转码整体架构说明书

版本	日期	作者	说明
1.0	2016-7-13		

1. 架构

一、控制架构：

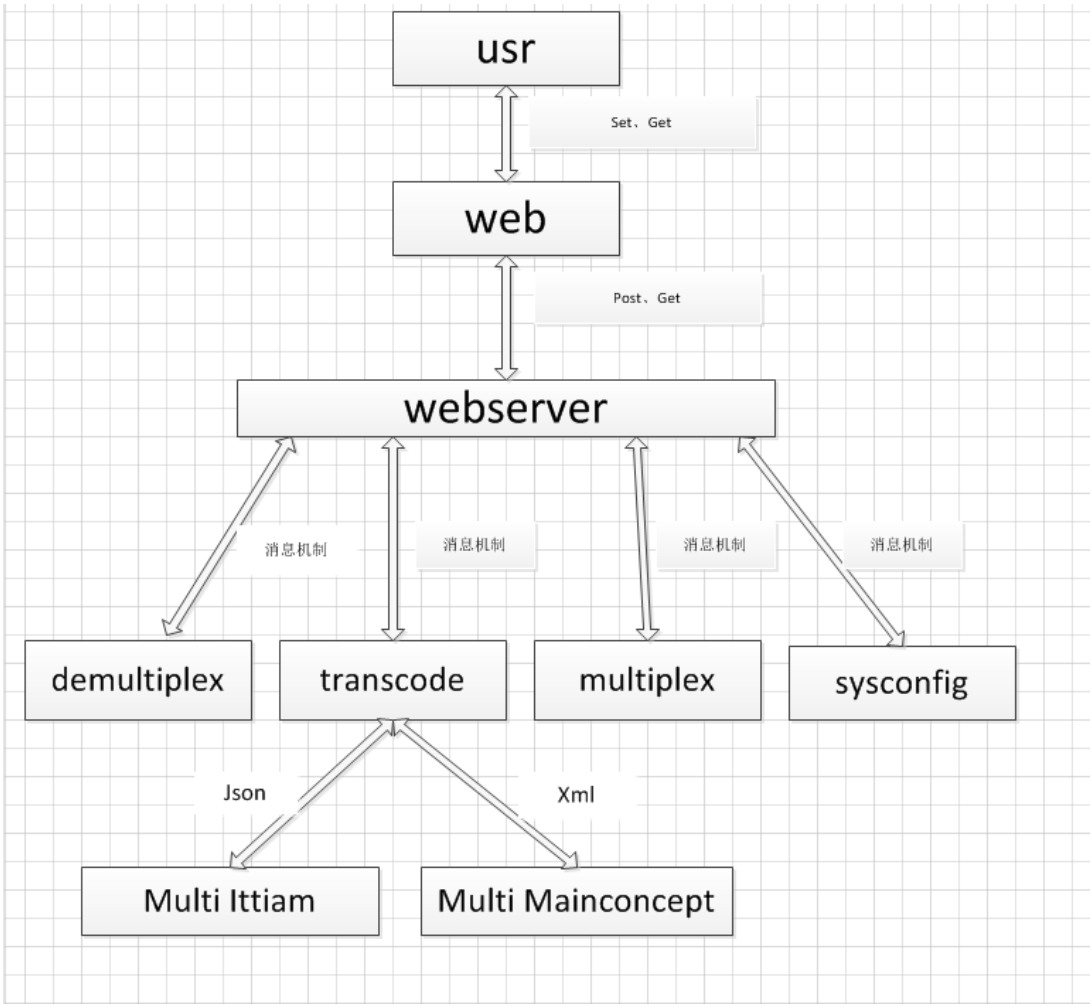


图 1 控制架构

注：webserver 控制其它进程通过消息机制进行通信（构建发送和接收消息队列）

二、数据流向：

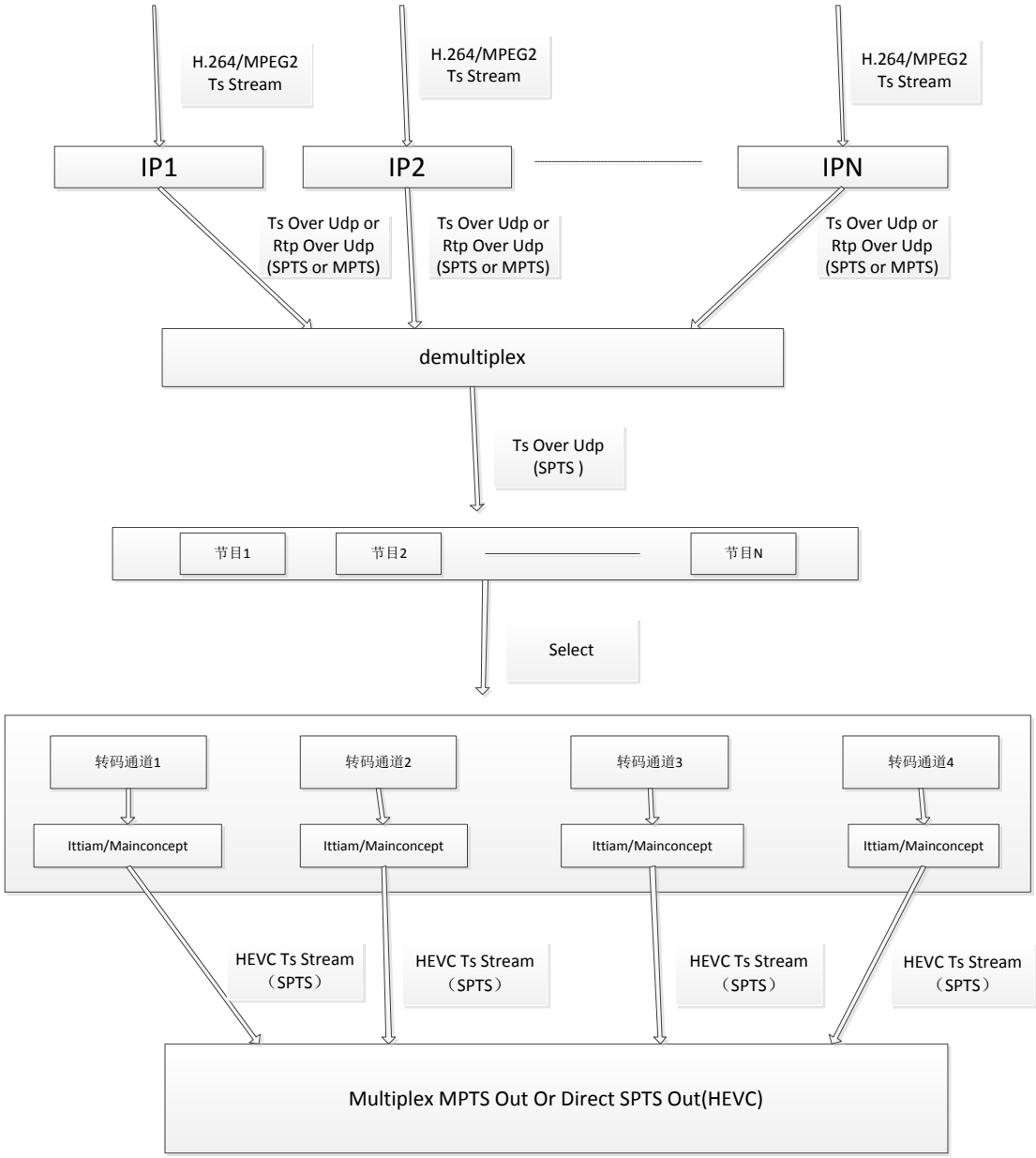


图 2 数据流向架构

