AircityCloudForLinuxVersion 安装文档

PDF版本文档	下载地址
0615[最新版]	点击查看下载
0614	点击查看下载
0606	点击查看下载
0529	点击查看下载
0511	点击查看下载

```
# 实测环境
系统：  Kylin Linux Advanced Server V10 (Sword)(SP2)
架构：  X86_64
用户：  freedo[具有sudo权限]
```

一、安装包准备

系统架构	组件	安装包名称	下载地址	备注
x86_64	NVIDIA 显卡驱动	NVIDIA-Linux-x86_64-525.116.03.run	https://installpackages.gbim.vip/Nvidia-Drivers/ForLinux/NVIDIA-Linux-x86_64-525.116.03.run	
x86_64	中继服务 coturn 镜像	coturn-latest-amd64-20230510.tar.gz	https://installpackages.gbim.vip/docker_update/coturn-latest-amd64-20230510.tar.gz	
x86_64	AirCityCloud	AirCityCloud_x86_0615.zip	https://pan.baidu.com/s/1oF8NCCcLr5-Bq5bdnB6bwQ?pwd=ytjy	
ARM64	NVIDIA 显卡驱动	NVIDIA-Linux-aarch64-525.116.04.run	https://installpackages.gbim.vip/Nvidia-Drivers/ForLinux/ARM64/NVIDIA-Linux-aarch64-525.116.04.run	
ARM64	中继服务 coturn 镜像	coturn-latest-arm64-20230511.tar.gz	https://installpackages.gbim.vip/docker_update/coturn-latest-arm64-20230511.tar.gz	
ARM64	AirCityCloud	AirCityCloud_arm64_0615.zip	https://pan.baidu.com/s/1e1NeQ4hQTiPi6eIZxPJuJg?pwd=rdiw	

二、初始化服务器

1. 创建 freedo 用户，并授予 sudo 权限

```
useradd -u 9988 freedo  #建立用户名为 freedo 的一般用户
passwd freedo #为用户 freedo 设置密码
#Changing password for user freedo.
#New UNIX password:    ← 输入密码（密码不会被显示）
#Retype new UNIX password:    ← 再次输入密码确认两次密码一致
#passwd: all authentication tokens updated successfully.  ← 密码设置成功

#添加freedo用户到wheel组
usermod -G wheel freedo
```

> **【以下步骤都登录到 freedo 用户进行操作】**

2. 【可选，如果网络配置正常请跳过此步骤】网卡自动启动

Kylin V10 Server SP2 安装成功之后，默认网卡是禁用状态，需要配置网卡自动启用。

```
# 查看网卡信息
sudo ifconfig

# 根据获取的网卡信息，把以下命令中的ens192改为实际中的网卡名称
sudo nmcli con mod ens192 connection.autoconnect yes
```

3. 处理系统防火墙

> 选择下面的任意一种方法，处理系统自带 firewalld 防火墙

可选方法一：关闭防火墙服务

```
sudo systemctl stop firewalld
sudo systemctl disable firewalld
```

可选方法二：通过防火墙放行相关端口

```
# 允许vnc访问
sudo firewall-cmd --zone=public --add-port=5900-5902/tcp --permanent

# 允许Cloud页面访问,默认8087/tcp端口，根据实际情况修改
sudo firewall-cmd --zone=public --add-port=8087/tcp --permanent

# 渲染程序连接的端口
sudo firewall-cmd --zone=public --add-port=8088/tcp --permanent
# 实例管理服务使用的端口
sudo firewall-cmd --zone=public --add-port=8089/tcp --permanent


# 允许P2P视频流传输
sudo firewall-cmd --zone=public --add-port=50000-65535/udp --permanent


# 允许中继服务访问,默认3478/tcp 3478/udp端口，根据实际情况修改
sudo firewall-cmd --zone=public --add-port=3478/tcp --permanent
sudo firewall-cmd --zone=public --add-port=3478/udp --permanent

# 重载配置，然后查看配置状态
sudo firewall-cmd --reload
sudo firewall-cmd --list-port
```

```
[f    @kylinV10S    er-01 dep]$ sudo firewall-cmd --zone=public --add-port=8087/tcp --permanent
success
[f    @kylinV10S    er-01 dep]$ sudo firewall-cmd --zone=public --add-port=3478/tcp --permanent
success
[f    @kylinV10S    er-01 dep]$ sudo firewall-cmd --zone=public --add-port=3478/udp --permanent
success
[f    @kylinV10S    er-01 dep]$ sudo firewall-cmd --reload
success
[f    @kylinV10S    er-01 dep]$ sudo firewall-cmd --list-port
5900-5902/tcp 8087/tcp 3478/tcp 3478/udp
```

