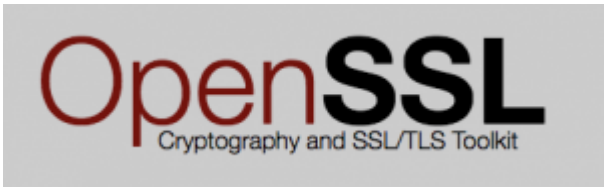
Scottyab's blog (https://scottyab.com/)

How to build and integrate OpenSSL into your Android NDK project



Recently I needed to work with OpenSSL in C/Cpp on Android and I couldn't find a simple way of including it. I looked at The Guardian project's openssl for Android (https://github.com/guardianproject/openssl-android) but it was very out of date. That's when I decided to go for compiling OpenSSL myself. This could of been a minefield but luckily there's a pre-configured build script that only requires a few modifications. This article aims to cover those modifications and how to integrate the compiled OpenSSL files into an NDK project.

n/tag/android/)

openssl_for_ios_and_android tools/script via github (https://github.com/leenjewel/openssl_for_ios_and_android)

I've upload my minor changes (no-zlib compile option) to this fork https://github.com/scottyab/openssl_for_ios_and_android

NOTE: for my purposes I only needed `lib-crypto` *and* `lib-ssl` *as I was focused on local only encryption. If you're looking to use networking in C/Cpp then you may also need to compile/include curl.*

Step 1: Downloads

- Clone https://github.com/leenjewel/openssl_for_ios_and_android (https://github.com/leenjewel/openssl_for_ios_and_androi)
- Download and install the Android NDK.

- Download and copy the downloaded openssl.tar file to `openssl_for_ios_and_android/tools` directory. I went for latest `openssl-1.1.0e.tar.gz` fromhttps://www.openssl.org/source/ (https://www.openssl.org/source/)

If you're new to the NDK check out this Intro to C for Android developers article (https://www.andriydruk.com/post/introduction-to-c-for-android-developers/) and the official docs (https://developer.android.com/ndk/guides/index.html)

Step 2: Prep build environment/script

Add `ANDROID_NDK` environment variable

Add the following line to ~/.bash_profile

```
export ANDROID_NDK=<path to NDK bundle>
```

update `build-openssl4android.sh` to use the downloaded openssl version

ramming/)

~line 20 `LIB_NAME="openssl-1.0.2k"` to the version you downloaded `LIB_NAME="openssl-1.1.0e"`

update `build-openssl4android.sh` to change the zlib compile option

Change the zlib compile option to no-zlib (if you are not using the scottyab (https://github.com/scottyab/openssl_for_ios_and_android) fork). Without this change I had build failure `app:externalNativeBuildDebug failed with vairous cmake errors i.e c_zlib.c:(.text+0xbc): undefined reference to` deflate`. Based on recommendations from this SO issue (http://stackoverflow.com/questions/18185618/static-linking-app-with-openssl-c-c-x86-x64).

~Line 53 `zlib \` to `no-zlib \`

Step 3: build

Start the build `$ ./build-openssl4android.sh`

Step 4: Copy output to your Android NDK project

- Copy the runtimes you want to support, i.e arm, x86, mips from `openssl_for_ios_and_android/output` to `<project root>/distribution/openssl`.
- Rename the directions to remove the `openssl-` prefix.

Step 5: update cmakerlists.txt file

I used this NDK Samples app Hello-libs (https://github.com/googlesamples/android-ndk/tree/master/hello-libs) as basis for my NDK project setup. Where the native-lib cpp file and cmakerlists.txt are already created/setup.

- Add the following lines (The references to `native-lib` is where your Cpp code will likely be)

rized/)

```
//configure import libs

set(distribution_DIR ${CMAKE_SOURCE_DIR}/../../../../distribution)

//add the open ssl crypto lib
add_library(libcrypto STATIC IMPORTED)
set_target_properties(libcrypto PROPERTIES IMPORTED_LOCATION
${distribution_DIR}/openssl/${ANDROID_ABI}/lib/libcrypto.a)

# add the open ssl ssl lib

add_library(libssl STATIC IMPORTED)

set_target_properties(libssl PROPERTIES IMPORTED_LOCATION

${distribution_DIR}/openssl/${ANDROID_ABI}/lib/libssl.a)

//add to target_include_directories

target_include_directories(native-lib PRIVATE

${distribution_DIR}/openssl/${ANDROID_ABI}/include)

//add to target_link_libraries

target_link_libraries( # Specifies the target library.

native-lib

# Links the openssl crypto

libcrypto

libssl

${log-lib} )
```

Step 6: Finish / Build in gradle

That's it you should be good to go and ready to start using openssl in your c and cpp files.

```
./gradlew assemble
```

Previous

What is the most well-designed app you've used?
(https://scottyab.com/2016/12/what-is-the-most-well-designed-app-youve-used/)

Next

How much does it cost to attend Google IO from the UK?
(https://scottyab.com/2017/05/how-much-does-it-cost-to-attend-io-from-uk/)