交叉编译带RTMP模块的Nginx到Android

 发表于 2017-07-11 |  更新于: 2018-07-31 |  [32 Comments](https://zhangtom.com/2017/07/11/%E4%BA%A4%E5%8F%89%E7%BC%96%E8%AF%91%E5%B8%A6RTMP%E6%A8%A1%E5%9D%97%E7%9A%84Nginx%E5%88%B0Android/#comments)

因为工作需要在Android上接受RTMP推流，只能想办法架设一个RTMP服务器。调研了一下决定用Nginx配合RTMP模块。不过目前为止，Nginx对交叉编译的支持还不是很友好，而且Android的C库bionic相比GNU的C库glibc还是有些差异(没有glob，没有crypt)。尽管参考了不少博客，一路上还是踩了不少坑，整理记录一下。

**2018.7.31更新**

1. macOS High Sierra 10.13.6, Android Studio 3.1.3 测试可用
2. 脚本目前只支持 ndk r15c（[这里下载](https://developer.android.com/ndk/downloads/older_releases)）

**主机环境**

系统：macOS Sierra 10.12.5

**准备工作**

为了方便，直接用的Android Studio 2.3.3，主要是用到了adb，还需要下好ndk。

在Android Studio的Preferences里直接搜索sdk，可以找到Android SDK Location，adb位于platform-tools目录内，而ndk-bundle就是ndk目录。

1. adb默认目录：

$HOME/Library/Android/sdk/platform-tools

1. ndk默认目录：

$HOME/Library/Android/sdk/ndk-bundle

**将adb所在目录加入环境变量，以便后续使用：**

|  |  |
| --- | --- |
| 1 | export PATH="$HOME/Library/Android/sdk/platform-tools:$PATH" |

[**直达总结 >>>**](https://zhangtom.com/2017/07/11/%E4%BA%A4%E5%8F%89%E7%BC%96%E8%AF%91%E5%B8%A6RTMP%E6%A8%A1%E5%9D%97%E7%9A%84Nginx%E5%88%B0Android/#%E6%80%BB%E7%BB%93)

**编译**

**交叉编译OpenSSL**

虽然Nginx支持指定OpenSSL源码编译，但并不是交叉编译的，最终链接时会有问题，因此需要先交叉编译OpenSSL。这部分参考OpenSSL的[官方wiki](https://wiki.openssl.org/index.php/Android" \t "_blank)。

**环境脚本**

先下载[Setenv-android.sh](https://wiki.openssl.org/images/7/70/Setenv-android.sh" \t "_blank)。

编辑该文件：

* 第18行的\_ANDROID\_NDK变量最终用于生成ANDROID\_NDK\_ROOT。之前我们已经知道ndk目录了，因此可以直接设置ANDROID\_NDK\_ROOT，将第11行（空行）替换为：

|  |  |
| --- | --- |
| 1 | export ANDROID\_NDK\_ROOT=$HOME/Library/Android/sdk/ndk-bundle |

* 等号右边部分就是ndk目录，按需更改
* 第25行的\_ANDROID\_EABI用于指定EABI，可以在$ANDROID\_NDK\_ROOT/toolchains下找到，常见的Android一般运行在arm架构的CPU上，因此填*arm-linux-androideabi*对应的那个，我这里是arm-linux-androideabi-4.9。如果是x86架构的，就应该填x86对应的那个，同时要记得改第30行的\_ANDROID\_ARCH变量
* 第39行的\_ANDROID\_API指定Android API等级，可以根据需要改，我改成了21
* 第122行的引号改为括号：

|  |  |
| --- | --- |
| 1 | ANDROID\_TOOLS=(arm-linux-androideabi-gcc arm-linux-androideabi-ranlib arm-linux-androideabi-ld) |

* 否则第132行的for循环会有问题。同样地，x86架构的需要将第125行的引号改为括号
* 第201行的==改为=
* 把行结束符改为当前系统对应的行结束符（例如在Sublime Text 3中，View > Line Endings可以切换），否则脚本不能正常执行。下载下来的是Windows行结束符。