4. 【可选】配置 tigervnc

1. 安装配置 vncserver 服务

```
# 安装tigervnc服务端
sudo dnf makecache
sudo dnf install tigervnc-server tigervnc-server-module

# 配置为系统服务
sudo cp /lib/systemd/system/vncserver@.service /etc/systemd/system/
sudo sed -i 's+<USER>+freedo+g' /etc/systemd/system/vncserver@.service
sudo systemctl daemon-reload
sudo systemctl enable vncserver@:1.service
```

2. 设置 freedo 用户的 vnc 密码

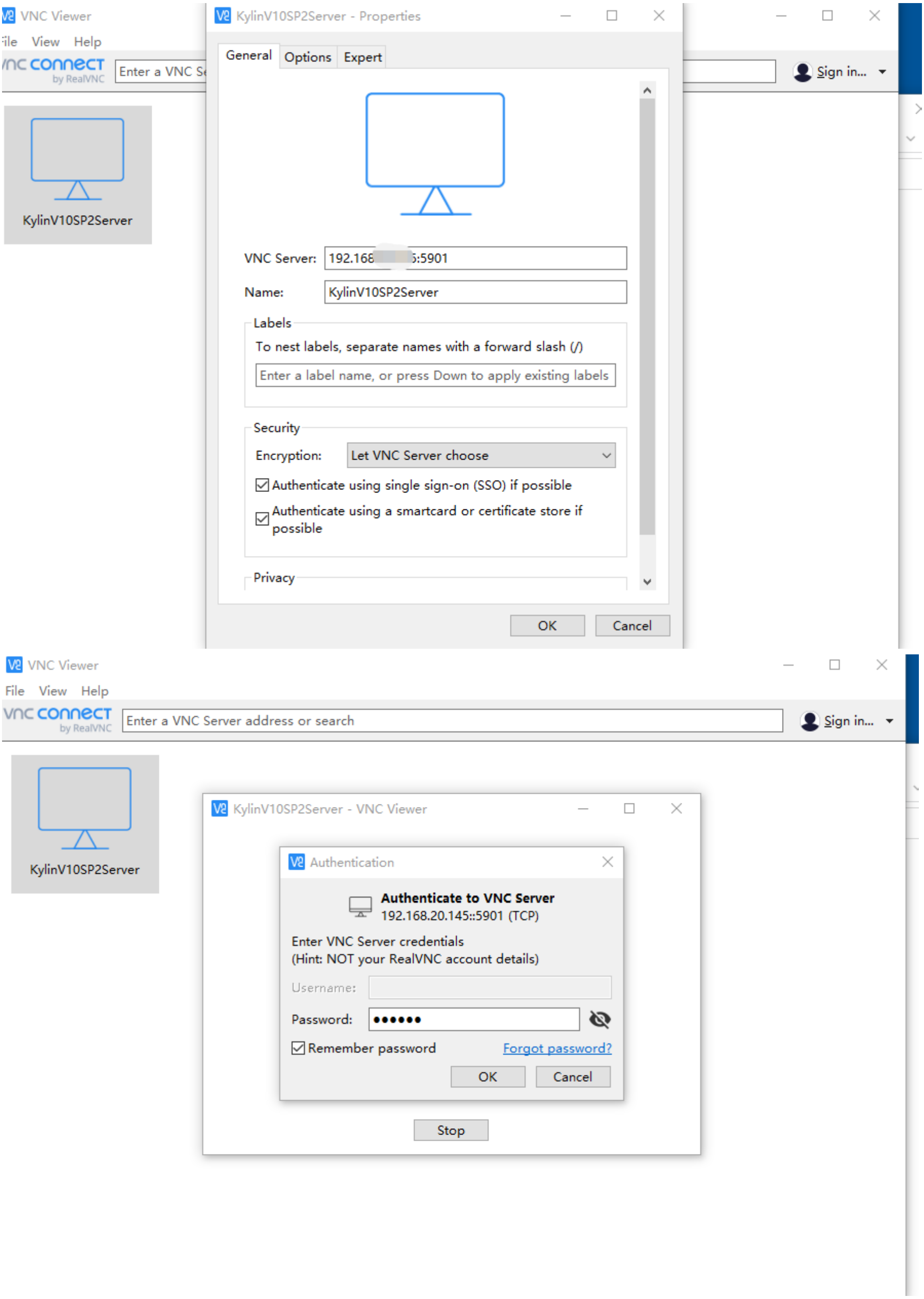
```
vncpasswd
#Password:
#Verify:
#Would you like to enter a view-only password (y/n)? n
```

