零声学院 音视频高级教程 ubuntu ffmpeg开发环境搭建
QQ 326873713 课程地址：5G时代-音视频高级开发
配置环境 ubuntu desktop 16.04 + ffmpeg 4.2.1
建议在《编译与安装》一节，都使用源码的方式编译和安装，本编译安装方式最终生成的皆为静态库。
部分命令说明：
git -C [git-command] 指定其它路径的仓库 执行命令 注意，-C 要在命令之前。

该教程的网页版本地址：https://www.yuque.com/docs/share/d4f0e320-4006-4038-8ab0-bba8585b397c?# 《零声学院 音视频高级教程 ubuntu ffmpeg开发环境搭建0.4》

1 创建目录

在home目录下创建
ffmpeg_sources：用于下载源文件
ffmpeg_build： 存储编译后的库文件
bin：存储二进制文件（ffmpeg，ffplay，ffprobe，X264，X265等）
cd ~
mkdir ffmpeg_sources ffmpeg_build bin

2 安装依赖

更新
sudo apt-get update
安装需要的组件
```
sudo apt-get -y install \
  autoconf \
  automake \
  build-essential \
  cmake \
  git-core \
  libass-dev \
  libfreetype6-dev \
  libsdl2-dev \
  libtool \
  libva-dev \
  libvdpau-dev \
  libvorbis-dev \
  libxcb1-dev \
  libxcb-shm0-dev \
  libxcb-xfixes0-dev \
  pkg-config \
  texinfo \
  wget \
  zlib1g-dev
```

3 编译与安装

本指南假定您要安装一些最常见的第三方库。每个小节为您提供安装该库所需的命令。
如果不需要某些功能，则可以跳过相关小节（如果不需要），然后在FFmpeg中删除相应的./configure选项。例如，如果不需要libvpx，请跳过该小节，然后从“ 安装FFmpeg”部分中删除--enable-libvpx。
提示：如果要在多核系统中加快编译速度，可以在每个make命令（例如make -j4）中使用-j选项。
建议都使用源码进行安装。

NASM

部分库使用到汇编程序。
使用源码进行安装

```
cd ~/ffmpeg_sources && \
wget https://www.nasm.us/pub/nasm/releasebuilds/2.14.02/nasm-2.14.02.tar.bz2 && \
tar xjvf nasm-2.14.02.tar.bz2 && \
cd nasm-2.14.02 && \
./autogen.sh && \
PATH="$HOME/bin:$PATH" ./configure --prefix="$HOME/ffmpeg_build" --bindir="$HOME/bin" && \
make && \
make install
```

Yasm

部分库使用到该汇编库
使用源码进行安装:

```
cd ~/ffmpeg_sources && \
wget -O yasm-1.3.0.tar.gz https://www.tortall.net/projects/yasm/releases/yasm-1.3.0.tar.gz && \
tar xzvf yasm-1.3.0.tar.gz && \
cd yasm-1.3.0 && \
./configure --prefix="$HOME/ffmpeg_build" --bindir="$HOME/bin" && \
make && \
make install
```

libx264

H.264视频编码器。更多信息和使用范例参考H.264 Encoding Guide
要求编译ffmpeg时配置：--enable-gpl --enable-libx264.
使用源码进行编译：

```
cd ~/ffmpeg_sources && \
git -C x264 pull 2> /dev/null || git clone --depth 1 https://gitee.com/mirrors_addons/x264.git && \
cd x264 && \
PATH="$HOME/bin:$PATH" PKG_CONFIG_PATH="$HOME/ffmpeg_build/lib/pkgconfig" ./configure --prefix="$HOME/ffmpeg_build" --bindir="$HOME/bin" --enable-static --enable-pic && \
PATH="$HOME/bin:$PATH" make && \
make install
```

libx265

H.265/HEVC 视频编码器， 更多信息和使用范例参考H.265 Encoding Guide。
要求编译ffmpeg时配置：--enable-gpl --enable-libx265.
使用源码进行编译：

```
sudo apt-get install mercurial libnuma-dev && \
cd ~/ffmpeg_sources && \
```

