SQLite 加密 -- SQLCipher

SQLite3 插件 github 下载地址

插件配置步骤地址

购买地址

其他加密方式介绍

SQLCipher API 地址

前言

应用使用 SQLite 来存储数据，很多时候需要对一部分的数据进行加密。常见的做法是对要存储的内容加密后存到数据库中，使用的时候对数据进行解密。这样就会有大量的性能消耗在数据的加密解密上。

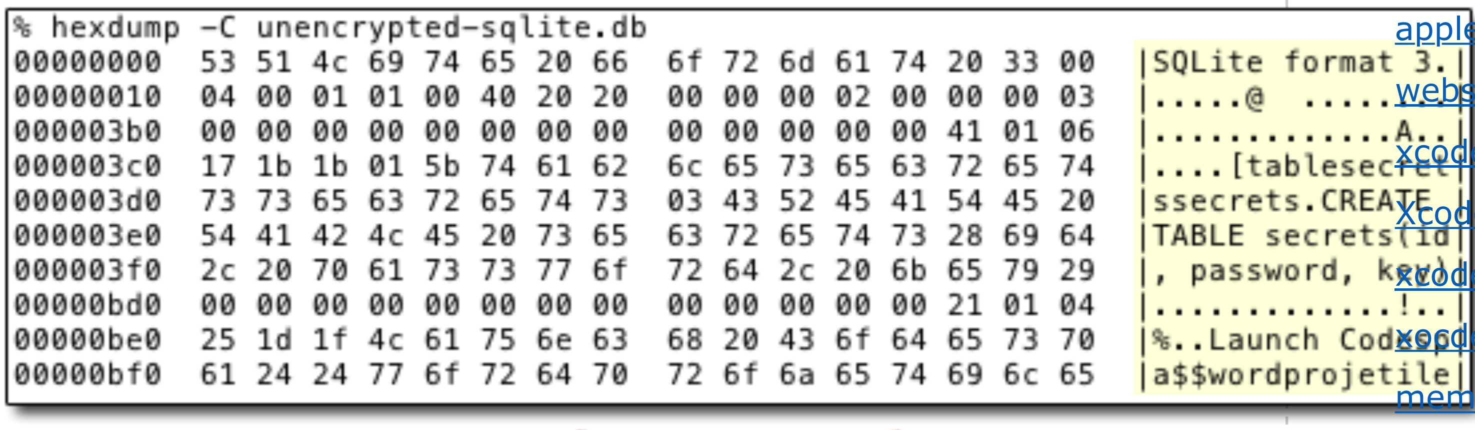
SQLite 本身是支持加密功能的 (免费版本不提供加密功能，商业版本是支持加密模块)。SQLCipher 是一个开源的 SQLite 加密的扩展，支持对 db 文件进行 256位的 AES 加密。

加密与非加密的数据库对比

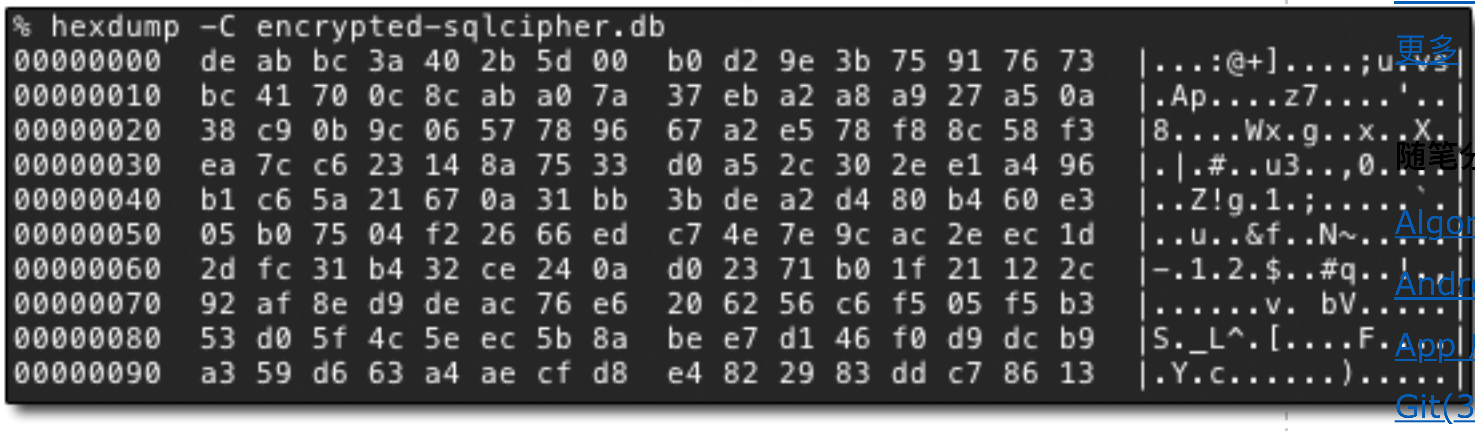
打开 Terminal 输入以下内容，

```
~ $ sqlite3 sqlcipher.db
sqlite> PRAGMA KEY=’test123’;
sqlite> CREATE TABLE t1(a,b);
sqlite> INSERT INTO t1(a,b) VALUES (‘one for the money’,
sqlite> .quit

~$ hexdump -C sqlite.db
```

结果：



```
% hexdump -C unencrypted-sqlite.db
00000000  53 51 4c 69 74 65 20 66  6f 72 6d 61 74 20 33 00  |SQLite format 3.|
00000010  04 00 01 01 00 40 20 20  00 00 00 02 00 00 00 03  |.....@  ........|
000003b0  00 00 00 00 00 00 00 00  00 00 00 00 00 41 01 06  |.............A..|
000003c0  17 1b 1b 01 5b 74 61 62  6c 65 73 65 63 72 65 74  |....[tablesecret|
000003d0  73 73 65 63 72 65 74 73  03 43 52 45 41 54 45 20  |ssecrets.CREATE |
000003e0  54 41 42 4c 45 20 73 65  63 72 65 74 73 28 69 64  |TABLE secrets(id|
000003f0  2c 20 70 61 73 73 77 6f  72 64 2c 20 6b 65 79 29  |, password, key)|
00000bd0  00 00 00 00 00 00 00 00  00 00 00 00 00 21 01 04  |.............!..|
00000be0  25 1d 1f 4c 61 75 6e 63  68 20 43 6f 64 65 73 70  |%..Launch Codesp|
00000bf0  61 24 24 77 6f 72 64 70  72 6f 6a 65 74 69 6c 65  |a$$wordprojetile|
```



```
% hexdump -C encrypted-sqlcipher.db
00000000  de ab bc 3a 40 2b 5d 00  b0 d2 9e 3b 75 91 76 73  |...:@+]....;u.vs|
00000010  bc 41 70 0c 8c ab a0 7a  37 eb a2 a8 a9 27 a5 0a  |.Ap....z7....'..|
00000020  38 c9 0b 9c 06 57 78 96  67 a2 e5 78 f8 8c 58 f3  |8....Wx.g..x..X.|
00000030  ea 7c c6 23 14 8a 75 33  d0 a5 2c 30 2e e1 a4 96  |.|.#..u3..,0....|
00000040  b1 c6 5a 21 67 0a 31 bb  3b de a2 d4 80 b4 60 e3  |..Z!g.1.;.....`.|
00000050  05 b0 75 04 f2 26 66 ed  c7 4e 7e 9c ac 2e ec 1d  |..u..&f..N~.....|
00000060  2d fc 31 b4 32 ce 24 0a  d0 23 71 b0 1f 21 12 2c  |-.1.2.$..#q..!.,|
00000070  92 af 8e d9 de ac 76 e6  20 62 56 c6 f5 05 f5 b3  |......v. bV.....|
00000080  53 d0 5f 4c 5e ec 5b 8a  be e7 d1 46 f0 d9 dc b9  |S._L^.[....F....|
00000090  a3 59 d6 63 a4 ae cf d8  e4 82 29 83 dd c7 86 13  |.Y.c......).....|
```

导航

博客园 首页 联系 订阅 管理

<			2017年7月			>
日	一	二	三	四	五	六
25	26	27	28	29	30	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

公告

昵称：EileenLeung
园龄：3年7个月
粉丝：3
关注：1
+加关注

统计

随笔 - 38 文章 - 0 评论 - 7

搜索

找找看

谷歌搜索

常用链接

我的随笔

我的评论

我的参与

最新评论

我的标签

我的标签

Mac osx(4)

applescript(3)

website(2)

xcode(2)

Xcode error warning(1)

xcode shortcuts(1)

xcode search path(1)

memory_management(1)

NSURLSession(1)

others(1)

