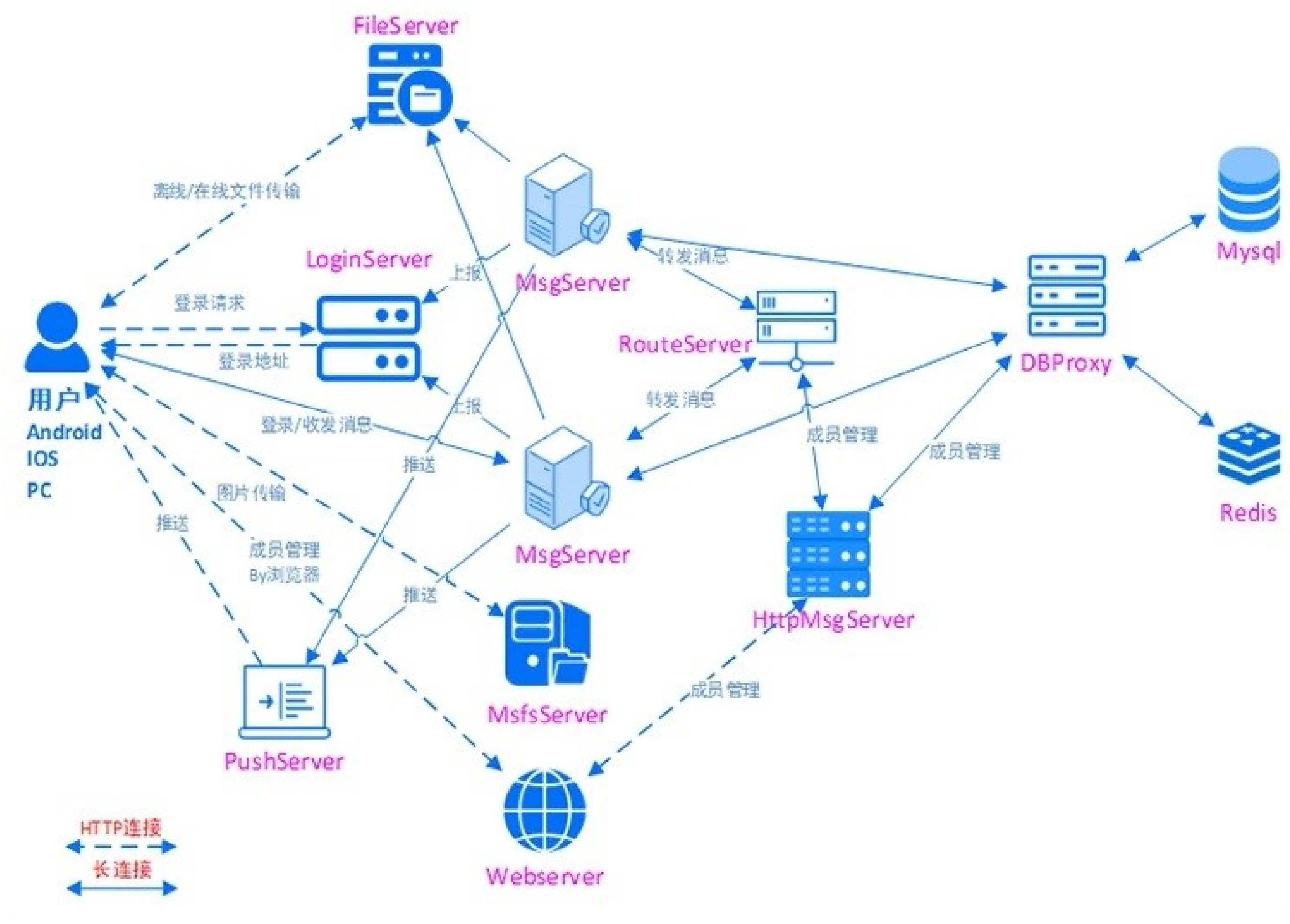
1-2 即时通讯项目编译和配置

零声学院 https://0voice.ke.qq.com
讲师 Darren老师 QQ326873713
班主任 柚子老师 QQ2690491738
2022年6月18日

课程连接：[C/C++Linux服务器开发/高级架构师【零声学院】(https://ke.qq.com/course/420945?tuin=137bb271)

网页链接：
https://www.yuque.com/docs/share/e848ccf0-ae02-4a46-9753-e0cb559ff568?# 《1-2 即时通讯项目编译和配置》



1. 简介

1. 零声学院即时通讯项目，基于开源项目TeamTalk二次开发。
2. TeamTalk目前的开源版本有较多的bug，功能也不够完善，零声学院始于TeamTalk但不止于TeamTalk，将会持续迭代改进该项目。

2. 目录说明

```
|-- android  Android客户端 (Android studio 3.2)
|-- auto_setup  启动目录
|-- doc     说明文档
|-- ios IOS客户端
|-- LICENSE
|-- mac 苹果客户端
|-- pb protocol buffer包格式
|-- php Web后台管理
|-- server 服务器代码 (centos7.0)
|-- win-client Windows客户端 (vs2015编译)
```

3. 编译部署说明（Ubuntu 16.04环境）

分为一下几个步骤

1. 安装mysql、nginx、redis、php等常用组件
2. 编译im
3. 部署im服务器+web管理后台
   安装方法如下所示

源码地址：源码压缩包： 0voice_im.tar.bz2

项目特别提示：

1. **保证redis正常运行；**
2. **保证mysql正常运行；**
3. **保证php正常运行；**
4. **保证nginx正常运行；**
5. **保证php正常运行，这里的php主要是web管理平台的使用；**
6. **protobuf版本库目前server端是2.0的版本，Android_av使用了3.0的版本**
7. **web服务器需要nginx支持；**
8. **服务不启动的时候主要查看log是否正常。**

如果已经安装了 redis、mysql和nginx则不需要重新安装。

3.1 安装常用组件

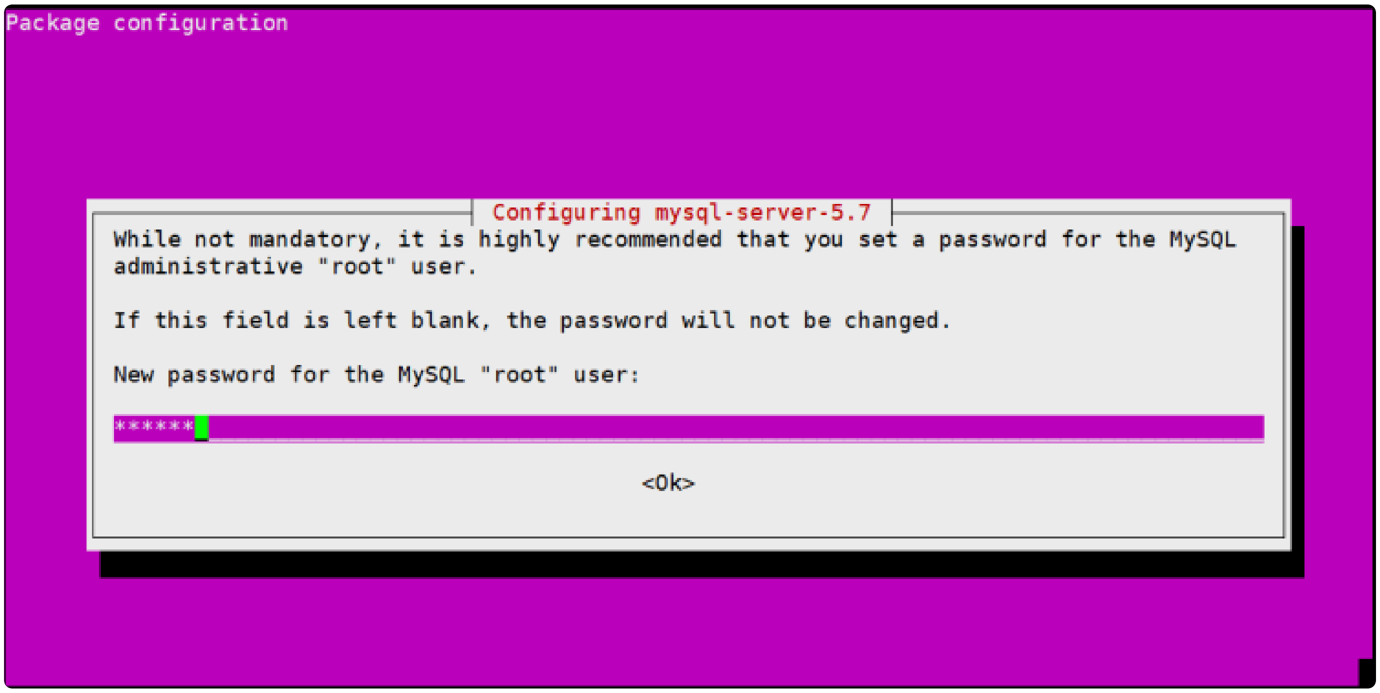
3.1.1 安装mysql

1. **安装mysql服务器端**

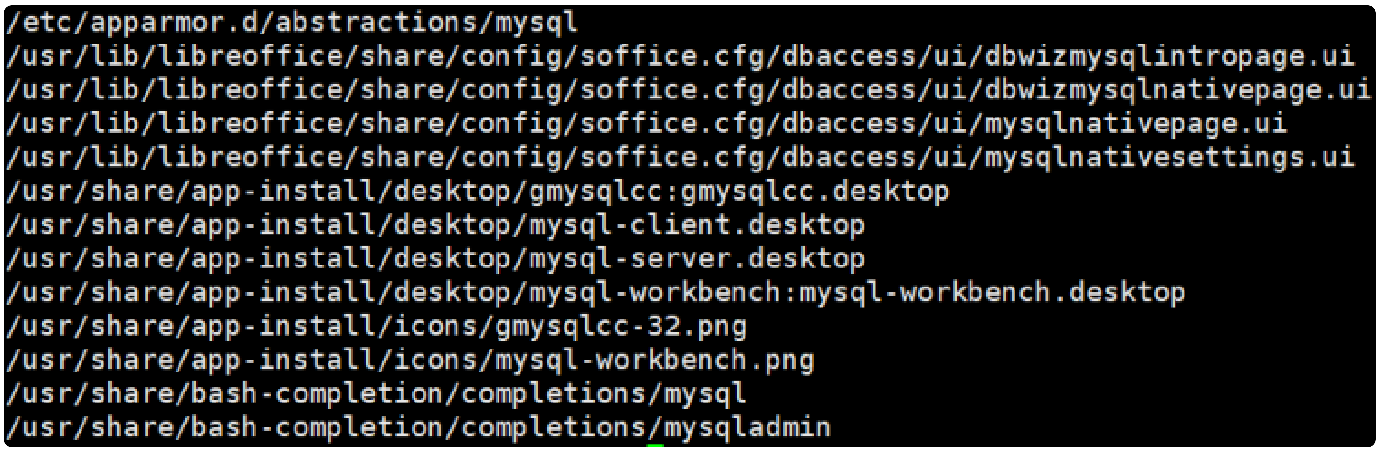
```bash
sudo apt-get -y install mysql-server
```