编辑后完整的脚本见[GitHub](https://github.com/tatowilson/Cross-Compile-Nginx-with-RTMP-Module-for-Android)，在终端中给脚本加上执行权限：

|  |  |
| --- | --- |
| 1 | chmod +x Setenv-android.sh |

**编译**

从[官网](https://www.openssl.org/source/" \t "_blank)下载最新LTS版本的OpenSSL源码（当前最新版是1.1.0f），解压到Setenv-android.sh脚本所在目录

依次执行以下命令：

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 | # 执行环境脚本，第一个句点不能省略 . ./Setenv-android.sh export OPENSSL\_DIR=/usr/local/ssl/$ANDROID\_API # 进入openssl源码目录，版本号按需更改 cd openssl-1.1.0f/ # 生成Makefile KERNEL\_BITS=32 ./config shared no-ssl2 no-ssl3 no-comp no-hw no-engine \  --openssldir=$OPENSSL\_DIR --prefix=$OPENSSL\_DIR make depend make all # -E 保留当前的环境变量给root用户 sudo -E make install CC=$ANDROID\_TOOLCHAIN/arm-linux-androideabi-gcc RANLIB=$ANDROID\_TOOLCHAIN/arm-linux-androideabi-ranlib |

**编译Nginx**

**准备Nginx源码**

下载最新稳定版Nginx的[源码](http://nginx.org/en/download.html" \t "_blank)、RTMP模块的[源码](https://github.com/arut/nginx-rtmp-module" \t "_blank)，放在同一目录下。由于Nginx的configure在执行过程中会编译一些测试程序来获取一些信息，而我们交叉编译出来的测试程序不可能在宿主机上运行，会导致获取信息有误，因此要做一些处理，通过adb在Android上执行测试程序来使其正常工作：

* 编辑nginx/auto/feature文件  
  在if [ -x $NGX\_AUTOTEST ]; then和case "$ngx\_feature\_run" in之间添加：

|  |  |
| --- | --- |
| 1 | adb push $NGX\_AUTOTEST /data/local/tmp 2>&1 >/dev/null |

* 并且在该case结束后（*esac之后*）添加：

|  |  |
| --- | --- |
| 1 | adb shell rm /data/local/tmp/$(basename $NGX\_AUTOTEST) |

* 再将/bin/sh -c $NGX\_AUTOTEST和`$NGX\_AUTOTEST`全部替换成：

|  |  |
| --- | --- |
| 1 | adb shell /data/local/tmp/$(basename $NGX\_AUTOTEST) |

* 这一步是将测试程序通过adb拷贝到Android的/data/local/tmp目录，在Android上执行测试程序，并在结束测试后删除测试程序，因此要事先确保adb能够正常工作
* 编辑nginx/auto/include文件  
  将ngx\_test="$CC -o $NGX\_AUTOTEST $NGX\_AUTOTEST.c"替换成:

|  |  |
| --- | --- |
| 1 | ngx\_test="$CC $CC\_AUX\_FLAGS -o $NGX\_AUTOTEST $NGX\_AUTOTEST.c" |

* 测试头文件时，要确保*--sysroot*参数正确，如果不正确就通过设置*CC\_AUX\_FLAGS*环境变量来指定，因此这里添加$CC\_AUX\_FLAGS，并且在执行configure之前执行：

|  |  |
| --- | --- |
| 1 | export CC\_AUX\_FLAGS="--sysroot=$ANDROID\_SYSROOT" |

* 编辑nginx/auto/types/sizeof文件  
  将ngx\_size=`$NGX\_AUTOTEST`替换为：

|  |  |
| --- | --- |
| 1 2 3 | adb push $NGX\_AUTOTEST /data/local/tmp 2>&1 >/dev/null ngx\_size=`adb shell /data/local/tmp/$(basename $NGX\_AUTOTEST)` adb shell rm /data/local/tmp/$(basename $NGX\_AUTOTEST) |

