在 Centos6 下搭建 Webrtc 的 Apprtc

Jonta
jonta@jonta.cn

1 Apprtc 简介

Apprtc 是谷歌对 Webrtc 的实现。废话不多说，为啥要搭建 Webrtc？因为它快啊，我在 WiFi 环境下测试直播延迟 200ms 左右，同时支持视频会议，也就是双向传输；只有这个优点么？当然不止，它支持浏览器直接播放，Chrome，Edge，Safari，Firefox，Opera 厉害吧；搭建个平台有啥用？那你知道它包含了 GCC 算法，TFRC 算法，卡尔曼滤波算法，AEC 算法，发送端宽带控制，接收端宽带预测等各种算法和 VP8，VP9，VP10 这样先进的编码技术，这么说吧，随便拿出个算法就够你毕业了，知道搭建它的意义了吧\(^o^)/。以下是在 Centos6 这样老得掉牙的系统上搭建，自然无比的心烦，如果你想要快速搭建，那么就用 Ubuntu 吧这个非洲的系统，它更新得太快了几个月一个版本，我是不想用的。如果能在 Centos6 下搭建成功，那么 Ubuntu 下简直小 Case。

2 搭建平台

操作系统：Centos6
Google webrtc 的服务器 Demo：详见 https://github.com/webrtc/apprtc
域名地址：例如我的域名 jonta.cn 或者 ip 地址

3 编译环境依赖软件安装

3.1 Centos6 软件必要的更新

由于 Centos6 软件版本太旧，必须更新 gcc 到 4.7 以上，python2.7 以上，npm3.10 以上，node6.14 以上，php7.0 以上，所以首先更新这些软件，以上除了 python2.7 源码编译安装比较顺利，其他相对困难，所以不要认为源码安装不就完了，  ，源码编译安装得把系统搞烂。

3.2 安装 gcc4.8

网上其他方法无一成功，在越过围墙之后发现这个网站的方法不错
https://unix.stackexchange.com/questions/125609/install-latest-gcc-on-rhel-6-x86-64#
于是借来一用，下面这条命令就是下载 yum 的源，接着就可以用 yum 安装 gcc 了

```
wget -O /etc/yum.repos.d/slc6-devtoolset.repo    http://linuxsoft.cern.ch/cern/devtoolset/slc6-devtoolset.repo
```

yum install devtoolset-2 可能等很久，提示没有签字，自行解决。在 Centos 里升级 gcc 基本不可能，太多依赖，这里的安装其实是创建一个最小 linux 系统里面仅仅包含新版本的 gcc，安装完成后再配置一下变量，写入/etc/profile 文件或者临时变量，whatever

```
export CC=/opt/rh/devtoolset-2/root/usr/bin/gcc
export CPP=/opt/rh/devtoolset-2/root/usr/bin/cpp
export CXX=/opt/rh/devtoolset-2/root/usr/bin/c++
```

这样原来的 gcc 没动，编译时调用新版 gcc。如果用 gcc4.4.7 在 npm 编译软件时会出现类似**../node_modules/nan/nan.h:41:3: error: #error This version of node/NAN/v8 requires a C++11 compiler** 这样的错误

3.3 更新 Nodej 和 NPM

与升级 gcc 一样，需要用 yum 安装就必须升级源，下面添加 node.js `yum` 库，

`curl -sL https://rpm.nodesource.com/setup_6.x | sudo -E bash`
接着需要删除原来的库，`yum erase nodejs npm` 嗯好了，再看看删干净没
`rpm -qa 'node|npm' | grep -v nodesource` 最后再安装 `yum install nodejs`，装好了没，不知道？那再确认下 node –v;npm -v

3.4 PHP7.0 的安装

Centos 什么都好就是软件版本跟不上，气人不。原版的 php 是 5.5 的，我们现在得把他升级到 7.0，不用想啦自带源里没有！来来来，保持队形，添加 php 软件仓库，rpm -Uvh `https://mirror.webtatic.com/yum/el6/latest.rpm`，其他源可不可以，只要你找得到都可以，接着安装 php 依赖呗。

```
yum install php71w.x86_64 php71w-cli.x86_64 php71w-common.x86_64
php71w-gd.x86_64 php71w-ldap.x86_64 php71w-mbstring.x86_64
php71w-mcrypt.x86_64 php71w-mysql.x86_64 php71w-pdo.x86_64
```

