直播流快速截图系统

——基于切片模式的截图服务

一、概览：

随着网络视频技术的发展，直播从专业领域已经开始在互联网上迅速的普及，越来越多的互联网公司开始搭建网络直播平台，直播越来越深入的渗透到人们生活的各个方面，基于直播形式的的商业模式也不断的涌现，在为网络直播不断的赋予新的功能和技术的同时，公共场所监控、交通监控、家庭监控等等基于摄像头的监控业务也在不断的扩大，每年都会有大量的摄像头投入运行，以前基于单点的小规模的应用，随着硬件成本的下降以业务的要求，逐步演变成数量多、覆盖面广、全天候监控等特点，随着监控设备数量的不断增多，监控范围的不断扩大，如何实时的了解设备的运行状态及当前的监控内容，变得尤其关键；在降低监管成本的同时，又能够保证监管的效果是急需要解决的一个难题。目前基于流方式的监测手段有很多，但其技术的核心都是依靠大带宽和充足的计算资源进行支撑，所以成本变得越来越不可控，稳定性也不高；如何彻底的解决这个问题，这就是本发明的目标；

基于流的方式进行直播的监测，前提要有充足的资源，投入大、成本高；为了解决这个问题，大家也尝试着使用截取图片的方式进行处理，首先基于流的方式进行截图，和针对于流直接进行监测的成本基本相同，并不能够降低整体的成本，还有可能会增加成本。本发明是基于切片模式的截图服务，根据切片服务(HLS:HLS 是Apple的动态码率自适应技术。主要用于PC和Apple终端的音视频服务。包括一个m3u8的索引文件和TS媒体分片文件)中生成文件(TS)生成的TS文件(ts是视频的封装格式，全称为MPEG2-TS。ts即"Transport Stream”的缩写。MPEG2-TS格式的特点就是要求从视频流的任一片段开始都是可以独立解码)进行截图，因为TS文件相对较小(每片大约为3-15秒左右，文件大小约1-3M)，所以读取的速度较快，处理的时间很短；

目前直播技术实现大都是支持HLS协议，所以本发明所适用的范围比较广泛，具有低成本、高可用、高并发等特点，系统的对接工作很低，后期的运营成本非常低，监测的效果与直播流基本相同，并且更加易于管理；

二、设计原则：

- **可靠性**

用软件系统规模越做越大越复杂，其可靠性越来越难保证。应用本身对系统运行的可靠性要求越来越高，软件系统的可靠性也直接关系到设计自身的声誉和生存发展竞争能力。软件可靠性意味着该软件在测试运行过程中避免可能发生故障的能力，且一旦发生故障后，具有解脱和排除故障的能力。本系统在设计阶段就考虑到这些因素，充分利用各种成熟的技术的产品，并实现集群化布署，能够实现对于高并发、高吞吐量的支撑，可以实现7x24小时不间断运行，具备很高的可靠性；

- **健壮性**

健壮性又称鲁棒性，是指软件对于规范要求以外的输入能够判断出这个输入不符合规范要求，并能有合理的处理方式。对于系统容错及兼容性做了非常多的处理，对于日常使用过程的数据错误，超时错误等都可以及时的进行解决。

