

# 服务器部署

- 服务器部署
  - 1 Jdk 配置
    - 1.1 下载 Jdk 1.8 版本压缩包解压
    - 1.2 编辑配置文件配置环境变量
    - 1.3 测试
  - 2 SELinux 安全策略
    - 2.1 关闭 SELinux 安全策略
  - 3 配置 Redis(离线)
    - 3.1 下载 redis 压缩包(无版本要求)解压
    - 3.2 编译安装
    - 3.3 测试运行
    - 3.4 设置开启自启动服务(systemd 管理)
      - 3.4.1 配置 redis.conf 文件
      - 3.4.2 创建并编辑 redis.service 文件
      - 3.4.3 redis 服务相关指令
  - 4 配置主服务自启动
    - 4.1 配置 nacos 服务自启动
      - 4.1.1 nacos 服务相关指令
    - 4.2 配置 wms 服务自启动(只配置一个其他配置相同)
      - 4.2.1 wms 服务相关指令
    - 4.3 配置 tomcat 服务自启动
      - 4.3.1 下载 tomcat 压缩包(无版本要求)解压
      - 4.3.2 编辑 startup.sh 启动脚本文件
      - 4.3.3 创建并编辑 tomcat.service 文件
      - 4.3.4 tomcat 服务相关指令
      - 4.3.5 访问前端页面
      - 4.3.6 端口占用问题
  - 5 开放防火墙端口
    - 5.1 相关指令
    - 5.2 开启服务器所需端口(暂定)
  - 6 数据库部署与备份
    - 6.1 安装数据库
    - 6.2 修改配置文件
    - 6.3 配置自启动服务

- 6.4 postgresql 服务相关指令
- 6.5 数据库定时备份(本地及其他服务器)
  - 6.5.1 SSH 密钥认证
  - 6.5.2 编辑 shell 脚本
  - 6.5.3 配置定时任务
- 7 大屏及报表(repoter)服务
  - 7.1 报表服务
  - 7.2 大屏服务
    - 7.2.1 IP摄像头显示
      - 7.2.1.1 配置Docker服务(离线)
      - 7.2.1.2 配置WebRTC-Streaer服务
- 8 配置网络
  - 8.1 配置电脑同时使用内外网(bash指令)
  - 8.2 配置虚拟机网络(WIFI->外网,网线->内网)
- 9 软件及配置文件

## 1 Jdk 配置

### 1.1 下载 Jdk 1.8 版本压缩包解压

```
# 进入对应目录 /path/为对应路径
cd /path/
# 解压
tar -zxvf xxx.tar.gz
```

### 1.2 编辑配置文件配置环境变量

```
vi /etc/profile
# 按下i进入编辑模式
# 添加如下语句配置Jdk环境变量; /path/为JDK安装位置
export JAVA_HOME=/path/
export PATH=$PATH:$JAVA_HOME/bin
# 按下esc 输入:wq!保存并强制退出
# 使配置文件生效
source /etc/profile
```

### 1.3 测试

```
# 输出Jdk版本
java -version
# 输出JAVA_HOME路径
echo $JAVA_HOME
```

## 2 SELinux 安全策略

### 2.1 关闭 SELinux 安全策略

```
# 查看SELinux状态
sestatus
# 临时关闭
setenforce 0
# 永久关闭 编辑/etc/selinux/config文件
vim /etc/selinux/config
# 按下i进入编辑模式 修改SELINUX=enforcing为SELINUX=disabled
SELINUX=disabled
# 按下esc,输入:wq退出并保存
# 重启系统使设置生效
reboot
```

## 3 配置 Redis(离线)

### 3.1 下载 redis 压缩包(无版本要求)解压

```
# 进入压缩包
cd /path/ /path/为对应路径
# 解压
tar -zxvf xxx.tar.gz
```

### 3.2 编译安装

```
# 进入解压后的目录 /path/为对应路径
cd /path/
# 编译
make
# 安装到指定目录 /path/为对应路径
make install PREFIX=/path/
```

### 3.3 测试运行

```
# 进入redis执行文件所在目录 /path/为redis安装目录
cd /path/bin/
# 运行 未使用systemd管理时
./redis-server
# 停止 未使用systemd管理时
./redis-cli shutdown
```

### 3.4 设置开启自启动服务(systemd 管理)

#### 3.4.1 配置 redis.conf 文件

```
# 编辑redis.conf文件,该文件通常位于redis目录下
vim /path/redis.conf
# 按下i进入编辑模式 找到NETWORK网络设置 修改bind 127.0.0.1 -:::1
# 添加本地地址和网络地址,以便远程访问使用
bind 127.0.0.1 ip(网络地址)
# 将守护模式改为no
protected-mode no
# 找到GENERAL设置,将守护进程改为yes 便于服务后台启动
daemonize yes
# 若redis无法启动可将配置文件中的系统日志打开
syslog-enabled yes logfile "" 此处设置日志生成位置
```

#### 3.4.2 创建并编辑 redis.service 文件

```
# 在/etc/systemd/system/目录下创建redis.service文件
vim /etc/systemd/system/redis.service
# 按下i进入编辑模式 输入以下内容
[Unit]
Description=Redis Service
After=network-online.target
[Service]
Type=forking
ExecStart=/path/bin/redis-server /path/redis.conf
ExecStop=/path/bin/redis-cli shutdown
Restart=always
[Install]
WantedBy=multi-user.target
# 此处Type需要为forking类型
# Description为描述信息
# After表示启动该服务之前必须启动某服务,此处如果redis.conf文件bind配置了网络ip,则需为
network-online.target,否则无法自启动
# ExecStart是服务启动的执行命令 /path/为redis的安装目录
# ExecStop是服务停止的执行命令
# WantedBy指定了服务所需的目标
# Restart为服务重启方式
# 按下esc键,输入:wq保存并退出
```

### 3.4.3 redis 服务相关指令

```
# 重新加载systemd配置
systemctl daemon-reload
# 启动/关闭服务为自启动
systemctl enable/disable redis
# 启动停止服务
systemctl start/stop redis
# 若未关闭SELinux,此时可能因为SELinux策略问题导致启动失败或自启动服务失败
# 查看服务状态
systemctl status redis
# 查看服务系统日志
journalctl -u redis
# 查看系统日志
journalctl -xe
# 查看端口占用情况
netstat -tulnp | grep port
```