这里好像有个包找不到，蛋四不影响啦。接着来，`yum install php71w-fpm.x86_64`。

3.5 Python2.7 和 pip，setuptools 安装

`pip` 和 **setuptools** 安装 `yum install python-pip python-setuptools`，为啥要说这个呢，这他么和安装的名字其他系统不一样非得加个前缀。Python2.7 安装呢，我不想说了。。。。编译的时候如果遇到这个问题，`Python build finished, but the necessary bits to build these modules were not found: _bsddb _curses _curses_panel _sqlite3 _ssl _tkinter bsddb185 bz2 dbm dl gdbm imageop readline sunaudiodev zlib To find the necessary bits, look in setup.py in detect_modules() for the module‘s name.`

先看看这个

模块	依赖	说明
_bsddb	bsddb	Interface to Berkeley DB library。Berkeley 数据库的接口
_curses	ncurses	Terminal handling for character-cell displays。
_curses_panel	ncurses	A panel stack extension for curses。
_sqlite3	sqlite	DB-API 2.0 interface for SQLite databases。SqlLite，CentOS 可以安装 sqlite-devel
_ssl	openssl-devel.i686	TLS/SSL wrapper for socket objects。
_tkinter	N/A	a thin object-oriented layer on top of Tcl/Tk。如果不使用桌面程序可以忽略 TKinter
bsddb185	old bsddb module	老的 bsddb 模块，可忽略。
bz2	bzip2-devel.i686	Compression compatible with bzip2。bzip2-devel
dbm	bsddb	Simple “database” interface。
dl	N/A	Call C functions in shared objects.Python2.6 开始，已经弃用。
gdbm	gdbm-devel.i686	GNU’s reinterpretation of dbm
imageop	N/A	Manipulate raw image data。已经弃用。
readline	readline-devel	GNU readline interface
sunaudiodev	N/A	Access to Sun audio hardware。这个是针对 Sun 平台的，CentOS 下可以忽略
zlib	Zlib	Compression compatible with gzip

安装 `readline-devel`，`sqlite-devel`，`bzip2-devel.i686`，`openssl-devel.i686`，

gdbm-devel.i686，libdbi-devel.i686，ncurses-libs，zlib-devel.i686，这些依赖包，重新编译，就一切 OK。有几个包找不到，没关系，看表就知道了。

4 Apprtc 的编译与安装

4.1 房间服务器搭建

终于可以开始进入正题了，确认 java,Python,nodejs,npm 都已安装，下面是一波命令：

```
git clone https://github.com/webrtc/apprtc.git

cd apprtc

npm install

npm -g install grunt-cli
```

你会发现 npm install 的时候 compiler-latest.tar.gz 下载失败，哦买噶，仔细观察他想打开 dl.google.com，呵呵，在天朝你想打开谷歌的连接，做梦去吧你，不能连咋办，用 vps 搭建 ssr？用 vpn 代理？跟你说这么说吧，我试了无数次没有成功，掩面哭泣。那么有没有替代 ip 呢，搜了下还真有，诺，就是这个 203.208.40.111 把他丢 hosts 里，别跟我说你不知道 hosts。然后就顺利多了。安装完 grunt-cli 以后，执行

```
grunt build
```

然后你会发现编译成功了，哈哈哈哈，有这么简单么，对，没有，你得修改几个文件，第一：修改 apprtc/src/app_engine/constants.py

```
TURN_BASE_URL = 'http://jonta.cn:80'; #本机内网地址就是你的 ip,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://jonta.cn: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 = ' jonta.cn: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: ' jonta.cn:8089',

WSS_INSTANCE_NAME_KEY: 'wsserver-std',

WSS_INSTANCE_ZONE_KEY: 'us-central1-a'

}, {

WSS_INSTANCE_HOST_KEY: ' jonta.cn:8089',

WSS_INSTANCE_NAME_KEY: 'wsserver-std-2',

WSS_INSTANCE_ZONE_KEY: 'us-central1-f'

}]
```

第二修改 apprtc/src/app_engine/apprtc.py

```
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
```

当然如果你有 ssl 认证就不需要改这个文件，影响是 google 浏览器必须用 https 连接，目前只有 firefox 支持 http 连接，你会说那就用 https 做连接呗，当然如果你是土豪当我没说，ssl 认证便宜的 5000 一年。测试不需要有 firefox 就够了。

OK，到这再

```
grunt build
```

完事儿。

既然编译完了，那咱们测试下呗，先等等，怎么运行呢，我们还需要个东西，那就是 Google App Engine SDK for Python 一定要下载 google_appengine_1.9.73.zip 这个版本，其他的没成功。解压位置随便，