- **可修改性**

提供完善和详细的设计与维护文档，供后期的人员进行维护和管理。

- **可测试性**

可以通过模拟直播环境进行功能的测试，也可以通过专用的手段进行大并发量的用户访问测试，产可以通过专用工具对于系统的关键模块进行极限测试。

- **效率性**

系统在支撑日常访问时，只受带宽和磁盘读写效率的硬件环境影响，对于CPU和内存的使用率非常低，能够大大的降低使用成本。

- **先进性**

系统基于成熟的协议及linux环境开发，并且经过算法和流程的优化，结构清晰，流程简单，具有非常高的可维护性；

- **可扩展性**

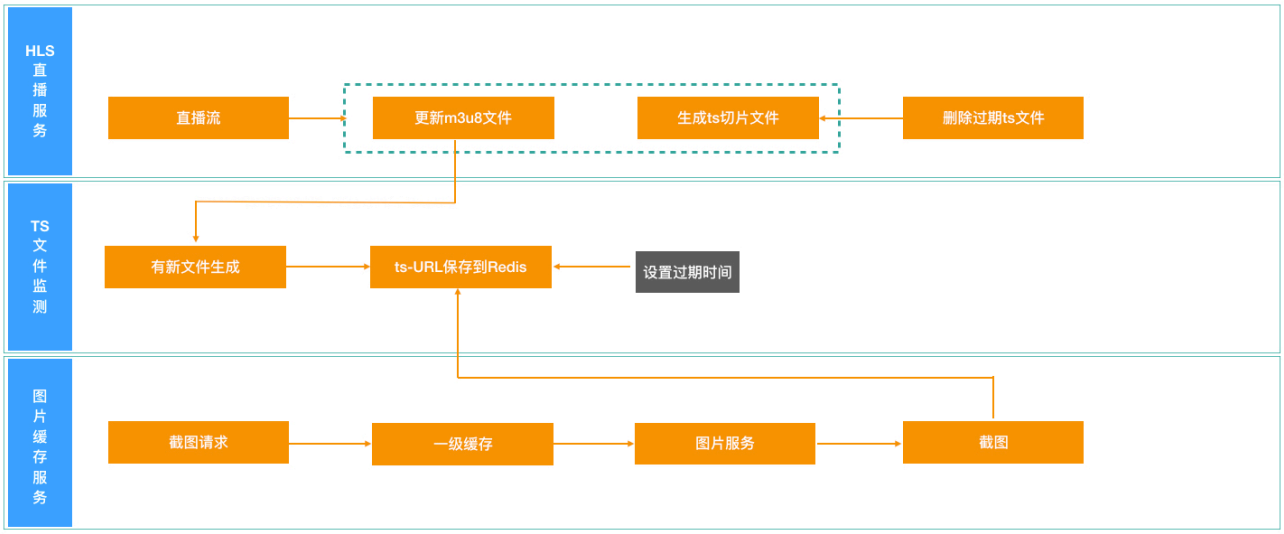
系统支持脚本开发环境，可以根据自已的需求进行调整和配置，能够随时扩展新的服务功能；

- **安全性**

系统安装的系统环境和开发的代码都经过严格的安全检测，并达到安全的要求；并可以对于用户鉴权系统进一步提升业务的安全级别，并可以随时进行软件的安全更新工作，不会影响业务的正常处理；

三、系统设计：

1) 系统架构



2) 架构说明

系统共划分为三个基础服务模块：HLS直播切片服务（可选）、文件状态监控服务、文件缓存服务；

HLS直播切片服务模块：

HLS (HTTP Live Streaming)是Apple的动态码率自适应技术。主要用于PC和Apple终端的音视频服务。包括一个m3u(8)的索引文件，TS媒体分片文件和key加密串文件。目前HLS服务已经成为了业界共用的一个http流协议，直播切片TS文件生成后，默认保存在本地，过期的内容会删除，直播截图服务就是利用在本地生成TS视频切片的特性而设计，一般TS切片的长度为3秒以上（也可以设置为更多，根据业务需要），本地保存的切片数最少为三片（可以多到十片左右），所以截图操作需要在切片删除前处理完成；

文件状态监控服务模块：

inotify-tools 是为linux下文件监控工具提供的一套c的开发接口库函数，同时还提供了一系列的命令行工具，这些工具可以用来监控文件系统的事件。 inotify-tools是用c编写的，除了要求内核支持inotify外，不依赖于其他。inotify-tools提供两种工具，一是 inotifywait，它是用来监控文件或目录的变化，二是inotifywatch，它是用来统计文件系统访问的次数。直播快速截图服务主要是使用inotifywait来实现；inotifywait通过监听指定目录（TS切片生成目标）的变化，主要是监听文件的