* 编辑nginx/auto/lib/openssl/conf文件  
  因为编译openssl时没有用.openssl子目录，所以要将.openssl/全部移除
* 编辑nginx/src/os/unix/ngx\_user.c源文件  
  由于Android的C库没有crypt，因此需要修改该文件中对crypt()函数的调用，改成调用OpenSSL中的DES\_crypt()方法  
  引入头文件：

|  |  |
| --- | --- |
| 1 | #include <openssl/des.h> |

* 将调用value = crypt((char \*) key, (char \*) salt);改为DES\_crypt():

|  |  |
| --- | --- |
| 1 | value = DES\_crypt((char \*) key, (char \*) salt); |

**准备glob**

由于Android的C库bionic没有glob，需要下载相应源码，[glob.h](https://raw.githubusercontent.com/white-gecko/TokyoCabinet/master/glob.h" \t "_blank)和[glob.c](https://raw.githubusercontent.com/white-gecko/TokyoCabinet/master/glob.c" \t "_blank)

* 编辑glob.h  
  删除以下代码：

|  |  |
| --- | --- |
| 1 2 3 4 5 6 | #include <sys/\_types.h>  #ifndef \_SIZE\_T\_DECLARED typedef \_\_size\_t size\_t; #define \_SIZE\_T\_DECLARED #endif |

* 编辑glob.c  
  删除对issetugid()的调用
* 将glob.h和glob.c文件移动或复制到nginx/src/os/unix目录
* 编辑nginx/auto/sources文件  
  在UNIX\_DEPS中添加glob.h的路径：

|  |  |
| --- | --- |
| 1 | src/os/unix/glob.h \ |

* 在UNIX\_SRCS中添加glob.c的路径：

|  |  |
| --- | --- |
| 1 | src/os/unix/glob.c \ |

* 编辑好的glob.h和glob.c见[GitHub](https://github.com/tatowilson/Cross-Compile-Nginx-with-RTMP-Module-for-Android" \t "_blank)

**设置、导出环境变量**

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 | START\_DIR=$PWD RTMP\_MODULE\_DIR=$START\_DIR/nginx-rtmp-module CROSS\_COMPILE\_GCC="$ANDROID\_TOOLCHAIN/$CROSS\_COMPILE"gcc # 放置编译完成后文件的目录 export DESTDIR=$START\_DIR # 在configure执行时的辅助参数，指定--sysroot，以确保测试程序能正常进行 export CC\_AUX\_FLAGS="--sysroot=$ANDROID\_SYSROOT" |

**生成Makefile**

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 | # 进入nginx目录，版本号按需更改 cd nginx-1.12.0  ./configure \ --crossbuild=android-arm \ --prefix=/sdcard/nginx \ --with-http\_ssl\_module \ --with-openssl=$OPENSSL\_DIR \ --without-http\_gzip\_module \ --without-pcre \ --without-http\_rewrite\_module \ --without-http\_proxy\_module \ --without-http\_userid\_module \ --without-http\_upstream\_zone\_module \ --without-stream\_upstream\_zone\_module \ --add-module=$RTMP\_MODULE\_DIR \ --with-cc=$CROSS\_COMPILE\_GCC \ --with-cc-opt="--sysroot=$ANDROID\_SYSROOT -Wno-sign-compare -pie -fPIE" \ --with-ld-opt="--sysroot=$ANDROID\_SYSROOT -pie -fPIE" |

**编译**

|  |  |
| --- | --- |
| 1 2 | make -j8 make install -j8 |

编译成功的文件位于$DESTDIR/sdcard/nginx目录

**总结**

为使用方便，整理了相关脚本，见[GitHub](https://github.com/tatowilson/Cross-Compile-Nginx-with-RTMP-Module-for-Android" \t "_blank)执行脚本前，确保adb和ndk都正常可以使用，并且下载好相关源码：

