Analysis and Design on Security of SQLite*

Liu Haiyan, Gong Yaowan

Department of Information Engineering Armored Force Engineering Institute, Beijing, China
lhy940@163.com, gogogyw@gmail.com

Abstract - This paper is to resolve the security problem of SQLite. The paper first analyzes some security methods commonly used in Database Area, as well as the current mechanism of SQLite. Then discusses the authentication, access control, encryption, audit, as well as the backup and restore mechanism of SQLite. At last, this paper combines authentication and encryption together, with the help of those interfaces prepared by SQLite, it implements an encrypting SQLite by adding codes of deriving keys, encryption and loging.

Index Terms - Database, Security, Enryption, SQLite.

1. Introduction

SQLite is an open source database system for small scale applications, which provides most support for standard SQL92, and has many advantages, such as high storage efficiency, fast query operation, small memory requested and so on. It has been widely used in embedded systems. Since SQLite is open source, it can be ported on to different operating systems easily. It's codes can also be modified to meet specific functional requirements.

Original SQLite basically does not provide any security control mechanisms. It uses a single file to store the entire content of the database. If only someone gets the database file, he can get the data stored just by using the SQLite tool, or even by using a generic text editor, e.g. notepad on Windows. To those systems who need security protections, this feature becomes the fatal defect of SQLite[1,2]. Basing on the analysis of the general database security mechanisms, the paper designs some security mechanisms for SQLite, which enhances its security while maintains its original advantages.

2. Security mechanism of database

A database is usually used to store various important information of a company, organization or a government department, so the security of database is critical to an information system. Whether for a single host application or a network application, a database may have to face various security threats[3,4]. The security mechanisms of database are those technical mehods adopted to protect the database and to prevent the illegal users from unauthorized access, stealing, changing or destroying the data stored.

2.1 some commonly usedsecurity mechanism of database

Nowadays, the mainstream databases all adopt some security measures. In addition to the essential mechanisms to guarantee data consistency and integrity, they usually adopt some other measures[3,4,5], such as access control, data encryption, authenticaiton and audition, etc.

1) Authentication

The user or application provides its own identity, and the database authenticates the validity of the user or application. Only legitimate users or applicaions can access the data in the database. User authentication is the base of all other security mechanisms, only after the authenticaion, the certification and audition can be carried out.

2) Access control

Access control of a database management system means to grant different permissions to different users, and to ensure that a user can only access the data authorized to him. Currently, some large-scale databases, such as Oracle, SQLServer, etc. adopt the role based access control mechanism, i.e. the system assigns different roles to users, such as db_owner, dbcreator, securityadmin,etc. and the database grant diffferent authority according to the roles, a user can perform operations only granted to the roles which he belongs to.

3) Encryption

The authentication and the access control mechanism are carriced out by the database management system database systems(DBMS), they achieve the access control mechanism of database. However they lack the effective protection to the stored data. For example, if the attacker make use of the vulnerability of the operating system or database, or by physical contact to the database system, they can steal or see the data by operate on the database file. To encrypt the data stored in the database is to protect the data against such threats.

4) Audition

Audition of a database is to monitor and record the user operaitons on the database, these records are used for post analysis of database to find out the attacker clues.

5) Backup and recovery

Backup and recovery of the database is used when the database is unrecoverable failed. First the database is backup regularly, then the database is restored to a former consistent state by used of this backup.

2.2 security mechanism of SQLite

The transaction processing of SQLite is quite reliable, it can always keep the consistency and integrity of data even when the system break down or power off. However, compared with some largescale database systems, its consideration on security is rather rare.

SQLite is not the kind of database which is based on service, so the database itself has no access control and

* This work is partially supported by 2012CJ061.

authorization mechanism. Since it stores the whole database in a single file, it relies on the file access control mechanism of the operating systems, that is to say, all the legitimate users of the operating system can access the database, as long as they have the read or write permission on the database file. If you can access the SQLite database file, you can use the program called sqlite3.exe, or use any text editor, such as notepad.exe, to view the content of the database.

In the source code of SQLite, there is interface reserved for encryption, while it does not provide any encryption implementations. For embedded systems, depending on the operating system and file access control mechanism is the first defense of SQLite, while encryption is the more critical defense mechanism.

