流媒体高级编程

STREAMING MEDIA DAY05

内容

上午	09:00 ~ 09:30	作业讲解和回顾
	09:30 ~ 10:20	屏幕流录制
	10:30 ~ 11:20	
	11:30 ~ 12:20	
下午	14:00 ~ 14:50	混合流录制
	15:00 ~ 15:50	
	16:00 ~ 16:50	
	17:00 ~ 17:30	总结和答疑

屏幕流录制

屏幕流录制

- 需求分析 — 需求分析
- 概要设计 — 概要设计
- 编码实现 — av_gettime

需求分析

需求分析

- 录制屏幕流
  - 抓取视频显示器上的动态影像保存到本地或推送至远程
    - ✓ 本地录制：ScreenRecorder screen.flv
    - ✓ 推送直播：ScreenRecorder rtmp://192.168.1.166/live/1

知识讲解

概要设计

概要设计

进程入口
main

录制
record

录制屏幕
recordScreen

打印源格式信息
printSrcFmt

打印目标格式信息
printDstFmt

打印解码包信息
printDecPkt

打印编码包信息
printEncPkt

打印视频帧信息
printVidFrm

打印FFmpeg错误
ffmpegError

事件处理
doEvent

编码实现

av_gettime

- 获取当前系统时间
  - `#include <libavutil/time.h>`
    ```
    int64_t av_gettime (void);
    ```
  - 返回始自1970年1月1日0点0分0秒，直到函数被调用时的总微秒数，1微秒=10^{-6}秒，即百万分之一秒

知识讲解

Tarena Technology 达内科技

ScreenRecorder

【参见：FFmpeg/Primer/ScreenRecorder】

课堂练习

- 录制屏幕流
  - 抓取视频显示器上的动态影像保存到本地或推送至远程
    - ✓ 本地录制：ScreenRecorder screen.flv
    - ✓ 推送直播：ScreenRecorder rtmp://192.168.1.166/live/1

混合流录制

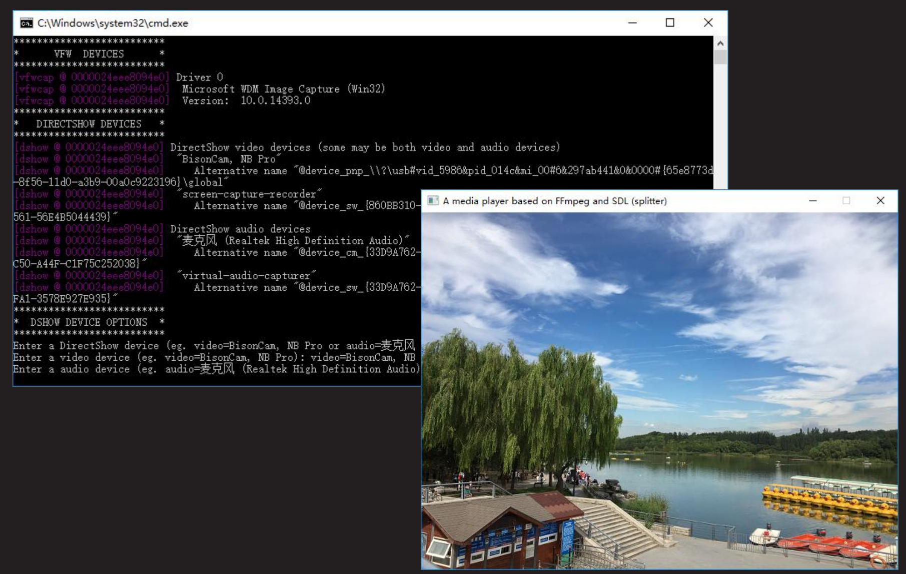
混合流录制

- 需求分析 — 需求分析
- 概要设计 — 概要设计
- 编码实现 — av_compare_ts

需求分析

需求分析

- 录制音视频混合流
  - 在Windows上列表显示所有的Video For Windows设备和DirectShow设备及其选项
  - 通过用户选择的采集设备，捕获音视频数据，保存或推送
  - 分别在独立的线程中，以并发的方式，采集音频和视频流，避免因捕获过程的相互等待，丢失帧数据
  - 借助独立的混流线程，将捕获到的音视频帧，按解码时间戳的升序，依次写入目标格式中