## 4 配置主服务自启动

### 4.1 配置 nacos 服务自启动

```
# 在/etc/systemd/system/系统服务目录下创建nacos.service文件
vim /etc/systemd/system/nacos.service
# 按下i进入编辑模式 输入以下内容
[Unit]
Description=Nacos Service
After=network.target
[Service]
Type=simple
ExecStart=/path/bin/java -jar /Jarpath/xxx.jar
Restart=always
[Install]
WantedBy=multi-user.target
# /path/为JDK安装位置 /Jarpath/为jar包位置
# 按下esc,输入:wq保存并退出
# 注:此处不直接使用java命令是因为不使用完整路径可能会报无法找到java命令,别处同理
```

#### 4.1.1 nacos 服务相关指令

```
# 重新加载systemd配置
systemctl daemon-reload
# 启动/关闭服务为自启动
systemctl enable/disable nacos
# 启动停止服务
systemctl start/stop nacos
# 若未关闭SELinux安全策略可能会导致启动失败
# 查看服务状态
systemctl status nacos
# 查看服务系统日志
journalctl -u nacos
# 查看系统日志
journalctl -xe
```

#### 4.2 配置 wms 服务自启动(只配置一个其他配置相同)

```
# 在/etc/systemd/system/系统服务目录下创建xxx.service文件
vim /etc/systemd/system/wmserver.service
# 按下i进入编辑模式 输入以下内容
[Unit]
Description=xxx Service
After=network.target nacos.service redis.service
Requires=nacos.service # nacos服务启动失败则不启动该服务
[Service]
Type=simple
ExecStartPre=/bin/bash -c 'until nc -z -w5 ip port; do sleep 1; done'
ExecStart=/path/bin/java -jar /JARpath/xxx.jar
Restart=always
RestartSec=5 # 重启间隔为 5 秒
StartLimitIntervalSec=60 # 重启间隔时间为 60 秒
StartLimitBurst=3 # 在 60 秒内最多重启 3 次
[Install]
WantedBy=multi-user.target
# After表示nacos和redis服务启动之后启动
# Requires表示如果nacos服务启动失败则不启动
# ExecStartPre检查是否可以连接数据库ip port 数据库的ip和端口
# /path/为JDK安装位置 /Jarpath/为jar包位置
# 按下esc,输入:wq保存并退出
```

#### 4.2.1 wms 服务相关指令

```
# 重新加载systemd配置
systemctl daemon-reload
# 启动/关闭服务为自启动
systemctl enable/disable wmserver
# 启动停止服务
systemctl start/stop wmserver
# 若未关闭SELinux安全策略可能会导致启动失败
# 查看服务状态
systemctl status wmserver
# 查看服务系统日志
journalctl -u wmserver
# 查看系统日志
journalctl -xe
```

### 4.3 配置 tomcat 服务自启动

#### 4.3.1 下载 tomcat 压缩包(无版本要求)解压

```
# 进入tomcat压缩包目录解压 /path/为tomcat压缩包所在目录
cd /path/
# 解压
tar xxx.tar.gz
```

#### 4.3.2 编辑 startup.sh 启动脚本文件

```
# 进入该文件所在目录 /path/为tomcat解压后目录
cd /path/bin/
# 编辑startup.sh文件,在文件适当位置,设置JAVA_HOME,不然可能报识别不出java命令
vim startup.sh
# 按下i进入编辑模式,设置JAVA_HOME /path/为Jdk路径
export JAVA_HOME=/path/
# 按下esc,输入:wq保存并退出
```

#### 4.3.3 创建并编辑 tomcat.service 文件

```
# 在/etc/systemd/system/系统服务目录下创建tomcat.service并编辑
vim /etc/systemd/system/tomcat.service
# 按下i进入编辑模式,输入以下内容
[Unit]
Description=Tomcat Service
After=network.target
[Service]
Type=forking
ExecStart=/path/bin/startup.sh & # /path/为tomcat安装目录路径 此处&表示后台运行
ExecStop=/path/bin/shutdown.sh
Restart=on-failure
RestartSec=10
[Install]
WantedBy=multi-user.target
# 按下esc,输入:wq保存并退出
```

#### 4.3.4 tomcat 服务相关指令



```
# 重新加载systemd配置
systemctl daemon-reload
# 启动/关闭服务为自启动
systemctl enable/disable tomcat
# 启动停止服务
systemctl start/stop tomcat
# 若未关闭SELinux安全策略可能会导致启动失败
# 查看服务状态
systemctl status tomcat
# 查看服务系统日志
journalctl -u tomcat
# 查看系统日志
journalctl -xe
```

#### 4.3.5 访问前端页面

将打包好的前端package放在/tomcat/webapps/目录下 在网页输入ip:port/你的package在webapps下的相对路径

本项目中有以下服务需放在tomcat中运行: 报表服务,前端PC,前端RF以及大屏led

#### 4.3.6 端口占用问题

```
# 查看端口占用
lsof -i :port # 该语句可以显示进程PID
netstat -tuln | grep :port # 该语句可以显示是否有监听的进程
# 终止进程
kill PID
kill -9 PID # 强制
```

## 5 开放防火墙端口

### 5.1 相关指令

```
# 关闭开启防火墙
systemctl stop/start firewalld
# 开放和关闭指定端口 可以采用xx-xx一次性开放多个连续端口
sudo firewall-cmd --permanent --add-port=your_port/tcp # 开放
sudo firewall-cmd --permanent --remove-port=your_port/tcp # 关闭
# 重载防火墙规则
sudo firewall-cmd --reload
# 查看防护墙状态
sudo systemctl status firewalld
# 列出所有TCP和UDP监听端口
firewall-cmd --list-all
```

5.2 开启服务器所需端口(暂定)

下述表格中的端口为本系统需开放的TCP端口

nacos 服务	wcs 服务	tomcat	redis	wms 服务	大屏
8849 9848 9849	38089 8888	8080	6379	18010 18020 18030 18030 18040 18050 18060	8050 8000

除此之外,大屏服务(WebRTC-Streamer服务)需开放针对IP摄像头的UDP端口,目前开放端口为:50000-20024

```
# 开启端口
sudo firewall-cmd --permanent --add/remove-port=your_port/tcp
# 重载防火墙规则
sudo firewall-cmd --reload
```

