驾驭开源库 WebRTC

第三章-编译 Android 版

作者：adam 鸣谢：老张、aaalaniz

日期：2015-4-6

版本：1.0.0

欢迎转载，有问题反馈 Q：2780113541，尽量完善系列教程。

depot_tools 代理设置参考老张《史上最牛逼的墙内下载 webrtc 代码说明_20150401》

编译定制参考 aaalaniz 的脚本：

https://github.com/pristineio/webrtc-build-scripts/blob/master/android/build.sh

0、官方源码 svn 自 2015-3-24 已迁至谷歌 git

一些使用 svn 地址的教程已不适用或需改进

1、部署 http(s)转 sock5 代理，具体参考第一章

Shadowsocks：代理地址 127.0.0.1:1080

Privoxy：代理地址 127.0.0.1:8118

2、编译环境准备

Ubuntu14.04x64 主机或虚拟机一台，root 帐户登陆，其他账户命令请加 sudo

本教程以/root 作为工作区，保证剩余空间 30G 以上，下来的总大小可能十几 G。

3、下载依赖库

终端输入，下面非黑色同色为一行：

cd /root

apt-get -y install wget git gnupg flex bison gperf build-essential zip curl subversion pkg-config clang libgtk2.0-dev
curl -o install-build-deps-android.sh https://src.chromium.org/svn/trunk/src/build/install-build-deps-android.sh
curl -o install-build-deps.sh https://src.chromium.org/svn/trunk/src/build/install-build-deps.sh

chmod 777 *.sh

sudo /bin/bash ./install-build-deps-android.sh

4、下载工具准备

cd /root

git config --global http.proxy http://127.0.0.1:8118

git config --global https.proxy https://127.0.0.1:8118

git clone https://chromium.googlesource.com/chromium/tools/depot_tools.git

5、设置下载工具路径至环境变量

echo 'export PATH=/root/depot_tools/:$PATH' >> ~/.bash_profile

source ~/.bash_profile

6、下载工具代理设置以及下载 webrtc

vi /root/depot_tools/http_proxy.boto，内容如下：

[Boto]

proxy = 127.0.0.1

proxy_port = 8118

cd /root

mkdir webrtc_android

cd webrtc_android

设置下载工具代理环境变量：

export http_proxy=http://127.0.0.1:8118

export https_proxy=http://127.0.0.1:8118

设置下载工具 git 代理：

git config --global http.proxy http://127.0.0.1:8118

或修改配置文件 %user_home%\.gitconfig [对特定仓库的话就是 .git/config]，增加：

[http]

proxy = http://127.0.0.1:8118

设置 BOTO 代理，解决 download google storage 失败问题：

export NO_AUTH_BOTO_CONFIG=/root/depot_tools/http_proxy.boto

export GYP_DEFINES="$GYP_DEFINES OS=android"

首次下载输入下载命令：

fetch webrtc_android

等待十几 G 下载完成，最好没报错。

报错说明网络或上边代理设置存在问题，排除问题然后执行如下命令:

下载代码:

git pull 或 git fetch

下载依赖项生成编译文件:

gclient sync

6、设置 android 的环境变量，ninja 编译

source src/build/android/envsetup.sh

Debug 版本：ninja -C out/Debug

Release 版本：ninja -C out/Release

如果没出过错，一次就编译成了，如果出错请检查前边那些设置有问题并重复 gclient sync

确保下载完整，重新编译。

编译好的库和 demo 在 src\out 下面。

----------进阶----------

1、代码和依赖项更新

cd /root/webrtc_android

export PATH=$PATH:/root/depot_tools

export GYP_DEFINES="$GYP_DEFINES OS=android"

同步下载源码：

git pull 或 git fetch

同步下载依赖项并根据 GYP_GENERATORS 的设置生成编译文件：

gclient sync

只同步下载依赖项不生成编译文件：

gclient sync --nohooks

只根据 GYP_GENERATORS 的设置生成编译文件：

gclient runhooks

2、VPS 编译打包下载

如果使用的是 Ubuntu 版 VPS 编译，整个 WEBRTC 是可以打包下来用的，解压时注意选项，不要使连接符号失效。

先备份 src 下的 out 目录，然后编译测试通过后，生成的 out 目录会很大。

删掉 out，用备份的 out 替换，打包（推荐 7z），这样你就有份完整的原始包了

Ubuntu 自带 lighthttp 服务，包放到/var/www，用支持续传的工具下载下来。

下不来就是网站有类型过滤，改个后缀为 jpg 就可以了。

3、定制编译

如下命令默认 armv7a 架构下的所有库和 demo 都编译

ninja -C out/Debug

实际上所有 android 架构都支持，比如 arm64、ia32 、x86、x64 等

比如我只想编 arm64 下的 WebRTCDemo：

```
cd /root/webrtc_android
source src/build/android/envsetup.sh
export GYP_GENERATORS="ninja"
export GYP_DEFINES="$GYP_DEFINES OS=android target_arch=arm64 target_subarch=arm64"
export GYP_GENERATOR_FLAGS="output_dir=out_android_arm64-v8a"
export GYP_CROSSCOMPILE=1
gclient runhooks
ninja -C out_android_arm64-v8a/Debug WebRTCDemo
```

参考：

http://www.webrtc.org/native-code/development

https://github.com/pristineio/webrtc-build-scripts/blob/master/android/build.sh