如果是ubuntu18.04/20.04则不会提示输入密码，默认是没有密码

配置root权限密码，然后按ok，根据提示再次输入密码即可



```
Package configuration

Configuring mysql-server-5.7
While not mandatory, it is highly recommended that you set a password for the MySQL
administrative "root" user.

If this field is left blank, the password will not be changed.

New password for the MySQL "root" user:

******

<Ok>
```



```
/etc/apparmor.d/abstractions/mysql
/usr/lib/libreoffice/share/config/soffice.cfg/dbaccess/ui/dbwizmysqlintropage.ui
/usr/lib/libreoffice/share/config/soffice.cfg/dbaccess/ui/dbwizmysqlnativepage.ui
/usr/lib/libreoffice/share/config/soffice.cfg/dbaccess/ui/mysqlnativepage.ui
/usr/lib/libreoffice/share/config/soffice.cfg/dbaccess/ui/mysqlnativesettings.ui
/usr/share/app-install/desktop/gmysqlcc:gmysqlcc.desktop
/usr/share/app-install/desktop/mysql-client.desktop
/usr/share/app-install/desktop/mysql-server.desktop
/usr/share/app-install/desktop/mysql-workbench:mysql-workbench.desktop
/usr/share/app-install/icons/gmysqlcc-32.png
/usr/share/app-install/icons/mysql-workbench.png
/usr/share/bash-completion/completions/mysql
/usr/share/bash-completion/completions/mysqladmin
```

2. **安装mysql客户端**

```bash
sudo apt-get -y install mysql-client
```

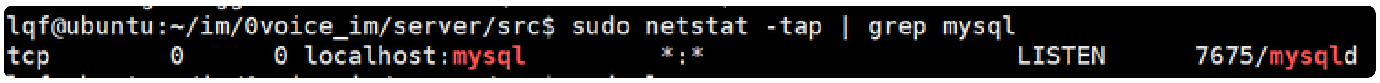
3. **安装mysql模块**

```bash
sudo apt-get -y install libmysqlclient-dev
```

4. **验证是否成功**

```bash
sudo netstat -tap | grep mysql
```

看到

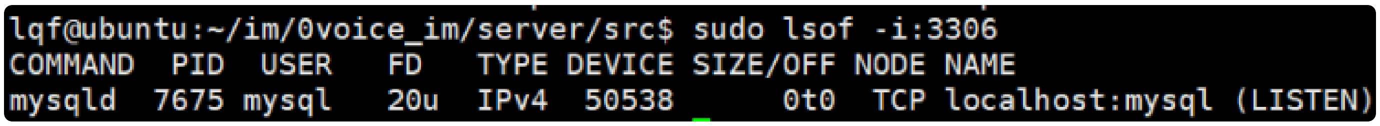


```
lqf@ubuntu:~/im/0voice_im/server/src$ sudo netstat -tap | grep mysql
tcp        0      0 localhost:mysql         *:*                     LISTEN      7675/mysqld
```

或者用lsof查看数据库默认端口3306

```bash
sudo lsof -i:3306
```

看到



```
lqf@ubuntu:~/im/0voice_im/server/src$ sudo lsof -i:3306
COMMAND  PID  USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
mysqld  7675 mysql   20u  IPv4  50538      0t0  TCP localhost:mysql (LISTEN)
```

5. **进入mysql**

Bash 复制代码

```bash
mysql -u root -p
```

如果是ubuntu18.04则以无密码的方式登录进去然后设置密码，**就得是在root权限下，比如加上sudo**

Plain Text 复制代码

```
sudo mysql -u root -p 直接按回车登录然后设置密码
mysql>use mysql;   然后敲回车(注意下面字母的大小写)
#  更新 plugin 及 authentication_string 字段，比如密码123456
mysql> UPDATE user SET plugin="mysql_native_password",
authentication_string=PASSWORD("123456") WHERE user="root";
#  输出以下结果
Query OK, 1 row affected, 1 warning (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 1

 #  保存更新结果
mysql> FLUSH PRIVILEGES;
# 退出并重启 mysql
mysql> exit;
sudo service mysql restart
```

需要特别注意的是ubuntu18.04版本在操作数据库的时候需要sudo权限

注意：Ubuntu20.4 的设置密码又不一样（mysql –V打印mysql版本）：参考：https://blog.csdn.net/delilahy123/article/details/124834998解决

6. 启动/停止/重启mysql

常用命令，这里不是让大家关闭mysql，只是告知有这么些命令。

> service mysql start //启动mysql
> service mysql restart //重新启动mysql
> service mysql stop //关闭mysql

3.1.2 安装redis和 hiredis（如果已经有就不用再安装）

以源码方式安装，
注意（hiredis的安装，在dbproxy要使用到）

1. **下载到合适的位置，自己指定**

> wget http://download.redis.io/releases/redis-6.2.5.tar.gz

2. **解压**

> tar -zxvf redis-6.2.5.tar.gz

3. **编译**

> cd redis-6.2.5
> make

4. **编译安装依赖文件**

> cd deps
> make hiredis linenoise lua jemalloc
> cd hiredis
> sudo make install
> cd ../lua
> sudo make install

5. **安装redis**

> cd ../../src
> sudo make install

6. **启动Redis后台运行**

修改配置文件：修改redis.conf文件：vi打开redis-server配置的redis.conf文件（在redis源码目录下），然后使用快捷匹配模式：/stop-writes-on-bgsave-error定位到stop-writes-on-bgsave-error字符串所在位置，接着把后面的yes设置为no即可。另外：daemonize no改为daemonize yes。

redis-server /绝对路径/to/redis.conf
比如：

目前这个项目 **redis不要设置授权信息，auth， 如果设置了auth导致 服务连不上redis。**

然后用sudo lsof -i:6379 查看是否有监听相应的6379 端口

3.1.3 安装nginx（如果已经安装则不需要）

3.1.3.1 安装nginx

1. 安装必要的第三方依赖包

```bash
sudo apt-get -y install libpcre3 libpcre3-dev
sudo apt-get -y install zlib1g-dev
sudo apt-get install openssl libssl-dev
```

2. 下载比较新的稳定版本，自己放到合适的位置

```bash
wget http://nginx.org/download/nginx-1.16.1.tar.gz
```

3. 解压配置编译安装

```bash
tar -zxvf nginx-1.16.1.tar.gz
cd nginx-1.16.1
./configure --prefix=/usr/local/nginx
make
sudo make install
```

安装后的执行文件路径：/usr/local/nginx/sbin/
配置文件路径：/usr/local/nginx/conf/
4. 创建配置文件子目录和在nginx.conf中包含conf.d

Bash 复制代码

```bash
sudo mkdir /usr/local/nginx/conf/conf.d
```

修改/usr/local/nginx/conf/nginx.conf

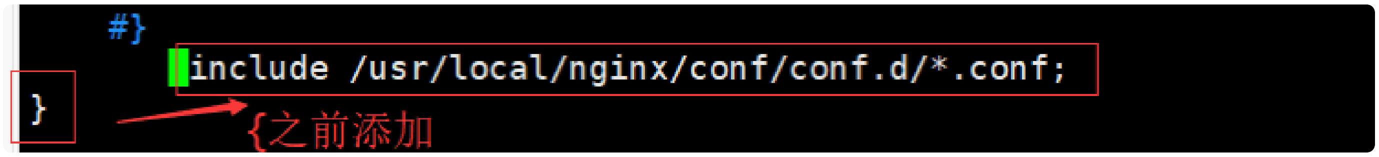
Bash 复制代码

```bash
sudo vim /usr/local/nginx/conf/nginx.conf
```

末尾{之前添加

> include /usr/local/nginx/conf/conf.d/*.conf;

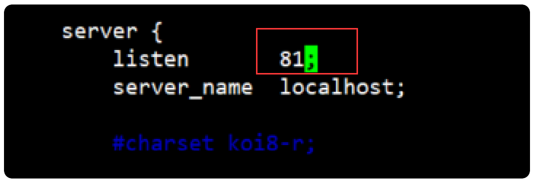
如图所示



```
    #}
        include /usr/local/nginx/conf/conf.d/*.conf;
}
```

目的是在部署即时通讯的网页配置端时将对应的.conf文件拷贝到 /usr/local/nginx/conf/conf.d目录时能够被nginx.conf包含。

5. 将默认的80端口改成81，因为即时通讯的web后台管理需要用到80端口。



```
    server {
        listen       81;
        server_name  localhost;

        #charset koi8-r;
```

3.1.3.2 配置启动nginx

1. 创建一个nginx.service

在 /lib/systemd/system/目录下面新建一个nginx.service文件。并赋予可执行的权限。

Bash 复制代码

```bash
sudo vim /lib/systemd/system/nginx.service
```

2. 编辑service内容

Plain Text 复制代码

```
[Unit]
Description=nginx - high performance web server
After=network.target remote-fs.target nss-lookup.target
[Install]
WantedBy=multi-user.target
[Service]
Type=forking
ExecStart=/usr/local/nginx/sbin/nginx -c /usr/local/nginx/conf/nginx.conf
ExecReload=/usr/local/nginx/sbin/nginx -s reload
ExecStop=/usr/local/nginx/sbin/nginx -s stop
```

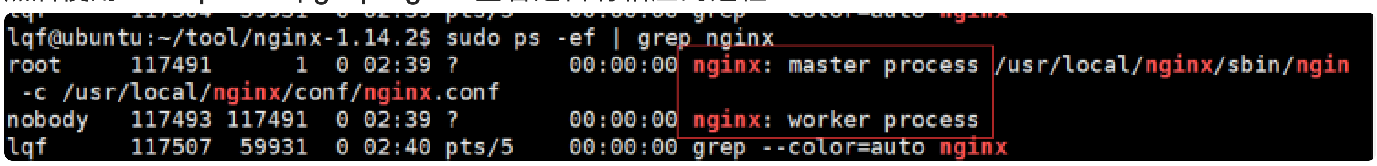
3. 启动服务

在启动服务之前，需要先重载systemctl命令

Bash 复制代码

```bash
sudo systemctl daemon-reload
# 启动nginx
sudo systemctl start nginx.service
```

然后使用**sudo ps -ef | grep nginx** 查看是否有相应的进程



```
lqf@ubuntu:~/tool/nginx-1.14.2$ sudo ps -ef | grep nginx
root      117491      1  0 02:39 ?        00:00:00 nginx: master process /usr/local/nginx/sbin/ngin
 -c /usr/local/nginx/conf/nginx.conf
nobody    117493 117491  0 02:39 ?        00:00:00 nginx: worker process
lqf       117507  59931  0 02:40 pts/5    00:00:00 grep --color=auto nginx
```

4. 常用命令

> 启动：sudo systemctl start nginx.service
>
> 停止：sudo systemctl stop nginx.service
>
> 重新加载：sudo systemctl reload nginx.service
>
> 显示nginx服务的状态：systemctl status nginx.service

> 在开机时启用nginx服务：sudo systemctl enable nginx.service
> 在开机时禁用nginx服务：sudo systemctl disable nginx.service

3.1.3.3 创建conf子目录

```bash
sudo mkdir -p /usr/local/nginx/conf/conf.d/
```

3.1.4 安装php（ubuntu16.04版本）

（如果是你是ubuntu18+版本直接看3.1.5）

1. 安装PHP7以及常用扩展

```bash
sudo apt-get -y install php7.0-fpm php7.0-mysql php7.0-common php7.0-mbstring php7.0-gd php7.0-json php7.0-cli php7.0-curl
```

2. 启动php7.0-fpm进程

```bash
sudo systemctl start php7.0-fpm
```

3. 查看php7.0-fpm运行状态。

```bash
systemctl status php7.0-fpm
或
/etc/init.d/php7.0-fpm status
```

