CSDN 首页 博客 学院 下载 GitChat TinyMind 论坛 问答 商城 ••• 搜博主文章 写博客 发Chat

gladsnow的专栏 RSS订阅

# 原 Webrtc服务器搭建(基于局域网环境)

2017年09月08日 23:05:26

0 写评论 目录 收藏 微信 微博 QQ

# ebrtc服务器搭建

## 基于局域网环境)

st Modified Date: 2017/8/2

## 录



## 1. 搭建平台

- **操作系统**：Ubuntu 16.04 server(64bits)
- **Google webrtc的服务器Demo**：详见https://github.com/webrtc/apprtc
- **IP地址：**局域网 192.168.6.54

## 2. 软件安装

- **安装JDK：**

```
add-apt-repository ppa:openjdk-r/ppa
apt-get update
apt-get install openjdk-8-jdk
```

- **安装nodejs相关包：**

```
apt-get install nodejs
apt-get install npm
apt-get install nodejs-legacy
npm -g install grunt-cli
```

- **安装Python和Python-webtest：**

```
apt-get install python
apt-get install python-webtest
```

**注： 若已安装过上述软件，可忽略；如上述未提及的软件需要安装，请自行安装。**

## 3. 搭建房间服务器(Room Server)

400电话怎么申请 python培训机构 如何申请400电话 网络舆情监控系统

登录 注册

```
git clone  https://github.com/webrtc/apprtc.git
cd apprtc
npm install
```

若npm install报错，请自行解决。

- **修改文件**
  1.**修改/root/apprtc/src/app_engine/constants.py**

```
    TURN_BASE_URL = 'http://192.168.6.54:80'; #本机内网地址192.168.6.54,此处的端口号与Nginx监听的端口号保持一致
        TURN_URL_TEMPLATE = '%s/turn.php?username=%s&key=%s'; #如果turn.php未实现，可使用默认配置
        CEOD_KEY = 'inesadt'   #此处后面turn配置的用户名保持一致

        ICE_SERVER_BASE_URL = 'http://192.168.6.54:80';#此处的端口号与Nginx监听的端口号保持一致
        ICE_SERVER_URL_TEMPLATE = '%s/iceconfig.php?key=%s'; #如果iceconfig.php未实现，可用默认配置，但是Android Apk会有问题

        WSS_INSTANCE_HOST_KEY = '192.168.6.54:8089'  #信令服务器端口号8089
        WSS_INSTANCE_NAME_KEY = 'vm_name'
        WSS_INSTANCE_ZONE_KEY = 'zone'
        WSS_INSTANCES = [{
        WSS_INSTANCE_HOST_KEY: '192.168.6.54:8089',
        WSS_INSTANCE_NAME_KEY: 'wsserver-std',
        WSS_INSTANCE_ZONE_KEY: 'us-central1-a'
        }, {
        WSS_INSTANCE_HOST_KEY: '192.168.6.54:8089',
        WSS_INSTANCE_NAME_KEY: 'wsserver-std-2',
        WSS_INSTANCE_ZONE_KEY: 'us-central1-f'
        }]
```

**修改/root/apprtc/src/app_engine/apprtc.py（若使用https,则不需修改此文件）**

```
if wss_tls and wss_tls == 'false':
        wss_url = 'ws://' + wss_host_port_pair + '/ws'
        wss_post_url = 'http://' + wss_host_port_pair
        else:
        wss_url = 'ws://' + wss_host_port_pair + '/ws'
        wss_post_url = 'http://' + wss_host_port_pair
```

- **编译（在apprtc目录下进行）**

```
grunt build
```

编译完成之后，会生成out目录，房间服务器编译完成。

**注（编译成功可忽略）：**此处编译需要翻墙，若编译时无法翻墙，可下载手动下载https://api.callstats.io/static/callstats.min.js，并把文件callstats.min.j
pprtc/out/app_engine/third_party/callstats/下。
然后修改/root/apprtc/build/build_app_engine_package.py文件：

```
    # Download callstats.
    ......
    ......
    response = requests.get(urls[fileName])
    #if response.status_code == 200:    #把此处注释掉
    print 'Downloading %s to %s...' % (urls[fileName], path)
    with open(path + fileName, 'w') as to_file:
       to_file.write(response.text)
    #else:  #把此处注释掉
    #  raise NameError('Could not download: ' + filename + ' Error:' + \ #把此处注释掉
    #str(response.status_code))  #把此处注释掉
```