更多

随笔分类

Algorithm

Android 基础

App启动(1)

Git(3)

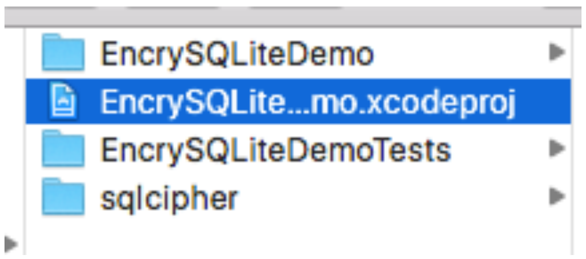
HTML && CSS

HTML5学习过程

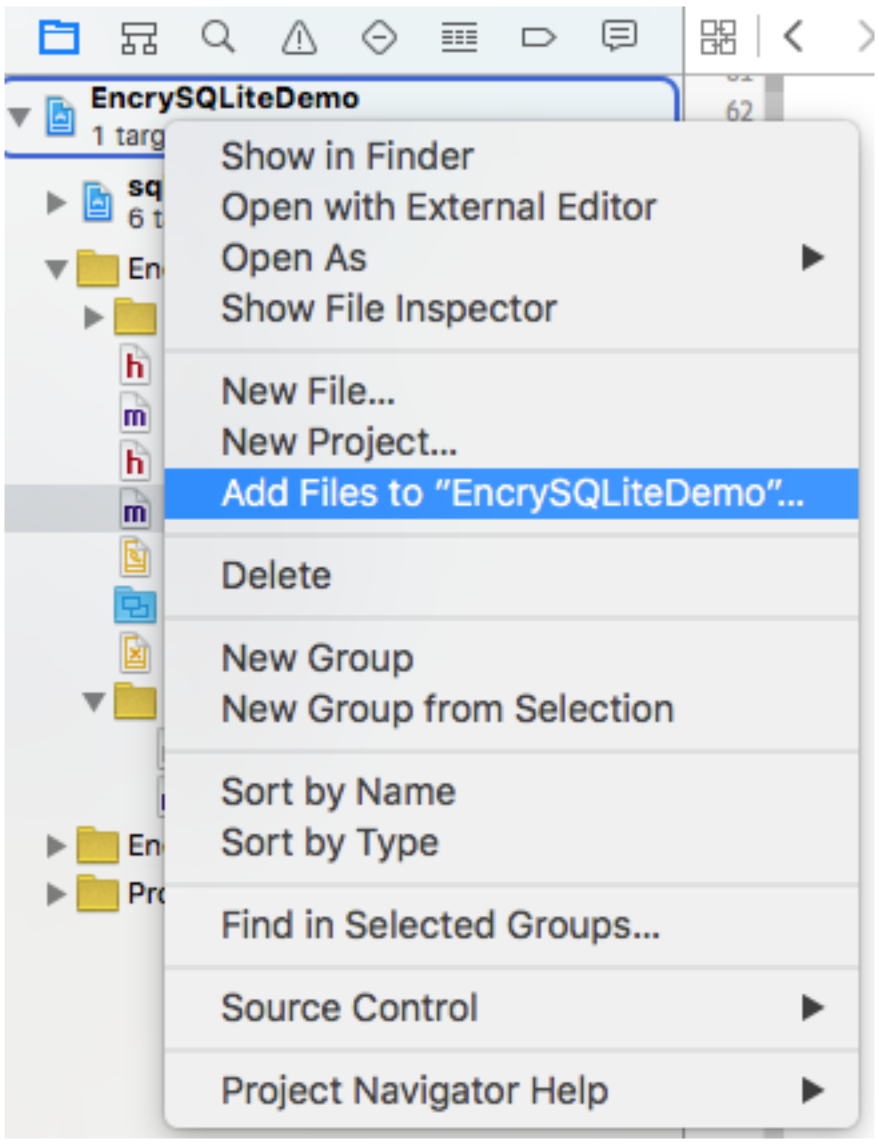
iOS(1)

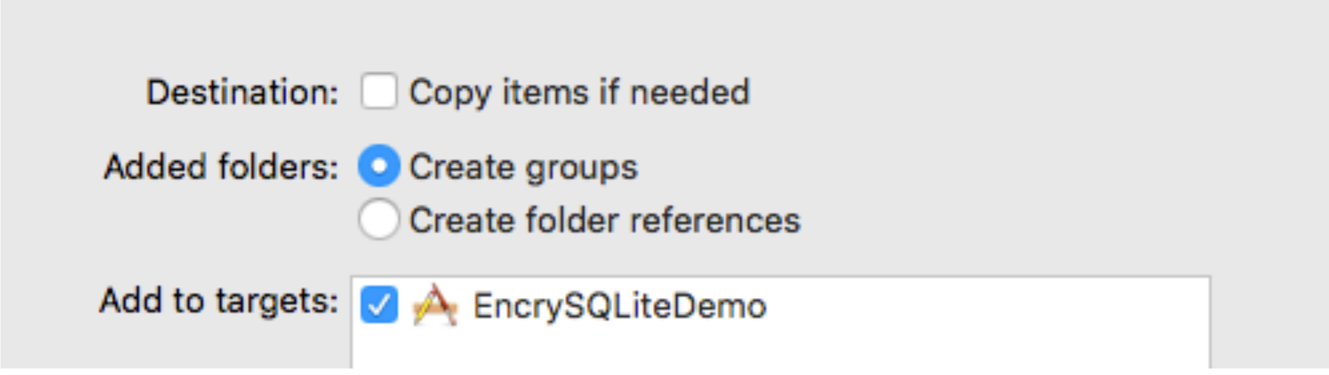
配置步骤

1、到 github 上下载 SQLCipher 插件，并存放到项目根目录下。



2、sqlcipher.xcodeproj 以 static library 的方式添加到项目里面。





Destination: □ Copy items if needed

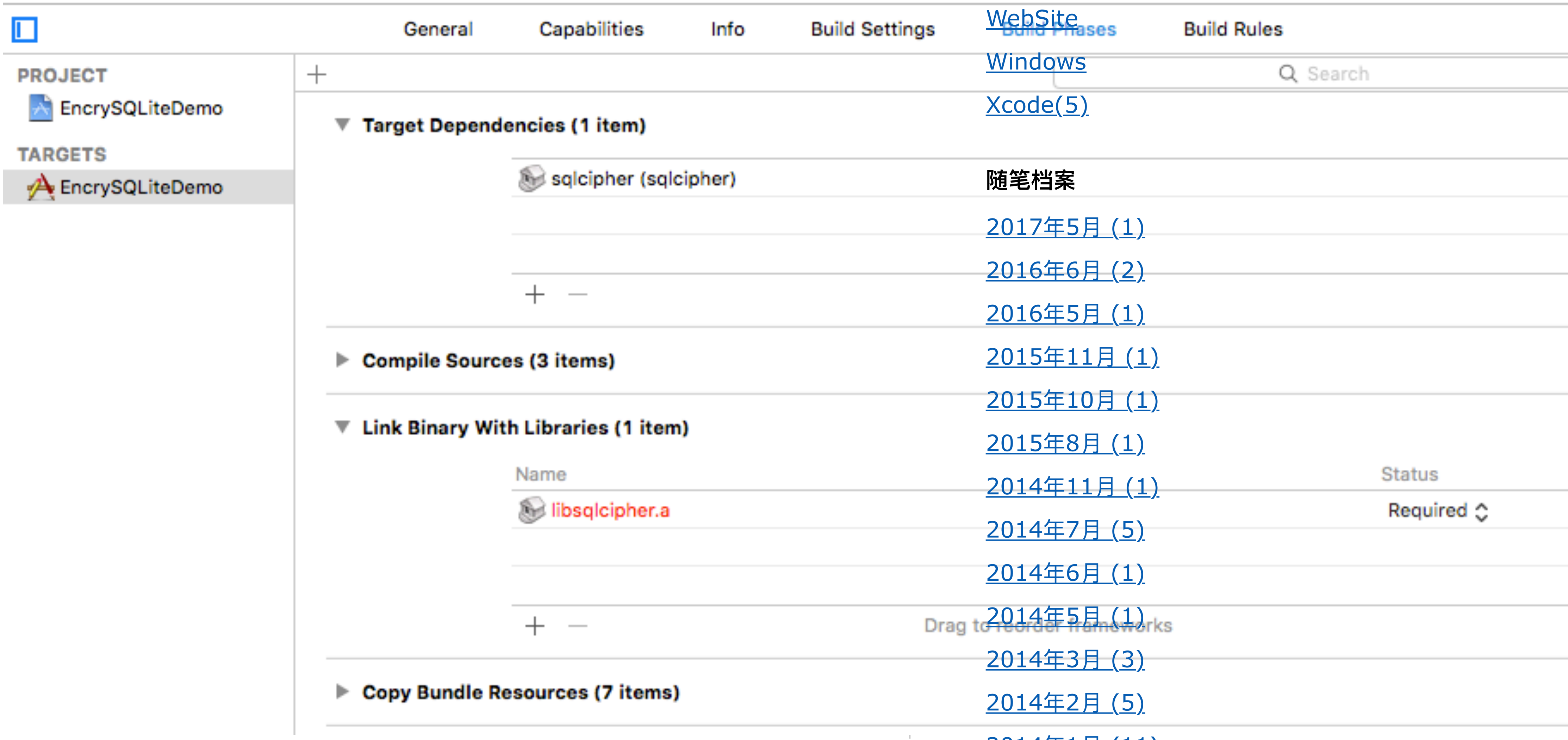
Added folders: ● Create groups

○ Create folder references

Add to targets: ☑ EncrySQLiteDemo

3、关联新添加的静态库 (注意，这里不能包含系统的 libsqlite3.dylib)