6 数据库部署与备份

6.1 安装数据库

```
# 下载postgresql压缩包解压 /path/为压缩包所在目录
cd /path/
tar -zxvf xxx.tar.gz
# 进入解压后的目录 编译安装 /postgresql-<version>/为解压后的目录
cd /postgresql-<version>/
./configure # 配置软件以适应当前系统环境
# 此步骤可能出现缺少软件包的情形,需配置镜像源,按照提示下载需要的软件包再执行./configure
make # 编译
make install # 安装 此处可以使用PREFIX=/path/ 指定安装目录
# 初始化数据库 注:这里建议修改配置文件后进行数据库初始化
/your_pgsql_path/bin/initdb -D /your_pgsql_path/data
# 启动数据库
/your_pgsql_path/bin/pg_ctl -D /your_pgsql_path/data start
# 停止数据库
/your_pgsql_path/bin/pg_ctl -D /your_pgsql_path/data stop
# 创建恢复数据库
# 建议使用数据库连接工具例如Navicat, Dategrip, Dbeaver等
```

6.2 修改配置文件

```
# 修改配置文件pg_hba.conf和postgresql.conf
# 在pg_hba.conf文件中找到如下行进行修改, 以下是示例
# TYPE      DATABASE      USER      ADDRESS      METHOD
# host      all            all       127.0.0.1/32  md5
# host:表示使用主机地址进行连接认证。
# all:表示适用于所有数据库和所有用户。
# 127.0.0.1/32:表示允许来自本地主机（即 127.0.0.1）的连接。
# md5:表示使用密码认证进行连接。
```

```
# TYPE DATABASE      USER      ADDRESS      METHOD

# "local" is for Unix domain socket connections only
local all            all         trust
# IPv4 local connections:
host  all            all        0.0.0.0/0     scram-sha-256
host  all            all        127.0.0.1/32  trust
# IPv6 local connections:
host  all            all        ::1/128       trust
# Allow replication connections from localhost, by a user with the
# replication privilege.
local replication all         trust
host  replication all        127.0.0.1/32  trust
host  replication all        ::1/128       trust
```

```
# 在postgresql.conf中找到并编辑listen_addresses项, 将其设置为 '*', 表示允许所有ip地址连接
listen_addresses = '*'
# 找到下面的port=5432和max_connections=xx 去掉注释符'#', 最大连接数按照实际修改
```

### # - Connection Settings -

```
listen_addresses = '*'      # what IP address(es) to listen on;
                             # comma-separated list of addresses;
                             # defaults to 'localhost'; use '*' for all
                             # (change requires restart)
port = 5432                 # (change requires restart)
max_connections = 2000      # (change requires restart)
```

## 6.3 配置自启动服务

```
# 在/etc/systemd/system/目录下创建postgresql.service文件并编辑
vim /etc/systemd/system/postgres.service
# 按下i进入编辑模式 输入一下内容
[Unit]
Description=postgresql.service
After=network.target
[Service]
Type=forking
#运行程序的用户和群组
User=postgres # 需提前用户和组需要提前创建
Group=postgres
# 启动命令
ExecStart=/your_pgsql_path/bin/pg_ctl -D /your_pgsql_path/data start
# 重新加载
ExecReload=/your_pgsql_path/bin/pg_ctl -D /your_pgsql_path/data restart
# 停止程序
ExecStop=/your_pgsql_path/bin/pg_ctl -D /your_pgsql_path/data stop
# 是否给服务分配独立的临时空间，需要
PrivateTmp=true
[Install]
WantedBy=multi-user.target
# 按下esc,输入:wq保存并退出
```

## 6.4 postgresql 服务相关指令

```
# 重新加载systemd配置
systemctl daemon-reload
# 启动/关闭服务为自启动
systemctl enable/disable postgresql
# 启动停止服务
systemctl start/stop postgresql
# 若未关闭SELinux,此时可能因为SELinux策略问题导致启动失败或自启动服务失败
# 查看服务状态
systemctl status postgresql
# 查看服务系统日志
journalctl -u postgresql
# 查看系统日志
journalctl -xe
```

## 6.5 数据库定时备份(本地及其他服务器)

### 6.5.1 SSH 密钥认证

```
# SSH 密钥认证 离线模式下通过ssh自动登录远程服务器
ssh-keygen -t rsa
```

```
[root@pgsql pgsql]# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa) db_keys
Enter passphrase (empty for no passphrase):
```

```
# 输入上述命令后会得到如下输出,红框为密钥公钥保存路径,不输入则默认在~/.ssh/目录下,后面一项
为密钥密码
# 在目标主机上根目录下创建./ssh目录和authorized_keys文件
mkdir -p ~/.ssh
cd /root/.ssh/
vim authorized_keys
# 按i进入编辑模式将ssh-keygen -t rsa生成的xxx.pub文件内容复制到authorized_keys文件中
# 按下esc,输入:wq保存退出
# 设置~/.ssh目录的权限为700,并设置authorized_keys文件的权限为 600
chmod 700 ~/.ssh
chmod 600 ~/.ssh/authorized_keys
# 第一次登录需要手动验证,输入以下命令验证 /path/to/db_keys为密钥路径 user@hostname为目标用
户和主机
ssh -i /path/to/db_keys user@hostname
# 注意 若需连接的远程主机发生更改进入/root/.ssh/known_hosts删除旧的密钥
```

## 6.5.2 编辑 shell 脚本

```

# 编辑脚本文件 此处只示例wms数据库脚本wmsnet操作相同
vim xxx.sh # 按i进入编辑模式
# 输入以下内容
#!/bin/bash
# 设置本地wms备份文件的目录
local_backup_wms_dir="/home/postgres/backups/wms"
# 设置本地数据库密码
db_password="timms@1234"
# 设置本地wms数据库用户
db_wms_user="l1276wms"
# 设置本地wms数据库
db_wms_name="l1276wms"
# 设置本机ip
ip="127.0.0.1"
# 设置本机数据库端口
port="5432"
#设置本机密钥位置
local_key="/home/postgres/db_key"
# 设置本地pg_dump可执行文件位置
local_pgdump="/home/postgres/postgresql-14.6/bin/pg_dump"
# 设置远程备份文件wms的目录
remote_backup_wms_dir="/home/l1276wms/wms/pgsql/wms_backups/"
# 设置远程主机用户和主机
remote_user="root@172.16.10.3"
# 设置远程服务器wms目录地址
remote_user_wms_dir="root@172.16.10.3:/home/l1276wms/wms/pgsql/wms_backups/"
# 设置保留备份文件的天数
retention_days=7
# 创建本地wms备份文件的名称, 包含当前日期时间
local_backup_file="$local_backup_wms_dir/wmsbackup_$(date +%Y%m%d_%H%M%S).sql"
# 备份 PostgreSQL 数据库到新的本地备份文件 此处不填写pg_dump完整路径会导致crontap执行脚本本失败
PGPASSWORD="$db_password" "$local_pgdump" -U "$db_wms_user" -h "$ip" -p "$port" -d "$db_wms_name" > "$local_backup_file"
# 为本地备份文件设置读写执行权限
chmod +rwx "$local_backup_file"
# 使用公钥密钥方式登录目标主机, 并使用 scp 将备份文件传输到远程服务器
scp -i "$local_key" "$local_backup_file" "$remote_user_wms_dir"
# 删除本地和远程服务器上旧备份文件, 保留一定天数内的备份文件
find "$local_backup_wms_dir" -type f -name "wmsbackup_*" -mtime +$retention_days -exec rm {} \;
ssh -i "$local_key" "$remote_user" "find $remote_backup_dir -type f -name 'wmsbackup_*' -mtime +$retention_days -exec rm {} \;"
# 按esc, 输入:wq保存退出