3. 重启系统，确认 vnc 服务正常启动

```
[freedo@kylinV10Sp2Server-01 ~]$ sudo systemctl status vncserver@\:1.service
● vncserver@:1.service - Remote desktop service (VNC)
   Loaded: loaded (/etc/systemd/system/vncserver@.service; enabled; vendor preset: disabled)
   Active: active (running) since Mon 2023-05-08 10:18:05 CST; 9min ago
  Process: 1730 ExecStartPre=/bin/sh -c /usr/bin/vncserver -kill :1 > /dev/null 2>&1 || : (code=exited, status=0/SUCCESS)
  Process: 1845 ExecStart=/usr/bin/vncserver -autokill :1 (code=exited, status=0/SUCCESS)
 Main PID: 1855 (Xvnc)
    Tasks: 221
   Memory: 519.4M
   CGroup: /system.slice/system-vncserver.slice/vncserver@:1.service
           ├─1855 /usr/bin/Xvnc :1 -auth /home/freedo/.Xauthority -desktop kylinV10Sp2Server-01:1 (freedo) -fp catalogue:/etc/X11/fontpath.d -geometry 1024x768 -pn -rfbauth /home/freedo/.vnc/passwd -rfbport 59>
           ├─2261 sh -c (/home/freedo/.vnc/xstartup; /usr/bin/vncserver -kill :1) >> '/home/freedo/.vnc/kylinV10Sp2Server-01:1.log' 2>&1 &
           ├─2262 /usr/bin/mate-session
           ├─2265 dbus-launch --exit-with-session /usr/bin/mate-session
           ├─2304 /usr/bin/dbus-daemon --syslog --fork --print-pid 6 --print-address 8 --session
           ├─2310 /usr/libexec/dconf-service
           ├─2314 gnome-keyring-daemon --start
           ├─2318 /usr/libexec/mate-settings-daemon
           ├─2323 marco
           ├─2345 caja
           ├─2349 mate-volume-control-applet
           ├─2351 /usr/bin/python3 /usr/local/lib64/mate-indicators/libexec/reset_applet_position.py
           ├─2355 nm-applet
           ├─2361 ukui-screensaver-backend
           ├─2363 /usr/libexec/gvfsd
           ├─2373 /usr/libexec/gvfsd-fuse /home/freedo/.cache/gvfs -f -o big_writes
           ├─2374 /usr/libexec/geoclue-2.0/demos/agent
           ├─2497 /usr/libexec/imsettings-daemon
           ├─2512 /usr/libexec/at-spi-bus-launcher
           ├─2517 /usr/bin/dbus-daemon --config-file=/usr/share/defaults/at-spi2/accessibility.conf --nofork --print-address 3
           ├─2525 /usr/libexec/gvfsd-trash --spawner :1.10 /org/gtk/gvfs/exec_spaw/0
           ├─2532 /usr/bin/pulseaudio --start --log-target=syslog
           ├─2571 /usr/libexec/gvfs-udisks2-volume-monitor
           ├─2605 /usr/libexec/at-spi2-registryd --use-gnome-session
           ├─2635 /usr/libexec/gvfs-goa-volume-monitor
           ├─2654 /usr/libexec/goa-daemon
           ├─2744 mate-panel
           ├─2802 /usr/bin/fcitx -D
           ├─2806 /usr/libexec/pulse/gsettings-helper
           ├─2811 /usr/libexec/pulse/gconf-helper
```

4. 使用 VNC Viewer 访问

> VNC Viewer 下载地址: https://www.realvnc.com/en/connect/download/viewer/windows/



5. 安装配置 nvidia 显卡驱动

> 驱动下载地址： `https://www.nvidia.com/Download/Find.aspx?lang=en-us`

---

1. 安装依赖包

```
sudo dnf install elfutils-libelf-devel libglvnd libglvnd-devel vulkan-loader vulkan-loader-devel
```