4. 测试PHP是否安装成功

```bash
php -v
或
php --version
```

5. **修改php配置文件**，修改www.conf

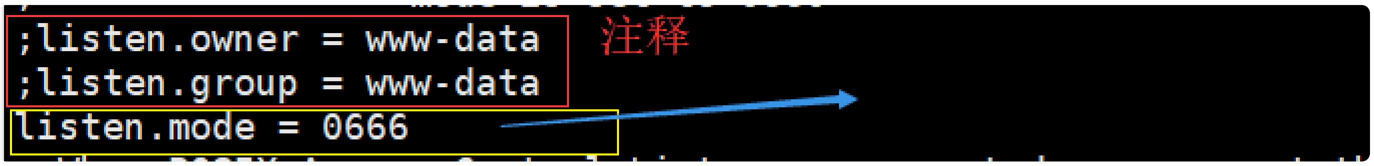
```bash
sudo vim /etc/php/7.0/fpm/pool.d/www.conf
```

做如下处理（大概49行）
注释掉（用;）

> ;listen.owner = www-data
> ;listen.group = www-data

将mode值修改为0666

> listen.mode = 0666



```
;listen.owner = www-data
;listen.group = www-data
listen.mode = 0666
```

6. 最后，执行sudo /etc/init.d/php7.0-fpm restart重启php-fpm服务

```bash
sudo /etc/init.d/php7.0-fpm restart
```

3.1.5 安装php（ubuntu18.04版本）

1. 安装PHP7以及常用扩展

Bash 复制代码

```bash
sudo apt-get -y install php7.2-fpm php7.2-mysql php7.2-common php7.2-mbstring php7.2-gd php7.2-json php7.2-cli php7.2-curl
```

2. 启动php7.2-fpm进程

Bash 复制代码

```bash
sudo systemctl start php7.2-fpm
```

3. 查看php7.2-fpm运行状态。

Bash 复制代码

```bash
systemctl status php7.2-fpm
或
/etc/init.d/php7.2-fpm status
```

4. 测试PHP是否安装成功

Bash 复制代码

```bash
php -v
或
php --version
```

5. **修改php配置文件**
   修改www.conf

Bash 复制代码

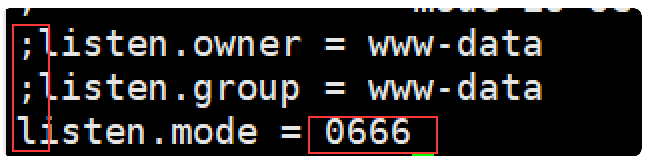
```bash
sudo vim /etc/php/7.2/fpm/pool.d/www.conf
```

做如下处理（大概47行）
注释掉

> ;listen.owner = www–data
>
> ;listen.group = www–data

将mode值修改为0666

> listen.mode = 0666



```
;listen.owner = www-data
;listen.group = www-data
listen.mode = 0666
```

6. 最后，执行sudo /etc/init.d/php7.2–fpm restart重启php–fpm服务

```bash
sudo /etc/init.d/php7.2–fpm restart
```

3.1.6 安装php（ubuntu 20.04版本）

1. 安装PHP7以及常用扩展

```bash
sudo apt-get -y install php7.4–fpm php7.4–mysql php7.4–common php7.4–mbstring php7.4–gd php7.4–json php7.4–cli php7.4–curl
```

2. 启动php7.4–fpm进程

```bash
sudo systemctl start php7.4–fpm
```

3. 查看php7.4–fpm运行状态。

Bash 复制代码

```bash
systemctl status php7.4-fpm
或
/etc/init.d/php7.4-fpm status
```

4. 测试PHP是否安装成功

Bash 复制代码

```bash
php -v
或
php --version
```

5. **修改php配置文件**
   修改www.conf

Bash 复制代码

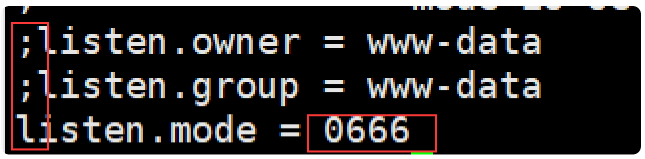
```bash
sudo vim /etc/php/7.4/fpm/pool.d/www.conf
```

做如下处理（大概47行）
注释掉

> ;listen.owner = www-data
> ;listen.group = www-data

将mode值修改为0666

> listen.mode = 0666



```
;listen.owner = www-data
;listen.group = www-data
listen.mode = 0666
```

6. **最后，执行sudo /etc/init.d/php7.4-fpm restart重启php-fpm服务**

```c
sudo /etc/init.d/php7.4-fpm restart
```

3.2 编译IM

先解压0voice_im

```bash
tar jxf 0voice_im.tar.bz2
```

进入目录0voice_im/server/src

1. 编译protocol buffer库（编译之前确保已经安装了g++编译器，一般云服务器默认没有安装g++的）

```bash
sudo ./make_protobuf.sh
```

2. 编译日志log4

```bash
sudo ./make_log4cxx.sh
```

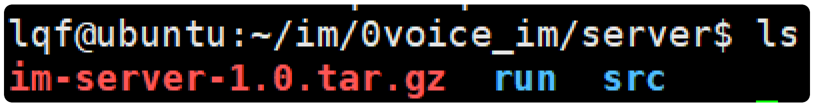
3. 编译IM模块

```bash
sudo ./build_ubuntu.sh version 1.0
```

4. 编译完成后返回上一级目录，即是回到0voice_im/server目录

```bash
cd ..
```

此时多了im-server-1.0.tar.gz压缩包（**注意是在0voice_im/server这一级目录**）



```
lqf@ubuntu:~/im/0voice_im/server$ ls
im-server-1.0.tar.gz  run  src
```

将im-server-1.0.tar.gz拷贝到0voice_im/auto_setup/im_server

> cp im-server-1.0.tar.gz ../auto_setup/im_server/

4 部署IM

进入0voice_im/auto_setup，准备部署im

4.1 导入teamtalk数据库

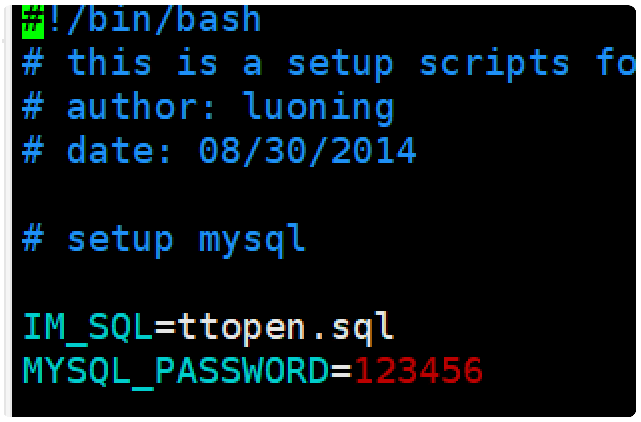
注意数据库的密码和在安装mysql时候一致

1. **在setup.sh里面修改数据库密码**

> cd auto_setup/mysql
> vim setup.sh

将数据库密码改为在mysql时配置的密码，我在配置时使用了123456，所以改为

> MYSQL_PASSWORD=123456

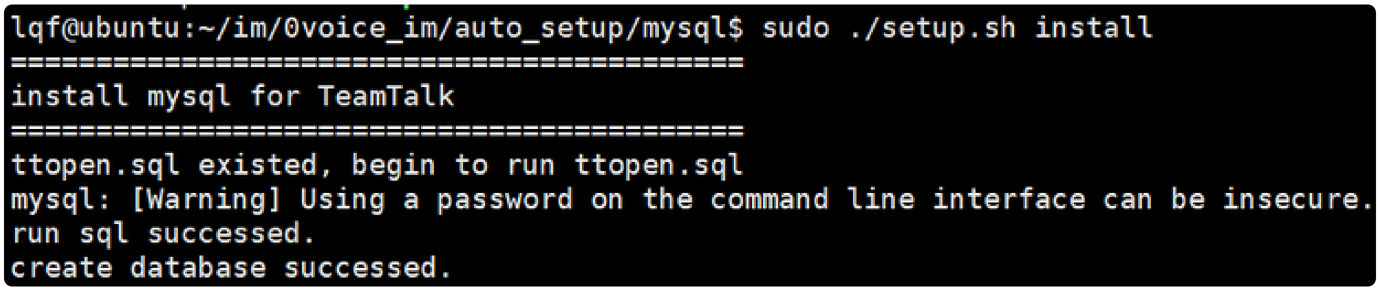


```
#!/bin/bash
# this is a setup scripts fo
# author: luoning
# date: 08/30/2014

# setup mysql

IM_SQL=ttopen.sql
MYSQL_PASSWORD=123456
```

2. 导入数据库

> sudo ./setup.sh install

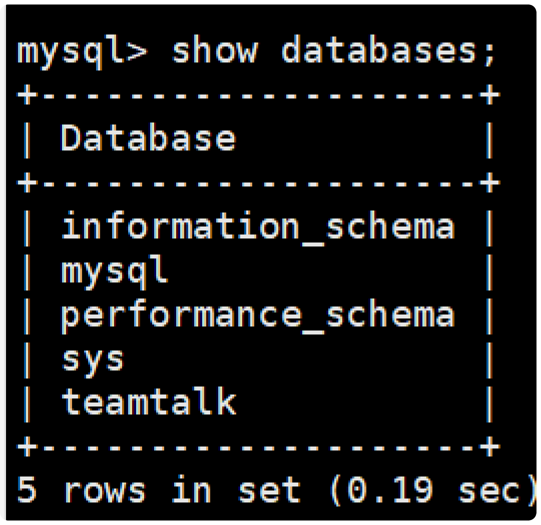
导入成功的结果



```
lqf@ubuntu:~/im/0voice_im/auto_setup/mysql$ sudo ./setup.sh install
============================================
install mysql for TeamTalk
============================================
ttopen.sql existed, begin to run ttopen.sql
mysql: [Warning] Using a password on the command line interface can be insecure.
run sql successed.
create database successed.
```

也可以进入mysql做二次确认

> mysql –u root –p
> show databases;



```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
| teamtalk           |
+--------------------+
5 rows in set (0.19 sec)
```

看到数据库有teamtalk

4.2 启动IM

4.2.1 先配置模块配置文件

我们先单机部署，主要是修改msgserver.conf、dbproxyserver.conf和loginserver.conf
先回到0voice_im目录

```
cd auto_setup/im_server
```

修改msgserver.conf

1. **修改msgserver.conf消息服务配置文件**

修改auto_setup/im_server/conf目录的配置文件，该目录存储了各个server对应的配置文件，如果配置错误系统将无法启动。

```
vim conf/msgserver.conf
```

将大概23行的FileServerIP改成FileServer所在的服务器的外网地址

```
#连接fileserver
FileServerIP1=192.168.206.131
#连接fileserver
FileServerPort1=8601
#FileServerIP2=localhost
#FileServerPort2=8601
```

将大概28行的IpAddr1和IpAddr2改为本机IP外网地址，如果是局域网部署则改为局域网ip，比如

```
#该IP 注册到LoginServer
IpAddr1=192.168.221.131 #电信IP
IpAddr2=192.168.221.131 #网通IP
```