iOS App 瘦身
iOS.Annimation
iOS.Block(1)
ios.CGContext
iOS.Date
iOS.Image
iOS.In App Purchase(1)
iOS.Memory Managerment
iOS.NavigationBar(1)
iOS.Network(3)
iOS.OBjective-C 语法基础
iOS.PDF
iOS.RunTime
iOS.SearchBar(1)
iOS.SNS
iOS.TabBar
iOS.UILabel
iOS.UIScrollView+UITableView
iOS.xib + storyBoard
iOS.编程习惯(1)
iOS.打包、安装(1)
iOS.第三方框架(1)
iOS.定位服务
iOS.宏
iOS.基本类型
iOS.集合
iOS.加解密
iOS.架构设计
iOS.面试
iOS.逆向工程(4)
iOS.视图布局
iOS.输入框
iOS.数据库(2)
iOS.推送通知
iOS.文本处理
iOS.文件管理(1)
iOS.系统功能访问权限
iOS.线程(1)
iOS.消息机制(1)
iOS.引擎
iOS.应用间通信
iOS.应用优化(1)
iOS适配(1)
iOS-文档翻译
Linux
Mac osx(4)
Mac.AppleScript(1)
Math
others(1)
Python
React Native
Swift(1)
TCP 协议



Target Dependencies (1 item)

sqlcipher (sqlcipher)

Compile Sources (3 items)

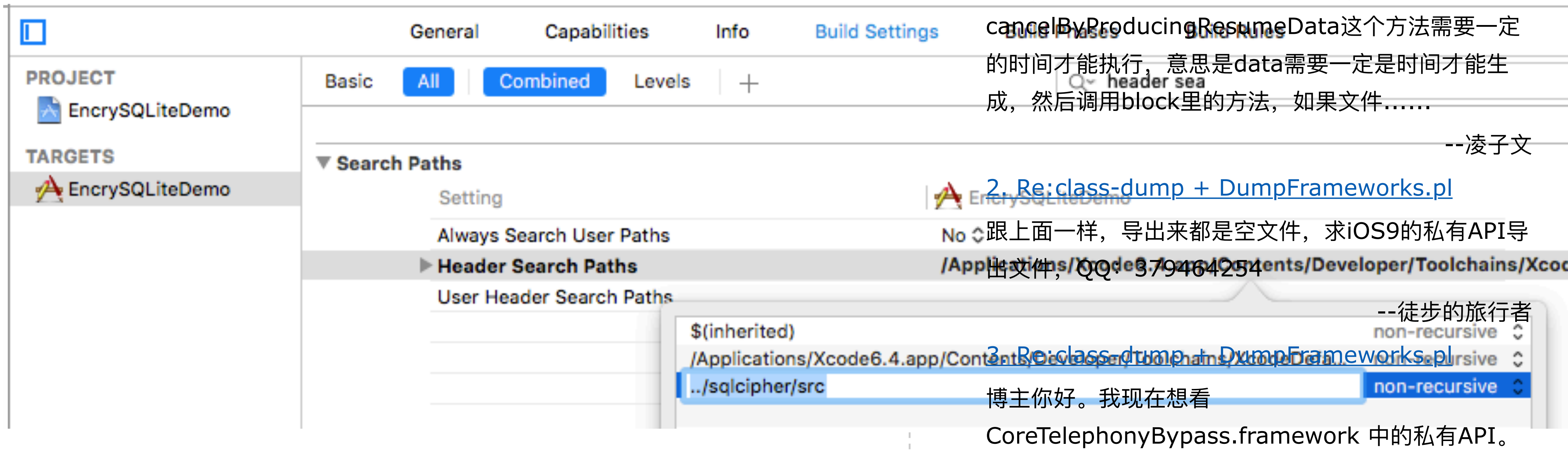
Link Binary With Libraries (1 item)

libsqlcipher.a Required

Copy Bundle Resources (7 items)

4、设置 Build Setting

"Header Search Path" 添加，"../sqlcipher/src"，这里需要注意路径的关系。



Search Paths

Always Search User Paths No

Header Search Paths /Applications/Xcode6.4.app/Contents/Developer/Toolchains/Xcod

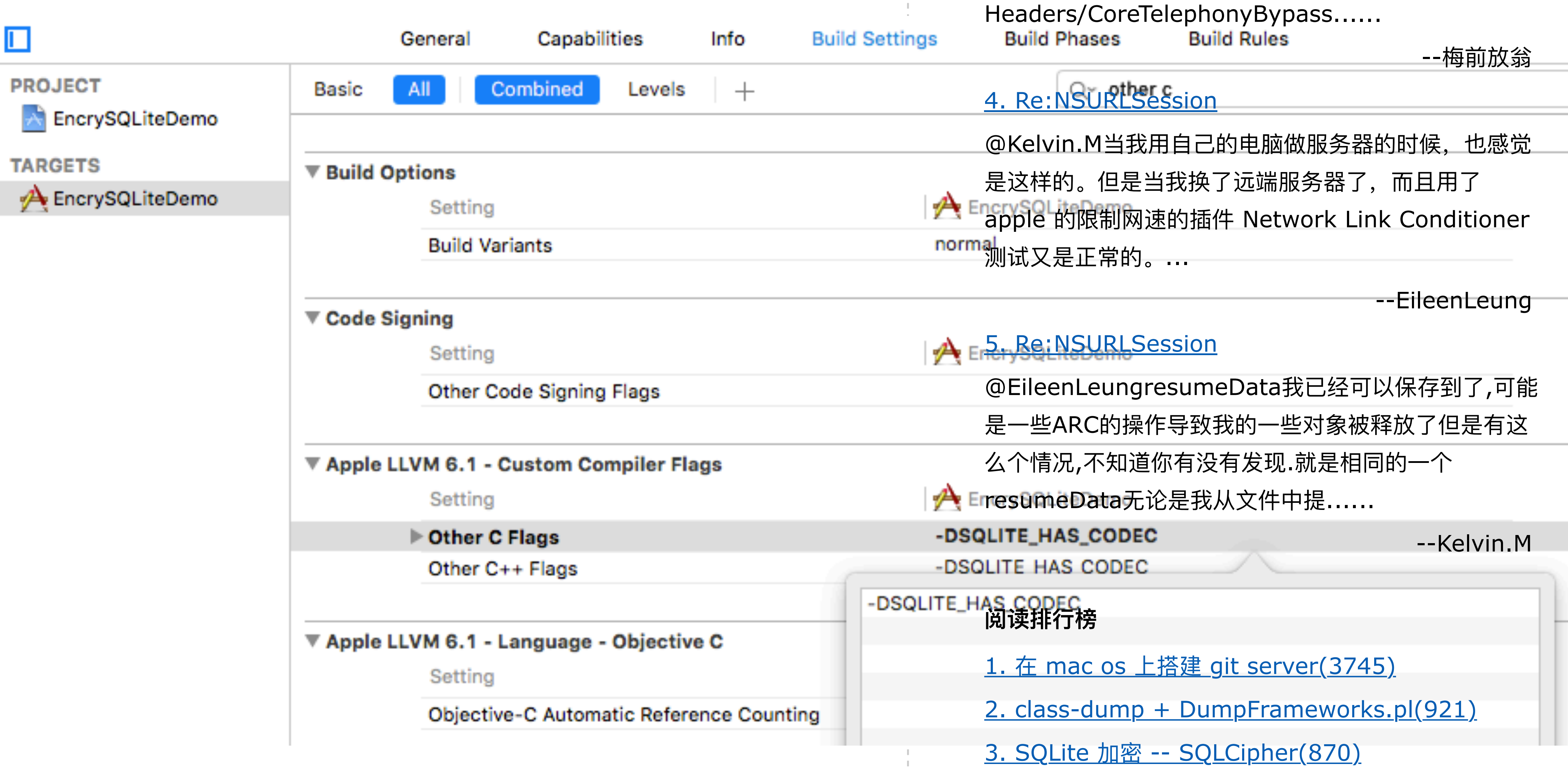
User Header Search Paths

$(inherited) non-recursive

/Applications/Xcode6.4.app/Contents/Developer/Toolchains/XcodeDefa... non-recursive