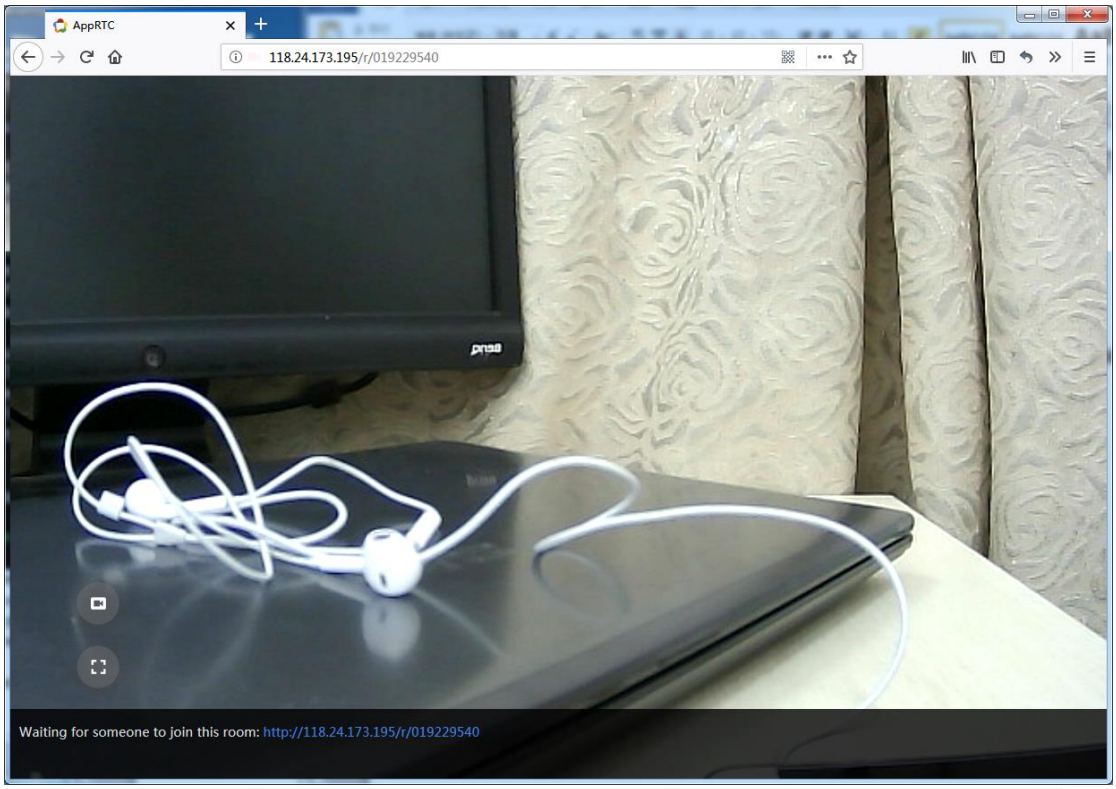
```
export PATH=$PATH:/xxx/google_appengine/
```

这样就好了，接着来测试一下咱们的房间，哈哈，可能会检查 App Engine SDK 版本更新，会连接到谷歌，别想了，所以我们在后面添加 --skip_sdk_update_check 跳过更新。

```
dev_appserver.py --host=jonta.cn apprtc/out/app_engine --skip_sdk_update_check
```

在电脑或安卓手机的 firefox 浏览器中访问 jonta.cn:8080

Like this



如果出现无法绑定 8080 端口，执行 fuser -k 8080/tcp 把占用 8080 端口程序 kill 掉。

4.2 信令服务器搭建 Collider Server

信令服务器是干嘛的？信令就是协调通讯的过程，为了建立一个 Webrtc 的通讯过程，客户端需要交换如下信息： 会话控制信息，用来开始和结束通话，即开始视频、结束视频这些操作指令。知道了吧，它是个指挥官。

Apprtc 源码里自带，下面进入到 collider

```
cd apprtc/src/collider
```

在编译它之前确认安装了 go 语言编译工具，在设置下 GOPATH 环境变量，这是为了指定编译后的可执行文件存放路径。

```
export GOPATH=/xxx/xxx/apprtc/src/collider
```