注：数据在各个进程之间的传输通过 IP 端口

三、软件架构：

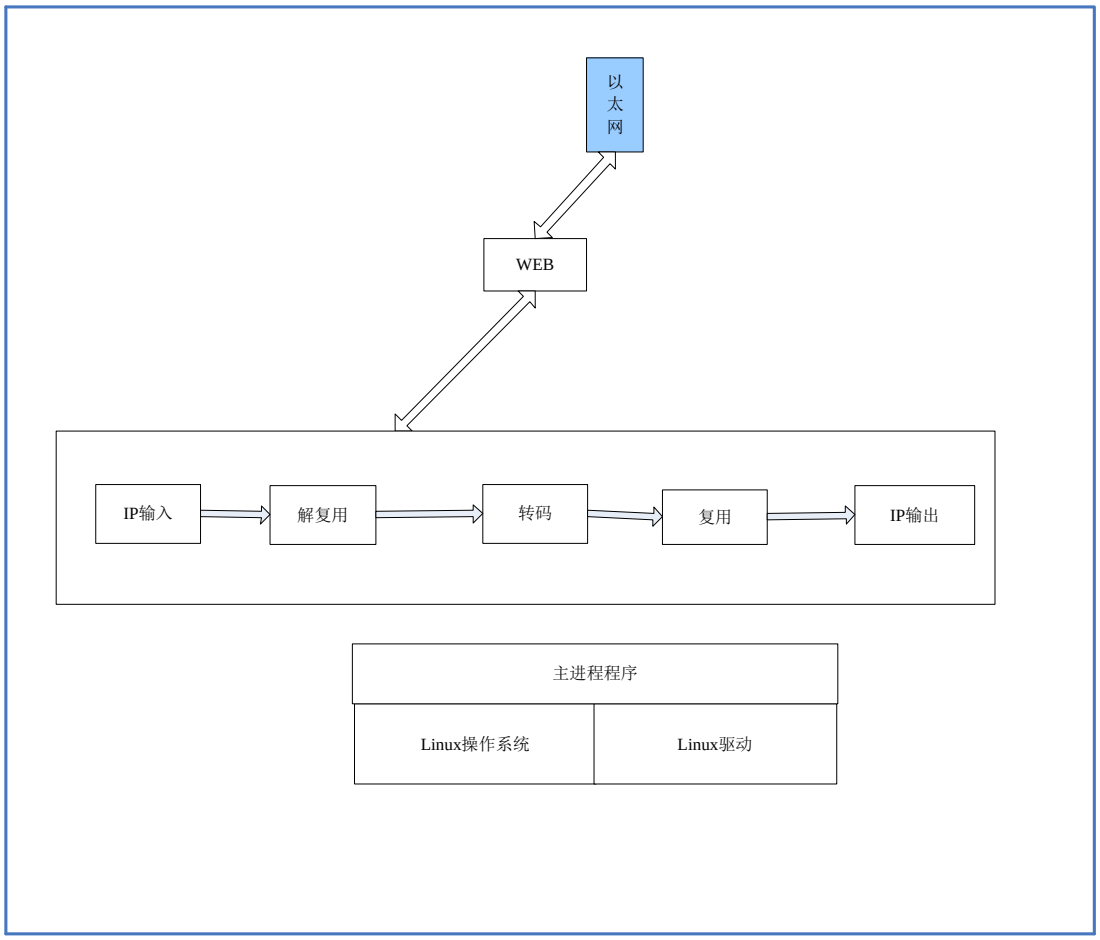
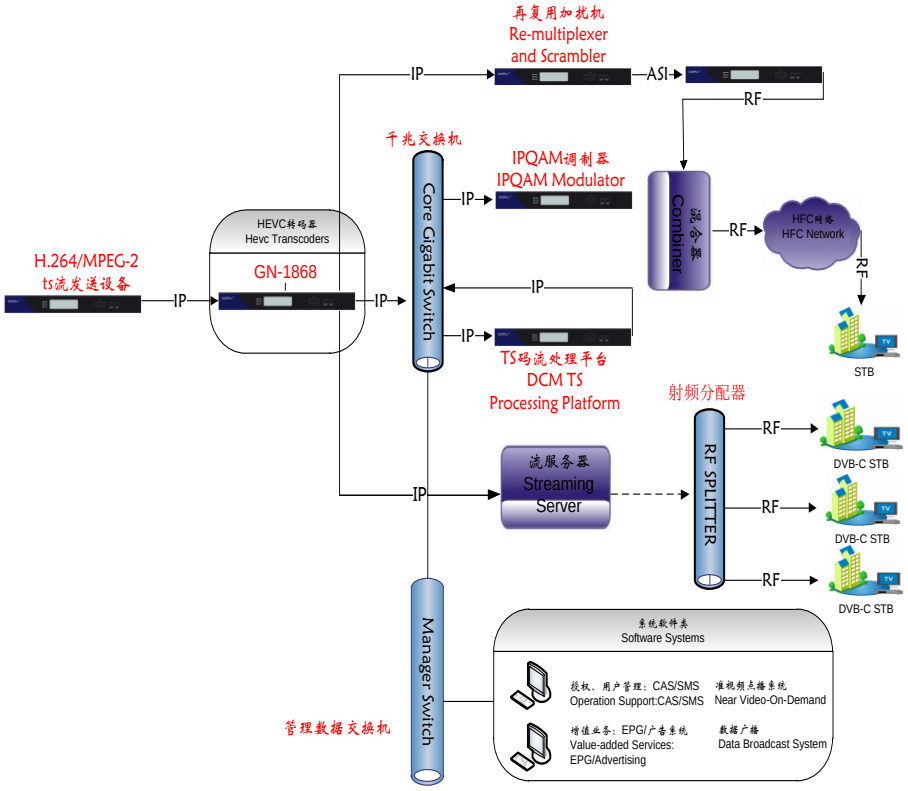


图 3 软件架构

四、物理架构：



图四　物理架构

2. 主要功能

主要功能：

1、输入 H.264/MPEG2 音视频 TS/RTP（MPTS/SPTS）流，输出 HEVC 音视频 TS（SPTS）流

3. 限制条件

1、 必须具有 root 权限

2、 内存大于 4G，CPU 为 4770K/R（替换 CPU 需要找 ittiam 替换相关配置文件）

3、 AC3 音频格式只支持 Bypass 输出。

4、 Ittiam 暂时只能支持一路高清（最高支持到 1920*1080P60），两路标清（720*576P25），短时间可转码三路标清。Mainconcept 暂时只能支持一路高清（最高支持到 1920*1080P30），两路标清（720*576P25）。

