.0.htm) (дата обращения 29.09.2015)