leenjewel / **openssl_for_ios_and_android**

OpenSSL Library for iOS and Android

71 commits | **1** branch | **4** releases | **3** contributors

Branch: **master** | **New pull request** | **Find file** | **Clone or download**

liyin update example — Latest commit `11b90af` 20 days ago

example/android	update example	20 days ago
tools	bug fix	2 months ago
.gitignore	Fix some bugs	9 months ago
README.md	Add explain	20 days ago

README.md

OpenSSL & cURL Library for iOS and Android

library	version	platform support	arch support	pull commit
openssl	1.1.0f	ios	armv7s armv7 i386 x86_64 arm64	aae1672
		android	armeabi armeabi-v7a arm64-v8a x86 x86_64 mips mips64	aae1672
curl	7.53.1	ios	armv7s armv7 i386 x86_64 arm64	aae1672
		android	armeabi armeabi-v7a arm64-v8a x86 x86_64 mips mips64	aae1672

English

Downloads

If you do not want to build it by yourself, you could download our prebuilt library from there

Android prebuilt library download from here!!!

OpenSSL Version

This a static library compile from openssl and cURL for iOS and Android.

- ~~openssl-1.0.2c.tar.gz~~
- openssl-1.1.0f.tar.gz
- https://github.com/openssl/openssl

cURL Version

- ~~curl-7.47.1.tar.gz~~

- curl-7.53.1.tar.gz
- https://github.com/curl/curl

Android NDK Version

- ~~android-ndk-r13b~~
- android-ndk-r14b
- ~~android-ndk-r15 (**Do not try to build use 15 It will fail**)~~

How to build

For iOS

Copy `openssl-1.1.0f.tar.gz` to `tools` file folder and run

```
cd tools
sh ./build-openssl4ios.sh
```

Copy `curl-7.53.1.tar.gz` to `tools` file folder and run

```
cd tools
sh ./build-curl4ios.sh
```

For Android

Set ENV `NDK_ROOT`

```
cd tools
sh ./build-openssl4android.sh
```

You could build it with ABI like

```
cd tools
sh ./build-openssl4android.sh android  # for armeabi
sh ./build-openssl4android.sh android-armeabi #for armeabi-v7a
sh ./build-openssl4android.sh android64-arm64 #for arm64_v8a
sh ./build-openssl4android.sh android-x86  #for x86
sh ./build-openssl4android.sh android64  #for x86_64
sh ./build-openssl4android.sh mips  #for mips
sh ./build-openssl4android.sh mips64 #for mips64
```

> **You must build openssl first**
>
> **else cURL HTTPS is disable (without ssl)**

OpenSSL for Android is build with `libz` support using dynamic link. `libz` is publically provided by Android system.

```
sh ./build-curl4android.sh
```

You could build it with ABI like

```
cd tools
sh ./build-curl4android.sh android  # for armeabi
sh ./build-curl4android.sh android-armv7 #for armeabi-v7a
sh ./build-curl4android.sh android64-arm64 #for arm64_v8a
sh ./build-curl4android.sh android-x86  #for x86
sh ./build-curl4android.sh android-x86_64  #for x86_64
sh ./build-curl4android.sh mips  #for mips
sh ./build-curl4android.sh mips64 #for mips64
```

How to use

For iOS

Copy `lib/libcrypto.a` and `lib/libssl.a` and `lib/libcurl.a` to your project.

Copy `include/openssl` folder and `include/curl` folder to your project.

Add `libcrypto.a` and `libssl.a` and `libcurl.a` to `Frameworks` group and add them to `[Build Phases] ====> [Link Binary With Libraries]`.

Add openssl include path and curl include path to your `[Build Settings] ====> [User Header Search Paths]`

About "__curl_rule_01__ declared as an array with a negative size" problem

When you build cURL for arm64 you will get this error.

You need to change `curlbuild.h` from :

```
#define CURL_SIZEOF_LONG 4
```

to :

```
#ifdef __LP64__
#define CURL_SIZEOF_LONG 8
#else
#define CURL_SIZEOF_LONG 4
#endif
```

For Android

Copy `lib/armeabi` folder and `lib/armeabi-v7a` folder and `lib/x86` to your android project `libs` folder.