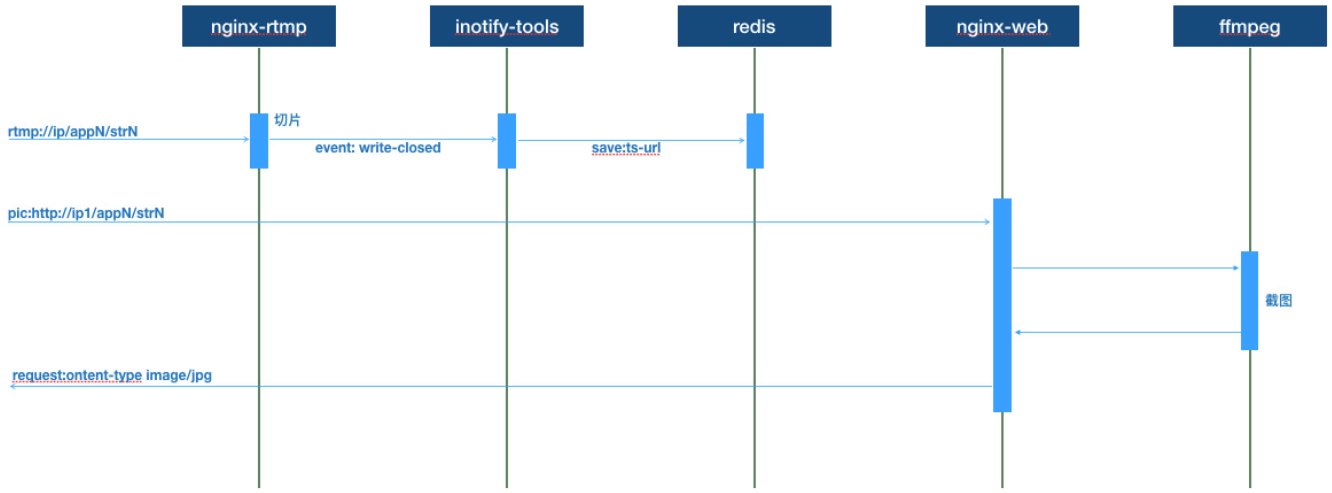
write_close事件，来确定是否有新TS文件生成，发现有新文件生成的事件后，则把生成的文件访问地址保存入redis中进行保存；

文件缓存服务：

Redis（全称：Remote Dictionary Server 远程字典服务）是一个开源的使用ANSI C语言编写、支持网络、可基于内存亦可持久化的日志型、Key-Value数据库，并提供多种语言的API；接收到截图访问请求后，会提取当前该直播流的最新文件访问地址进行截图（截图处理时间为100MS左右）；直播截图内容会被临时存储到Redis缓存数据库中，并提供标准的http访问；

四、主要流程介绍

1) 主要流程



2) 主要流程说明

nginx-rtmp直播流处理流程：

Nginx (engine x) 是一个高性能的HTTP和反向代理web服务器，同时也提供了IMAP/POP3/SMTP服务。Nginx是由伊戈尔·赛索耶夫开发的，第一个公开版本0.1.0发布于2004年10月4日，其将源代码以类BSD许可证的形式发布，因它的稳定性、丰富的功能集、示例配置文件和低系统资源的消耗而闻名。Nginx-rtmp模块是Nginx服务器中一个专门处理rtmp直播流媒体的模块，主要处理直播类的业务而设计，能够实现基于rtmp协议的直播流的播放、转换、切片和收录等功能；

编码器推流进入nginx-rtmp模块后，服务器进行流切片处理，系统的默认切片大小为5s (可根据实际需要调整，不要小于2秒)， 切片后会在本地目录中生成后缀为m3u8的播放List文件和后缀为TS的媒体切片内容文件；

inotify事件监控：

Inotify 是一个 Linux特性，它监控文件系统操作，比如读取、写入和创建。Inotify 反应灵敏，用法非常简单，并且比 cron 任务的繁忙轮询高效得多。inotify-wait是inotify提供的监控目录和文件变化的工具，通过监控指定位置的信息文件，来推送监控信息，切片模式的截图服务通过监控HLS服

务生成目录的变化，来判断是否有新的TS文件生成，如果有新的TS文件生成，则把该文件的访问地址保存入redis中；

FFmpeg截图

FFmpeg是一套可以用来记录、转换数字音频、视频，并能将其转化为流的开源计算机程序。采用LGPL或GPL许可证。它提供了录制、转换以及流化音视频的完整解决方案。切片模式的截图服务把FFmpeg做为视频截图的操作命令，通过输入视频文件地址，输出截取的图片文件内容；

Redis缓存服务

Redis是一个开源的使用ANSI C语言编写、遵守BSD协议、支持网络、可基于内存亦可持久化的日志型、Key-Value数据库，并提供多种语言的API，它通常被称为数据结构服务器。切片模式的截图服务把Redis做为图片缓存的第一级缓存使用，并设定过期的时间，过期后由Redis自动进行删除，以达到自动运维的目标，如果需要支撑大并发量的访问，可以使用nginx或其它技术，设置二级或多级缓存；