知识讲解

概要设计

概要设计

知识讲解


进程入口
main
录制(主线程)
record
打印设备信息
printDevices
录制视频(子线程)
recordVideo
录制音频(子线程)
recordAudio
视频队列
音频队列
混流(子线程)
mux
字符编码转换
gbk2utf8
打印源格式信息
printSrcFmt
打印解码包信息
printDecPkt
打印视频帧信息
printVidFrm
打印目标格式信息
printDstFmt
打印编码包信息
printEncPkt
打印音频帧信息
printAudFrm
打印FFmpeg错误
ffmpegError
打印SDL错误
sdlError
事件处理
doEvent

编码实现

av_compare_ts

- 比较时间戳
  - #include <libavutil/mathematics.h>

```
int av_compare_ts (
  int64_t    ts_a,  // 时间戳a
  AVRational tb_a,  // 时间戳a的单位
  int64_t    ts_b,  // 时间戳b
  AVRational tb_b); // 时间戳b的单位
```

  - 若时间戳a早先于时间戳b，则返回-1；
    若时间戳a迟晚于时间戳b，则返回1；
    若时间戳a同时于时间戳b，则返回0

AVRecorder

【参见：FFmpeg/Primer/AVRecorder】

课堂练习

- 录制音视频混合流
  - 在Windows上列表显示所有的Video For Windows设备和DirectShow设备及其选项
  - 通过用户选择的采集设备，捕获音视频数据，保存或推送
  - 分别在独立的线程中，以并发的方式，采集音频和视频流，避免因捕获过程的相互等待，丢失帧数据
  - 借助独立的混流线程，将捕获到的音视频帧，按解码时间戳的升序，依次写入目标格式中

附录

基于Nginx+RTMP的流媒体系统(下)