```

### 6.5.3 配置定时任务

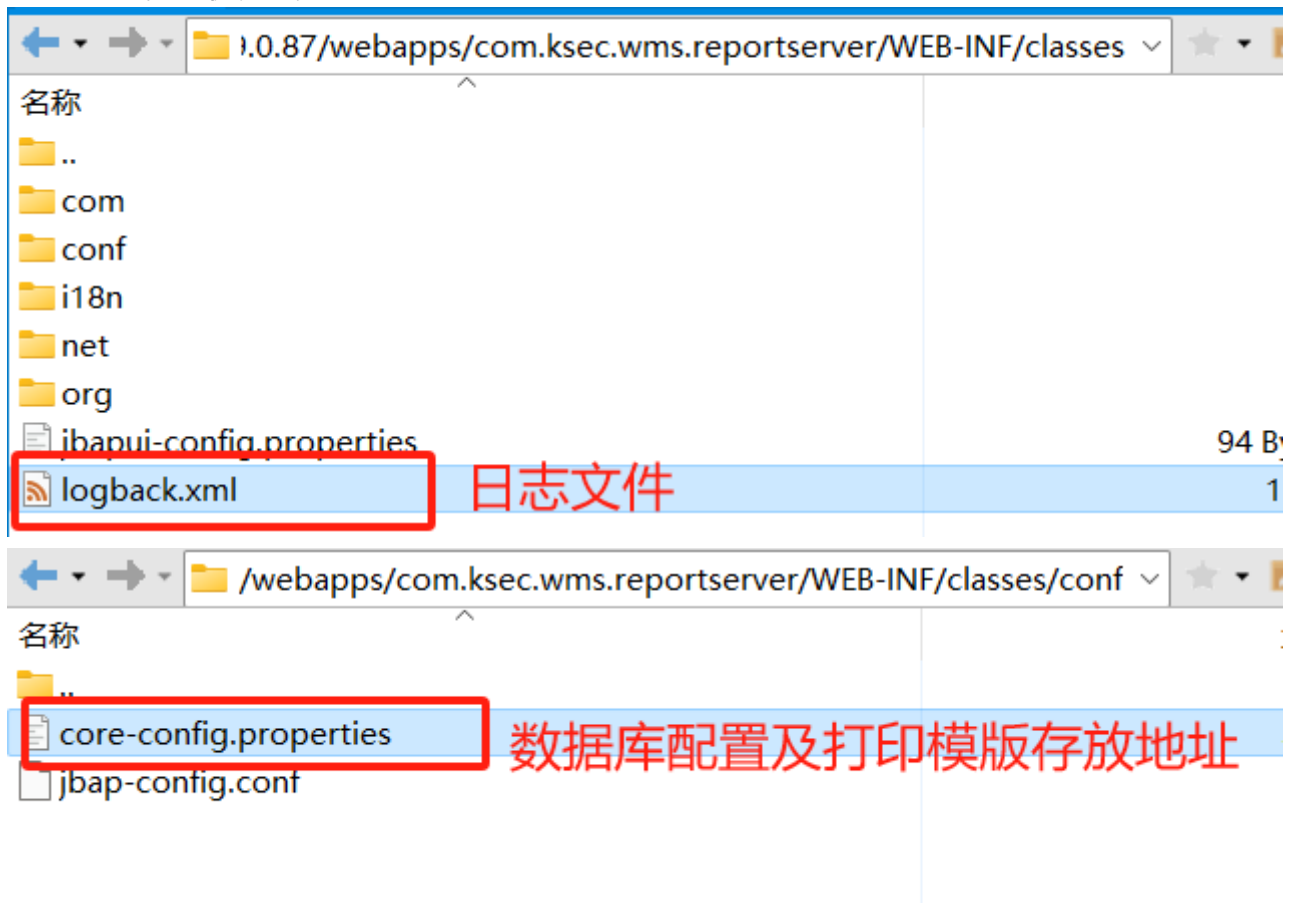
```
# 配置定时任务
crontab -e
# 输入一下内容(示例)
15 10 * * * /your_shell_script
# 15表示分钟 10表示小时,每天10:15备份
# 保存退出
```

