Математические структуры и моделирование
2000, вып. 6, с. 139–142

УДК 681.3.053

ШИФРОВАНИЕ ДАННЫХ В DBF-ФАЙЛАХ

В.В. Ворошилов, О.Л. Епанчинцева

The technology of coding database files based on the RC5-algorithm is given

Некоторые современные СУБД, например Microsoft Access, обеспечены встроенными средствами защиты данных, а некоторые, не менее популярные, например Visual FoxPro, их не имеют. Проблемы защиты информации в базах данных становятся особенно актуальными по мере развития сети организации. Не все базы данных размещены на выделенном сервере. В силу разных причин пользователь продолжает работать с данными локально, но наличие одноранговой Windows-сети и недостаточная квалификация пользователя делают возможным просмотр DBF-файлов внешними программами, минуя предусмотренное в разработанном приложении разграничение доступа.

В частности, подобная проблема защиты данных встала при разработке в среде Visual FoxPro 5.0 программы автоматической генерации тестов для вступительных испытаний. Таблицы Visual FoxPro хранятся в незащищенном формате, что позволяет без особого труда просмотреть все записи внешними программами, например Far Manager.

В качестве средства защиты предлагается шифрование полей с конфиденциальной информацией.

Но информация хранится не только в полях таблиц. Если осуществляется индексирование данных по некоторому полю, то информация из ВСЕХ записей этого поля также в незащищенном виде будет храниться в индексном файле. Применить шифрование к индексному файлу невозможно, поскольку порядок зашифрованных полей отличен от порядка исходных. Поэтому необходимо создавать индексы непосредственно перед работой, как и устанавливать связи между таблицами. Нужно учитывать ограничение на тип данных, а именно: можно шифровать только символьные поля.

Предложенный вариант защиты не позволяет восстановить данные после случайного или преднамеренного изменения кода шифрованного поля. Поэтому рекомендуется хранить копию зашифрованной базы данных.

Остается нерешенной и проблема получения несанкционированного доступа при некорректном выходе из приложения (например при сбое питания), поскольку система не сможет определить факт завершения работы для шифрования информации.

© **2000 В.В. Ворошилов, О.Л. Епанчинцева**

E-mail: m81voros@math.omsu.omskreg.ru, eol@univer.omsk.su
Омский государственный университет

Так как в Visual FoxPro отсутствуют какие бы то ни было процедуры шифрования информации, в первую очередь было необходимо создать библиотеку криптографических функций, на которой в дальнейшем можно строить всю систему шифрования. Для основы данной библиотеки рассматривались следующие криптоалгоритмы с симметричным ключом:

- Blowfish
- DES
- ГОСТ 281547-89
- IDEA
- RC5

Таблица 1. Описание алгоритмов шифрования

Название алгоритма	Описание
Blowfish	Разработан: в 1993 году Брюсом Шнайнером. Размер блока: 64б. Размер ключа: 32-448б. Патент: не запатентован. Особенности: Непригоден для использования в системах, где ключ часто меняется и на каждом ключе шифруются небольшие по объему данные. Алгоритм лучше всего подходит для систем, в которых на одном и том же ключе шифруются большие массивы данных.
DES (Data Encryption Format). Федеральный стандарт шифрования США.	Разработан: в 1972-1975 годах в исследовательской лаборатории корпорации IBM. В качестве федерального стандарта США принят в 1977 году. Размер блока: 64б. Размер ключа: 56б (64б, 8 не используются). Патент: не запатентован. Особенности: Алгоритм очень сложен в плане программной реализации. Длина ключа (56б) явно недостаточна для обеспечения приемлемой стойкости, что неоднократно было доказано перебором ключа на спецпроцессорах или в распределенной сети.
ГОСТ 281547-89. Стандарт Российской Федерации на шифрование и имитозащиту данных.	Разработан: предположительно во второй половине 70-х годов в бывшем 8-м Главном Управлении КГБ СССР. Авторы не известны. Размер блока: 64б. Размер ключа: 256б. Патент: не запатентован. Особенности: Алгоритм оптимизирован для программной реализациии и значительно превосходит DES в криптостойкости.