if cd x265 2> /dev/null; then git pull && cd ..; else git clone https://gitee.com/mirrors_videolan/x265.git; fi && \
cd x265/build/linux && \
PATH="$HOME/bin:$PATH" cmake -G "Unix Makefiles" -DCMAKE_INSTALL_PREFIX="$HOME/ffmpeg_build" -DENABLE_SHARED=off ../../source && \
PATH="$HOME/bin:$PATH" make && \
make install

libvpx

VP8/VP9视频编解码器。 更多信息和使用范例参考VP9 Video Encoding Guide 。
要求编译ffmpeg时配置： --enable-libvpx.
使用源码进行编译：
cd ~/ffmpeg_sources && \
git -C libvpx pull 2> /dev/null || git clone --depth 1 https://github.com/webmproject/libvpx.git && \
cd libvpx && \
PATH="$HOME/bin:$PATH" ./configure --prefix="$HOME/ffmpeg_build" --disable-examples --disable-unit-tests --enable-vp9-highbitdepth --as=yasm --enable-pic && \
PATH="$HOME/bin:$PATH" make && \
make install

libfdk-aac

AAC音频编码器. 更多信息和使用范例参考AAC Audio Encoding Guide。
要求编译ffmpeg时配置：--enable-libfdk-aac (如果你已经配置了 --enable-gpl则需要加上--enable-nonfree).
使用源码进行编译：
cd ~/ffmpeg_sources && \
git -C fdk-aac pull 2> /dev/null || git clone --depth 1 https://github.com/mstorsjo/fdk-aac && \
cd fdk-aac && \
autoreconf -fiv && \
./configure CFLAGS=-fPIC --prefix="$HOME/ffmpeg_build" && \
make && \
make install

libmp3lame

MP3音频编码器.
要求编译ffmpeg时配置：--enable-libmp3lame.

使用源码进行编译：
cd ~/ffmpeg_sources && \
git clone --depth 1 https://gitee.com/hqiu/lame.git && \
cd lame && \
PATH="$HOME/bin:$PATH" ./configure --prefix="$HOME/ffmpeg_build" --bindir="$HOME/bin" --enable-nasm --with-pic && \
PATH="$HOME/bin:$PATH" make && \
make install

libopus

Opus音频编解码器.
要求编译ffmpeg时配置：--enable-libopus.
使用源码进行编译：

```
cd ~/ffmpeg_sources && \
git -C opus pull 2> /dev/null || git clone --depth 1 https://github.com/xiph/opus.git && \
cd opus && \
./autogen.sh && \
./configure --prefix="$HOME/ffmpeg_build"  -with-pic&& \
make && \
make install
```

FFmpeg

注意注意：如果要安装debug版本，请参考第6章节 《6 支持FFmpeg 代码 Debug》配置ffmpeg。

```
cd ~/ffmpeg_sources && \
wget -O ffmpeg-4.2.1.tar.bz2 https://ffmpeg.org/releases/ffmpeg-4.2.1.tar.bz2 && \
tar xjvf ffmpeg-4.2.1.tar.bz2 && \
cd ffmpeg-4.2.1 && \
PATH="$HOME/bin:$PATH" PKG_CONFIG_PATH="$HOME/ffmpeg_build/lib/pkgconfig" CFLAGS="-O3 -fPIC" ./configure \
  --prefix="$HOME/ffmpeg_build" \
  --pkg-config-flags="--static" \
  --extra-cflags="-I$HOME/ffmpeg_build/include" \
  --extra-ldflags="-L$HOME/ffmpeg_build/lib" \
  --extra-libs="-lpthread -lm" \
  --bindir="$HOME/bin" \
  --enable-gpl \
  --enable-libass \
  --enable-libfdk-aac \
  --enable-libfreetype \
  --enable-libmp3lame \
  --enable-libopus \
  --enable-libvorbis \
  --enable-libvpx \
  --enable-libx264 \
  --enable-libx265 \
  --enable-pic \
  --enable-shared   \
  --enable-nonfree && \
PATH="$HOME/bin:$PATH" make && \
make install && \
hash -r
```