5、 不能超过系统允许的最大消息队列数量，可以用 ipcs –l 命令查看，不够需要更改系统支持的最大消息队列数量。

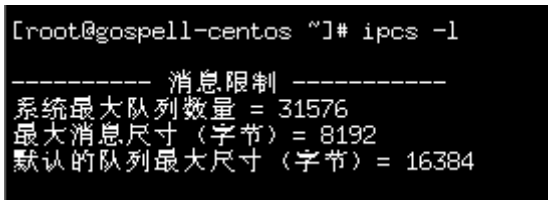


图 4 消息队列限制

4. 输入

1、 H.264/MPEG2 音视频 TS/RTP（MPTS/SPTS）流

5. 输出

1、 输出 HEVC 音视频 TS（SPTS）流

6. 目录结构

一、目录

build 目录：存放整个转码项目执行和停止所需要的所有文件，将在下面做详解

demultiplex 目录：解复用目录，MPTS IN -->> SPTS OUT

multiplex 目录：复用目录，SPTS IN –>>SPTS/MPTS OUT

transcode 目录：转码进程启动、异常检测、重启

ittiiam 目录：启动 Ittiam 转码进程

webserver 目录：配置和查询其它各个进程的参数

third_party_lib:第三方库，在下面做简单解释

complie.sh 文件：一键编译文件，编译 demultiplex、multiplex、transcode、ittiiam、webserver、third_party_lib 目录

二、build 目录（按名称）

1、bin 目录：media-engine.jar，ittiam 可执行文件的打包文件

2、config 目录：ittiam media-engine 的参数配置文件

3、dependencies 目录：一些 ittiam media-engine 执行所需要的动态库文件和 shell 文件

4、lib 目录：一些 ittiam media-engine 执行所需要的动态库文件，与 CPU 相关

5、pid 目录：转码进程运行的 pid，用于 stop.sh 文件停止相关进程

6、web 目录：web 相关文件

7、code_cfg_*.json：ittiam media-engine 执行所需参数配置文件

8、code_cfg_*.xml：mainconcept 执行所需参数配置文件

9、demultiplex_x86：解复用可执行文件，32 位

10、ittiam_x64：启动 media-engine 可执行文件，64 位

11、MAINCONCEPT：mainconcept 转码可执行文件，64 位

12、multiplex_x64：复用可执行文件，64 位

13、multiplex_x86：复用可执行文件，32 位

14、run.sh：整个转码项目的启动 shell 文件

15、start_wme.sh：ittiam 专属的 shell 文件，用于检测必要的动态库是否存在

16、stop.sh：停止整个转码项目的 shell 文件

17、sys_log*：转码进程异常产生的日志文件

18、sysconfig_x64：系统配置可执行文件，64 位

19、transcode_x64：转码配置可执行文件，64 位

20、umconfig.txt：web 的相关配置文件

21、webserver.ini：进程间交互所需消息队列文件

22、webserver_x64：web 服务可执行文件，64 位

23、webserver_x86：web 服务可执行文件，32 位

三、third_party_lib 目录（按顺序）

1、 Inc 目录：相关头文件

2、 Lib 目录：相关库文件

1、 libcjson_x64.a：json 库文件，64 位

2、 libmsg_x64.a：消息构造库文件，64 位

3、 libmsg_x86.a：消息构造库文件，32 位

4、 libmxml_x64.a：xml 构造库文件，64 位

5、 libmxml_x86.a：xml 构造库文件，32 位

6、 libplatform_x86.a：gospell 私有平台库文件，32 位

7、 libts_x64.a：ts 解析库文件，64 位

8、 libts_x86.a：ts 解析库文件，32 位

9、 libuser_x64.a：user 库文件，64 位

10、 libwebs_x64.a：web 库文件，64 位

11、 libwebs_x86.a：web 库文件，32 位

12、 libwebserver_x64.a：webserver 库文件，64 位

3、 Src 目录：相关源码目录

1、 cjson 目录：json 相关文件

2、 fileupgrade：升级文件

3、 fileupload：上传文件

4、 msg:消息文件

5、 mxml:xml 文件

6、 ts:ts 流解析相关文件

7、 user:user 相关文件

8、 webs:web 相关文件

7. 流程

一、手动执行：

1、 cd 到 build 目录

2、 切换到 root
控制台键入 su 命令后输入 root 用户的密码

3、 修改 run.sh 权限
控制台键入 chmod +x run.sh

4、 运行 run.sh
./run.sh

8. 鸣谢

感谢之前对这个模块有过贡献的同事们。