## 7 大屏及报表(repoter)服务

### 7.1 报表服务

将报表服务的Web服务放在tomcat目录下的webapps目录下,启动tomcat

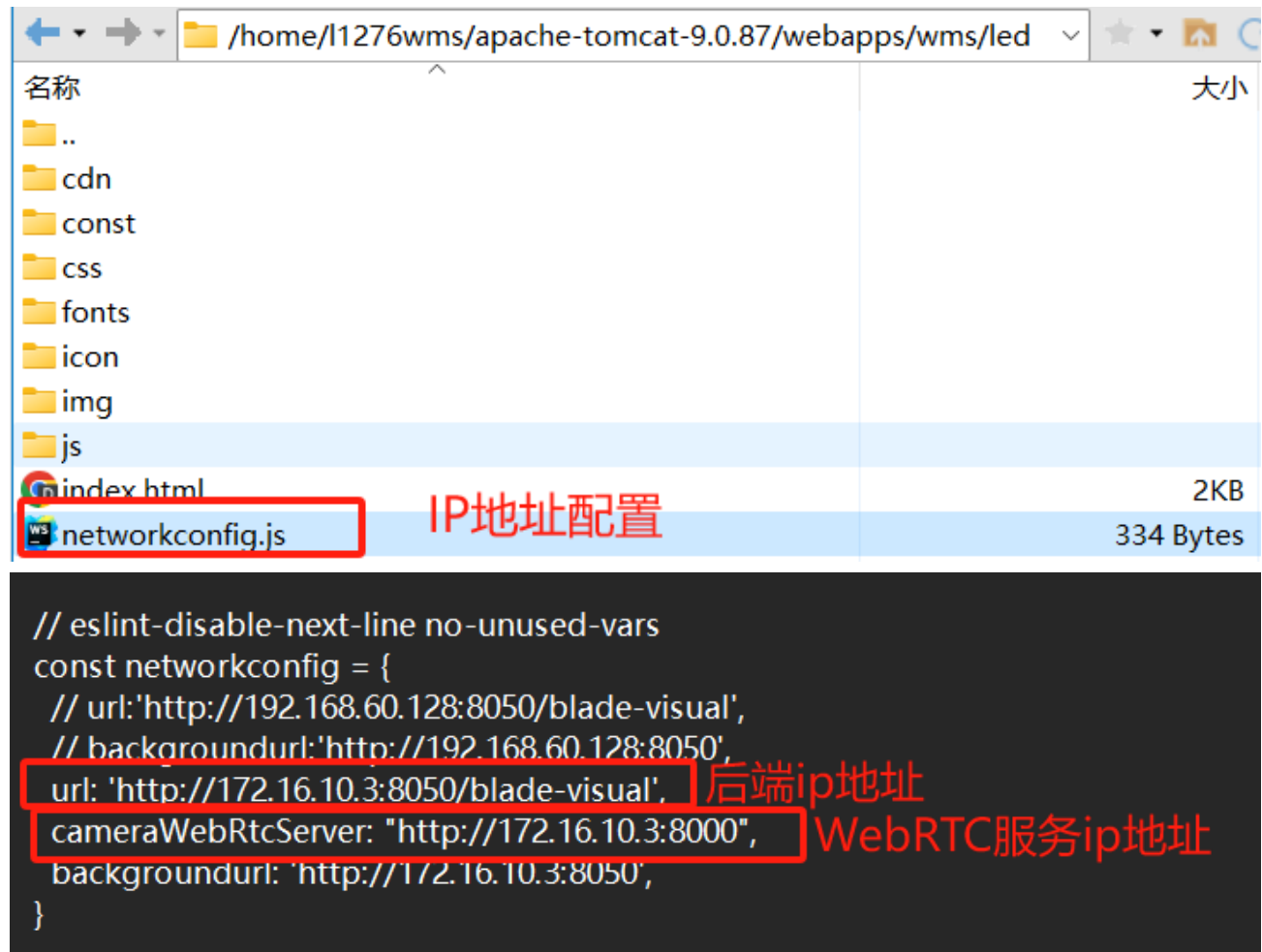
根据需要修改报表服务的配置文件



### 7.2 大屏服务

将大屏服务的Web服务放在tomcat目录下的webapps目录下,启动tomcat

根据需要修改配置文件



名称 大小

- ..
- cdn
- const
- css
- fonts
- icon
- img
- js
- index.html 2KB
- networkconfig.js 334 Bytes**

**IP地址配置**

```
// eslint-disable-next-line no-unused-vars
const networkconfig = {
  // url:'http://192.168.60.128:8050/blade-visual',
  // backgroundurl:'http://192.168.60.128:8050',
  url: 'http://172.16.10.3:8050/blade-visual',
  cameraWebRtcServer: "http://172.16.10.3:8000",
  backgroundurl: 'http://172.16.10.3:8050',
}
```

**后端ip地址**  
**WebRTC服务ip地址**

后端大屏jar包服务自启动配置参考主服务wmserver配置  
需修改jar包中的数据库地址以及日志等配置

### 7.2.1 IP摄像头显示

声明:由于系统问题,此处配置WebRTC-Streamer服务采用Docker容器化技术部署

#### 7.2.1.1 配置Docker服务(离线)



```
# 下载docker压缩包,进入压缩包目录解压
cd xxx
# 解压到/usr/local/bin/目录下,忽略文件路径第一个组件,即直接放在/usr/local/bin/目录下
tar xzf docker-<版本号>.tgz -C /usr/local/bin/ --strip-components=1
# 配置环境变量
vim /etc/profile
# 末尾添加以下代码 保存退出
export PATH=$PATH:/usr/local/bin/docker
# 重新加载系统变量
source /etc/profile
# 测试
docker --version
```

```
# 配置docker服务自启动
vim /etc/systemd/system/docker.service
# 添加如下内容,保存退出
[Unit]
Description=Docker Service
#Documentation=https://docs.docker.com
After=network-online.target
Wants=network-online.target
[Service]
Type=simple
ExecStart=/usr/local/bin/dockerd
ExecReload=/bin/kill -s HUP $MAINPID
TimeoutSec=0
RestartSec=2
Restart=always
[Install]
WantedBy=multi-user.target
```

```
# 重新加载系统服务
systemctl daemon-reload
# 添加到系统开机启动服务
systemctl enable docker.service
# 启动停止服务(建议先重启虚拟机)
systemctl start/stop docker
# 查看服务状态
systemctl status docker
journalctl -u docker
```

### 7.2.1.2 配置WebRTC-Streaer服务

## 方法一

使用docker pull mpromonet/webrtc-streamer拉取服务镜像

- 配置虚拟机上网,参考[配置虚拟机上网](#)
- 设置docker镜像代理(未实践过)
  - 创建/etc/docker/daemon.json文件,格式如下,重启docker  

```
{ "registry-mirrors": ["https://your-registry-mirror"] }
```
- docker pull mpromonet/webrtc-streamer拉取镜像

## 方法二

使用有网的计算机拉取mpromonet/webrtc-streamer镜像,打包重新加载

- 打包
  - docker save -o webrtc-streamer.tar mpromonet/webrtc-streamer
- 加载
  - docker load -i webrtc-streamer.tar

```
# 编辑WebRTC-Streamer服务配置文件
vim /usr/local/bin/conf.json # 地址为docker安装位置便于查看
# 添加数据(示例)
{
  "urls" :
  {
    "85" :
    {
      "video" :
      "rtsp://admin:jxgzfsk@001@172.16.1.85:554/Streaming/Channels/101"
    },
  }
}
```

```
# 创建镜像并加载配置文件及监听端口
docker create -p 8000:8000 --name=webrtc-streamer --net=host -v
/usr/local/bin/config.json:/app/config.json -it mpromonet/webrtc-streamer -C
/app/config.json -R 50000:50024
# 略微解释
-p 8000:8000 -> 将容器的8000端口映射到主机的8000端口
--name=webrtc-streamer -> 指定容器名称
--net=host -> 使用主机的网络命名空间,使得容器可以直接使用主机的网络
-v xxx -> 将conf.json文件挂载在容器内
-it -> 分配一个交互式终端
-C -> 指定容器启动时加载的配置文件
-R -> 设置WebRTC端口范围(UDP)
```

```
# 配置镜像自启动
vim /etc/systemd/system/webrtc.service
# 添加如下内容
[Unit]
Description=WebRtc-Streamer Service
Requires=docker.service
After=docker.service
[Service]
Type=simple
ExecStart=/usr/local/bin/docker start -a webrtc-streamer
ExecStop=/usr/local/bin/docker stop -t 2 webrtc-streamer
TimeoutSec=0
RestartSec=2
Restart=always
[Install]
WantedBy=multi-user.target
```

```
# 重载系统服务
systemctl daemon-reload
# 注册开机自启服务
systemctl enable webrtc
# 启动或停止服务(建议重启虚拟机)
systemctl start/stop webrtc
# 查看服务状态
systemctl status webrtc
journalctl -u webrtc
```