然后重新登录系统或者在当前shell会话执行如下命令以识别新安装ffmpeg的位置：

```
source ~/.profile
```

现在已经完成编译和安装ffmpeg (also ffplay, ffprobe, lame, x264, & x265) 。该文档剩余章节主要讲如何更新和删除ffmepg。
编译完成后，ffmpeg_build ffmpeg_sources bin目录的大体情况

```
lqf@ubuntu:~/ffmpeg_build$ ls
bin  include  lib  share
```

lqf@ubuntu:~/ffmpeg_sources$ ls
fdk-aac lame-3.100.tar.gz opus yasm-1.3.0.tar.gz
ffmpeg-4.2.1 libvpx x264
ffmpeg-4.2.1.tar.bz2 nasm-2.14.02 x265
lame nasm-2.14.02.tar.bz2 yasm-1.3.0

lqf@ubuntu:~/bin$ ls
ffmpeg ffplay ffprobe lame nasm ndisasm vsyasm x264 yasm ytasm

4 使用

现在，您可以打开一个终端，输入ffmpeg命令，它应该执行新的ffmpeg。
如果你需要多个用户能同时使用你新编译的ffmpeg，则可以移动或者拷贝ffmpeg二进制文件从~/bin到/usr/local/bin。
测试ffplay是否可以使用（需要在图形方式进行测试）
ffplay rtmp://media3.scctv.net/live/scctv_800
如果能够正常播放则说明 ffplay能够编译成功使用。播放的时候要等等画面。

5 文档

你可以使用 man ffmpeg以本地的方式访问文档:
echo"MANPATH_MAP $HOME/bin $HOME/ffmpeg_build/share/man" >> ~/.manpath
你可能必须注销系统然后重新登录man ffmpeg才生效。
HTML 格式的文档位于 ~/ffmpeg_build/share/doc/ffmpeg.
你也可以参考在线文档 online FFmpeg documentation,

6 支持FFmpeg 代码 Debug

刚才的工程可以运行，但不能debug。解决此问题，首先认定一点，生成的可执行程序中，ffmpeg 不包含调试信息，调试信息在 ffmpeg_g 中,debug 要选择 ffmpeg_g。
另外，./config选项也是确定包含调试信息的核心，需要在config中添加：

- --enable-debug=3：开启debug调试
- --disable-asm：禁用 asm 优化
- --disable-optimizations：禁用优化，以便调试时按函数顺序执行。
- --disable-stripping：禁用剥离可执行程序和共享库，即调试时可以进入到某个函数进行单独调试。

采用以下命令重新config:
先clean

```
make clean
```

再重新config

```
./configure \
  --prefix="$HOME/ffmpeg_build" \
  --pkg-config-flags="--static" \
  --extra-cflags="-I$HOME/ffmpeg_build/include" \
  --extra-ldflags="-L$HOME/ffmpeg_build/lib" \
  --extra-libs="-lpthread -lm" \
  --bindir="$HOME/bin" \
```

```
  --enable-gpl \
  --enable-libass \
  --enable-libfdk-aac \
  --enable-libfreetype \
  --enable-libmp3lame \
  --enable-libopus \
  --enable-libvorbis \
  --enable-libvpx \
  --enable-libx264 \
  --enable-libx265 \
  --enable-pic \
  --enable-shared   \
  --enable-nonfree \
  --enable-debug=3 \
  --disable-optimizations \
  --disable-asm \
  --disable-stripping && \
PATH="$HOME/bin:$PATH" make && \
make install && \
hash -r
```

一些注意事项：

1. 在使用 ffplay 播放生成 h264 格式的视频时，播放速度会加快，解决方式：不要使用 FFmpeg 转码生成纯 h264 格式的视频，要使用一种容器包含 h264 视频，即生成一种音视频流格式，也就是不要生成纯粹的 h264 码流，而是生成诸如 mkv 等格式的文件。