另外：

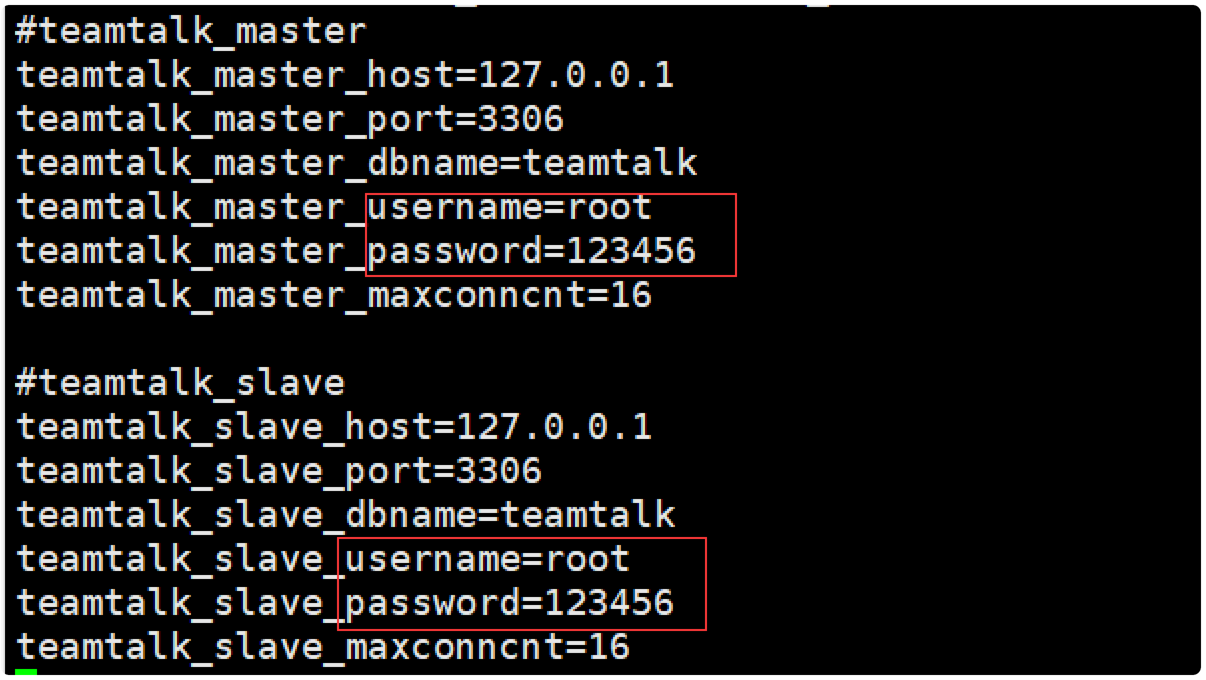
- DBServerIP1对应 db_proxy_server如果不在同一个服务器，则也改成db_proxy_server所在服务器的地址。
- LoginServerIP1对应 login_server如果不在同一个服务器，则也改成login_server所在服务器的地址。

修改dbproxyserver.conf

2. 修改dbproxyserver.conf存储中间件配置文件

修改访问db的密码（master为写入，slave为读取）

> vim conf/dbproxyserver.conf



```
#teamtalk_master
teamtalk_master_host=127.0.0.1
teamtalk_master_port=3306
teamtalk_master_dbname=teamtalk
teamtalk_master_username=root
teamtalk_master_password=123456
teamtalk_master_maxconncnt=16

#teamtalk_slave
teamtalk_slave_host=127.0.0.1
teamtalk_slave_port=3306
teamtalk_slave_dbname=teamtalk
teamtalk_slave_username=root
teamtalk_slave_password=123456
teamtalk_slave_maxconncnt=16
```

修改loginserver.conf

3. 修改loginserver.conf

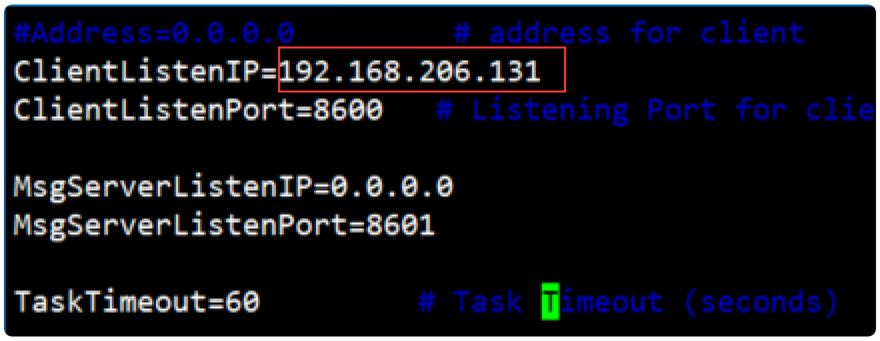
> vim conf/loginserver.conf

主要是修改msfs对应的ip，比如： msfs=http://192.168.221.131:8700/

如果msfs的地址不对，则不能传输图片。

修改fileserver.conf

修改为**外网ip地址，供客户端可以访问**。



```
#Address=0.0.0.0          # address for client
ClientListenIP=192.168.206.131
ClientListenPort=8600    # Listening Port for clie

MsgServerListenIP=0.0.0.0
MsgServerListenPort=8601

TaskTimeout=60          # Task Timeout (seconds)
```

4.2.2 启动服务器模块

1. **启动相应的服务**

> sudo ./setup.sh install

2. **查看相应的服务是否都已经启动**

查找大部分后缀为_server的模块

```bash
sudo ps -ef | grep _server
root        580      1  0 17:06 ?        00:00:00 ./login_server
root        584      1  0 17:06 ?        00:00:01 ./route_server
root        588      1  0 17:06 ?        00:00:01 ./msg_server
root        592      1  0 17:06 ?        00:00:00 ./file_server
root        601      1  0 17:06 ?        00:00:01 ./http_msg_server
root        605      1  0 17:06 ?        00:00:00 ./push_server
root        609      1  0 17:06 ?        00:00:10 ./db_proxy_server
```

查找msfs的模块

```bash
sudo ps -ef | grep msfs
ubuntu     4124 22368  0 18:29 pts/3    00:00:00 grep --color=auto msfs
root      19990      1  0 Mar25 ?        00:00:18 ./msfs
```

如果有相应的模块没有起来，则需要通过log去排除。
比如进入到0voice_im/auto_setup/im_server/im-server-1.0目录
使用tail目录

```bash
tail -n 30 -f msg_server/log/default.log
```

查看是否启动失败。

可以单独重启某个服务模块，比如
lqf@ubuntu:~/im/0voice_im/auto_setup/im_server/im-server-1.0$ sudo ./restart.sh login_server

3. 端口说明，可以使用sudo lsof -i:端口号（或者sudo netstat -tunlp查看所有端口） 查看相应进程是否起来

如果是云服务器有防火墙的服务，一定要记得开放端口，**建议在云服务器部署的时候先开放以下所有的端口。**
8100,8080,8000,10600,8200,8400,8500,8600,8601,8700

服务	说明	端口	备注(端口放行)
login_server	登录服务器，负责身份验证，负责给登录成功的客户端分配msg_server。 这个服务监听在两个端口，一个是tcp端口8100，用于和后端的服务器交互，另一个是http端口8080，需对外开放	8100 8080	8100需要对外开放 8080需要对外开放
msg_server	消息服务器，用户登录成功后，就和指定的消息服务器交互。	8000	8000需对外开放
db_proxy_server	数据库中间件，后端为存储层，mysql和redis	10600	
route_server	消息转发，不同msg_server上用户交互需要中转站来转发消息。	8200	
http_msg_server	主要提供对外的web api	8400	
push_server	消息推送服务（暂时先不用）	8500	
file_server	文件中转站，临时存储	8600 8601	需对外开放 需对外开放
msfs	小文件永久存储，聊天的图片、表情等，端口	8700	需对外开放

4.2.3 单独启动某个模块

可以单独重启某个服务模块，比如

lqf@ubuntu:~/im/0voice_im/auto_setup/im_server/im-server-1.0$ sudo ./restart.sh login_server

5 部署web管理后台

1. **回到0voice_im目录**

将php拷贝一份为tt压缩后拷贝到0voice_im/auto_setup/im_web/

> cd 0voice_im 回到根目录
> cp –ar php tt
> zip –r tt.zip tt
> cp tt.zip auto_setup/im_web/
> cd auto_setup/im_web

2. 修改web的配置文件

主要是ip地址和数据库用户名和密码以及nginx的配置

（1）ip地址修改

> vim conf/config.php

修改为自己的外网地址

Plain Text 复制代码
```
    $config['msfs_url'] = 'http://192.168.221.131:8700/';
    $config['http_url'] = 'http://192.168.221.131:8400';
```

（2）数据库修改

> vim conf/database.php

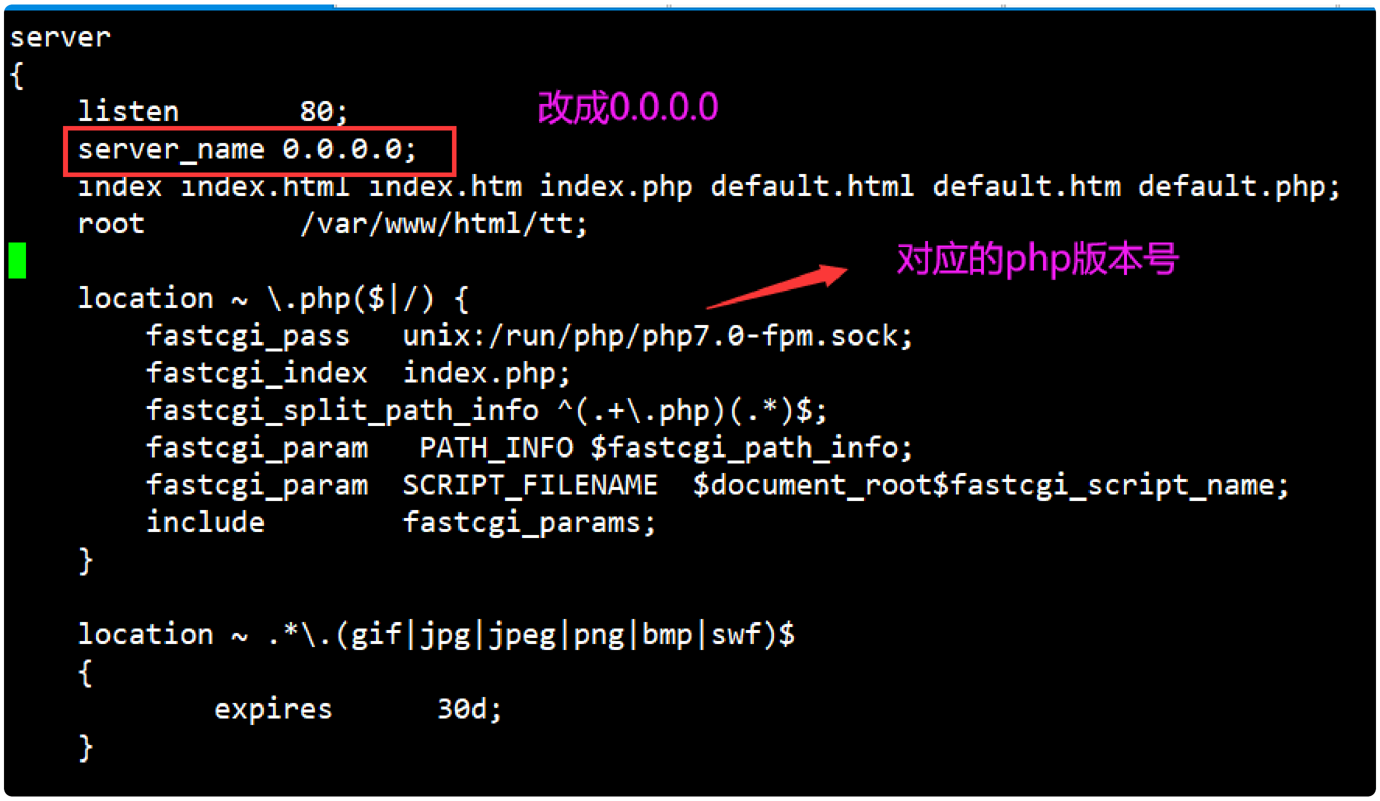
修改为：

Plain Text 复制代码
```
  $db['default']['username'] = 'root';
  $db['default']['password'] = '123456';
```

