SQLite Encryption - Technical Solution

Objective

The objective for this project is to update sqlite3 to be able to read / write to encrypted databases in realtime as it is already available in closed source extensions like SEE (https://www.sqlite.org/see/doc/trunk/www/readme.wiki)

Technical Solution

The technical solution is to change the functions that are pointed to, in the sqlite3 struct below, which controls all io operations on sqlite3, they are located

```
struct sqlite3_io_methods {
  int iVersion;
  int (*xClose)(sqlite3_file*);
  int (*xRead)(sqlite3_file*, void*, int iAmt, sqlite3_int64 iOfst);
  int (*xWrite)(sqlite3_file*, const void*, int iAmt, sqlite3_int64 iOfst);
  int (*xTruncate)(sqlite3_file*, sqlite3_int64 size);
  int (*xSync)(sqlite3_file*, int flags);
  int (*xFileSize)(sqlite3_file*, sqlite3_int64 *pSize);
  int (*xLock)(sqlite3_file*, int);
  int (*xUnlock)(sqlite3_file*, int);
  int (*xCheckReservedLock)(sqlite3_file*, int *pResOut);
  int (*xFileControl)(sqlite3_file*, int op, void *pArg);
  int (*xSectorSize)(sqlite3_file*);
  int (*xDeviceCharacteristics)(sqlite3_file*);
  /* Methods above are valid for version 1 */
  int (*xShmMap)(sqlite3_file*, int iPg, int pgsz, int, void volatile**);
  int (*xShmLock)(sqlite3_file*, int offset, int n, int flags);
  void (*xShmBarrier)(sqlite3_file*);
  int (*xShmUnmap)(sqlite3_file*, int deleteFlag);
  /* Methods above are valid for version 2 */
  int (*xFetch)(sqlite3_file*, sqlite3_int64 iOfst, int iAmt, void **pp);
  int (*xUnfetch)(sqlite3_file*, sqlite3_int64 iOfst, void *p);
  /* Methods above are valid for version 3 */
  /* Additional methods may be added in future releases */
};
```

in the file sqlite3.c file on the SQLite.NET.2015 project (https://system.data.sqlite.org/index.html/doc/trunk/www/index.wiki), and then we will point to functions that will provide the original functionality, but also using encryption methods before each read/write.

The encryption will work at **Page Level**, so each page (defaults to 4k) will be individually encrypted, ***actually the Page encryption is the only modification to be done on the functions above***, in windows the function pointers point to:

```
static const sqlite3_io_methods winIoMethod = {
  3,                              /* iVersion */
  winClose,                       /* xClose */
  winRead,                        /* xRead */
  winWrite,                       /* xWrite */
  winTruncate,                    /* xTruncate */
  winSync,                        /* xSync */
  winFileSize,                    /* xFileSize */
  winLock,                        /* xLock */
  winUnlock,                      /* xUnlock */
  winCheckReservedLock,           /* xCheckReservedLock */
  winFileControl,                 /* xFileControl */
  winSectorSize,                  /* xSectorSize */
  winDeviceCharacteristics,       /* xDeviceCharacteristics */
  winShmMap,                      /* xShmMap */
  winShmLock,                     /* xShmLock */
  winShmBarrier,                  /* xShmBarrier */
  winShmUnmap,                    /* xShmUnmap */
  winFetch,                       /* xFetch */
  winUnfetch                      /* xUnfetch */
};
```

A new pragma statement (ENCRYPT_METHOD) will be added on the function:

```
SQLITE_PRIVATE void sqlite3Pragma(
  Parse *pParse,
  Token *pId1,        /* First part of [schema.]id field */
  Token *pId2,        /* Second part of [schema.]id field, or NULL */
  Token *pValue,      /* Token for <value>, or NULL */
  int minusFlag       /* True if a '-' sign preceded <value> */
){
```

that will control the encryption method to be used, and another pragma statement (ENCRYPT_KEY) will be added to set the key to be used on the specified file.

Crypto Library

I used many years ago on the university a crypto library that worked very well for windows:

```
https://github.com/weidai11/cryptopp
```

I have successfully compiled it on Visual Studio 2015 and ran the unit tests, it creates a static library that will be added to the sqlite3 DLL build process,

Unit Tests

The plan is to run the sqlite3 unit tests that come with the sqlite3 solution, and verify if the resulting file is encrypted or not, comparing the binary file generated with the ENCRYPT_METHOD set with the binary file generated with such pragma unset.