下面开始编译

```
go get collidermain

go install collidermain
```

当执行 go get collidermain 时报错：unrecognized import path “golang.org/x/net/websocket”xxxx，咋办，不要急，这都归功于咱么伟大的中国墙，go 语言的官方网站墙内访问不了。翻墙下载 golang.org/x/net 这个包，或者找国内镜像下载。解压到 apprtc/src/collider 命名为 golang.org,重新编译。你会发现 apprtc/src/collider 里面多了三个文件夹 bin pkg src 其中 bin 里面就是我们要的信令服务器执行文件。虽然成功了，还有一步要做，那就是修改 apprtc/src/collidermain/main.go 文件指定房间服务器地址。

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

重新编译 collider，最后运行

```
apprtc/src/collider/bin/collidermain -port=8089 -tls=false
```

4.3 搭建 Coturn 服务器

此服务器是穿透服务器，穿透服务器则是用来打洞，使得处于不同网络环境的客户端之间可以直接通信，即所谓的 P2P。

```
wget http://turnserver.open-sys.org/downloads/v4.2.3.1/turnserver-4.2.3.1-CentOS6.5-x86_64.tar.gz

cd turnserver-4.2.3.1-CentOS6.5-x86_64(即上一步的文件名)

./install

或者

git clone https://github.com/coturn/coturn

cd coturn

./configure

make

make install
```

然后编辑文件/etc/turnserver.conf,把以下内容加入到文件最后，

```
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=jonta #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=jonta:0x7e3a2ed35d3cf7f19e2f8b015a186f54

user=jonta:jonta
```

```
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 jonta -r north.gov -p jonta
```

`coturn` 的证书生成（即配置文件中 `cert` 和 `pkey`）

```
openssl req -x509 -newkey rsa:2048 -keyout /usr/local/etc/turn_server_pkey.pem -out
/usr/local/etc/turn_server_cert.pem -days 99999 -nodes
```

启动它

```
turnserver
```

5 配置 Nginx 服务器

5.1 Nginx 安装与配置

直接点

`yum install nginx`

编辑配置文件`/etc/nginx/nginx.conf`

```
upstream roomserver {
server jonta.cn:8080;
}
server {
listen 80;
root /var/www/html;
index index.html index.htm index.nginx-debian.html index.php; #此处添加 index.php
server_name _;

location ~ \.php$ {
include snippets/fastcgi-php.conf;
fastcgi_pass unix:/run/php/php7.0-fpm.sock;
}
location / {
proxy_pass http://roomserver$request_uri;
proxy_set_header Host $host;
}
}
```

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
$jsonObj->uris=array("stun:jonta.cn:3478","turn: jonta.cn:3478?transport=udp","turn: jonta.cn:3478?transport=tcp");
echo json_encode($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 = "";
public $ttl = "";
public $uris = array("");
}
?>
```

iceconfig.php 文件内容

```
<?php
        $request_username = "jonta";  //配置成自己的 turn 服务器用户名
        if(empty($request_username)) {
            echo "username == null";
            exit;
        }
        $request_key = "inesadt";  //配置成自己的 turn 服务器密码
        $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 = "CEOD_KEY";//constants.py 中 CEOD_KEY