2. 屏蔽自带驱动

```
sudo touch /etc/modprobe.d/blacklist-nvidia-nouveau.conf
sudo bash -c "cat > /etc/modprobe.d/blacklist-nvidia-nouveau.conf" << EOF
blacklist nouveau
options nouveau modeset=0
EOF
```

3. 显卡支持

```
sudo touch /etc/modprobe.d/nvidia.conf
sudo bash -c "cat > /etc/modprobe.d/nvidia.conf" << EOF
options nvidia NVreg_OpenRmEnableUnsupportedGpus=1
EOF
```

4. 重建 initramfs image

```
sudo mv /boot/initramfs-$(uname -r).img /boot/initramfs-$(uname -r).img.bak
sudo dracut /boot/initramfs-$(uname -r).img $(uname -r)
```

5. 重启

```
sudo systemctl reboot
```

6. 停止 x windows 相关服务

```
sudo systemctl stop lightdm vncserver@\:1
```

7. 上传下载后的驱动文件到服务器

略

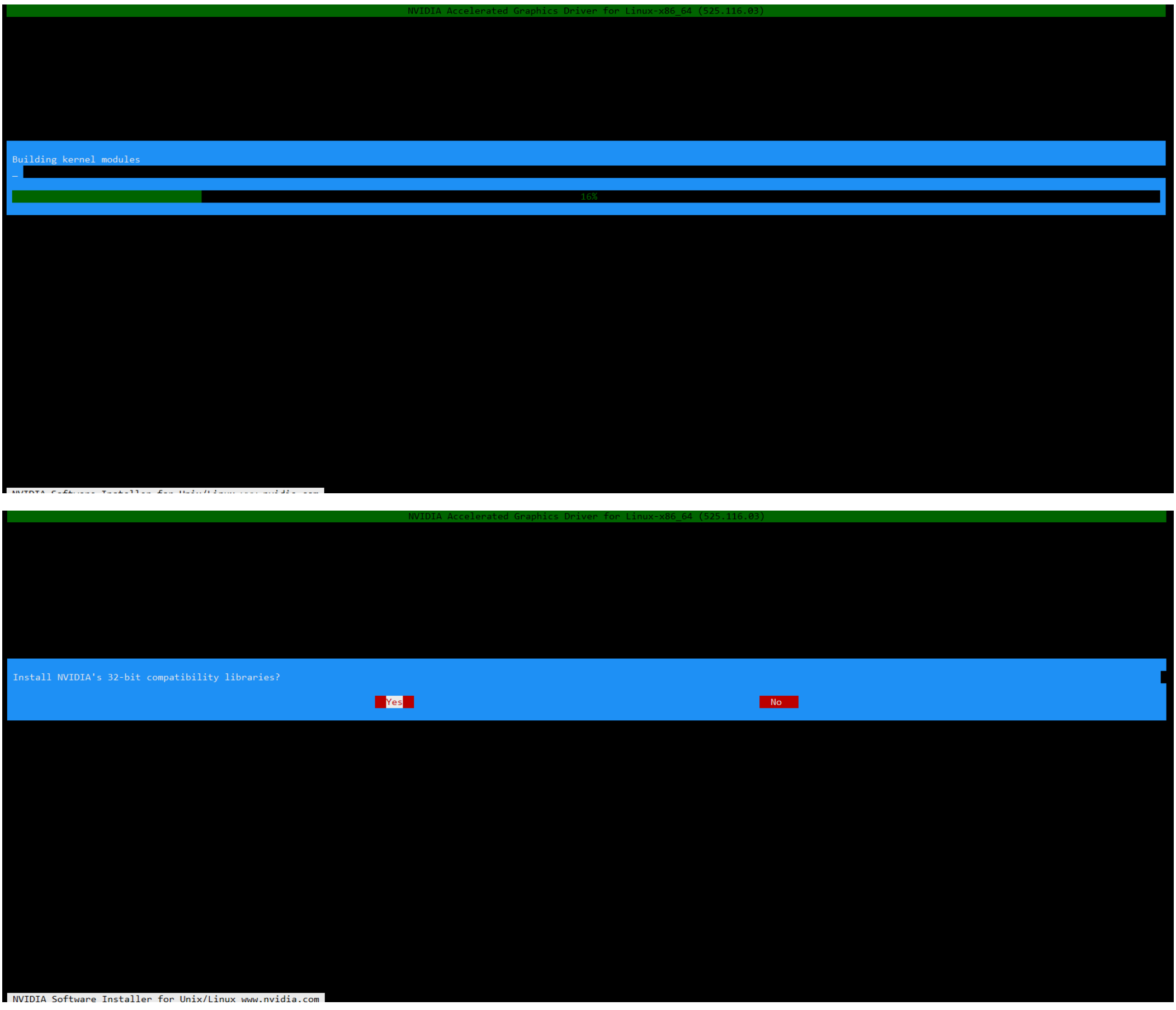
8. 为驱动文件添加执行权限

```
sudo chmod +x NVIDIA-Linux-x86_64-525.116.03.run
```