Tarena Technology 达内科技

知识讲解

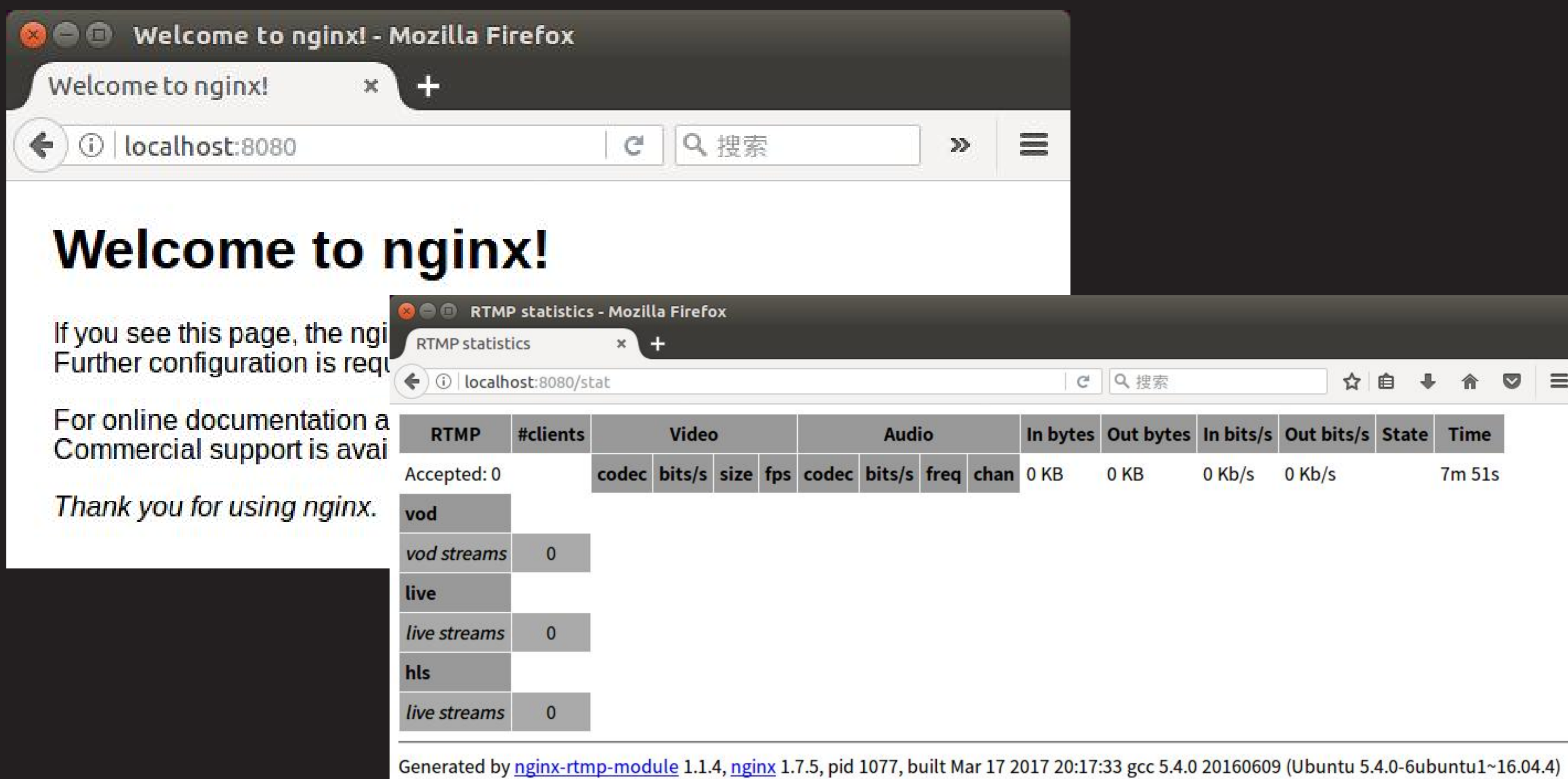
13. 在服务器上启动或重启nginx服务
    - $ sudo service nginx start
      或
      $ sudo service nginx restart
14. 在服务器上更改防火墙，允许1935/tcp端口
    - $ sudo ufw allow 1935/tcp
15. 在服务器上用netstat检查端口侦听情况
    - $ netstat –ltn

```
Proto Recv-Q Send-Q Local Address Foreign Address State
tcp        0      0 0.0.0.0:1935  0.0.0.0:*       LISTEN
tcp        0      0 0.0.0.0:8080  0.0.0.0:*       LISTEN
```

基于Nginx+RTMP的流媒体系统(下)

16.在服务器上通过浏览器检查nginx服务状态

– http://localhost:8080
– http://localhost:8080/stat



知识讲解

基于Nginx+RTMP的流媒体系统(下)