../sqlcipher/src non-recursive

"Other C Flags" 添加 "-DSQLITE_HAS_CODEC"



Build Options

Build Variants normal

Code Signing

Other Code Signing Flags

Apple LLVM 6.1 - Custom Compiler Flags

Other C Flags -DSQLITE_HAS_CODEC

Other C++ Flags -DSQLITE_HAS_CODEC

-DSQLITE_HAS_CODEC

Apple LLVM 6.1 - Language - Objective C

Objective-C Automatic Reference Counting

由于 SQLCipher 是支持 Mac OSX, iOS, WatchOS, TVOS 等多个平台,

WebSite

Windows

Xcode(5)

随笔档案

2017年5月 (1)

2016年6月 (2)

2016年5月 (1)

2015年11月 (1)

2015年10月 (1)

2015年8月 (1)

2014年11月 (1)

2014年7月 (5)

2014年6月 (1)

2014年5月 (1)

2014年3月 (3)

2014年2月 (5)

2014年1月 (11)

2013年12月 (4)

最新评论

1. Re:NSURLSession

楼主，我写到这个地方碰到一个问题，那就是cancelByProducingResumeData这个方法需要一定的时间才能执行，意思是data需要一定是时间才能生成，然后调用block里的方法，如果文件......

--凌子文

2. Re:class-dump + DumpFrameworks.pl

跟上面一样，导出来都是空文件，求iOS9的私有API导出文件，QQ：379464254

--徒步的旅行者

3. Re:class-dump + DumpFrameworks.pl

博主你好。我现在想看CoreTelephonyBypass.framework 中的私有API。但是按照你的方法操作之后，确实在主目录下生成了Headers/CoreTelephonyBypass......

--梅前放翁

4. Re:NSURLSession

@Kelvin.M当我用自己的电脑做服务器的时候，也感觉是这样的。但是当我换了远端服务器了，而且用了 apple 的限制网速的插件 Network Link Conditioner 测试又是正常的。...

--EileenLeung

5. Re:NSURLSession

@EileenLeungresumeData我已经可以保存到了,可能是一些ARC的操作导致我的一些对象被释放了但是有这么个情况,不知道你有没有发现.就是相同的一个resumeData无论是我从文件中提......

--Kelvin.M

阅读排行榜

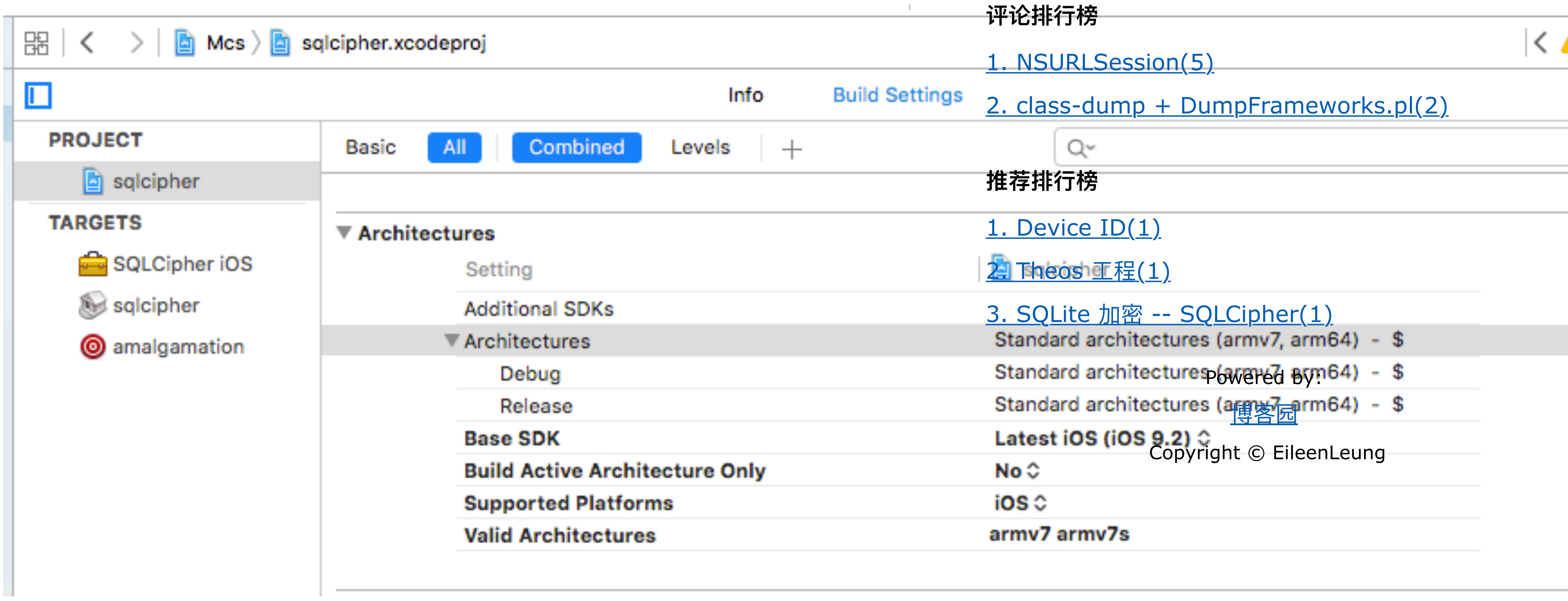
1. 在 mac os 上搭建 git server(3745)

2. class-dump + DumpFrameworks.pl(921)

3. SQLite 加密 -- SQLCipher(870)

4. UINavigationBar 总结(688)

所以必要的时候，需要修改 sqlcipher.project 的配置文件，否则会引起编译或者 linking 错误，修改如下：



▼ Architectures

Setting

Additional SDKs

▼ Architectures — Standard architectures (armv7, arm64) - $

Debug — Standard architectures (armv7, arm64) - $

Release — Standard architectures (armv7, arm64) - $

Base SDK — Latest iOS (iOS 9.2)

Build Active Architecture Only — No

Supported Platforms — iOS

Valid Architectures — armv7 armv7s

项目中使用示例

```
#import <sqlite3.h>

- (void)openDB2 {
  NSString *documentPath = NSSearchPathForDirectoriesI
  NSString *db2Path = [documentPath stringByAppendingP

  if (sqlite3_open([db2Path UTF8String], &database2) =
    const char* key = [@"eileen" UTF8String];
    sqlite3_key(database2, key, strlen(key));
//     if (sqlite3_exec(database2, (const char*) "CRE
//       NSLog(@"password is correct, or, database
//
//     } else {
//       NSLog(@"incorrect password!");
//     }
//     sqlite3_close(database2);

     if (sqlite3_exec(database2, "INSERT INTO t1(a,
       NSLog(@"密码正确");
     }
     else {
       NSLog(@"密码错误");
     }


    sqlite3_stmt *statement = NULL;
    sqlite3_prepare_v2(database2, "SELECT a,b FROM t

    while (sqlite3_step(statement) == SQLITE_ROW) {
      char *field0 = (char*)sqlite3_column_text(st
      NSString *field0Str = @"";
      if (field0) {
        field0Str = [NSString stringWithUTF8Stri
      }

      char *field1 = (char*)sqlite3_column_text(st
```

5. 编译运行错误汇集(493)

评论排行榜

1. NSURLSession(5)
2. class-dump + DumpFrameworks.pl(2)

推荐排行榜

1. Device ID(1)
2. Theos 工程(1)
3. SQLite 加密 -- SQLCipher(1)

Powered by:
博客园

```
            NSString *field1Str = @"";
            if (field1) {
                field1Str = [NSString stringWithUTF8Stri
            }
            NSLog(@"a = %@, b = %@;", field0Str, field1S
        }
        sqlite3_finalize(statement);
    }
    else {
      sqlite3_close(database2);
    }
}
```