（3）nginx配置

> vim conf/im.com.conf

以及将php改成你对应的版本号（ubuntu18.x改为7.2）。

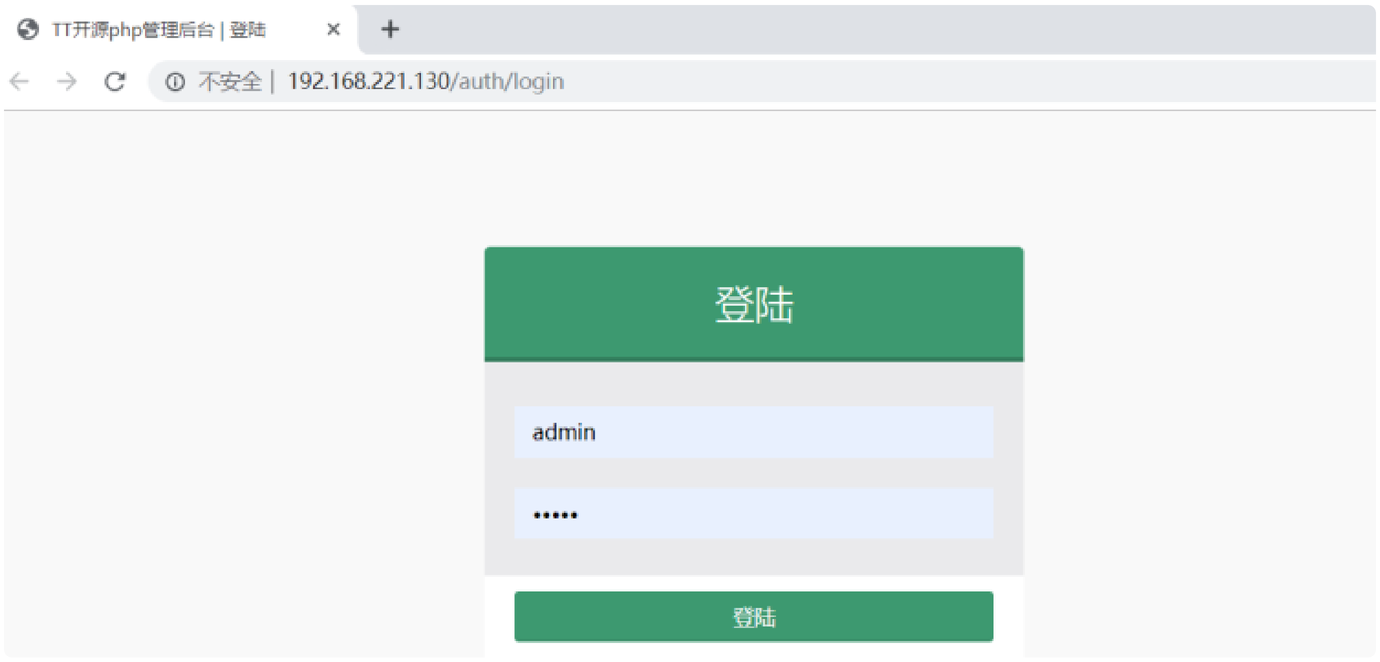


```
server
{
    listen       80;
    server_name 0.0.0.0;
    index index.html index.htm index.php default.html default.htm default.php;
    root         /var/www/html/tt;

    location ~ \.php($|/) {
        fastcgi_pass   unix:/run/php/php7.0-fpm.sock;
        fastcgi_index  index.php;
        fastcgi_split_path_info ^(.+\.php)(.*)$;
        fastcgi_param   PATH_INFO $fastcgi_path_info;
        fastcgi_param  SCRIPT_FILENAME  $document_root$fastcgi_script_name;
        include        fastcgi_params;
    }

    location ~ .*\.(gif|jpg|jpeg|png|bmp|swf)$
    {
            expires      30d;
    }
```

3. **部署web**

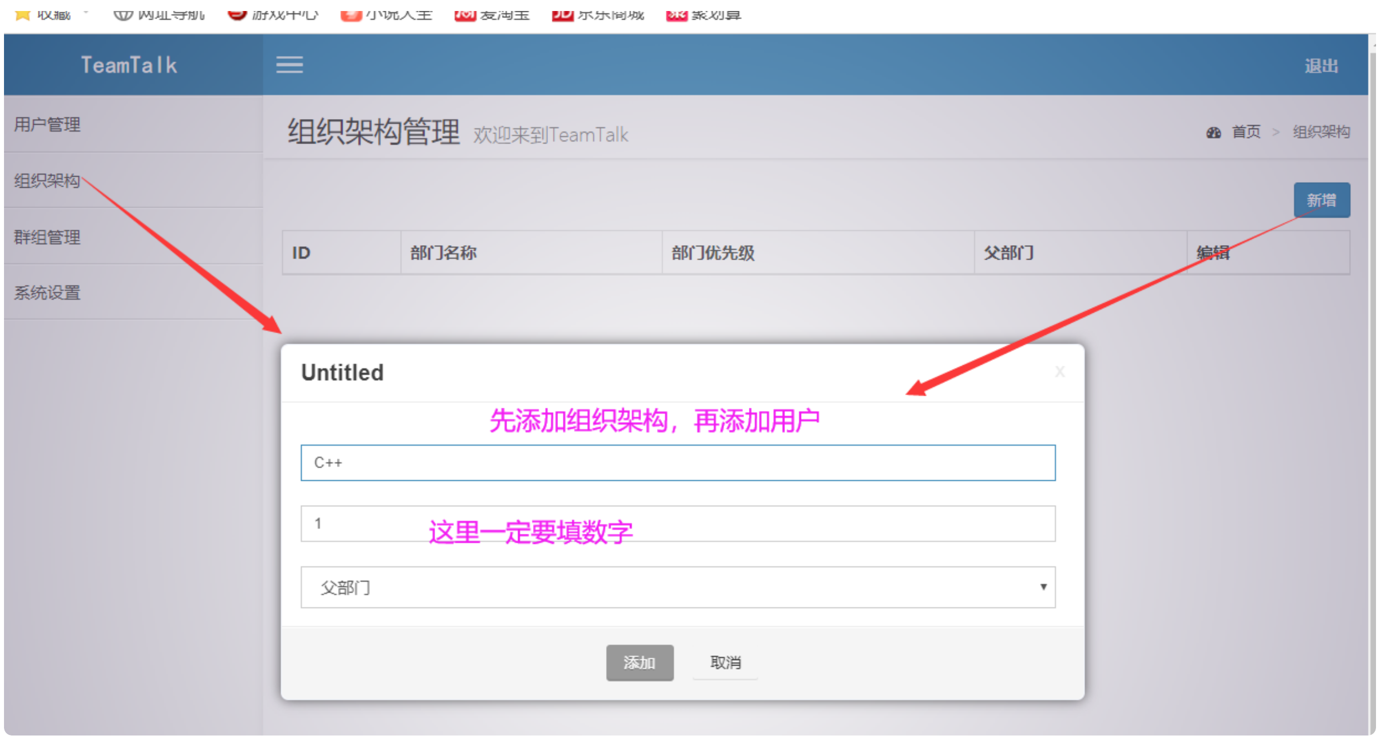
> sudo ./setup.sh install

4. 务必使用**谷歌浏览器**（其他浏览器有兼容性问题）打开服务器的ip地址，比如192.168.221.131，然后用户名和密码都为**admin**

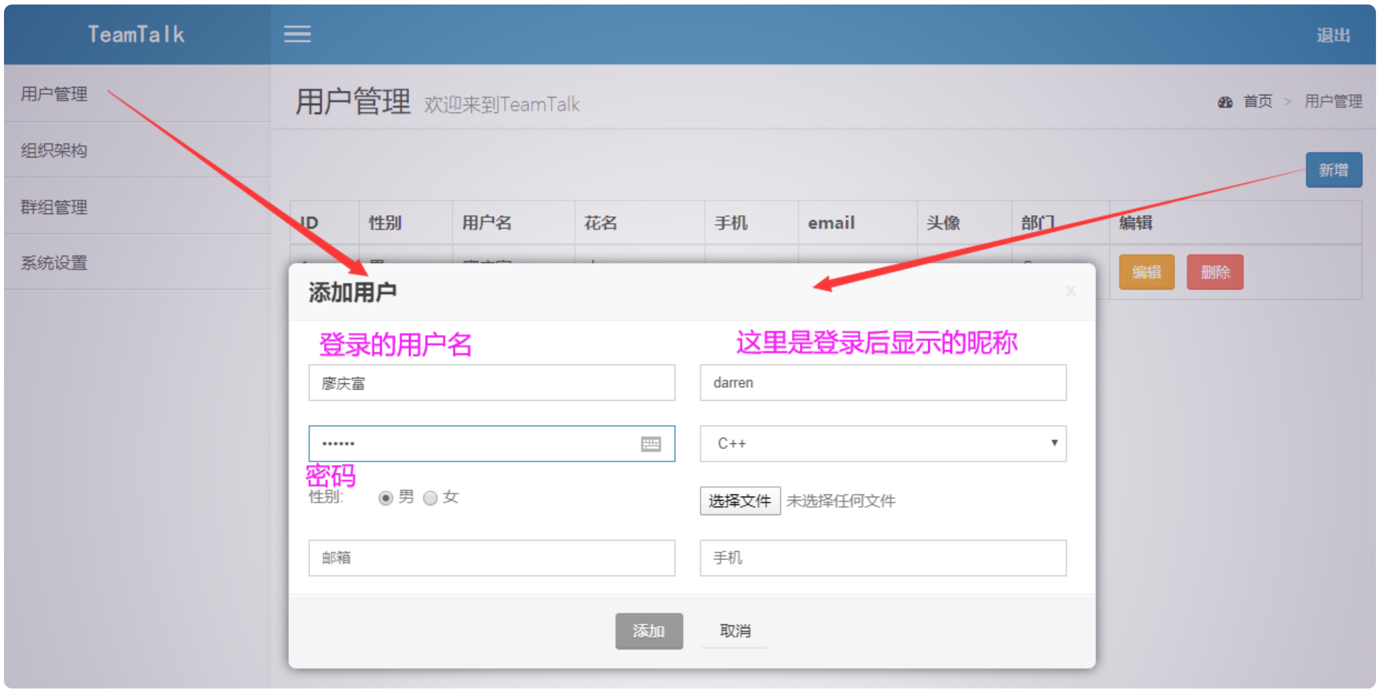


登录进去后则可以配置部门和成员。
比如添加c++部门和成员darren、king、mark，密码都为123456

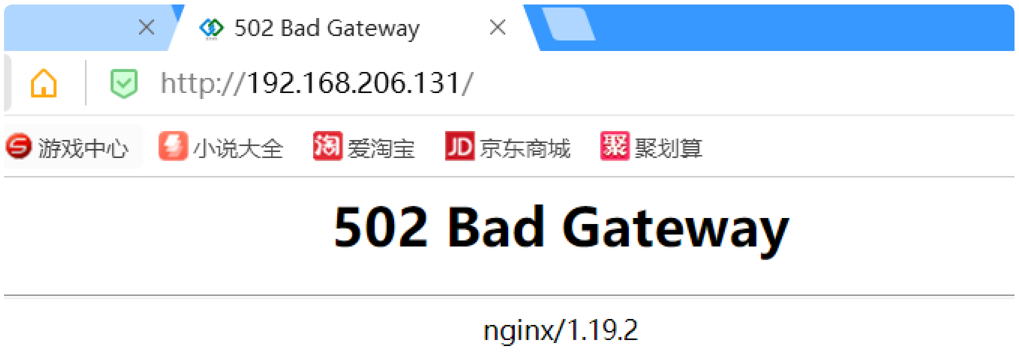
（1）添加组织架构，



（2）添加用户(**添加后手动刷新页面显示添加的信息**)