知识讲解

17. 在服务器上将视频点播文件(1.mp4 ...)拷贝到~/Videos目录下，保证任何用户对其可读

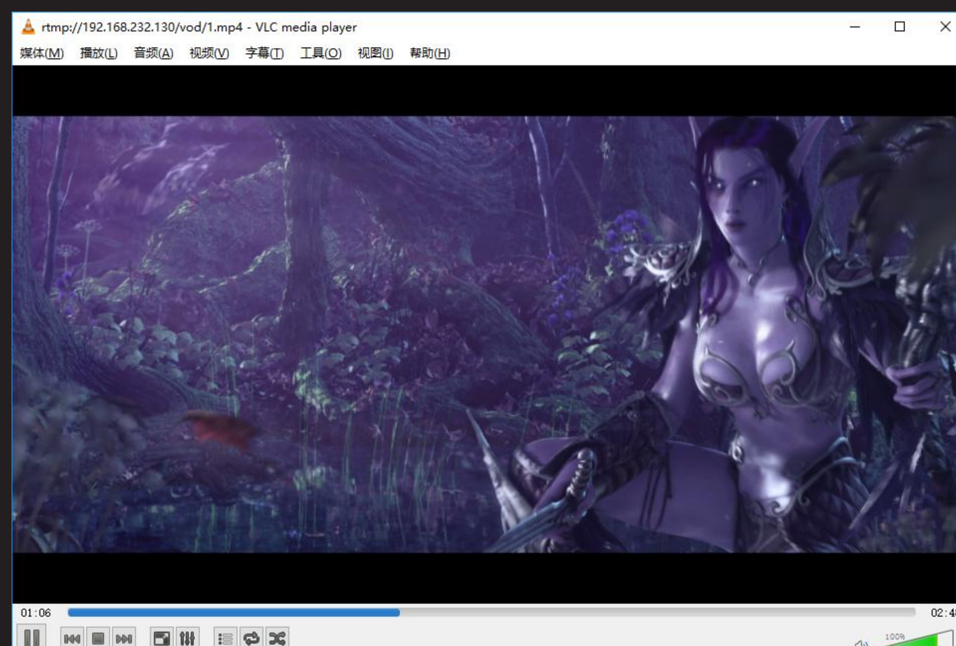
```
minwei@ubuntu:~/Videos$ pwd
/home/minwei/Videos
minwei@ubuntu:~/Videos$ ls -l
总用量 3135604
-rw-r--r-- 1 minwei minwei 417170370 8月   12  2014 1.mp4
-rw-r--r-- 1 minwei minwei 297089682 8月   12  2014 2.mp4
-rw-r--r-- 1 minwei minwei 373161819 8月   12  2014 3.mp4
-rw-r--r-- 1 minwei minwei 361059118 8月   12  2014 4.mp4
-rw-r--r-- 1 minwei minwei 423277865 8月   12  2014 5.mp4
-rw-r--r-- 1 minwei minwei 960192458 8月   15  2014 6.mp4
-rw-r--r-- 1 minwei minwei 169194964 1月   24 21:20 7.mp4
-rw-r--r-- 1 minwei minwei 109132009 11月   7  2015 8.mp4
-rw-r--r-- 1 minwei minwei 100544991 1月   22 19:05 9.mp4
minwei@ubuntu:~/Videos$
```

基于Nginx+RTMP的流媒体系统(下)

知识讲解

18.测试点播媒体文件

- 播放端需要先从http://www.videolan.org下载并安装VLC media player
- 在播放端用VLC media playe打开网络串流：rtmp://192.168.232.130/vod/1.mp4

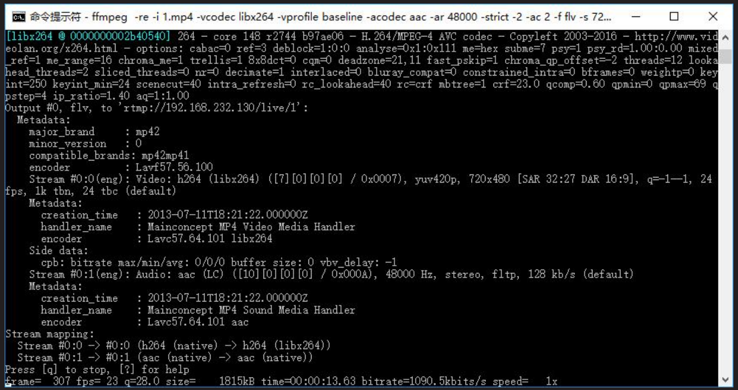


基于Nginx+RTMP的流媒体系统(下)

知识讲解

19.测试RTMP直播媒体文件

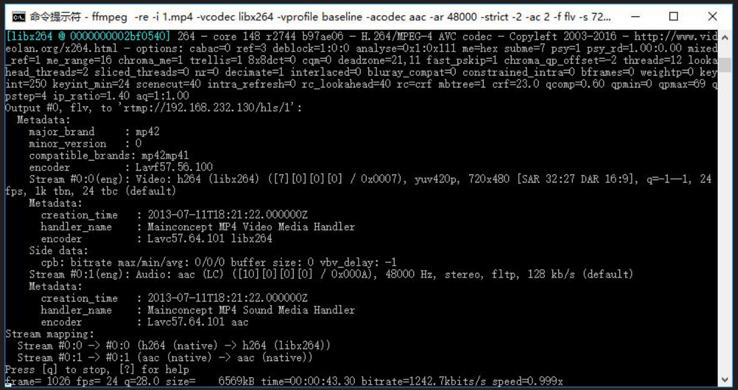
- 推送端需要先从http://ffmpeg.org下载并安装FFmpeg
- 在推送端用FFmpeg推送媒体流：
  ffmpeg -re -i 1.mp4 -vcodec libx264 -vprofile baseline -acodec aac -ar 48000 -strict -2 -ac 2 -f flv -s 720x480 -q 10 rtmp://192.168.232.130/live/1
- 在播放端用VLC media player打开网络串流：
  rtmp://192.168.232.130/live/1