Copy `include/openssl` folder and `include/curl` to your android project.

Android Makefile

Add openssl include path to `jni/Android.mk`.

```
#Android.mk

include $(CLEAR_VARS)

LOCAL_MODULE := curl
LOCAL_SRC_FILES := Your cURL Library Path/$(TARGET_ARCH_ABI)/libcurl.a
include $(PREBUILT_STATIC_LIBRARY)


LOCAL_C_INCLUDES := \
        $(LOCAL_PATH)/Your Openssl Include Path/openssl \
        $(LOCAL_PATH)/Your cURL Include Path/curl

LOCAL_STATIC_LIBRARIES := libcurl

LOCAL_LDLIBS := -lz
```

CMake

Define `ssl`, `crypto`, `curl` as *STATIC IMPORTED* libraries.

```
add_library(crypto STATIC IMPORTED)
set_target_properties(crypto
  PROPERTIES IMPORTED_LOCATION ${CMAKE_SOURCE_DIR}/libs/${ANDROID_ABI}/libcrypto.a)

add_library(ssl STATIC IMPORTED)
set_target_properties(ssl
  PROPERTIES IMPORTED_LOCATION ${CMAKE_SOURCE_DIR}/libs/${ANDROID_ABI}/libssl.a)
```

```
add_library(curl STATIC IMPORTED)
set_target_properties(curl
  PROPERTIES IMPORTED_LOCATION ${CMAKE_SOURCE_DIR}/libs/${ANDROID_ABI}/libcurl.a)
```

Then link these libraries with your target, e.g.

```
target_link_libraries( # Specifies the target library.
                       native-lib

                       curl
                       ssl
                       crypto
                       )
```

About "libcrypto.a(ui_openssl.o):ui_openssl.c:function read_string_inner: error: undefined reference to 'signal' " problem

when you get these error

```
libcrypto.a(ui_openssl.o):ui_openssl.c:function read_string_inner: error: undefined reference to 'signal'

libcrypto.a(ui_openssl.o):ui_openssl.c:function read_string_inner: error: undefined reference to
'tcsetattr'
```

You need to rebuild OpenSSL static library with NDK API level 16 or earlier

If you build OpenSSL with API level 16 or earlier you may not build it for arch 64 bit only support 32 bit

more information :

https://github.com/openssl/openssl/issues/988

http://stackoverflow.com/questions/37122126/whats-the-exact-significance-of-android-ndk-platform-version-compared-to-api-le

中文

这是一个适用于 iOS 平台和 Android 平台的 Openssl 静态链接库。基于 openssl-1.1.0f 版本编译生成。

后来又加入了适用于 iOS 平台和 Android 平台且支持 SSL 的 cURL 静态链接库。基于 curl-7.53.1 版本编译生成。

> Our build script default use API 16 to build OpenSSL

下载

如果你不想自己构建，那么你可以使用我们已经预先构建好的版本，请在这里下载

Android 编译好的库从这里下载!!!

在 iOS 工程中使用

将 `lib/libcrypto.a` 和 `lib/libssl.a` 还有 `lib/libcurl.a` 三个静态链接库文件拷贝到你的 iOS 工程下面合适的位置。

将 `include/openssl` 文件夹和 `include/curl` 文件夹拷贝到你的 iOS 工程下面合适的位置。注意，所有的头文件均要放置到 `openssl` 文件夹下，不要随意更改文件夹名称。

将 `libcrypto.a` 和 `libssl.a` 还有 `lib/libcurl.a` 三个静态链接库文件通过 `[Build Phases] ====> [Link Binary With Libraries]` 引入你的 iOS 工程。

将包含所有头文件的 `openssl` 文件夹和含有头文件的 `curl` 文件夹设置到头文件搜索路径中。即在 `[Build Settings] ====> [User Header Search Paths]` 中设置好。

关于 "__curl_rule_01__ declared as an array with a negative size" 的问题解决办法

当你在 iOS 的 arm64 架构环境下编译 cURL 静态库时会遇到这个问题。解决的办法是修改 `curlbuild.h` 头文件，将下面这行：

```
#define CURL_SIZEOF_LONG 4
```