Nginx图片服务

Nginx (engine x) 是一个高性能的HTTP和反向代理web服务器，同时也提供了IMAP/POP3/SMTP服务。Nginx是由伊戈尔·赛索耶夫开发的，第一个公开版本0.1.0发布于2004年10月4日，其将源代码以类BSD许可证的形式发布，因它的稳定性、丰富的功能集、示例配置文件和低系统资源的消耗而闻名。切片模式的截图服务把Nginx做为截图的访问服务器，根据流访问地址，转换成为截图访问地址，如:rtmp://a.b.com/appN/streamN转换成 http://[a.b.com]/appN/streamN即可访问该直播流的当前截图；

3) 关键程序说明

文件监测 inotify_ts.sh

```
#!/bin/bash
#设定本机的IP地址和端口号，供截图服务在调用TS文件截图时使用；
redip=192.168.1.81 #需要访问的redis服务的地址,如果需要指定端口，直接在后面加即可；
ip=192.168.1.10   #本机的外部访问地址
port=80              #本机的web端口，ip+port 提供其它机器访问生成的TS文件；
tspath=/tmp/live/   #本机生成ts目录的地址，注意路径后要加“/“
#以上的这些设置项需要根据实际的环境进行设置；
inotifywait -mrq --timefmt '%d/%m/%y %H:%M'  --format '%T %w %f %e'  --event close_write  --exclude '^(.*\.m3u8|.*\.bak|.*\.jpg)$' $tspath | awk -F' ' \
'{len=split($3, array, "/"); \
```

```
    cmd= "redis-cli -h ""'$redip'"" set "array[len-2]"_"array[len-1]" http://""'$ip'"":""'$port'""/
getts/"array[len-2]"/"array[len-1]"/"$4" ex 20 >/dev/null 2>&1"
    #print cmd;
    system(cmd);
    #now=systime();print now;
    }'
```

在初始设置的时候需要设定HLS切片的指定目录，如果使用nginx-rtmp进行切片处理，可参考nginx.conf中的切片路径设置，如例子中的/tmp/live/, 注意需要在后面加上“/ ”；

redip=192.168.1.81 #需要访问的redis服务的地址,如果需要指定端口，直接在后面加即可；

ip=192.168.1.10 #本机的外部访问地址

port=80 #本机的web端口，ip+port 提供其它机器访问生成的TS文件；

tspath=/tmp/live/ #本机生成ts目录的地址，注意路径后要加“/“

[redis-cli -h localhost expire "array[len-2]"_"array[len-1]" 20]中的20秒为缓存过期的时间，因为每个TS片完成后会被删除，所以redis的记录也需要随时更新，每条记录有效期为20秒，一般TS片不超过10秒钟，所以会在数据过期前进行更新。