基于Nginx+RTMP的流媒体系统(下)

20. 测试HLS直播媒体文件

- 在推送端用FFmpeg推送媒体流：
  ffmpeg -re -i 1.mp4 -vcodec libx264 -vprofile baseline -acodec aac -ar 48000 -strict -2 -ac 2 -f flv -s 720x480 -q 10 rtmp://192.168.232.130/hls/1
- 在播放端用VLC media player打开网络串流：
  rtmp://192.168.232.130/hls/1

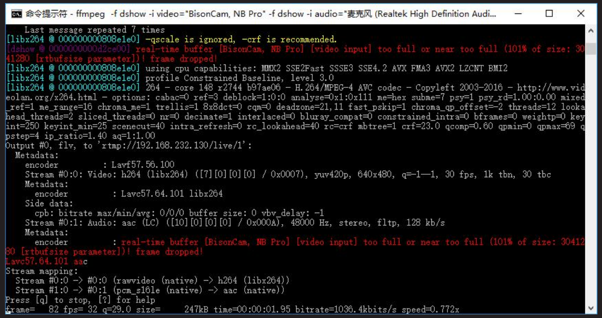


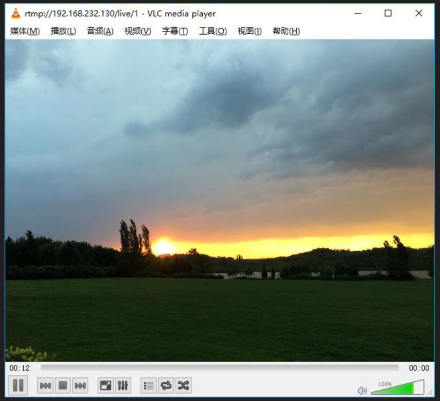
基于Nginx+RTMP的流媒体系统(下)

知识讲解

21.测试RTMP直播摄像头和麦克风

- 在推送端用FFmpeg推送媒体流：
  ffmpeg -f dshow -i video="BisonCam, NB Pro" -f dshow -i audio="麦克风 (Realtek High Definition Audio)" -pix_fmt yuv420p -vcodec libx264 -vprofile baseline -acodec aac -ar 48000 -strict -2 -ac 2 -f flv -s 640x480 -q 10 rtmp://192.168.232.130/live/1
- 在播放端用VLC media player打开网络串流：
  rtmp://192.168.232.130/live/1

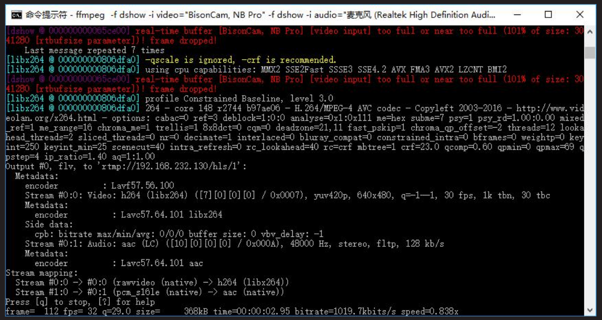


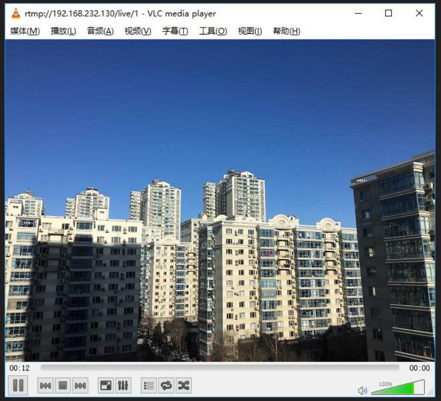


基于Nginx+RTMP的流媒体系统(下)