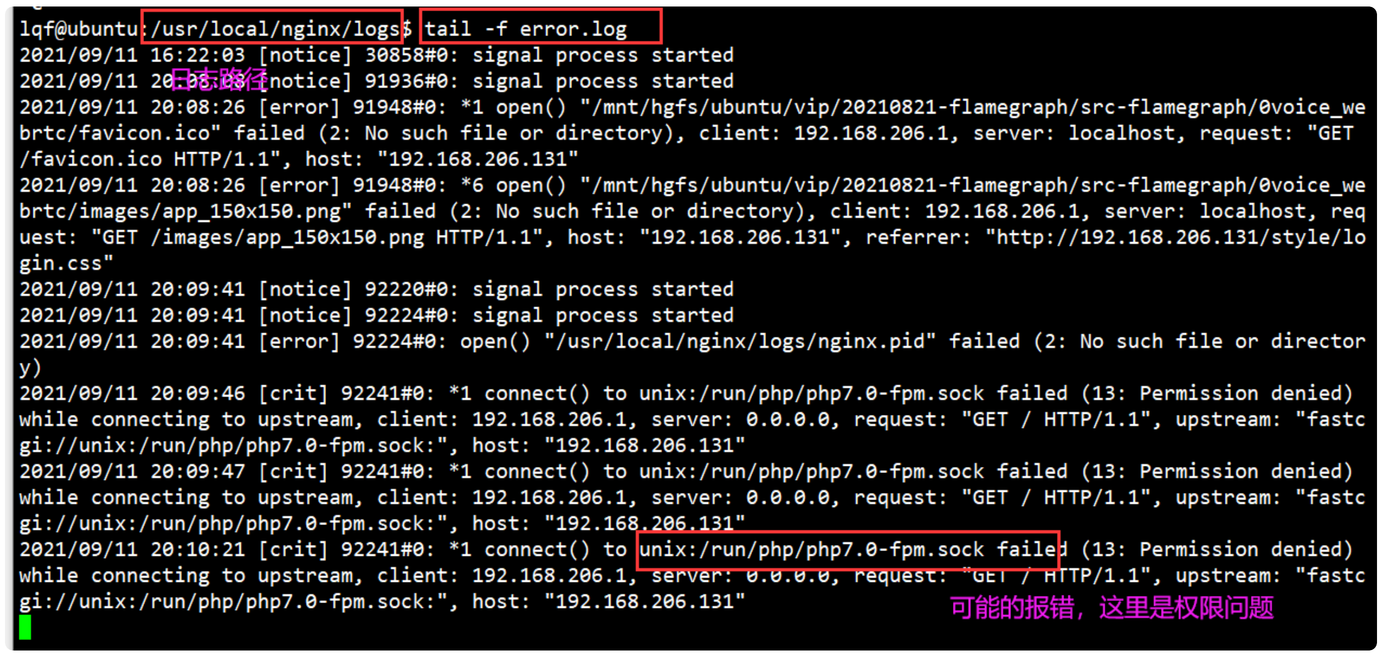


异常情况



502 Bad Gateway

nginx/1.19.2



```
lqf@ubuntu:/usr/local/nginx/logs$ tail -f error.log
2021/09/11 16:22:03 [notice] 30858#0: signal process started
2021/09/11 20:08:26 [notice] 91936#0: signal process started
2021/09/11 20:08:26 [error] 91948#0: *1 open() "/mnt/hgfs/ubuntu/vip/20210821-flamegraph/src-flamegraph/0voice_we
brtc/favicon.ico" failed (2: No such file or directory), client: 192.168.206.1, server: localhost, request: "GET
/favicon.ico HTTP/1.1", host: "192.168.206.131"
2021/09/11 20:08:26 [error] 91948#0: *6 open() "/mnt/hgfs/ubuntu/vip/20210821-flamegraph/src-flamegraph/0voice_we
brtc/images/app_150x150.png" failed (2: No such file or directory), client: 192.168.206.1, server: localhost, req
uest: "GET /images/app_150x150.png HTTP/1.1", host: "192.168.206.131", referrer: "http://192.168.206.131/style/lo
gin.css"
2021/09/11 20:09:41 [notice] 92220#0: signal process started
2021/09/11 20:09:41 [notice] 92224#0: signal process started
2021/09/11 20:09:41 [error] 92224#0: open() "/usr/local/nginx/logs/nginx.pid" failed (2: No such file or director
y)
2021/09/11 20:09:46 [crit] 92241#0: *1 connect() to unix:/run/php/php7.0-fpm.sock failed (13: Permission denied)
while connecting to upstream, client: 192.168.206.1, server: 0.0.0.0, request: "GET / HTTP/1.1", upstream: "fastc
gi://unix:/run/php/php7.0-fpm.sock:", host: "192.168.206.131"
2021/09/11 20:09:47 [crit] 92241#0: *1 connect() to unix:/run/php/php7.0-fpm.sock failed (13: Permission denied)
while connecting to upstream, client: 192.168.206.1, server: 0.0.0.0, request: "GET / HTTP/1.1", upstream: "fastc
gi://unix:/run/php/php7.0-fpm.sock:", host: "192.168.206.131"
2021/09/11 20:10:21 [crit] 92241#0: *1 connect() to unix:/run/php/php7.0-fpm.sock failed (13: Permission denied)
while connecting to upstream, client: 192.168.206.1, server: 0.0.0.0, request: "GET / HTTP/1.1", upstream: "fastc
gi://unix:/run/php/php7.0-fpm.sock:", host: "192.168.206.131"
```

是因为php我们修改了www.conf配置文件后没有重启php。**（注意自己的php版本）**

Bash 复制代码

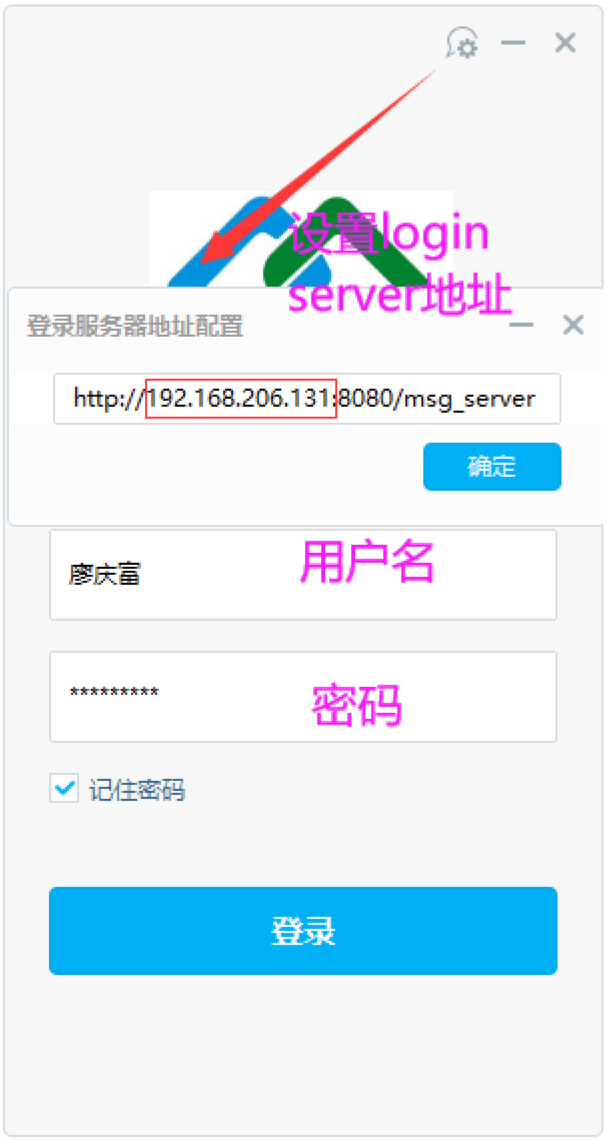
```bash
sudo /etc/init.d/php7.0-fpm restart
```

6 客户端登录

已经编译好的客户端可执行文件路径：0voice_im/win-client/bin/teamtalk/Release/0voice_im.exe，点击可以运行。

注意配置成自己msg_server的IP地址

注意是用户名登录，不是花名。

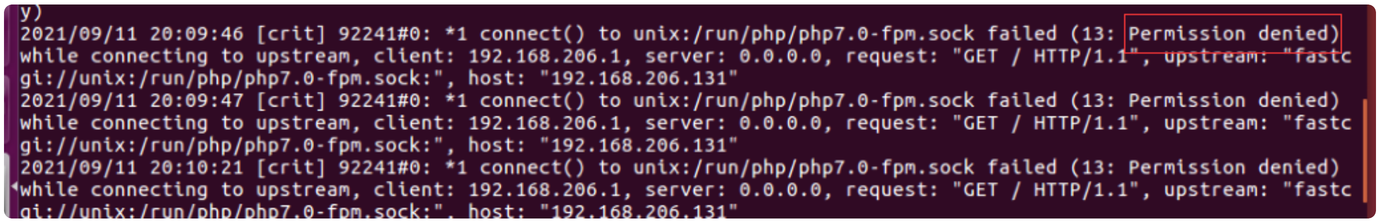


清空原有用户数据库 C:\ProgramData\TeamTalkOpen

7 异常排查

查看nginx日志

tail –n 30 –f /usr/local/nginx/logs/error.log

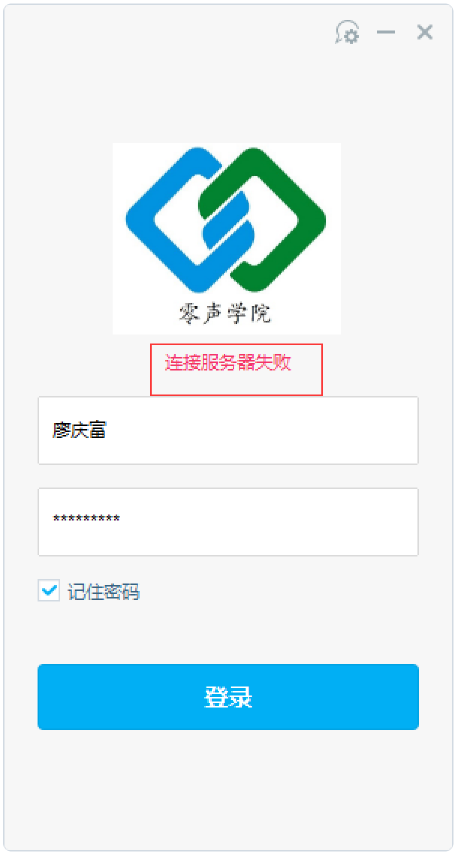


```
y)
2021/09/11 20:09:46 [crit] 92241#0: *1 connect() to unix:/run/php/php7.0-fpm.sock failed (13: Permission denied)
while connecting to upstream, client: 192.168.206.1, server: 0.0.0.0, request: "GET / HTTP/1.1", upstream: "fastc
gi://unix:/run/php/php7.0-fpm.sock:", host: "192.168.206.131"
2021/09/11 20:09:47 [crit] 92241#0: *1 connect() to unix:/run/php/php7.0-fpm.sock failed (13: Permission denied)
while connecting to upstream, client: 192.168.206.1, server: 0.0.0.0, request: "GET / HTTP/1.1", upstream: "fastc
gi://unix:/run/php/php7.0-fpm.sock:", host: "192.168.206.131"
2021/09/11 20:10:21 [crit] 92241#0: *1 connect() to unix:/run/php/php7.0-fpm.sock failed (13: Permission denied)
while connecting to upstream, client: 192.168.206.1, server: 0.0.0.0, request: "GET / HTTP/1.1", upstream: "fastc
gi://unix:/run/php/php7.0-fpm.sock:", host: "192.168.206.131"
```