SQLite does not provide any audit mechanism either. The backup and restore of the database file have to rely on manual copy of the database file.

3. Design the security machanism for SQLite

3.1 Level of the security in database

There are three ways to access a SQLite database.

1) Writing a program which calles the API functions to access the database.

2) Using the tools provided by SQLite such as sqlite.exe or other tools to access a SQLite database.

3) use any other file reading or writing tools, such as GEDIT on Linux, Notepad on Windows to edit the database file directly.

According to the structure of database as well as the applcations of database, the implementation of security mechanism in database can be achieved on two levels.

1) in the database level, i.e. to increase security control in the codes of the Database Management System (DBMS).

2) in the application level, i.e. to add security control in applications who use the database. For this kind of methods, all of the three access ways of the database must be considered, otherwise only partial access methods add security control, and other methods can not use the data correctly anymore.

These two implementation levels have their own advantages and disadvantages. The former needs to modify the database source code, while the latter needs to modify the application layer codes. For SQLite, these two schema can both be implemented, user can chose any one schema or combine them together according to the specific needs.

3.2 Security mechanism for SQLite

Security of SQLite can be enhanced in several ways, such as access control, data encryption, audit, backup and restore, etc. Next this paper will illustrate how these methods work.

1) access control

A SQLite database is an ordinary file, so access to it must be permitted by the file access control mechanism first. Further more, authentication can be added to the DBMS. A user or an application must provide correct password when he want to access the database. Only after the authenticaion, the database can be accessed, e.g. created, queried, modified, inserted, deleted, modified, and so on.

2) data encryption

There are two data encryption schema can be implemented for SQLite. One is to encrypt on the DBMS level, i.e. perform encryption or decryption while reading from the database or writing to the database. Another scheme belongs to the application layer, where encryption or decrypton can be operated on some fields of the records, while what the DBMS faces are cipher fields encrypted.

Compared with the second sheme, the first scheme can provide stronger encryption, where the encryption function is embed into the DBMS, and the encryption and decryption process is transparent to users. Nervertheless, this method increases the load of the DBMS, and the source code of DBMS must be modified. The latter method needs the application program to encrypt the data before writing data to the database, and decrypt data after reading from the database, thus increasing the burden of application programs.

The SQLite source code reserves encryption interface, such as sqlite3_key, sqlite3_rekey, etc. which can provide DBMS level encryption. There have been some encryption implementations on SQLite[1,2], but these are based on earlier versions of SQLite, while SQLite develops very fast. At present the latest version of SQLite is sqlite3.6.17. The functions and data structures have changed a lot which makes up of early defects and make functions more perfect.

3) audit mechanism

Audit mechanism of SQLite can be implemented with the logging mechanism provided by the operating system. For example, on Linux system, the syslog system call can be used to log inportant operations. Audit mechanism in DBMS can also be implemented in application layer. In DBMS, applcation program interfaces funcitons(API) can be provided to log important operations. Either of these two methods needs to modify the source code of SQLite and enable the multithread options at the same time.

4) backup and recovery mechanism of SQLite

SQLite uses a single file to store the complete content of the database, so the backup and recovery can easily be realize just by file copy.

In this paper, authentication, encryption and audit mechanism are all realized by modifying the source code SQLite in DBMS level. The authentication and encryption mechanism are combine together into the unity of the encryption mechanism, i.e. not only the data are encrypted but also the database structure is encrypted. Password must be provided before any access to the database. If the password is not correct, then the structure of the database can not be correctly understood, neither does it allow for the follow-up access. As for the backup and retore mechanism, API functions are defined to backup and to restore a specified database.The following paragraph will illustrate how the encryption mechanism is implemented in detail.

4. Implementation of encryption in SQLite

4.1 Interface of encryption in SQLite

In the SQLite source code, there has been encryption interface reserved which defines the prototypes for encryption related functions but without any implementation codes. The main interface is make up of several functions. In file sqlite3.h, there are function prototypes namely sqlite3_key and sqlite3_rekey, the former is used to specify the key of a database, and the latter is used to reset the key. In file attach.c, there are some encryption related functions, for example, function sqlite3CodecGetKey is used to get the currrent key of a database, function sqlite3CodecAttach is used to relate the key to the database.