IDEA (International Decryption-Encryption Algorithm)	Разработан 1990-1991 г. Цюрих, Щвейцария. Авторы: Xuejia Lai, James Masse. Размер блока: 64б. Размер ключа: 128б. Патент: Компания Аском (Ascom), Берн, Швейцария. Запатентован в США (патент №5214703, действует с 16.05.1991 по 25.05.2010), Европе (патент №0482154, действует с 16.05.1991 по 16.05.2011) и Японии (патент №508119/1991, действует с 16.05.1991). Особенности: целиком «арифметический» алгоритм. Оптимизирован для 16-разрядных процессоров с быстрой командой умножения.
RC5	Разработан: Роном Ривестом для RSA laboratories. Размер блока: 32, 64, 128. Размер ключа: переменный. Патент: не запатентован. Особенности: При вводе ключа вводятся также количество проходов, размер блока и т.д. RC5 с размером блока 64 бита и 12 или более проходов обеспечивает хорошую стойкость против дифференциального и линейного криптоанализов. Алгоритм параметричен , используется блочный шифр.

Таблица 1 подготовлена по материалам сайта Андрея Винокурова [1] и сайта компьютерного клуба Запорожского Института экономики и информационных технологий [2].

Критериями отбора алгоритма являлись:

1. Надежность шифрования.
2. Скорость.
3. Простота реализации.
4. Возможность коммерческого использования.

Из рассмотренных алгоритмов **Blowfish** не пригоден для систем, где на каждом ключе шифруются небольшие по объему данные, поэтому отброшен. **DES** довольно сложен для программной реализации, к тому же, был неоднократно взломан. Алгоритм **IDEA** запатентован и, таким образом, не доступен для коммерческой реализации. **ГОСТ 281547-89** отброшен, т. к. менее устойчив и более сложен по сравнению с RC5. Наиболее удачным оказался для реализации алгоритм **RC5**, как наиболее стойкий к взлому и простой в реализации.

На основе выбранного криптоалгоритма была написана (на C++ с ассемблеровской вставкой для повышения производительности) динамическая библиотека crypto.dll, содержащая в себе единственную функцию шифрования / дешифрования, параметрами которой являются шифруемая строка, строка с паролем, их длины и флаг, определяющий необходимую операцию (0-шифрование, 1-дешифрование).

Библиотека написана на Microsoft Visual C++ 6.0. Для повышения производительности главный цикл шифрования был оформлен в виде вставки на языке Assembler.

Таблица 2. Оценка скорости работы библиотеки

Вариант алгоритма	Шифрование(сек.)	Дешифрование(сек.)
C++ со вставкой на Assembler	5.503	5.756
Только C++	9.103	10.567

Необходимо отметить несколько побочных эффектов, связанных с реализацией в виде внешней dll библиотеки:

1) строка передается в процедуру по ссылке;
2) длина шифруемой строки должна быть кратна 8.

На длину пароля никаких ограничений не накладывается.

Пример:

```
declare rc5_32new in crypto.dll
STRING string, INTEGER stlen,;
STRING password, INTEGER paslen,
INTEGER flag
String='1234567812345678'
Password='87654321'
rc5_32new(@string,LEN(string),password,LEN(password),0)
clear dlls
```

или

```
declare rc5_32 in crypto.dll
STRING string, INTEGER stlen,;
STRING password, INTEGER paslen,;
INTEGER flag
String='1234567812345678'
Password='87654321'
result=encrypt(string,password)
string=decrypt(result,password)
clear dlls
procedure encrypt
   parameters string,password
   rc5_32new(@string,LEN(string),password,LEN(password),0)
   return string
procedure decrypt
  parameters string, password
  rc5_32new(@string,LEN(string),password,LEN(password),1)
  return string
```

Оценка скорости работы алгоритма проводилась на таблице из 10000 записей с длиной шифруемого поля 248 символов, размещенной на локальном диске компьютера с процессором Celeron (400 МГц) и объемом памяти 64Мб.

Как видно из таблицы, процесс шифрования/дешифрования не замедляет работу пользователя, обеспечивет необходимую конфиденциальность данных.

Литература

1. Винокуров А. – `http://www.halyava.ru/crypto/`
2. Компьютерный клуб Запорожского Института экономики и информационных технологий. – `http://www.ecomon.zp.ua/crypto/`