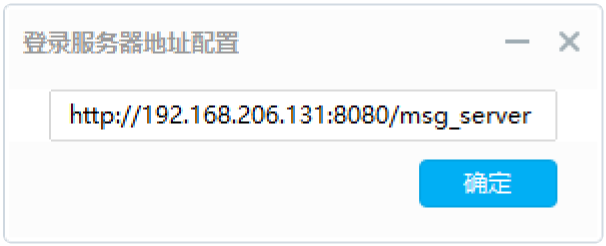
从这里可以看出来php权限有问题，根据3.1.4的方法进行修改，**而且要注意是否重启了php**。

即时通讯日志

客户端提示**连接服务器失败**

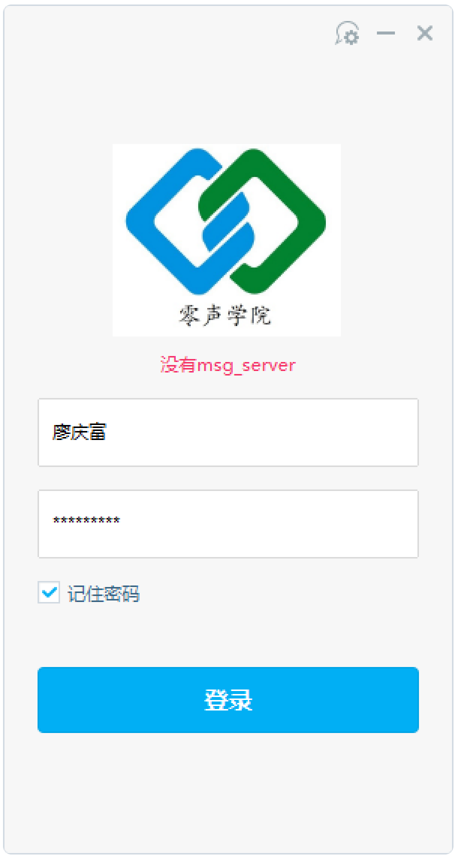


说明

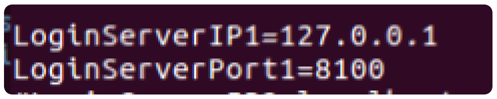


设置有误，或者login_server没有启动。

客户端提示**没有msg_server**

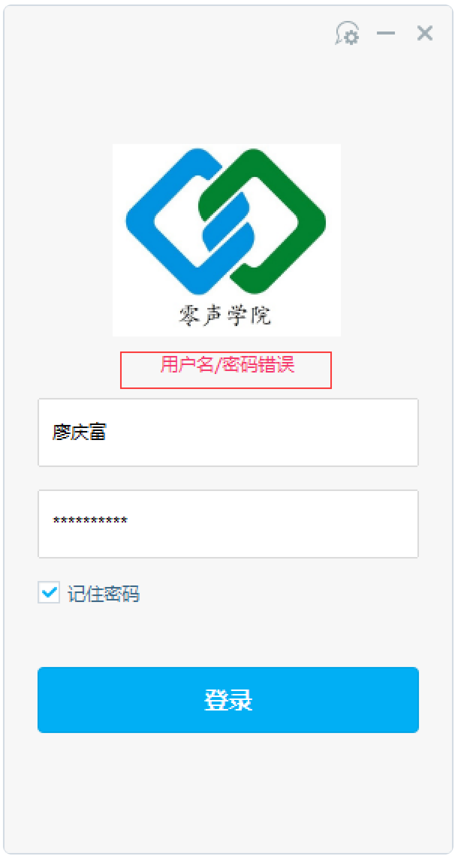


检测msg_server是否启动。
如果已经启动则检测msg_server是否正确连接了login_server，检测配置文件

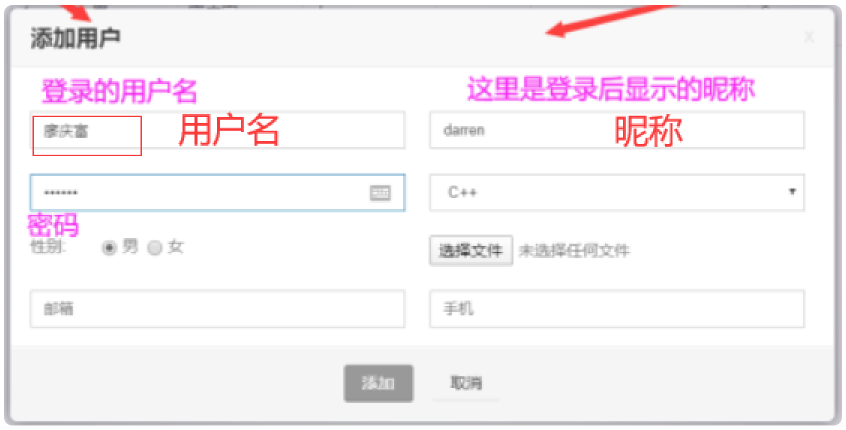


```
LoginServerIP1=127.0.0.1
LoginServerPort1=8100
```

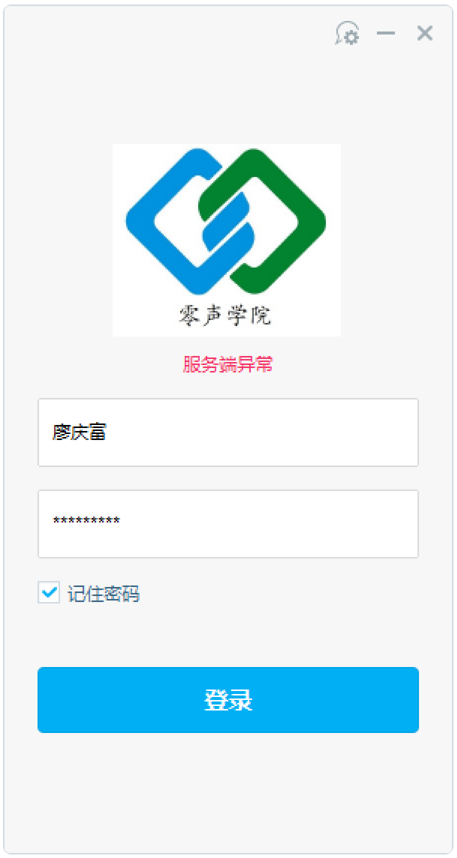
客户端提示用户名/密码错误



用户名和密码不匹配，用户名是注册时的用户名，不是昵称。

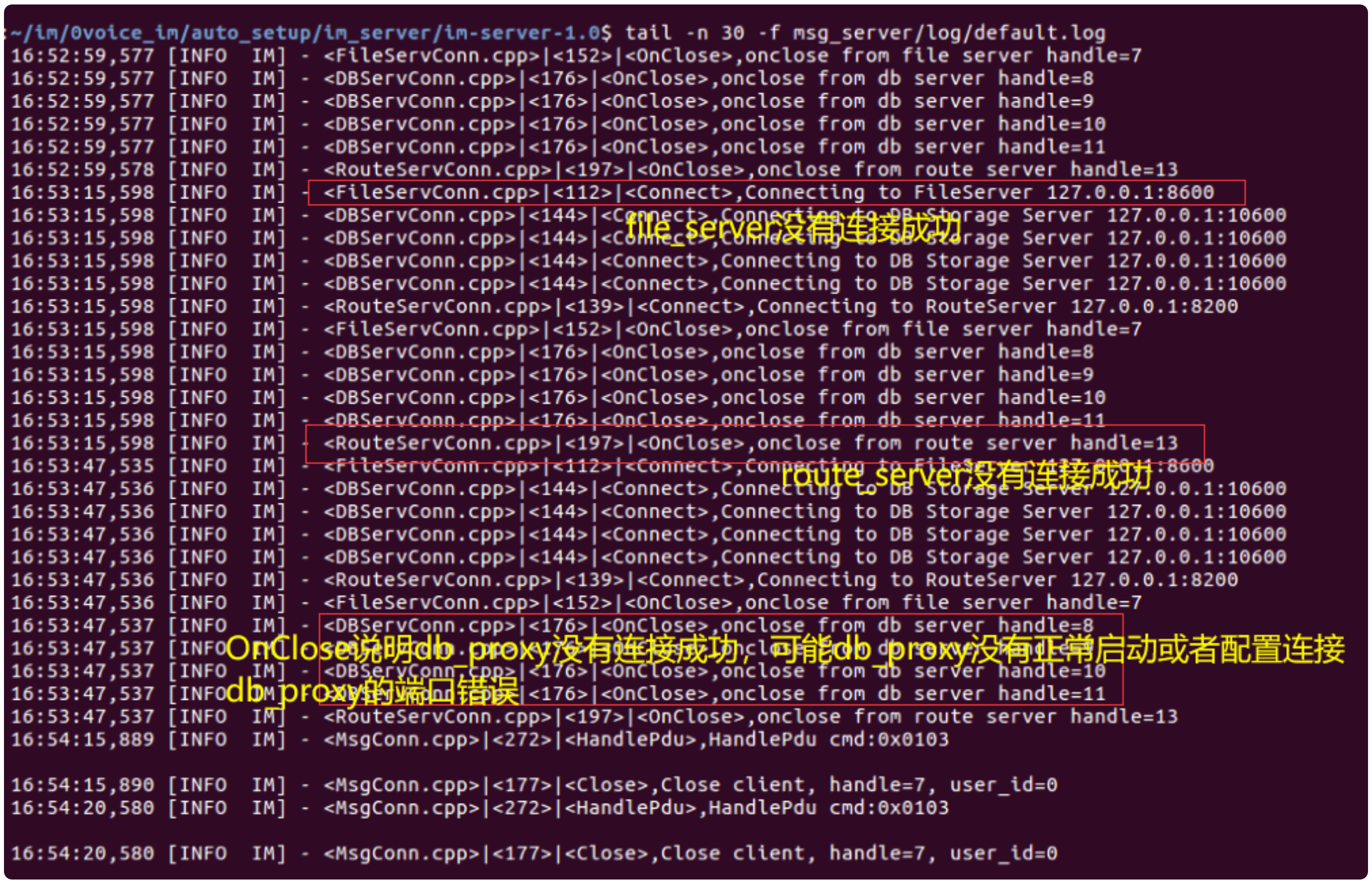


客户端提示服务端异常



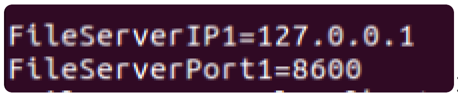
查看msg_server日志

```
lqf@ubuntu:~/im/0voice_im/auto_setup/im_server/im-server-1.0$ tail -n 30 -f msg_server/log/default.log
```