To implement the encryption operation, all read and write operations must be connected with the encrypt and decrypt operations, the funtion sqlite3pager_set_codec is defined for this task. The prototype of sqlite3pager_set_codec is defined as follows.

```
void sqlite3pager_set_codec( Pager *pPager,
        void *(*xCodec)(void*,void*,Pgno,int),
        void *pCodecArg)
{
  pPager->xCodec = xCodec;
  pPager->pCodecArg = pCodecArg;
} ;
```

This funciton associates the coding funtions with each page, thus the coding function is executed automatically when reading and write a page.

It can be concluded from above analysis, to implement the encryption funtion, some functions must be implemented, e.g. sqlite3_key, sqlite3_rekey, sqlite3CodecGetKey, sqlite3CodecAttach as well as the coding function.

4.2 Encryption algorithm

A SQLite database file is stored where a page is the minimum unit, page size can be customized. When encrypt a database, the encryption algorithm should fit for the page size. As if well known, encryption algorithms are classified into block encryption algorithm and sequence encryption algorithm. In block algorithm, plain texts are divided into fixed length blocks, a block is encrypted or decrypted together. In sequence algorithm, the key is a sequence generated, each byte (or bit) of the plaintext is encrypted with the the corresponding byte in the sequence. If a block encryption algorithm is used, and if the plaintext is not of integer times of the block size, then the plaintext must to be filled with random data. So the ciphertext may be larger than the original plaintext, and the original page may not be large enough to store the encrypted data, and thus may need to adjust the stored page. In this paper, we choose the sequence encryption algorithm RC4, which avoids the page adjustment problem of block algorithms.

4.3 Key generation

Since the password input by users is not of fixed length, and from the security point of view, if password input is used directly as the encryption key, then the key may be revealed from the database. Therefore in this paper, a MD5 operation is exectuted on the input password, and the abstraction computed is used as the source of the RC4 key.

4.4 implementation

Basing on the source codes of SQLite3.6.17, We implements the function of authenticaion, data encryption, simple loging mechanism, backup and restore functions with standard C language. The architecture of the encryption source can be shown in Fig. 1, where the arrow indicates the relaitonship among some important funcitons.

4.5 Testing the encryption function

New database system must be tested to see if SQLite modified as above works in proper way. The test items include 1) the original application programs can run correctly on the new database system. 2) the encryption function of the new database is correct. 3)logging mechanism,backup and restore function is correct. Since encryption is the main functions added to the database, next will describe the testing process of the encryption funciton.

First write an application program which accesses a SQLite database. The program is very simple and easy to get its meaning. It Open a database named test2.db, create a table named user, and insert two records to the table. The source code of this program is as follows.

```
main(){
    sqlite3 *db;
    char *zErrMsg=0;
    sqlite3_open("test2.db",&db);
    if(db==NULL)
        return -1;
    sqlite3_exec(db,"create table user (name varchar(64)
                UNIQUE, age text);",0,0,&zErrMsg);
    sqlite3_exec(db,"insert into user values('Li Nana ',
                '20');",0,0,&zErrMsg);
    sqlite3_exec(db,"insert into user values('Wang Haihu',
                '10');",0,0,&zErrMsg);
    sqlite3_free(zErrMsg);
    sqlite3_close(db);
}
```

Since this program is simple and the program is correct, there is no error dealing operations in the program. In real applications, statements on checking errors must be inserted after each statement accessing the database.

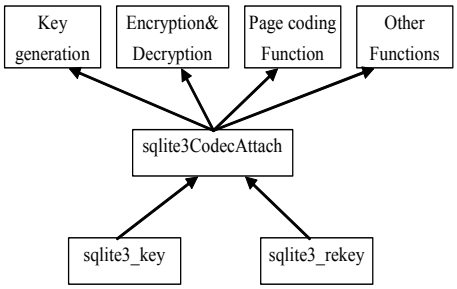

Fig. 1 Architecture of the encryption source code

Fig. 2 is the content of file test2.db when opening by notepad. It can be seen that the table structure and the data of the records all can be read. So when a database is not encrypted, one can get the content by access the database file directly.

Then modify the above program, insert one line after the statement sqlite3_open("test2.db",&db):

Sqlite3_key(db, “123456”,6);

The meaning of this statement is that to encypt the database by key “123456”, and the length of the key is 6.