然后继续进行编译即可。

- **安装和配置google app engine**

  **1.下载google app engine**

**2.配置google app engine 路径**
解压google_appengine_1.9.50.zip

```
unzip google_appengine_1.9.50.zip
```

编辑/etc/profile文件，在文件最后添加语句：

```
export PATH="$PATH:/root/google_appengine/"
```

（当前安装目录是/root/google_appengine,请根据自己的安装目录进行配置）
保存profile文件，进行以下操作生效

```
source /etc/profile
```

- **运行房间服务器（room server)**

在目录/root/google_appengine目录下找到dev_appserver.py脚本,执行以下语句

```
./dev_appserver.py --host=192.168.6.54 /root/apprtc/out/app_engine
```

想后台运行，则执行

```
nup ./dev_appserver.py --host=192.168.6.54 /root/apprtc/out/app_engine &
```

- **在浏览器中访问房间服务器**

http://192.168.6.54:8080

---

## 搭建信令服务器(Collider Server)

- **安装go语言编译器**

```
apt-get install golang-go
```

- **复制collider源代码**
（此源码在房间服务器源码目录下/root/apprtc/src/collider/)
在/root目录下新建文件夹

```
mkdir -p goWorkspace/src
```

配置编译环境，此配置是暂时有效的

```
export GOPATH=/root/goWorkspace/
```

把/root/apprtc/src/collider/目录下的三个目录（collider、collidermain、collidertest）复制到/root/goWorkspace/src/目录下

```
cp -rf /root/apprtc/src/collider/* /root/goWorkspace/src
```

- **修改代码**
编辑文件/root/goWorkspace/src/collidermain/main.go，修改房间服务器的地址

```
var roomSrv = flag.String("room-server", "http://192.168.6.54:8080", "The origin of the room server")
```

- **编译信令服务器**
进入目录/root/goWorkspace/src/,此处编译需要翻墙。

```
go get collidermain
go install collidermain
```

编译成功后，在/root/goWorkspace/下会生成bin和pkg目录。
若此处编译无法翻墙，可手动下载需要的文件。在/root/goWorkspace/src/目录下，

```
mkdir -p golang.org/x
cd golang.org/x/
git clone https://github.com/golang/net
```

然后再进行编译即可。

- **运行信令服务器**
  进入/root/goWorkspace/bin/目录，运行信令服务器

```
./collidermain -port=8089 -tls=false
```

若想后台运行，则执行

```
nohup ./collidermain -port=8089 -tls=false &
```

0
写评论
目录
收藏
微信
微博
QQ

## 搭建STUN\TURN服务器

- **安装coturn**

```
apt-get install coturn
```

- **进行相关配置**

  1. 编辑文件/etc/default/coturn,把TURNSERVER_ENABLED=1的注释去掉。
  2. 编辑文件/etc/turnserver.conf,把以下内容加入到文件最后（或者在文件中找到相应的选项，进行配置）

```
listening-device=eth0  #此处eth0是电脑网卡名称
listening-port=3478       #turn服务器的端口号
relay-device=eth0 #此处eth0是电脑网卡名称
min-port=49152
max-port=65535
Verbose
fingerprint
lt-cred-mech
use-auth-secret
static-auth-secret=inesadt    #此处要和房间服务器配置时constants.py文件中的CODE_KEY保持一致。
user=inesadt:0x7e3a2ed35d3cf7f19e2f8b015a186f54
user=inesadt:inesadt
stale-nonce
cert=/usr/local/etc/turn_server_cert.pem
pkey=/usr/local/etc/turn_server_pkey.pem
no-loopback-peers
no-multicast-peers
mobility
no-cli
```

上述文件中 0x7e3a2ed35d3cf7f19e2f8b015a186f54的生成方法：

```
turnadmin -k -u inesadt -r north.gov -p inesadt
```

```
-k 表示生成一个long-term credential key
-u 表示用户名
-p 表示密码
-r 表示Realm域（这个值的设置可能会有影响）。
```

coturn的证书生成（即配置文件中cert和pkey)

```
sudo openssl req -x509 -newkey rsa:2048 -keyout /usr/local/etc/turn_server_pkey.pem -out /usr/local/etc/turn_server_cert.pem -day
```