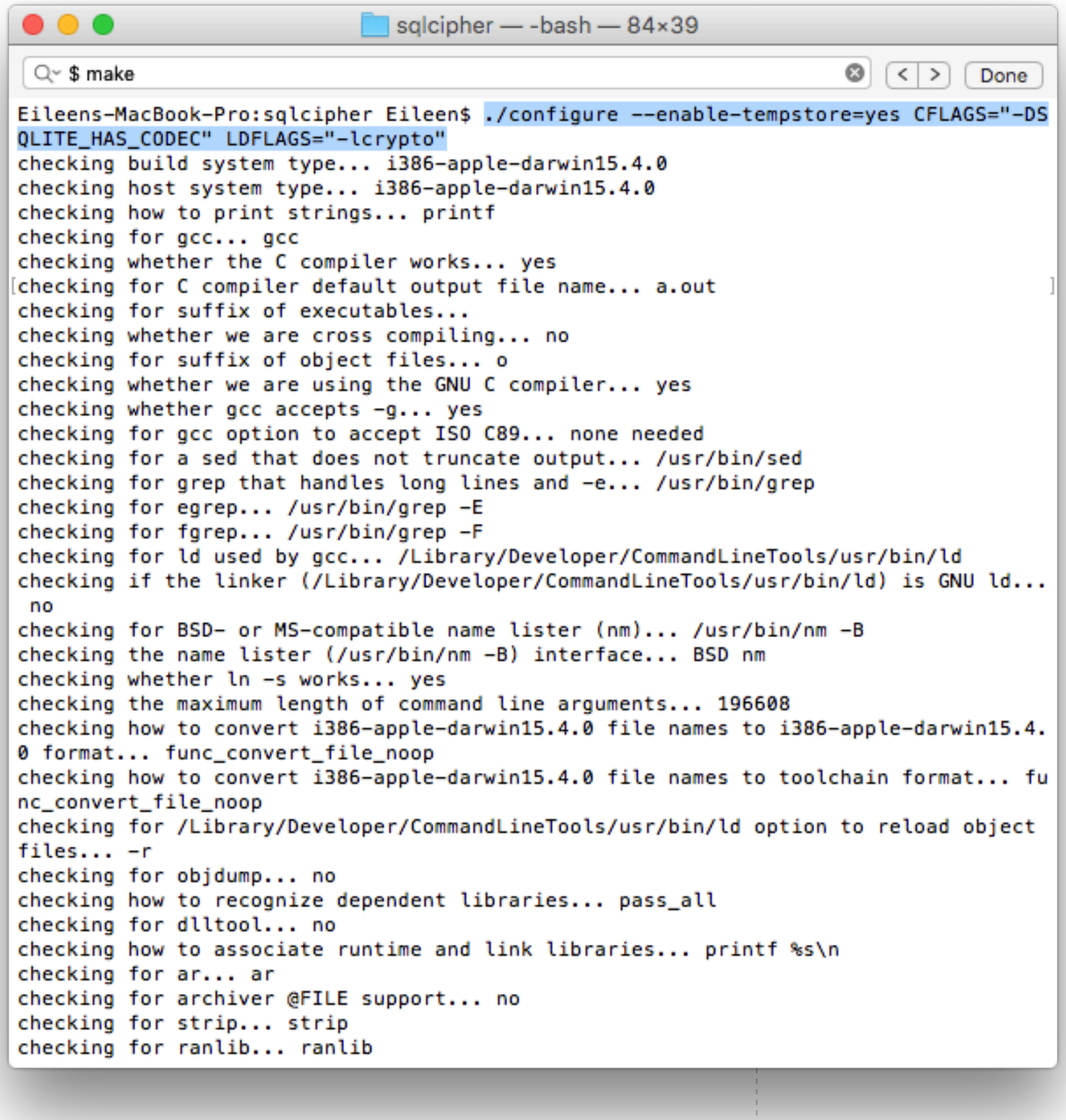
Terminal 上安装 SQLCipher

总的来说在 Terminal 执行以下 2句即可，参照：

```
$ ./configure --enable-tempstore=yes CFLAGS="-DSQLITE_HAS
$ make    ;#  Run the makefile.
```

1、cd 到下载好的 sqlcipher 目录下，并执行

```
$ ./configure --enable-tempstore=yes CFLAGS="-DSQLITE_HAS
```



```
Eileens-MacBook-Pro:sqlcipher Eileen$ ./configure --enable-tempstore=yes CFLAGS="-DS
QLITE_HAS_CODEC" LDFLAGS="-lcrypto"
checking build system type... i386-apple-darwin15.4.0
checking host system type... i386-apple-darwin15.4.0
checking how to print strings... printf
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking for a sed that does not truncate output... /usr/bin/sed
checking for grep that handles long lines and -e... /usr/bin/grep
checking for egrep... /usr/bin/grep -E
checking for fgrep... /usr/bin/grep -F
checking for ld used by gcc... /Library/Developer/CommandLineTools/usr/bin/ld
checking if the linker (/Library/Developer/CommandLineTools/usr/bin/ld) is GNU ld...
 no
checking for BSD- or MS-compatible name lister (nm)... /usr/bin/nm -B
checking the name lister (/usr/bin/nm -B) interface... BSD nm
checking whether ln -s works... yes
checking the maximum length of command line arguments... 196608
checking how to convert i386-apple-darwin15.4.0 file names to i386-apple-darwin15.4.
0 format... func_convert_file_noop
checking how to convert i386-apple-darwin15.4.0 file names to toolchain format... fu
nc_convert_file_noop
checking for /Library/Developer/CommandLineTools/usr/bin/ld option to reload object
files... -r
checking for objdump... no
checking how to recognize dependent libraries... pass_all
checking for dlltool... no
checking how to associate runtime and link libraries... printf %s\n
checking for ar... ar
checking for archiver @FILE support... no
checking for strip... strip
checking for ranlib... ranlib
```