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

        $arrayObj = array();
        $arrayObj[0]['username'] = $response_username;
        $arrayObj[0]['credential'] = $response_password;
        //配置成自己的 stun/turn 服务器
        $arrayObj[0]['urls'][0] = "stun:jonta.cn:3478";
        $arrayObj[0]['urls'][1] = "turn:jonta.cn:3478?transport=tcp";
        $arrayObj[0]['uris'][0] = "stun:jonta.cn:3478";
        $arrayObj[0]['uris'][1] = "turn:jonta.cn:3478?transport=tcp";
        $jsonObj = new Response();
        $jsonObj->lifetimeDuration = "300.000s";
        $jsonObj->iceServers = $arrayObj;
        echo json_encode($jsonObj);
```

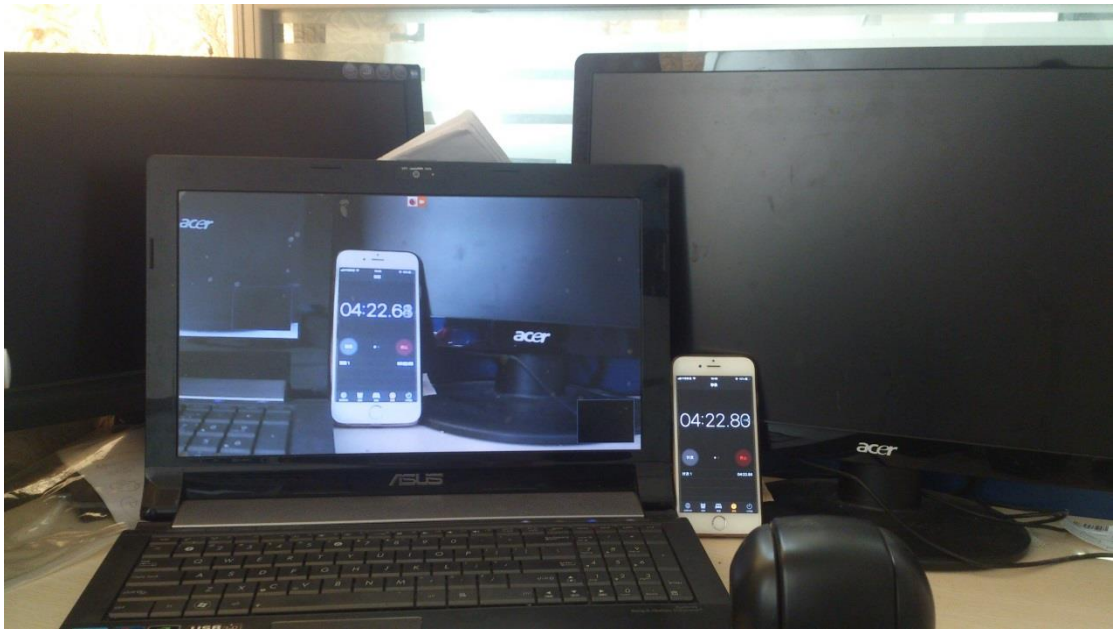
```
        /**
        * 使用 HMAC-SHA1 算法生成签名值
        *
        * @param $str 源串
        * @param $key 密钥
        *
        * @return 签名值
        */
        function getSignature($str, $key) {
            $signature = "";
            if (function_exists('hash_hmac')) {
                $signature = base64_encode(hash_hmac("sha1", $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 $lifetimeDuration = "";
                public $iceServers = array("");
        }
    ?>
```

service nginx restart

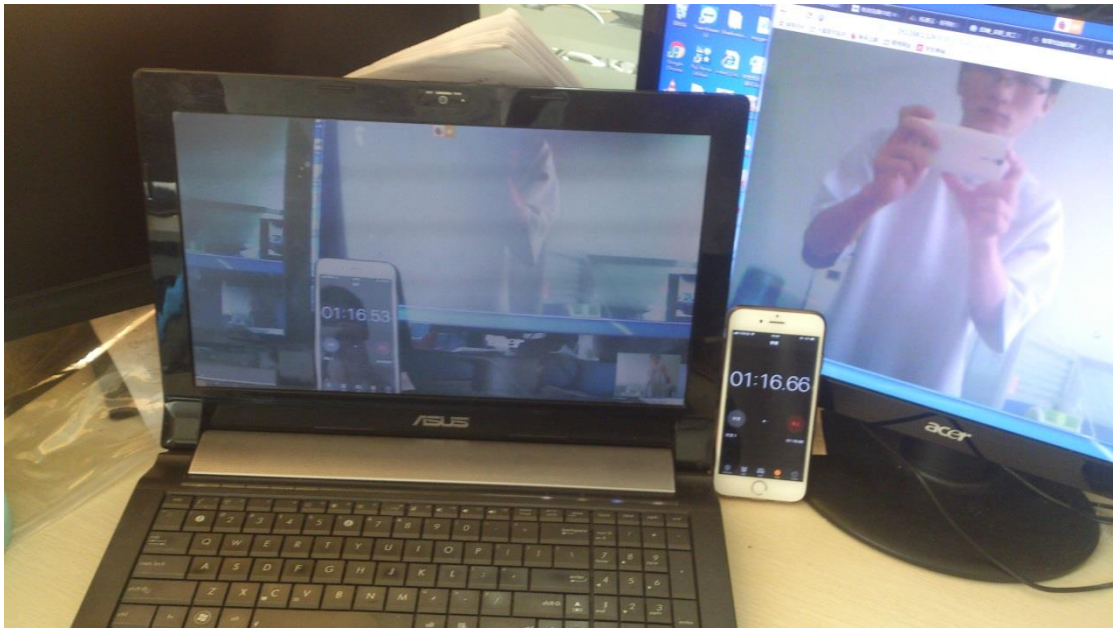
service php7.0-fpm restart

打开 firefox 浏览器输入

jonta.cn



再来一张视频聊天，请忽略我的脸(●'◡'●)



到此就结束啦，看看延时是不是很低。

By Jonta

2018 年 8 月 3 日