该程序一般需要增加进/usr/bin中，并且执行权限为777；

获取截图模块 getshot.lua

```
    --redis服务器的访问地址
    --local ip = "127.0.0.1"
    --local port = 6379

    ip = ngx.var.red_ip
    port = ngx.var.red_port+0
    --截图保存临时地址，上传后就会删除，但需要保证Nginx进程需要有读写权限；
    path= ngx.var.pic_path

    --切割字符串
    function mysplit(inputstr, sep)
        if sep == nil then
            sep = "%s"
        end
        local t={} ; i=1
        for str in string.gmatch(inputstr, "([^"..sep.."]+)") do
            t[i] = str
```

```
                i = i + 1
            end
            return t
        end

        --关闭redis数据库
        local function close_redis(red)
            if not red then
                return
            end
            local ok, err = red:close()
            if not ok then
                ngx.status = 404
                ngx.say("close redis error : ", err)
                ngx.exit(404)
            end
        end

        --根据ts的访问地址，调用ffmpeg进行截图；
        local function ffmpeg_shot(tsurl, appN, streamN)

            ngx.log(ngx.ERR, "popen begin!")
            cmd = 'ffmpeg -y -i '..tsurl..' -f image2 -ss 0 -t 0.001 -vframes 1 '..path..appN..'_'..streamN..'.jpg >/dev/null
2>&1&& redis-cli -h '..ip..' -p '..port..
            ' -x set '..appN..'_'..streamN..'_pic <'..path..appN..'_'..streamN..'.jpg >/dev/null 2>&1 && redis-cli -h '..ip..' -p
'..port..
            ' expire '..appN..'_'..streamN..'_pic 20 >/dev/null && rm -rf '..path..appN..'_'..streamN..'.jpg&'
            ngx.log(ngx.ERR, ip)
            ngx.log(ngx.ERR, port)
            ngx.log(ngx.ERR, path)
            ngx.log(ngx.ERR, cmd)
            t = io.popen(cmd)
            --ngx.log(ngx.ERR, "popen end1!")
            --popen系统调用是非阻塞，所以会存在没有截图完成时即返回结果的现像，为了保证在截图后进行后续操作，进
程增加了300ms的延时，不够精确，但应该能够满足大部分需求；
            ngx.sleep(0.3)
            --ngx.log(ngx.ERR, "sleep end1!")
        end

        local function getKey(p_app, p_stream)

            redkey = p_app.."_"..p_stream
            return redkey
        end
```

```
--获得uri,uri的
local uri = ngx.var.uri

local sp =  mysplit(uri, "/")
if sp == nil then
    ngx.status = 404
    ngx.say("url error!", err)
    return ngx.exit(404)

end
local app= sp[1]
local stream= sp[2]

local redis = require("resty.redis")

--创建实例
local red = redis:new()
--设置超时（毫秒）
red:set_timeout(1000)

--建立连接
local ok, err = red:connect(ip, port)
if not ok then
    close_redis(red)
    ngx.status = 404
    ngx.say("connect to redis error : ", err)
    return ngx.exit(404)
end

ngx.header.content_type = "text/plain"

--根据appName和streamName来提取当前ts片的访问地址，然后再进行截图；
local tsurl, err = red:get(getKey(app, stream))

if tsurl == ngx.null then
    close_redis(red)
    ngx.status = 404
    ngx.say("The tsurl not exit!: ", err)
    return ngx.exit(404)
end

ffmpeg_shot(tsurl, app, stream)
```

```
--截图完成后根据appName和streamName对于截图的内容进行提取；

local img, err = red:get(getKey(app, stream.."_pic"))
close_redis(red)
if img == ngx.null then

    ngx.status = 404
    ngx.say("The picture not exit!")
    return ngx.exit(404)

else
    ngx.header.content_type = "image/jpeg"
    ngx.say(img)
    ngx.exit(200)
end
```

该程序嵌入到nginx中进行运行，通过流的appName和streamName两个信息来提取相应的截图内容；因为截图内容是每个切片截取一张头关键帧图片，所以图片的截取间隔大致等于切片的时间长度；

该系统实现根据rtmp的地址，来获得该流的实时截图；

直播地址 : rtmp://ip/appName/streamName

实时截图 : http://ip1/appName/streamName

五、安装与部署

1）、系统环境：

直播服务器：在直播系统的基础上进行了扩充，所以所有的服务器都是直播的服务器，只是在软件上做相应的调整即可，无需新增服务器；

操作系统：linux

软件环境：nginx1.9.4 (需要提供ts文件的网络访问能力)

redis-client（需要本机访问redis缓存服务器)

图片服务及截图服务器：1+台（可根据需要配置）

操作系统：linux

软件环境：nginx1.9.4+lua

ffmpeg(用户截图）

redis-client(需要本机访问redis缓存服务器)

Redis数据库服务器：1+台

操作系统：linux

软件环境：redis-server

2）、直播服务器安装与配置

安装软件环境：

安装nginx+nginx_rtmp_moudle（如果以前支持，不需要重新安装）；

配置nginx.conf：

启用rtmp中的切片功能，例：

```
application myapp {
        live on;

        hls on;
        hls_nested on; #直播切片内容按照appName进行分目录存放，很重要；
        hls_path /home/thumbnail/myapp;#视频流存放地址
        hls_fragment_naming system;
```

```
            hls_fragment 5s;
            hls_playlist_length 30s;
            hls_continuous on; #连续模式。
            hls_cleanup off;   #对多余的切片进行删除。
            hls_nested on;    #嵌套模式。
        }
    }
```

安装文件监控软件：

1、安装inotify-tools（如果以前支持，不需要重新安装）；

2、安装redis-cli. 需要shell访问redis服务，所以需要通过调用redis-cli命令；

3、拷备inotify_ts.sh 到/usr/bin 下(根据实际配置情况);