Fig. 3 is the content of file test2.db when opening by notepad after it is modified. As can be seen that the content is mess up, no one can get the content of the database, thus it keeps the secret of the data.

Now write a new application program, which can read the data from the database after it is encrypted. The content of this program is as follows.

```
int SelectTable( void * para, int n_column, char ** column_value, char **column_name );
void main(){
    sqlite3 *db;
    char *zErrMsg=0;
    sqlite3_open("test2.db",&db);
    if(db==NULL)
        return;
    sqlite3_key(db,"123456",6);
    sqlite3_exec( db, "select * from user",
                  SelectTable, NULL, &zErrMsg);
    printf("errmsg is:%s",zErrMsg);
    sqlite3_free(zErrMsg);
    sqlite3_close(db);
}
int SelectTable( void * para, int n_column,
          char ** column_value, char **column_name )
{
    printf(" cloname=%s, colvalue=%s,cloname=%s,
       colvalue=%s\n", *column_name, *column_value,
       *(column_name+1), *(column_value+1) );
    printf( "----------\n" );
    return 0;
}
```

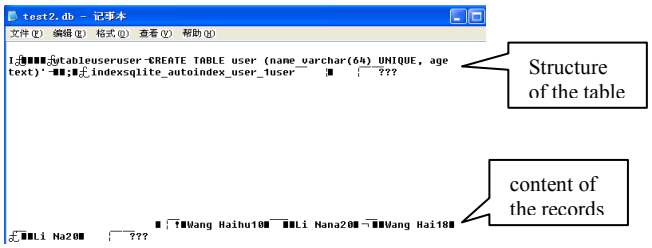

Fig. 2 database file before encryption

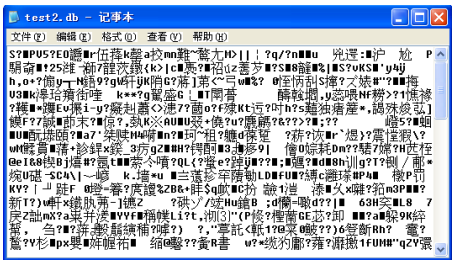

Fig. 3 database file after encryption

where statement sqlite3_key(db,"123456",6) provides the key for the encrypted database. Function SelectTable is a callback funciton who takes charge of reading data from the database one record after another.

The result of the program is shown in Fig. 4. So if the password is correct, then the program can get the original data before encryption.

If the password is not matched, for example if the statement is modified to:

sqlite3_key(db,"111111", 6);

then the statement accessing the database will fail, the erro message is “file is encrypted or is not a database” , as shown in Fig. 5.

So the encryption implemented in this paper can provide the funcion of data encryption and authentication.

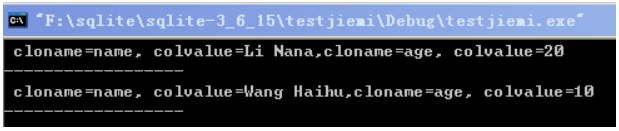

Fig. 4 result of the program when password is matched

Fig. 5 error messages when password is not matched

5. Conclusion

A database is a centralized structure to store data for information systems, where data security is critical. Basing on the analysis of the current database security mechanisms, this paper design security mechanisms for SQLite, illustrate how to modify the source code to enhance its security. The modifed SQLite inherits the advantages of the original system, and promote its security by authentication, encryption, logging, backup and restore mechanisms.

References

[1] Zhao Yuehua and Zhu Weiling, “Design and realization of database encrypt module based on SQLite”, Computer Engineering and Design, Vol.29, No.16, pp.4132-4134, August 2008.

[2] Liao Shunhe and Le Jiajin, “Analysis and research of the encryption method for SQLite”, Computer Applications and Software, Vol. 25, No.10, pp.70-72, October 2008.

[3] Tian Xiuxia, Wang Xiaoling, Gao Ming and Zhou Aoying, “Database as a Service–Security and Privacy Preserving”, Journal of Software, Vol.21, No.5, pp.991-1006, May 2010.

[4] Jiao Yan, “With Regard to the Status of the Database System Security Research”, Computer Security, 2010(5), pp.45-47, May 2010.

[5] Wen Xubo and Bai Haijuan. “Security tactic of Oracle database”, Information Technology, 2009(8), pp.91-93, August 2009.