2、输入

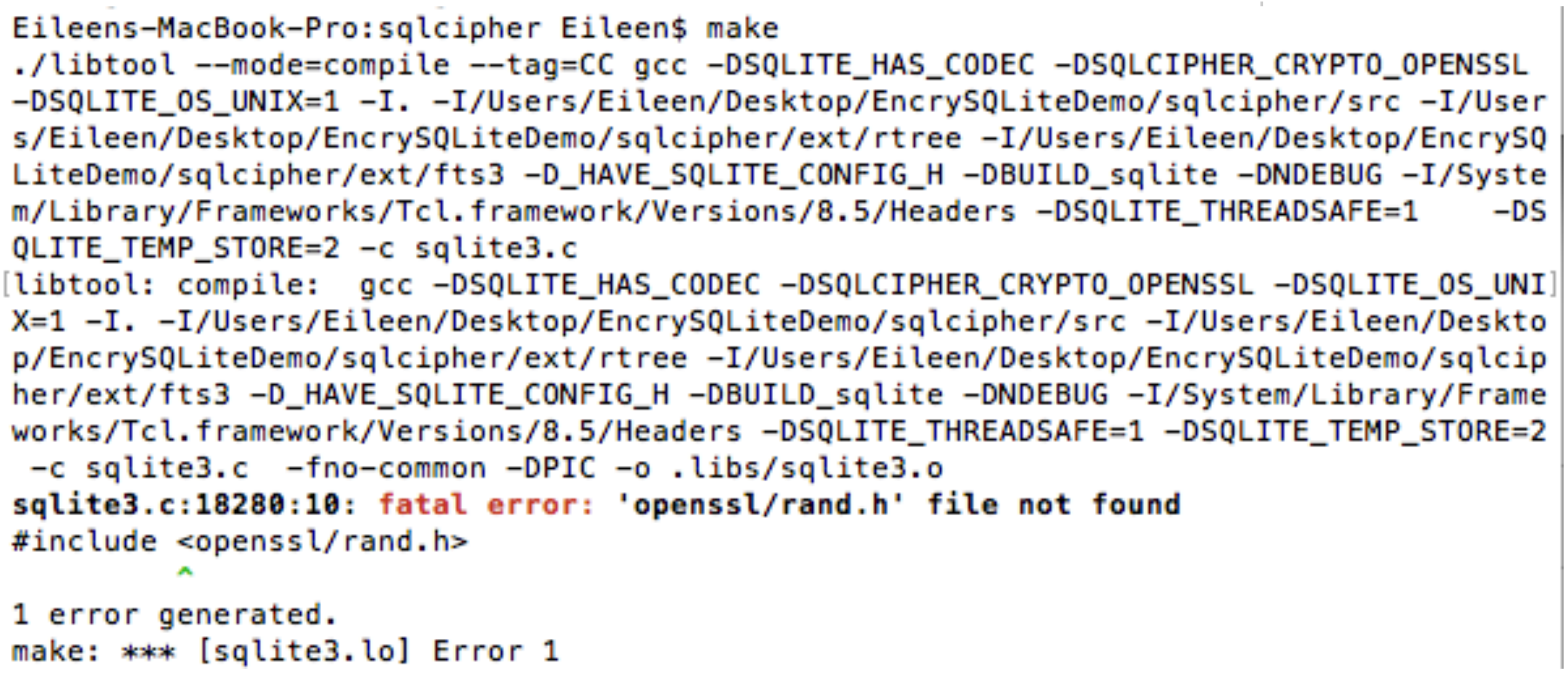
```
$ make
```

2.1、发生了错误，

sqlite3.c:18280:10: fatal error: 'openssl/rand.h' file not found

#include <openssl/rand.h>

见下图：



```
Eileens-MacBook-Pro:sqlcipher Eileen$ make
./libtool --mode=compile --tag=CC gcc -DSQLITE_HAS_CODEC -DSQLCIPHER_CRYPTO_OPENSSL
-DSQLITE_OS_UNIX=1 -I. -I/Users/Eileen/Desktop/EncrySQLiteDemo/sqlcipher/src -I/User
s/Eileen/Desktop/EncrySQLiteDemo/sqlcipher/ext/rtree -I/Users/Eileen/Desktop/EncrySQ
LiteDemo/sqlcipher/ext/fts3 -D_HAVE_SQLITE_CONFIG_H -DBUILD_sqlite -DNDEBUG -I/Syste
m/Library/Frameworks/Tcl.framework/Versions/8.5/Headers -DSQLITE_THREADSAFE=1    -DS
QLITE_TEMP_STORE=2 -c sqlite3.c
libtool: compile:  gcc -DSQLITE_HAS_CODEC -DSQLCIPHER_CRYPTO_OPENSSL -DSQLITE_OS_UNI
X=1 -I. -I/Users/Eileen/Desktop/EncrySQLiteDemo/sqlcipher/src -I/Users/Eileen/Deskto
p/EncrySQLiteDemo/sqlcipher/ext/rtree -I/Users/Eileen/Desktop/EncrySQLiteDemo/sqlcip
her/ext/fts3 -D_HAVE_SQLITE_CONFIG_H -DBUILD_sqlite -DNDEBUG -I/System/Library/Frame
works/Tcl.framework/Versions/8.5/Headers -DSQLITE_THREADSAFE=1 -DSQLITE_TEMP_STORE=2
 -c sqlite3.c  -fno-common -DPIC -o .libs/sqlite3.o
sqlite3.c:18280:10: fatal error: 'openssl/rand.h' file not found
#include <openssl/rand.h>
         ^
1 error generated.
make: *** [sqlite3.lo] Error 1
```

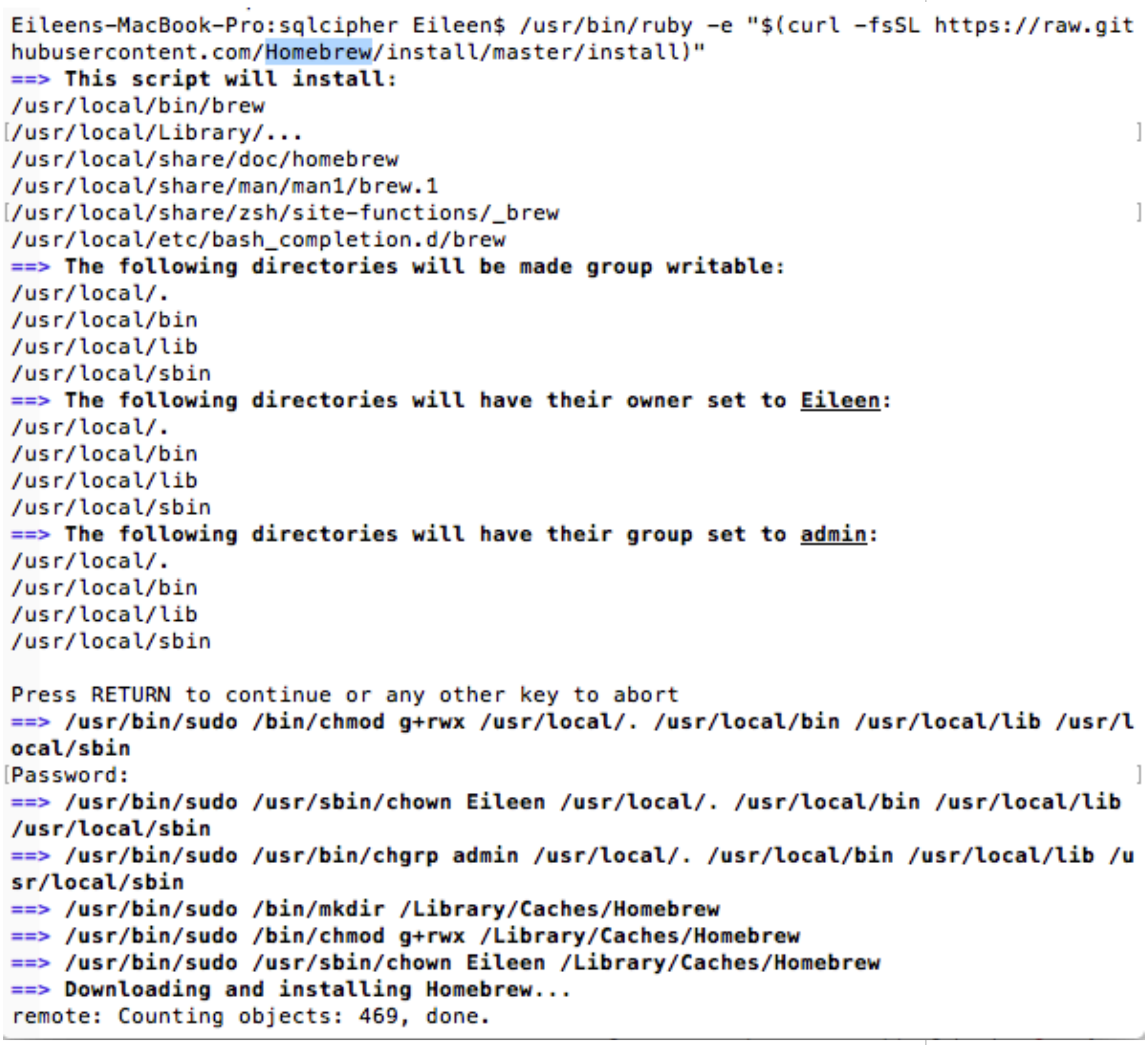
解决方法，输入：

```
$ brew link openssl --force
```

2.2、发生错误，“-bash: brew: command not found”，证明 OS 尚未安装 Homebrew。(安装 Homebrew 的前提下是安装了 Xcode, 并且 Command Line Tools 已安装, Terminal 输入 "gcc --version" 检查)

解决方法，输入：

```
$ -e "$(curl -fsSL https://raw.githubusercontent.com/Home
```



```
Eileens-MacBook-Pro:sqlcipher Eileen$ /usr/bin/ruby -e "$(curl -fsSL https://raw.git
hubusercontent.com/Homebrew/install/master/install)"
==> This script will install:
/usr/local/bin/brew
/usr/local/Library/...
/usr/local/share/doc/homebrew
/usr/local/share/man/man1/brew.1
/usr/local/share/zsh/site-functions/_brew
/usr/local/etc/bash_completion.d/brew
==> The following directories will be made group writable:
/usr/local/.
/usr/local/bin
/usr/local/lib
/usr/local/sbin
==> The following directories will have their owner set to Eileen:
/usr/local/.
/usr/local/bin
/usr/local/lib
/usr/local/sbin
==> The following directories will have their group set to admin:
/usr/local/.
/usr/local/bin
/usr/local/lib
/usr/local/sbin

Press RETURN to continue or any other key to abort
==> /usr/bin/sudo /bin/chmod g+rwx /usr/local/. /usr/local/bin /usr/local/lib /usr/l
ocal/sbin
Password:
==> /usr/bin/sudo /usr/sbin/chown Eileen /usr/local/. /usr/local/bin /usr/local/lib
/usr/local/sbin
==> /usr/bin/sudo /usr/bin/chgrp admin /usr/local/. /usr/local/bin /usr/local/lib /u
sr/local/sbin
==> /usr/bin/sudo /bin/mkdir /Library/Caches/Homebrew
==> /usr/bin/sudo /bin/chmod g+rwx /Library/Caches/Homebrew
==> /usr/bin/sudo /usr/sbin/chown Eileen /Library/Caches/Homebrew
==> Downloading and installing Homebrew...
remote: Counting objects: 469, done.
```