- **启动coturn服务器**

```
service coturn start
```

## 6. 配置Nginx服务器

- 安装Nginx

```
apt-get install nginx
```

- 安装php和php-fpm

```
apt-get install php
apt-get install php7.0-fpm
```

- 编辑配置文件/etc/nginx/sites-available/default

0
写评论
目录
收藏
微信
微博
QQ

```
upstream roomserver {
            server 192.168.6.54:8080;
        }
        server {
            #listen 80 default_server;
            #listen [::]:80 default_server;

            listen 80;
            # SSL configuration
            #
            # listen 443 ssl default_server;
            # listen [::]:443 ssl default_server;
            #
            # Note: You should disable gzip for SSL traffic.
            # See: https://bugs.debian.org/773332
            #
            # Read up on ssl_ciphers to ensure a secure configuration.
            # See: https://bugs.debian.org/765782
            #
            # Self signed certs generated by the ssl-cert package
            # Don't use them in a production server!
            #
            # include snippets/snakeoil.conf;

            root /var/www/html;

            # Add index.php to the list if you are using PHP
            index index.html index.htm index.nginx-debian.html index.php; #此处添加index.php

            server_name _;

            #   location / {
                # First attempt to serve request as file, then
                # as directory, then fall back to displaying a 404.
                #   try_files $uri $uri/ =404;
            #   }

            # pass the PHP scripts to FastCGI server listening on 127.0.0.1:9000
            #
            location ~ \.php$ {
                include snippets/fastcgi-php.conf;
                # With php7.0-cgi alone:
                #   fastcgi_pass 127.0.0.1:9000;
                # With php7.0-fpm:
                fastcgi_pass unix:/run/php/php7.0-fpm.sock;
            }

            location / {
                proxy_pass http://roomserver$request_uri;
                proxy_set_header Host $host;
            }
            # deny access to .htaccess files, if Apache's document root
            # concurs with nginx's one
            #
            #location ~ /\.ht {
                #   deny all;
```

```
            #}
        }
```

- **编写turn.php文件和iceconfig.php文件，并把文件放到目录/var/www/html/目录下**

  turn.php文件内容

```
<?php
            $request_username = $_GET["username"];
            if(empty($request_username)) {
                echo "username == null";
                exit;
            }
            $request_key = $_GET["key"];
            $time_to_live = 600;
            $timestamp = time() + $time_to_live;//失效时间
            $response_username = $timestamp.":".$_GET["username"];
            $response_key = $request_key;
            if(empty($response_key))
            $response_key = "code_key"; //constants.py中CEOD_KEY

            $response_password = getSignature($response_username, $response_key);

            $jsonObj = new Response();
            $jsonObj->username = $response_username;
            $jsonObj->password = $response_password;
            $jsonObj->ttl = 86400;
            //此处需配置自己的服务器
            $jsonObj->uris= array("stun:192.168.6.54:3478","turn:192.168.6.54:3478?transport=udp","turn:192.168.6.54:3478?transpor

            echo json_encode($jsonObj);

        /**
         * 使用HMAC-SHA1算法生成签名值
         *
         * @param $str 源串
         * @param $key 密钥
         *
         * @return 签名值
         */
        function getSignature($str, $key) {
        $signature = "";
        if (function_exists('hash_hmac')) {
        $signature = base64_encode(hash_hmac("sha1", $str, $key, true));
        } else {
        $blocksize = 64;
        $hashfunc = 'sha1';
        if (strlen($key) > $blocksize) {
        $key = pack('H*', $hashfunc($key));
        }
        $key = str_pad($key, $blocksize, chr(0x00));
        $ipad = str_repeat(chr(0x36), $blocksize);
        $opad = str_repeat(chr(0x5c), $blocksize);
        $hmac = pack(
        'H*', $hashfunc(
        ($key ^ $opad) . pack(
        'H*', $hashfunc(
        ($key ^ $ipad) . $str
        )
        )
        )
        );
        $signature = base64_encode($hmac);
        }
            return $signature;
            }

            class Response {
                public $username = "";
                public $password = "";
```