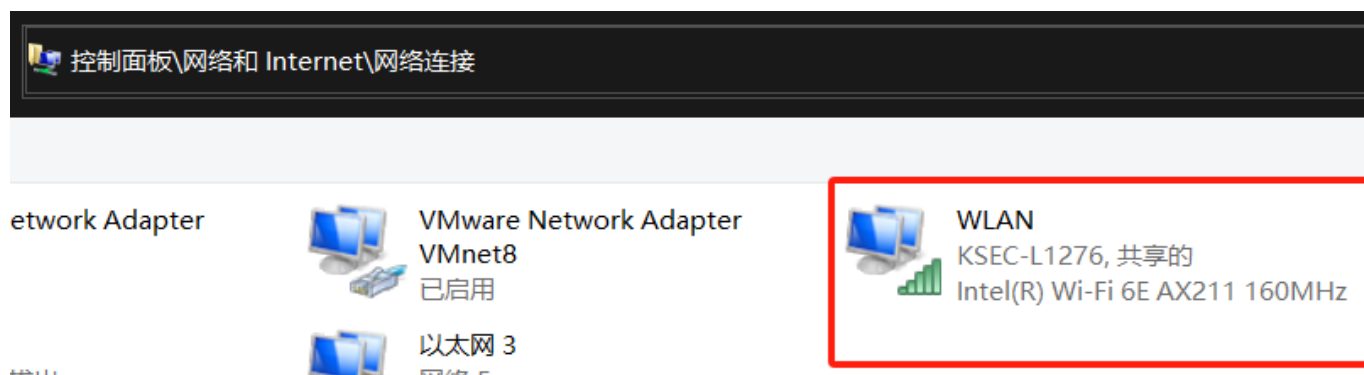
## 8 配置网络

### 8.1 配置电脑同时使用内外网(bash指令)

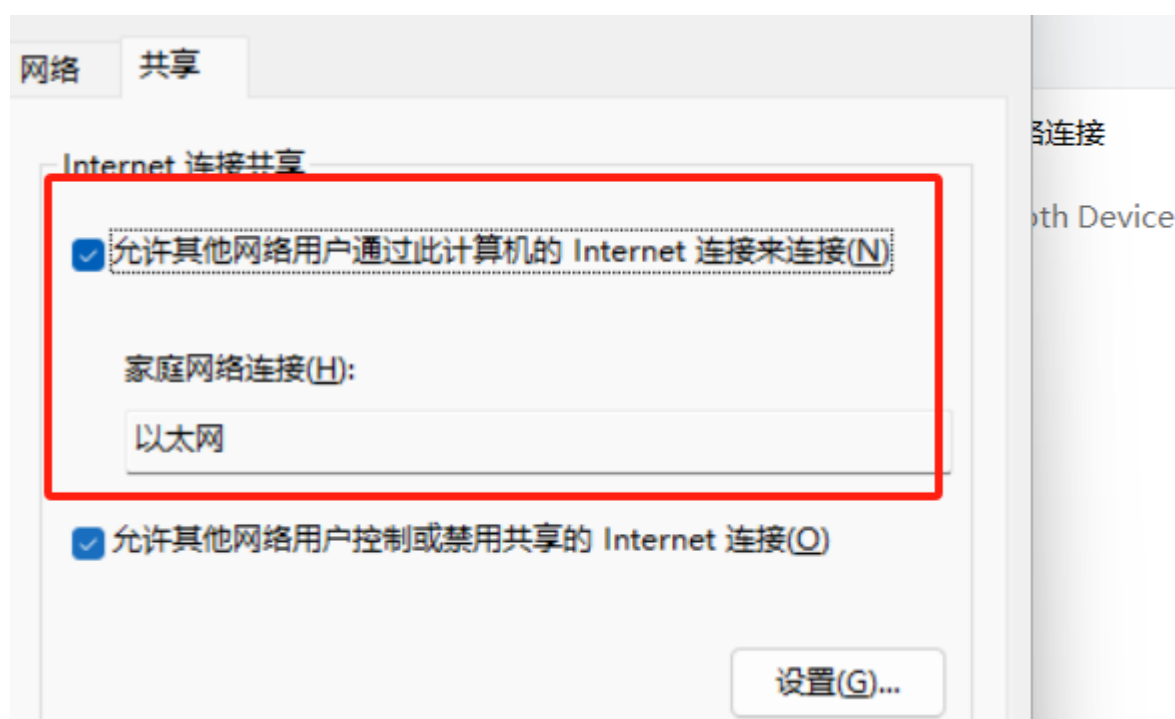
```
# 查看本机ip
ipconfig
# 查看网络配置route print
# 删除之前的网络配置
route delete 0.0.0.0
route delete 172.16.0.0 # 172.16.0.0 为内网网络地址
# 配置将所有流量路由到ip1(默认路由),将172.16.0.0网段的流量路由到ip2
route add -p 0.0..0 mask 0.0.0.0 ip1 # 此处ip为本机外网网关
route add -p 172.16.0.0 mask 255.255.0.0 ip2 # 此处为内网网关
```