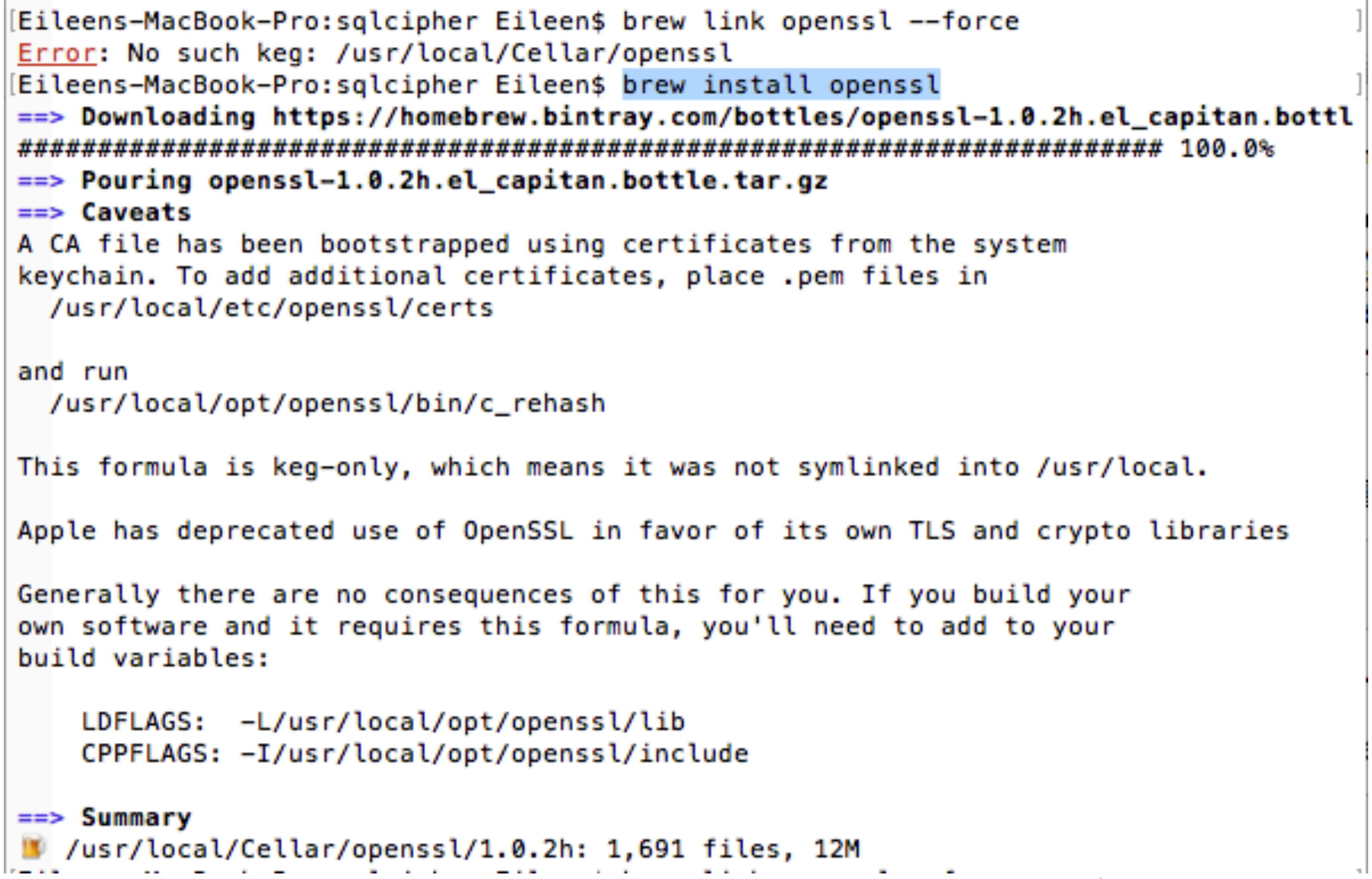
2.3、安装好 Homebrew 后，重新执行

```
$ brew link openssl --force
```

发生了错误：Error: No such keg: /usr/local/Cellar/openssl

解决方法，使用 brew 安装 openssl，输入：

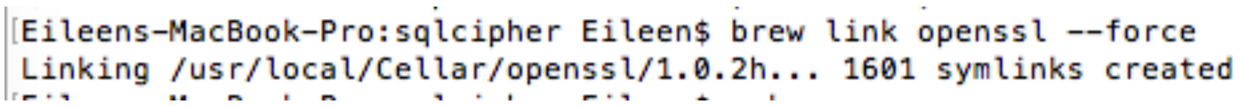
```
$ brew install openssl
```



```
Eileens-MacBook-Pro:sqlcipher Eileen$ brew link openssl --force
Error: No such keg: /usr/local/Cellar/openssl
Eileens-MacBook-Pro:sqlcipher Eileen$ brew install openssl
==> Downloading https://homebrew.bintray.com/bottles/openssl-1.0.2h.el_capitan.bottl
######################################################################## 100.0%
==> Pouring openssl-1.0.2h.el_capitan.bottle.tar.gz
==> Caveats
A CA file has been bootstrapped using certificates from the system
keychain. To add additional certificates, place .pem files in
  /usr/local/etc/openssl/certs

and run
  /usr/local/opt/openssl/bin/c_rehash

This formula is keg-only, which means it was not symlinked into /usr/local.

Apple has deprecated use of OpenSSL in favor of its own TLS and crypto libraries

Generally there are no consequences of this for you. If you build your
own software and it requires this formula, you'll need to add to your
build variables:

    LDFLAGS:  -L/usr/local/opt/openssl/lib
    CPPFLAGS: -I/usr/local/opt/openssl/include

==> Summary
🍺 /usr/local/Cellar/openssl/1.0.2h: 1,691 files, 12M
```

2.4、安装完 openssl 后，重新执行

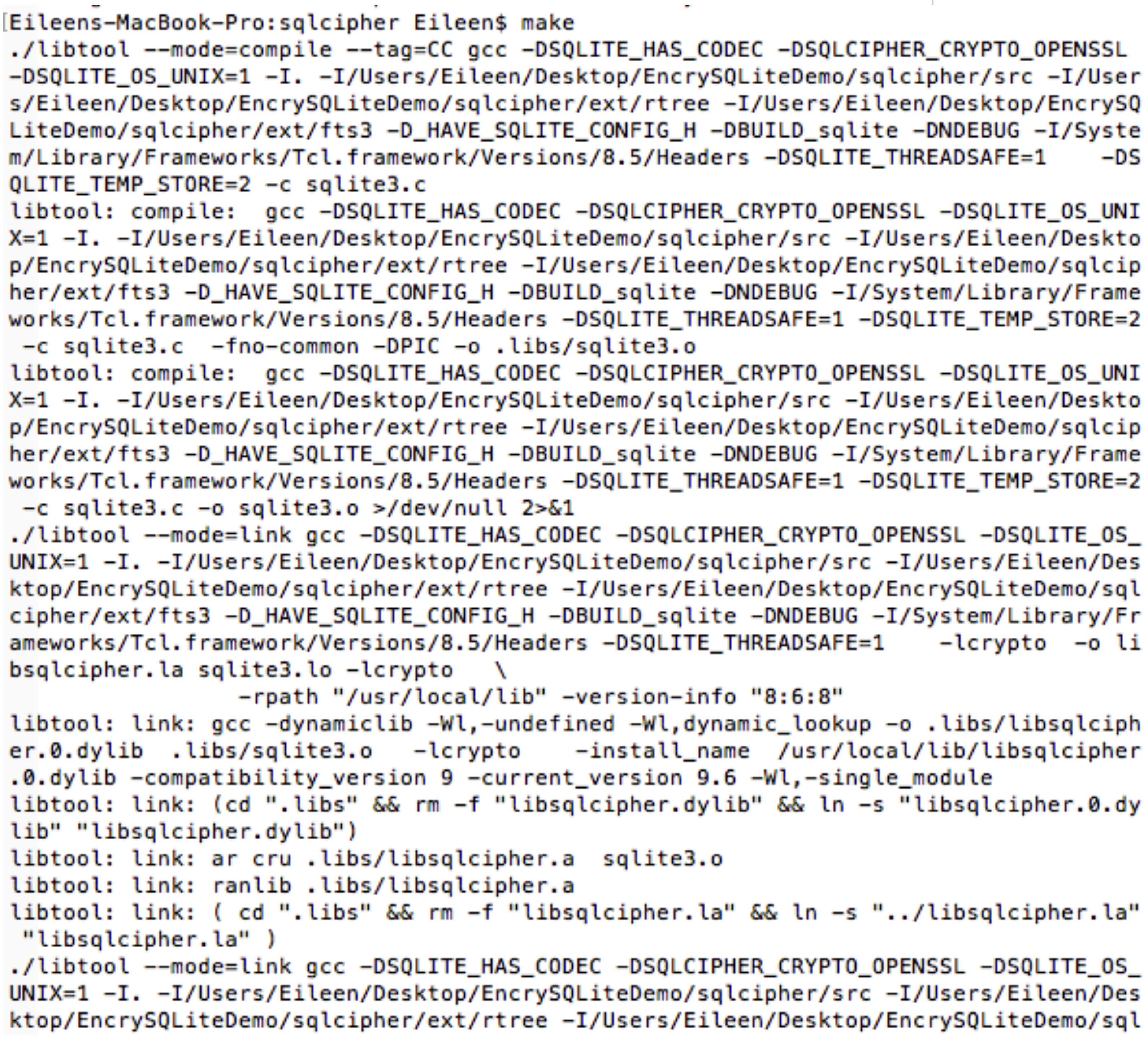
```
$ brew link openssl --force
```



```
Eileens-MacBook-Pro:sqlcipher Eileen$ brew link openssl --force
Linking /usr/local/Cellar/openssl/1.0.2h... 1601 symlinks created
```