配置inotify_ts.sh, 需要修改以下几个内容：

redip=192.168.1.81 #需要访问的redis服务的地址,如果需要指定端口，直接在后面加即可；

ip=192.168.1.10 #本机的外部访问地址

port=80 #本机的web端口，ip+port 提供其它机器访问生成的TS文件；

tspath=/tmp/live/ #本机生成ts目录的地址，注意路径后要加“/“

因为HLS生成的ts文件需要被切片服务器通过http访问，所以要在本机建立web服务；在本机安装nginx服务，并设定目录getts服务；

配置nginx.conf：

```
        server {
          listen   80;
          location /getts {
                    alias /tmp;
          }
```

测试安装是否正常：curl http://本机IP/appName/streamName/xxxxx.ts，如果能够访问到文件内容，则配置正常；

3）、图片服务及截图服务器安装

1、安装LuaJIT

```
wget http://luajit.org/download/LuaJIT-2.0.2.tar.gz
tar -zxvf LuaJIT-2.0.2.tar.gz
cd LuaJIT-2.0.2
```

```
make install PREFIX=/usr/local/LuaJIT

export LUAJIT_LIB=/usr/local/LuaJIT/lib
export LUAJIT_INC=/usr/local/LuaJIT/include/luajit-2.0
```

为了保证每次配置生效，最好把上面两行写到/etc/profile中；

2、 下载ngx_devel_kit和lua-nginx-module

```
cd /opt/download

wget https://github.com/simpl/ngx_devel_kit/archive/v0.3.0.tar.gz

wget https://github.com/openresty/lua-nginx-module/archive/v0.10.9rc7.tar.gz
```

3、 重新编译Nginx

```
cd /opt/download

wget http://nginx.org/download/nginx-1.14.0.tar.gz
tar -zxvf nginx-1.14.0.tar.gz
cd nginx-1.14.0

./configure --add-module=/opt/download/ngx_devel_kit-0.3.0 --add-module=/opt/
download/lua-nginx-module-0.10.9rc7

# 如若出现:
# error: the HTTP gzip module requires the zlib library.
# yum install zlib zlib-devel 一下即可

make -j 4 && make install
echo "/usr/local/LuaJIT/lib" >> /etc/ld.so.conf
# 然后执行如下命令:
ldconfig
```

4、安装ffmpeg

简单安装方法：apt -y install ffmpeg

源码安装：可上网查找方法；

```
5、安装lua-resty-redis
git clone https://github.com/openresty/lua-resty-redis.git
cd lua-resty-redis
make && make install
```

6、修改nginx配置文件,支持Lua脚本
在http内添加 lua_package_path “/usr/local/lib/lua/?.lua;;";

7、安装配置 getshot.lua脚本

```
$ mkdir /usr/local/nginx/lua
$ cp getshot.lua /usr/local/nginx/lua
$ vi /usr/local/nginx/conf/nginx.conf
```

```
#user  nobody;
worker_processes  1;

events {
    worker_connections  1024;
}

http {
    include       mime.types;
    default_type  application/octet-stream;

    #log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
    #                  '$status $body_bytes_sent "$http_referer" '
    #                  '"$http_user_agent" "$http_x_forwarded_for"';

    #access_log  logs/access.log  main;

    keepalive_timeout  65;
    #lua_package_path "/usr/local/nginx/lua/lua-resty-redis/lib/?.lua;;";
    lua_package_path “/usr/local/lib/lua/?.lua;;";

    server {
        listen       80;
        server_name  localhost;
        set $red_ip "127.0.0.1";
        set $red_port "6397";
        set $pic_path "/tmp/";
        location / {
            root   html;
            content_by_lua_file lua/getshot.lua;
        }
    }
}
```

需要在nginx.conf中设置三个变量：

```
        set $red_ip "127.0.0.1";
        set $red_port "6397";
        set $pic_path "/tmp/";
```

red_ip、red_port 为 redis 服务器访问地址和端口号；
`pic_path 是本机保存截图的临时路径，该目录需要nginx具有读写权限；注意后面一定要加“/”;`

4）、Redis服务器安装：

简单安装:apt install redis-server

集群安装可参考网上；

六、验证与测试：

安装完毕后，启用编码器进行直播推流，如: rtmp://192.168.1.2/app/live

查看相应的截图内容，如：http://192.168.1.2/app/live

此文档访问地址：github.com/xiaobo0903/streamshot