### 8.2 配置虚拟机网络(WIFI->外网,网线->内网)

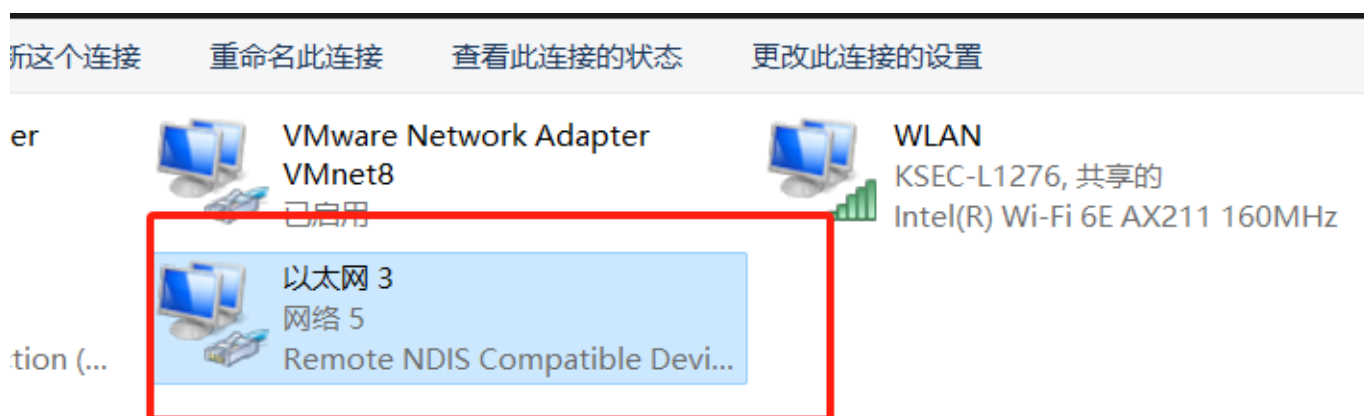
进入到以下位置,右键WLAN,点击属性



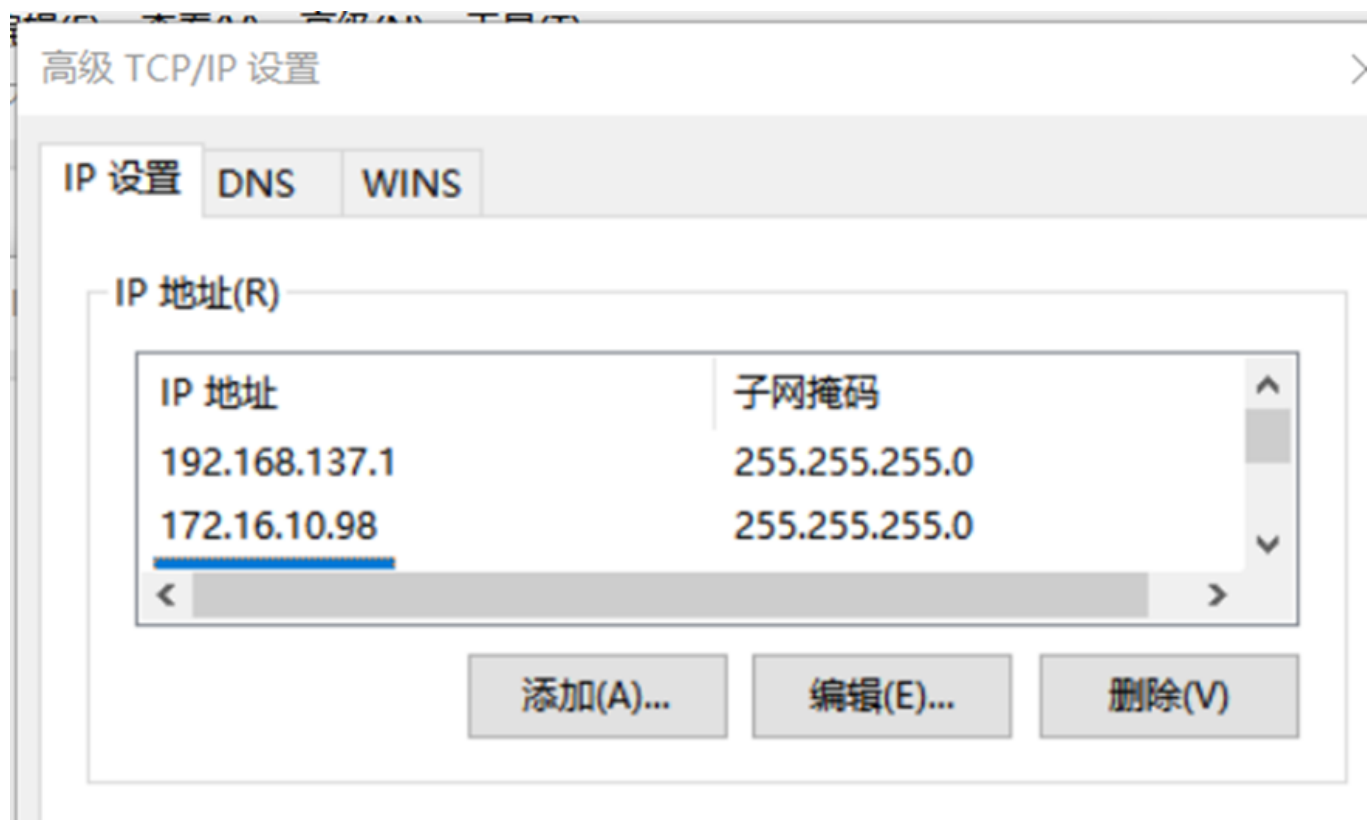
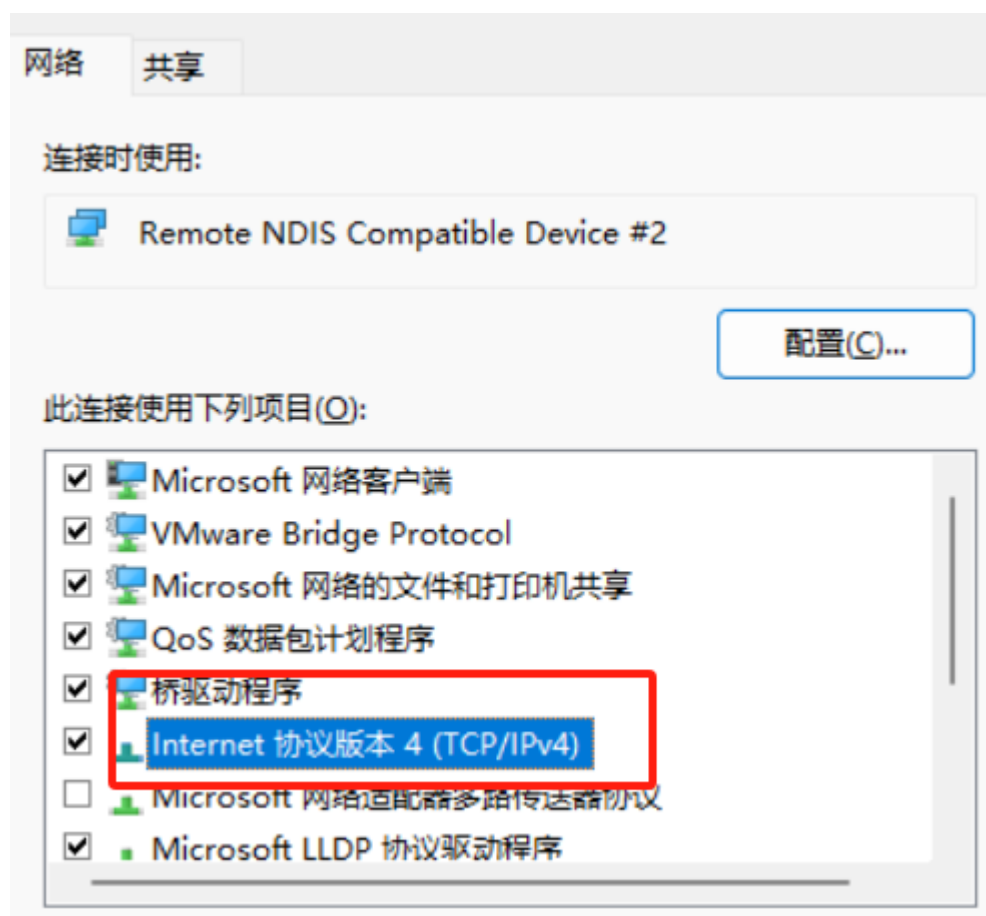
点击共享,共享给以太网