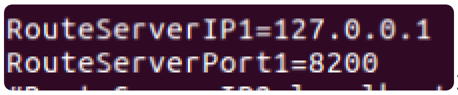
```
:~/im/0voice_im/auto_setup/im_server/im-server-1.0$ tail -n 30 -f msg_server/log/default.log
16:52:59,577 [INFO  IM] - <FileServConn.cpp>|<152>|<OnClose>,onclose from file server handle=7
16:52:59,577 [INFO  IM] - <DBServConn.cpp>|<176>|<OnClose>,onclose from db server handle=8
16:52:59,577 [INFO  IM] - <DBServConn.cpp>|<176>|<OnClose>,onclose from db server handle=9
16:52:59,577 [INFO  IM] - <DBServConn.cpp>|<176>|<OnClose>,onclose from db server handle=10
16:52:59,577 [INFO  IM] - <DBServConn.cpp>|<176>|<OnClose>,onclose from db server handle=11
16:52:59,578 [INFO  IM] - <RouteServConn.cpp>|<197>|<OnClose>,onclose from route server handle=13
16:53:15,598 [INFO  IM] - <FileServConn.cpp>|<112>|<Connect>,Connecting to FileServer 127.0.0.1:8600
16:53:15,598 [INFO  IM] - <DBServConn.cpp>|<144>|<Connect>,Connecting to DB Storage Server 127.0.0.1:10600
16:53:15,598 [INFO  IM] - <DBServConn.cpp>|<144>|<Connect>,Connecting to DB Storage Server 127.0.0.1:10600
16:53:15,598 [INFO  IM] - <DBServConn.cpp>|<144>|<Connect>,Connecting to DB Storage Server 127.0.0.1:10600
16:53:15,598 [INFO  IM] - <DBServConn.cpp>|<144>|<Connect>,Connecting to DB Storage Server 127.0.0.1:10600
16:53:15,598 [INFO  IM] - <RouteServConn.cpp>|<139>|<Connect>,Connecting to RouteServer 127.0.0.1:8200
16:53:15,598 [INFO  IM] - <FileServConn.cpp>|<152>|<OnClose>,onclose from file server handle=7
16:53:15,598 [INFO  IM] - <DBServConn.cpp>|<176>|<OnClose>,onclose from db server handle=8
16:53:15,598 [INFO  IM] - <DBServConn.cpp>|<176>|<OnClose>,onclose from db server handle=9
16:53:15,598 [INFO  IM] - <DBServConn.cpp>|<176>|<OnClose>,onclose from db server handle=10
16:53:15,598 [INFO  IM] - <DBServConn.cpp>|<176>|<OnClose>,onclose from db server handle=11
16:53:15,598 [INFO  IM] - <RouteServConn.cpp>|<197>|<OnClose>,onclose from route server handle=13
16:53:47,535 [INFO  IM] - <FileServConn.cpp>|<112>|<Connect>,Connecting to FileServer 127.0.0.1:8600
16:53:47,536 [INFO  IM] - <DBServConn.cpp>|<144>|<Connect>,Connecting to DB Storage Server 127.0.0.1:10600
16:53:47,536 [INFO  IM] - <DBServConn.cpp>|<144>|<Connect>,Connecting to DB Storage Server 127.0.0.1:10600
16:53:47,536 [INFO  IM] - <DBServConn.cpp>|<144>|<Connect>,Connecting to DB Storage Server 127.0.0.1:10600
16:53:47,536 [INFO  IM] - <DBServConn.cpp>|<144>|<Connect>,Connecting to DB Storage Server 127.0.0.1:10600
16:53:47,536 [INFO  IM] - <RouteServConn.cpp>|<139>|<Connect>,Connecting to RouteServer 127.0.0.1:8200
16:53:47,536 [INFO  IM] - <FileServConn.cpp>|<152>|<OnClose>,onclose from file server handle=7
16:53:47,537 [INFO  IM] - <DBServConn.cpp>|<176>|<OnClose>,onclose from db server handle=8
16:53:47,537 [INFO  IM] - <DBServConn.cpp>|<176>|<OnClose>,onclose from db server handle=9
16:53:47,537 [INFO  IM] - <DBServConn.cpp>|<176>|<OnClose>,onclose from db server handle=10
16:53:47,537 [INFO  IM] - <DBServConn.cpp>|<176>|<OnClose>,onclose from db server handle=11
16:53:47,537 [INFO  IM] - <RouteServConn.cpp>|<197>|<OnClose>,onclose from route server handle=13
16:54:15,889 [INFO  IM] - <MsgConn.cpp>|<272>|<HandlePdu>,HandlePdu cmd:0x0103

16:54:15,890 [INFO  IM] - <MsgConn.cpp>|<177>|<Close>,Close client, handle=7, user_id=0
16:54:20,580 [INFO  IM] - <MsgConn.cpp>|<272>|<HandlePdu>,HandlePdu cmd:0x0103

16:54:20,580 [INFO  IM] - <MsgConn.cpp>|<177>|<Close>,Close client, handle=7, user_id=0
```

file_server没有连接成功
route_server没有连接成功
OnClose说明db_proxy没有连接成功，可能db_proxy没有正常启动或者配置连接db_proxy的端口错误

1. <FileServConn.cpp>|<152>|<OnClose>,onclose from file server handle=7， 说明 file server可能没有启动成功，或者msg_server没有正确配置连接 file server的ip+port。
   a. sudo ps –ef | grep file_server 检测file server是否已经启动
   b. 

      ```
      FileServerIP1=127.0.0.1
      FileServerPort1=8600
      ```

      地址是否配置正确

2. <RouteServConn.cpp>|<197>|<OnClose>,onclose from route server handle=13， 说明 route_server可能没有启动成功，或者msg_server没有正确配置连接route_server的ip+port。
   a. sudo ps –ef | grep route_server 检测route_server是否已经启动
   b. 

      ```
      RouteServerIP1=127.0.0.1
      RouteServerPort1=8200
      ```

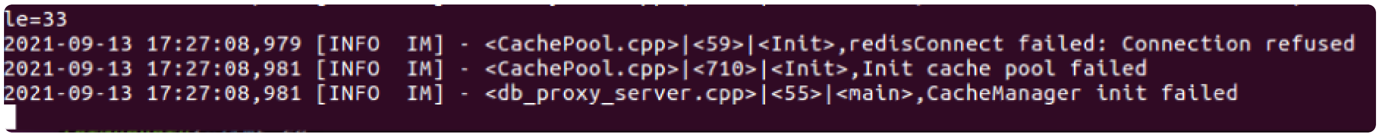
      地址是否配置正确

服务器db_proxy启动后客户端还是不能正常登录

需要查看日志，确保db_proxy是正常启动的。

~/im/0voice_im/auto_setup/im_server/im-server-1.0$ tail -n 30 -f db_proxy_server/log/default.log

redis server连接问题



```
le=33
2021-09-13 17:27:08,979 [INFO  IM] - <CachePool.cpp>|<59>|<Init>,redisConnect failed: Connection refused
2021-09-13 17:27:08,981 [INFO  IM] - <CachePool.cpp>|<710>|<Init>,Init cache pool failed
2021-09-13 17:27:08,981 [INFO  IM] - <db_proxy_server.cpp>|<55>|<main>,CacheManager init failed
```

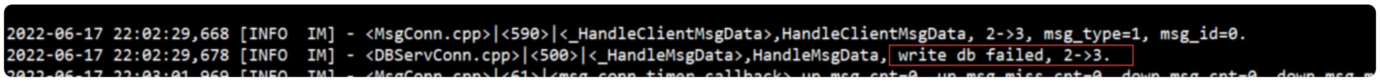
redisConnect failed: Connection refused 连接redis server失败。

1. 查看redis-server是否已经启动；
   sudo ps -ef | grep redis-server
2. 使用redis-cli连接redis-server测试是否正常工作；
3. 查看conf文件的redis配置是否正确；

发消息失败

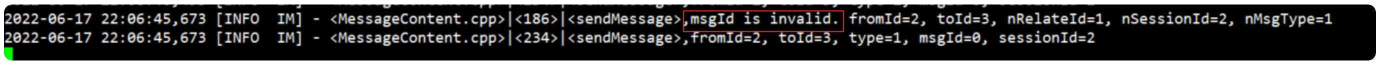
先排查msg_server, 再排查

ubuntu@im_server/im-server-1.0$ **tail -f msg_server/log/default.log**



```
2022-06-17 22:02:29,668 [INFO  IM] - <MsgConn.cpp>|<590>|<_HandleClientMsgData>,HandleClientMsgData, 2->3, msg_type=1, msg_id=0.
2022-06-17 22:02:29,678 [INFO  IM] - <DBServConn.cpp>|<500>|<_HandleMsgData>,HandleMsgData, write db failed, 2->3.
```

ubuntu@im_server/im-server-1.0$ **tail -f db_proxy_server/log/default.log**



```
2022-06-17 22:06:45,673 [INFO  IM] - <MessageContent.cpp>|<186>|<sendMessage>,msgId is invalid. fromId=2, toId=3, nRelateId=1, nSessionId=2, nMsgType=1
2022-06-17 22:06:45,673 [INFO  IM] - <MessageContent.cpp>|<234>|<sendMessage>,fromId=2, toId=3, type=1, msgId=0, sessionId=2
```

msgId is invalid， 消息id是redis生成，需要排查redis。

先登录redis，确定redis是否可以读写操作。

```bash
redis-cli
127.0.0.1:6379> set name darren
(error) MISCONF Redis is configured to save RDB snapshots, but it is currently not able to persist on disk. Commands that may modify the data set are disabled, because this instance is configured to report errors during writes if RDB snapshotting fails (stop-writes-on-bgsave-error option). Please check the Redis logs for details about the RDB error.

```

写入字段报错
有两种修改方法，一种是通过redis命令行修改，另一种是直接修改redis.conf配置文件

1. 命令行方式
命令行修改方式示例：
127.0.0.1:6379> config set stop-writes-on-bgsave-error no

2. 修改配置文件
修改redis.conf文件：vi打开redis-server配置的redis.conf文件，然后使用快捷匹配模式：/stop-writes-on-bgsave-error定位到stop-writes-on-bgsave-error字符串所在位置，接着把后面的yes设置为no即可。

查看服务器的日志

```
~/im/0voice_im/auto_setup/im_server/im-server-1.0$ tail -n 30 -f file_server/log/default.log
~/im/0voice_im/auto_setup/im_server/im-server-1.0$ tail -n 30 -f msfs/log/default.log
~/im/0voice_im/auto_setup/im_server/im-server-1.0$ tail -n 30 -f
route_server/log/default.log
~/im/0voice_im/auto_setup/im_server/im-server-1.0$ tail -n 30 -f
db_proxy_server/log/default.log
~/im/0voice_im/auto_setup/im_server/im-server-1.0$ tail -n 30 -f
http_msg_server/log/default.log
~/im/0voice_im/auto_setup/im_server/im-server-1.0$ tail -n 30 -f login_server/log/default.log
~/im/0voice_im/auto_setup/im_server/im-server-1.0$ tail -n 30 -f msg_server/log/default.log
```

```
du -h --max-depth=1
```