22. 测试HLS直播摄像头和麦克风

- 在推送端用FFmpeg推送媒体流：
  ffmpeg -f dshow -i video="BisonCam, NB Pro" -f dshow -i audio="麦克风 (Realtek High Definition Audio)" -pix_fmt yuv420p -vcodec libx264 -vprofile baseline -acodec aac -ar 48000 -strict -2 -ac 2 -f flv -s 640x480 -q 10 rtmp://192.168.232.130/hls/1
- 在播放端用VLC media player打开网络串流：
  rtmp://192.168.232.130/hls/1





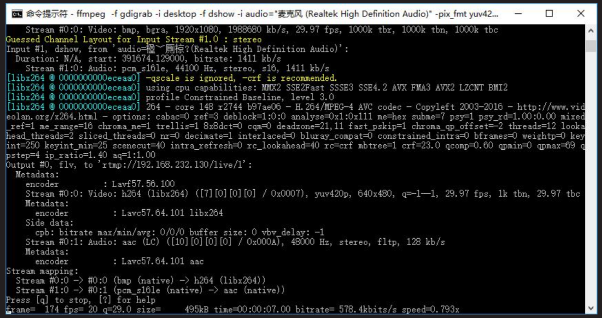
基于Nginx+RTMP的流媒体系统(下)

Tarena Technology 达内科技

知识讲解

23.测试RTMP直播屏幕和麦克风

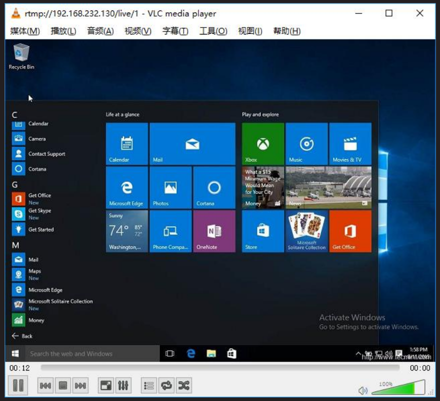
- 在推送端用FFmpeg推送媒体流：
  ffmpeg -f gdigrab -i desktop -f dshow -i audio="麦克风 (Realtek High Definition Audio)" -pix_fmt yuv420p -vcodec libx264 -vprofile baseline -acodec aac -ar 48000 -strict -2 -ac 2 -f flv -s 640x480 -q 10 rtmp://192.168.232.130/live/1
- 在播放端用VLC media player打开网络串流：
  rtmp://192.168.232.130/live/1

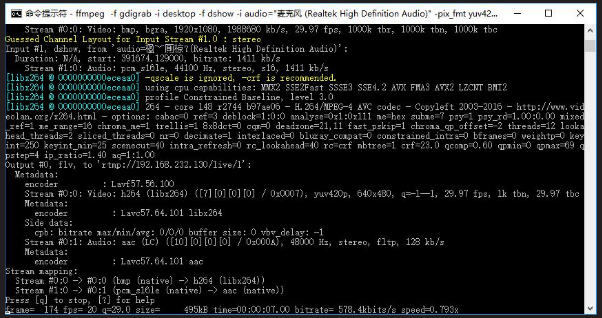


基于Nginx+RTMP的流媒体系统(下)

24.测试HLS直播屏幕和麦克风

- 在推送端用FFmpeg推送媒体流：
  ffmpeg -f gdigrab -i desktop -f dshow -i audio="麦克风 (Realtek High Definition Audio)" -pix_fmt yuv420p -vcodec libx264 -vprofile baseline -acodec aac -ar 48000 -strict -2 -ac 2 -f flv -s 640x480 -q 10 rtmp://192.168.232.130/hls/1
- 在播放端用VLC media player打开网络串流：
  rtmp://192.168.232.130/hls/1





总结和答疑