执行完后，再重新执行

```
$ make
```



```
Eileens-MacBook-Pro:sqlcipher Eileen$ make
./libtool --mode=compile --tag=CC gcc -DSQLITE_HAS_CODEC -DSQLCIPHER_CRYPTO_OPENSSL
-DSQLITE_OS_UNIX=1 -I. -I/Users/Eileen/Desktop/EncrySQLiteDemo/sqlcipher/src -I/User
s/Eileen/Desktop/EncrySQLiteDemo/sqlcipher/ext/rtree -I/Users/Eileen/Desktop/EncrySQ
LiteDemo/sqlcipher/ext/fts3 -D_HAVE_SQLITE_CONFIG_H -DBUILD_sqlite -DNDEBUG -I/Syste
m/Library/Frameworks/Tcl.framework/Versions/8.5/Headers -DSQLITE_THREADSAFE=1    -DS
QLITE_TEMP_STORE=2 -c sqlite3.c
libtool: compile:  gcc -DSQLITE_HAS_CODEC -DSQLCIPHER_CRYPTO_OPENSSL -DSQLITE_OS_UNI
X=1 -I. -I/Users/Eileen/Desktop/EncrySQLiteDemo/sqlcipher/src -I/Users/Eileen/Deskto
p/EncrySQLiteDemo/sqlcipher/ext/rtree -I/Users/Eileen/Desktop/EncrySQLiteDemo/sqlcip
her/ext/fts3 -D_HAVE_SQLITE_CONFIG_H -DBUILD_sqlite -DNDEBUG -I/System/Library/Frame
works/Tcl.framework/Versions/8.5/Headers -DSQLITE_THREADSAFE=1 -DSQLITE_TEMP_STORE=2
 -c sqlite3.c  -fno-common -DPIC -o .libs/sqlite3.o
libtool: compile:  gcc -DSQLITE_HAS_CODEC -DSQLCIPHER_CRYPTO_OPENSSL -DSQLITE_OS_UNI
X=1 -I. -I/Users/Eileen/Desktop/EncrySQLiteDemo/sqlcipher/src -I/Users/Eileen/Deskto
p/EncrySQLiteDemo/sqlcipher/ext/rtree -I/Users/Eileen/Desktop/EncrySQLiteDemo/sqlcip
her/ext/fts3 -D_HAVE_SQLITE_CONFIG_H -DBUILD_sqlite -DNDEBUG -I/System/Library/Frame
works/Tcl.framework/Versions/8.5/Headers -DSQLITE_THREADSAFE=1 -DSQLITE_TEMP_STORE=2
 -c sqlite3.c -o sqlite3.o >/dev/null 2>&1
./libtool --mode=link gcc -DSQLITE_HAS_CODEC -DSQLCIPHER_CRYPTO_OPENSSL -DSQLITE_OS_
UNIX=1 -I. -I/Users/Eileen/Desktop/EncrySQLiteDemo/sqlcipher/src -I/Users/Eileen/Des
ktop/EncrySQLiteDemo/sqlcipher/ext/rtree -I/Users/Eileen/Desktop/EncrySQLiteDemo/sql
cipher/ext/fts3 -D_HAVE_SQLITE_CONFIG_H -DBUILD_sqlite -DNDEBUG -I/System/Library/Fr
ameworks/Tcl.framework/Versions/8.5/Headers -DSQLITE_THREADSAFE=1    -lcrypto  -o li
bsqlcipher.la sqlite3.lo -lcrypto    \
                -rpath "/usr/local/lib" -version-info "8:6:8"
libtool: link: gcc -dynamiclib -Wl,-undefined -Wl,dynamic_lookup -o .libs/libsqlciph
er.0.dylib  .libs/sqlite3.o   -lcrypto    -install_name  /usr/local/lib/libsqlcipher
.0.dylib -compatibility_version 9 -current_version 9.6 -Wl,-single_module
libtool: link: (cd ".libs" && rm -f "libsqlcipher.dylib" && ln -s "libsqlcipher.0.dy
lib" "libsqlcipher.dylib")
libtool: link: ar cru .libs/libsqlcipher.a  sqlite3.o
libtool: link: ranlib .libs/libsqlcipher.a
libtool: link: ( cd ".libs" && rm -f "libsqlcipher.la" && ln -s "../libsqlcipher.la"
 "libsqlcipher.la" )
./libtool --mode=link gcc -DSQLITE_HAS_CODEC -DSQLCIPHER_CRYPTO_OPENSSL -DSQLITE_OS_
UNIX=1 -I. -I/Users/Eileen/Desktop/EncrySQLiteDemo/sqlcipher/src -I/Users/Eileen/Des
ktop/EncrySQLiteDemo/sqlcipher/ext/rtree -I/Users/Eileen/Desktop/EncrySQLiteDemo/sql
```

多次执行 make 操作会发生错误 “make: Nothing to be done for `all'”，解决方法，输入：

```
$ make clean

// 重新执行

$ make
```

3、执行完前面 2 步，sqlcipher 目录下会多了一个 sqlcipher 文件，用于 Terminal 中管理数据库。

Terminal 查看和修改数据库的密码管理

cd 到刚才新生成的 sqlcipher 文件的目录下，执行以下的操作，参照。

1、使用 SQLCipher 加密已经存在的数据库

```
$ ./sqlcipher plaintext.db
sqlite> ATTACH DATABASE 'encrypted.db' AS encrypted KEY '
sqlite> SELECT sqlcipher_export('encrypted');
sqlite> DETACH DATABASE encrypted;
```

2、解除使用 SQLCipher 加密的数据库密码

```
$ ./sqlcipher encrypted.db
sqlite> PRAGMA key = 'testkey';
sqlite> ATTACH DATABASE 'plaintext.db' AS plaintext KEY '
sqlite> SELECT sqlcipher_export('plaintext');
sqlite> DETACH DATABASE plaintext;
```

注意

有些软件的加密方式是不公开的，例如 Mac SQLiteManager 生成的加密的 .db 文件没法在程序里面解密打开。程序里面生成的加密的 .db 文件也没法用 Mac 上的 SQLiteManager 打开。

免费版本的项目代码不提供以下的功能：

- 数据库创建的时候，没有使用 sqlite3_key 设置密码，之后不能添加加密码管理；
- 对创建时已经设置了密码管理的数据库，不能取消其密码管理，只能重新设置新的密码；

分类：iOS.数据库

EileenLeung
关注 - 1
粉丝 - 3
+加关注

1
推荐

0
反对

« 上一篇：iOS9 适配(杂七杂八)
» 下一篇：保护网络传输的实现

posted on 2016-05-05 08:39 EileenLeung 阅读(870) 评论(0) 编辑 收藏

刷新评论 刷新页面 返回顶部

努力加载评论框中...

最新IT新闻:
· 前腾讯员工说：腾讯的核心能力在马化腾身上
· 中国正在研制新一代原子钟，或用于下一代北斗导航卫星
· Adobe已宣布Flash结束，开源它是最好的方案吗？
· 微信“电子发票”三大升级：实在太给力了
· 学习下蒂姆·库克给予的12条商业建议
» 更多新闻...

最新知识库文章:
· 小printf的故事：什么是真正的程序员？
· 程序员的工作、学习与绩效
· 软件开发为什么很难
· 唱吧DevOps的落地，微服务CI/CD的范本技术解读
· 程序员，如何从平庸走向理想？
» 更多知识库文章...