* 下载[Nginx最新稳定版源码压缩包](http://nginx.org/en/download.html" \t "_blank)（make\_nginx.sh会自动解压）http://nginx.org/en/download.html
* 下载[nginx-rtmp-module最新稳定版源码](https://github.com/arut/nginx-rtmp-module" \t "_blank) https://github.com/arut/nginx-rtmp-module
* 下载[openssl最新稳定LTS版源码、 openssl-fips最新版源码](https://www.openssl.org/source/" \t "_blank)，并解压<https://www.openssl.org/source/>

<https://www.openssl.org/source/openssl-1.1.0j.tar.gz>

https://www.openssl.org/source/openssl-fips-2.0.16.tar.gz

* 下载[GitHub](https://github.com/tatowilson/Cross-Compile-Nginx-with-RTMP-Module-for-Android" \t "_blank)仓库代码，并组织成如下目录结构：

|  |  |
| --- | --- |
| 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 | . ├── Setenv-android.sh ├── glob（glob.c和glob.h文件由make\_nginx.sh脚本在执行过程中复制到nginx/src/os/unix目录下） │   ├── glob.c │   └── glob.h ├── make\_nginx.sh ├── make\_openssl.sh │ ├── nginx-1.12.0/（该目录由make\_nginx.sh脚本自动解压对应的压缩包生成） │   └── ... ├── nginx-1.12.0.tar.gz（Nginx源码压缩包，版本号可以不一样） ├── nginx-rtmp-module/ │   └── ... ├── openssl-1.1.0f/（openssl源码，版本号可以不一样） │   └── ... ├── openssl-fips-2.0.16/（openssl-fips源码，版本号可以不一样） │   └── ... └── sdcard（该目录在编译成功后会自动生成）    └── nginx |

依次执行脚本即可

|  |  |
| --- | --- |
| 1 2 3 | # 第一个句点不能省略，用于保留脚本之间的环境变量 . ./make\_openssl.sh . ./make\_nginx.sh |

**参考资料**

* [编译nginx到Android上踩坑了](http://daozhao.goflytoday.com/2015/04/compile-nginx-on-android/)
* [nginx-1.0.12交叉编译](http://www.cppblog.com/van201314/articles/165661.html)
* [在64位Ubuntu下面使用android NDK编译nginx](http://blog.csdn.net/zangcf/article/details/22688873)
* [拥抱Android：编译nginx搭建移动平台](http://blog.csdn.net/prog_6103/article/details/49744633)
* [交叉编译php5,、nginx、squid方法](http://www.cnblogs.com/tolimit/p/4371995.html)
* [nginx 交叉编译 ( checking for C compiler found but is not working )](http://blog.csdn.net/fish43237/article/details/40515897)
* [Cross compile nginx 1.11 for arm](http://blog.xuite.net/raibura/dorama/454878569-Cross+compile+nginx+1.11+for+arm+)

二、遇到的问题

（1）问题内容：

checking for C compiler ... found but is not working

./configure error : C compiler gcc is not found

（2）原因分析：

configure首先会编译一个小测试程序，通过测试其运行结果来判断编译器是否能正常工作，由于交叉编译器所编译出的程序是无法在编译主机上运行的，故而产生此错误。

（3）解决办法：

编辑auto/cc/name文件，将21行的“exit 1”注释掉（令测试程序不会报错）。

### 运行报错：

./sbin/nginx -p /data/nginx/ -t

nginx: [emerg] getpwnam("1000") failed (2: No such file or directory) in /data/nginx/conf/nginx.conf:3

修改nginx.conf，将#user nobody;改为user nobody;

# 运行方式

Server:

sed -i 's/#user nobody/user root/g' sdcard/nginx/conf/nginx.conf

adb push sdcard/nginx /data

adb shell

Stb

cd /data/nginx

**./sbin/nginx -p /data/nginx/ -t**

./sbin/nginx -p /data/nginx -c conf/nginx.conf

./sbin/nginx -p /data/nginx -c conf/nginx.conf -s reload