改成：

```
#ifdef __LP64__
#define CURL_SIZEOF_LONG 8
#else
#define CURL_SIZEOF_LONG 4
#endif
```

在 **Android** 工程中使用

将 `lib/armeabi` 和 `lib/armeabi-v7a` 还有 `lib/x86` 文件夹拷贝到你的 Android 工程下面的 `libs` 文件夹中。

将 `include/openssl` 文件夹和 `include/curl` 文件夹拷贝到你的 Android 工程下面合适的位置。注意，所有的头文件均要放置到 `openssl` 文件夹下，不要随意更改文件夹名称。

Android Makefile 系统

修改 `jni/Android.mk` 文件，将头文件路径加入到搜索路径中，例如：

```
# Android.mk

include $(CLEAR_VARS)

LOCAL_MODULE := curl
LOCAL_SRC_FILES := Your cURL Library Path/$(TARGET_ARCH_ABI)/libcurl.a
include $(PREBUILT_STATIC_LIBRARY)


include $(CLEAR_VARS)

LOCAL_MODULE := curl
LOCAL_SRC_FILES := Your cURL Library Path/$(TARGET_ARCH_ABI)/libcurl.a
include $(PREBUILT_STATIC_LIBRARY)


LOCAL_C_INCLUDES := \
        $(LOCAL_PATH)/Your Openssl Include Path/openssl \
        $(LOCAL_PATH)/Your cURL Include Path/curl

LOCAL_STATIC_LIBRARIES := libcurl

LOCAL_LDLIBS := -lz
```

CMake 系统

把 `ssl`，`crypto`，`curl` 定义成 *STATIC IMPORTED* 库。

```
add_library(crypto STATIC IMPORTED)
set_target_properties(crypto
  PROPERTIES IMPORTED_LOCATION ${CMAKE_SOURCE_DIR}/libs/${ANDROID_ABI}/libcrypto.a)

add_library(ssl STATIC IMPORTED)
set_target_properties(ssl
  PROPERTIES IMPORTED_LOCATION ${CMAKE_SOURCE_DIR}/libs/${ANDROID_ABI}/libssl.a)

add_library(curl STATIC IMPORTED)
set_target_properties(curl
  PROPERTIES IMPORTED_LOCATION ${CMAKE_SOURCE_DIR}/libs/${ANDROID_ABI}/libcurl.a)
```

把它们和你的目标链接，如：

```
target_link_libraries( # Specifies the target library.
                       native-lib
```

```
curl
ssl
crypto
)
```

关于 "libcrypto.a(ui_openssl.o):ui_openssl.c:function read_string_inner: error: undefined reference to 'signal' " 报错解决

当你将编译好的 OpenSSL 静态库链接进你的安卓工程时, 可能会遇到如下报错

```
libcrypto.a(ui_openssl.o):ui_openssl.c:function read_string_inner: error: undefined reference to 'signal'

libcrypto.a(ui_openssl.o):ui_openssl.c:function read_string_inner: error: undefined reference to
'tcsetattr'
```

这时你需要使用低版本的 API 重新编译 OpenSSL 静态库, 推荐使用 android-16 或以下版本的 API 来重新编译

用低版本 API 编译的副作用是你无法编译出支持 64 位架构的静态库, 只能编译出支持 32 位架构的静态库

具体信息可以参考:

https://github.com/openssl/openssl/issues/988

http://stackoverflow.com/questions/37122126/whats-the-exact-significance-of-android-ndk-platform-version-compared-to-api-le

> 目前我们的编译脚本默认设置已经更改为使用 API 16 编译 32 位的 OpenSSL 静态库了

Reference / 参考资料

> 《How-To-Build-openssl-For-iOS》
>
> 《Compiling the latest OpenSSL for Android》
>
> 《在 Cocos2d-x 中使用 OpenSSL》
>
> 《using curl on iOS, I am unable to link with multiple architectures, CurlchkszEQ macro failing》
>
> 《porting libcurl on android with ssl support》