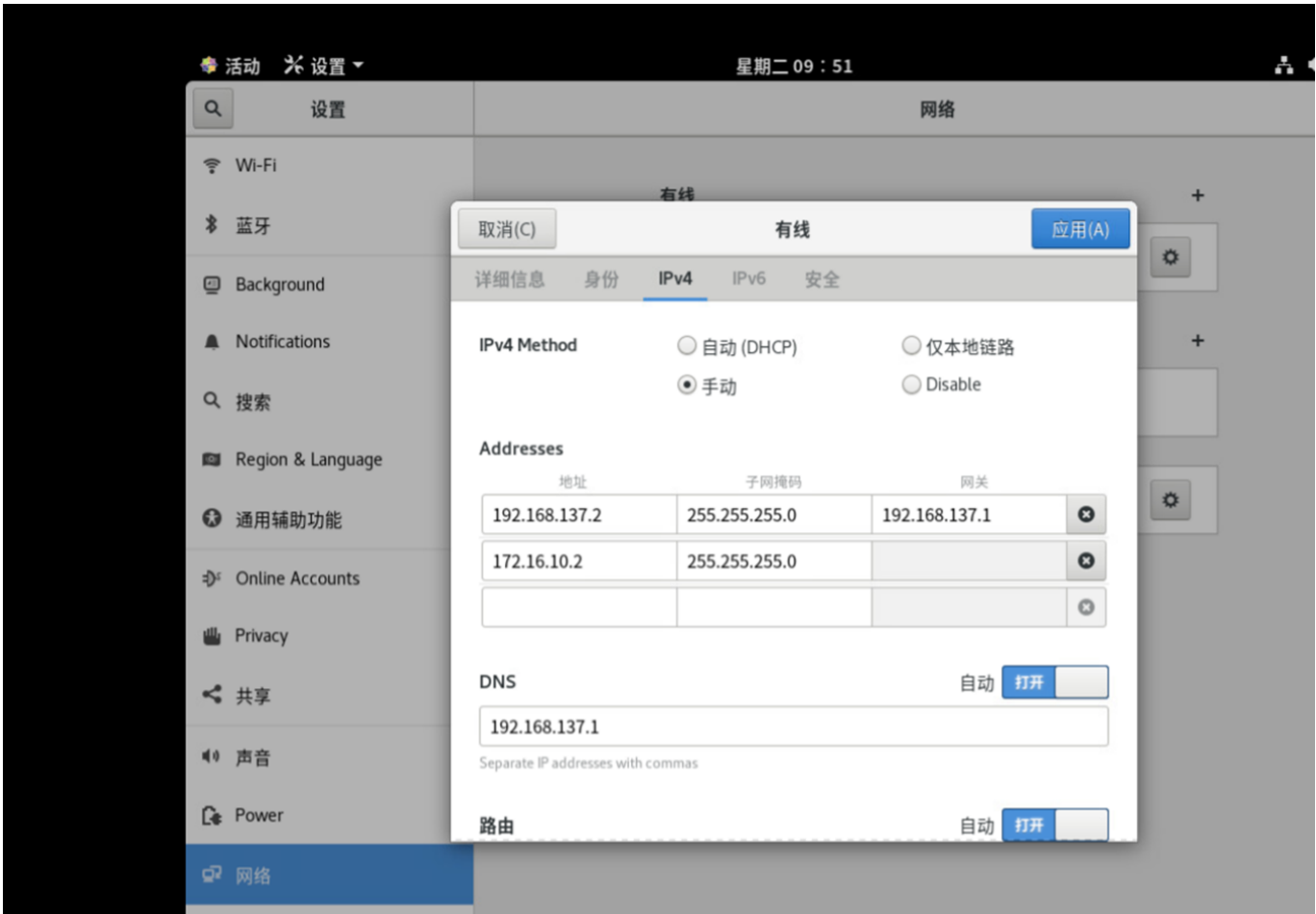
回到第一张图位置,点击通过网线连接到内网的以太网,右键点击属性



点击IPv4网络配置,此处计算机自动配置了192.168.137.1的地址,点击高级,添加内网ip,网关(本机内网IP)








进入虚拟机网络配置,配置192.168.137.x的网段,多开关网卡几次使配置生效(如未生效自行Search解决)



9 软件及配置文件

<https://github.com/moncheris/Ksec-L276>

含以下软件包及部署文档

名称	压缩后大小	原始大小	多
 apache-tomcat-9.0.87.tar.gz	11,746,152	11,743,322	C
 docker-24.0.6.tgz	69,717,000	69,797,795	B
 jdk-8u401-linux-x64.tar.gz	141,110,308	141,600,542	C
 redis-7.2.4.tar.gz	3,423,559	3,424,072	C
 webrtc-streamer-0.8.5.tar.gz	3,932,159	3,931,133	C

 .gitattributes	第一次提交
 README.md	Initial commit
 服务器部署.pdf	文档提交
 软件包.zip	第一次提交