0 写评论 目录 收藏 微信 微博 QQ

```
64                 public $ttl = "";
65                 public $uris = array("");
66             }
67 
68         ?>
```

iceconfig.php文件内容

```
1  <?php
2              $request_username = "inesadt";  //配置成自己的turn服务器用户名
3              if(empty($request_username)) {
4                  echo "username == null";
5                  exit;
6              }
7              $request_key = "inesadt";  //配置成自己的turn服务器密码
8              $time_to_live = 600;
9              $timestamp = time() + $time_to_live;//失效时间
10             $response_username = $timestamp.":".$_GET["username"];
11             $response_key = $request_key;
12             if(empty($response_key))
13             $response_key = "CEOD_KEY";//constants.py中CEOD_KEY
14 
15             $response_password = getSignature($response_username, $response_key);
16 
17             $arrayObj = array();
18             $arrayObj[0]['username'] = $response_username;
19             $arrayObj[0]['credential'] = $response_password;
20             //配置成自己的stun/turn服务器
21             $arrayObj[0]['urls'][0] = "stun:192.168.6.54:3478";
22             $arrayObj[0]['urls'][1] = "turn:192.168.6.54:3478?transport=tcp";
23             $arrayObj[0]['uris'][0] = "stun:192.168.6.54:3478";
24             $arrayObj[0]['uris'][1] = "turn:192.168.6.54:3478?transport=tcp";
25             $jsonObj = new Response();
26             $jsonObj->lifetimeDuration = "300.000s";
27             $jsonObj->iceServers = $arrayObj;
28             echo json_encode($jsonObj);
29 
30             /**
31             * 使用HMAC-SHA1算法生成签名值
32             *
33             * @param $str 源串
34             * @param $key 密钥
35             *
36             * @return 签名值
37             */
38             function getSignature($str, $key) {
39                 $signature = "";
40                 if (function_exists('hash_hmac')) {
41                     $signature = base64_encode(hash_hmac("sha1", $str, $key, true));
42                 } else {
43                     $blocksize = 64;
44                     hashfunc = 'sha1';
45                     if (strlen($key) > $blocksize) {
46                         $key = pack('H*', $hashfunc($key));
47                     }
48                     $key = str_pad($key, $blocksize, chr(0x00));
49                     $ipad = str_repeat(chr(0x36), $blocksize);
50                     $opad = str_repeat(chr(0x5c), $blocksize);
51                     $hmac = pack(
52                     'H*', $hashfunc(
53                             ($key ^ $opad) . pack(
54                                     'H*', $hashfunc(
55                                             ($key ^ $ipad) . $str
56                                     )
57                             )
58                         )
59                     );
60                     $signature = base64_encode($hmac);
61                 }
62                 return $signature;
```

0 写评论 目录 收藏 微信 微博 QQ

```
63            }
64
65             class Response {
66                    public $lifetimeDuration = "";
67                    public $iceServers = array("");
68             }
69          ?>
```

**注：**关于turn.php和iceconfig.php文件实现的必要性，如果是http局域网，即使不实现这两个文件，在浏览器之间也可实现视频通信，但是如果使用Goo k作为客户端，则可能会存在问题。经过测试，实现iceconfig.php即可，turn.php文件可不实现。

- **重启Nginx服务器和php7.0-fpm**

```
service nginx restart
service php7.0-fpm restart
```

0 写评论 目录 收藏 微信 微博 QQ

## 运行测试

- **PC浏览器（Android手机浏览器）之间的视频通信测试**
  访问http://192.168.6.54:8080
  1.PC浏览器：Firefox 54.0.1(64bits)，Android手机浏览器：Firefox 54.0.1

  > 测试OK

  2.PC浏览器：Google Chrome 59.0.3071.115(64bits)，Android手机浏览器：Google Chrome 59.0.3071.125

  > 测试失败
  > 失败原因：Error getting user media: Only secure origins are allowed。

- **Android APK客户端之间以及客户端与浏览器之间**
  1.获取Android APK

  > 下载webrtc源码，在源码目录下webrtc/examples/androidapp，进行编译即可生成Android APK

  2.测试Android APK客户端之间

  > 测试OK

  2.测试Android APK客户端与浏览器（Firefox)之间

  > 测试OK

## 附录

- 运行过程中的问题
  1. Failed to start signaling: Failed to execute 'pushState' on 'History': A history state object with URL 'http://192.168.6.54/r/198676628' cannot be created in gin 'https://192.168.6.54' and URL 'https://192.168.6.54/

> **解决方法1：**
> 房间服务器编译完成后，在/root/apprtc/out/app_engine/js/apprtc.debug.js文件中找到window.history.pushState({'roomId': roomId, 'roomLink': roomLink}, roomId, room 注释掉，重新运行即可。（如果重新编译，需要重新修改）
> **解决方法2：**
> 在/root/apprtc/src/web_app/js/appcontroller.js文件中找到window.history.pushState({'roomId': roomId, 'roomLink': roomLink}, roomId, roomLink)，把这句话注释掉，然后 运行房间服务器即可。