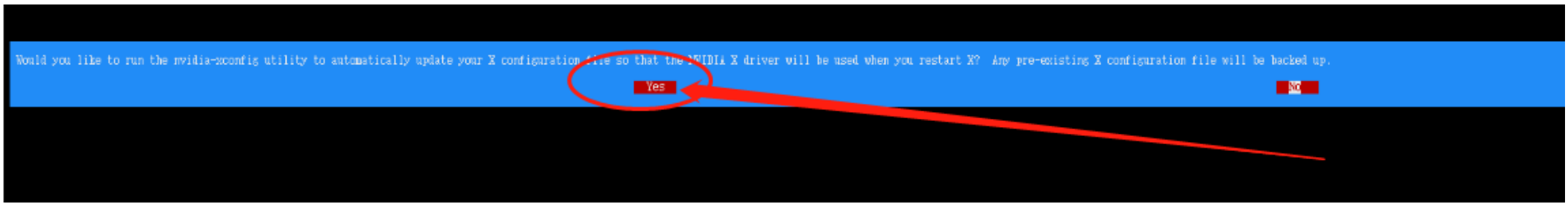
9. 运行文件安装驱动

```
sudo ./NVIDIA-Linux-x86_64-525.116.03.run -m=kernel-open
```

```
[freedo@kylinV10Sp2Server-01 ~]$ sudo chmod +x NVIDIA-Linux-x86_64-525.116.03.run
[freedo@kylinV10Sp2Server-01 ~]$ sudo ./NVIDIA-Linux-x86_64-525.116.03.run -m=kernel-open
Verifying archive integrity... OK
Uncompressing NVIDIA Accelerated Graphics Driver for Linux-x86_64 525.116.03.....................................................................................................
.....................................................................................................
.....................................................................................................
```



注意： 最后一步，提示是否自动配置 *X Window* 的时候，请选择"*yes*"，否则会导致 *vnc* 无法调用独立显卡，从而无法运行 *AirCityExplorer*



10. 再次重启

```
sudo systemctl reboot
```

11. 重启之后执行命令`nvidia-smi`确定显卡驱动状态

```
[freedo@kylinV10Sp2Server-01 ~]$ nvidia-smi
Mon May  8 10:18:56 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 525.116.03   Driver Version: 525.116.03   CUDA Version: 12.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA GeForce ...  Off  | 00000000:13:00.0 Off |                  N/A |
| 39%   44C    P8    22W / 350W |      5MiB / 24576MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      1833      G   /usr/libexec/Xorg                   4MiB |
+-----------------------------------------------------------------------------+
```

三、上传安装包到服务器任意目录并解压

1. 创建部署目录

```
sudo mkdir /deploy
sudo chown -R freedo.freedo /deploy
```

2. 上传安装包 AirCityCloud_x86_0615.zip 到/deploy

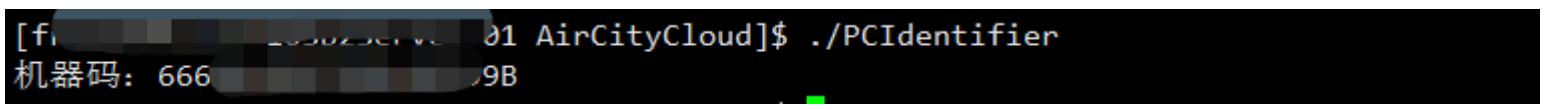
3. 解压安装包

```
unzip AirCityCloud_x86_0615.zip
```

四、申请 License

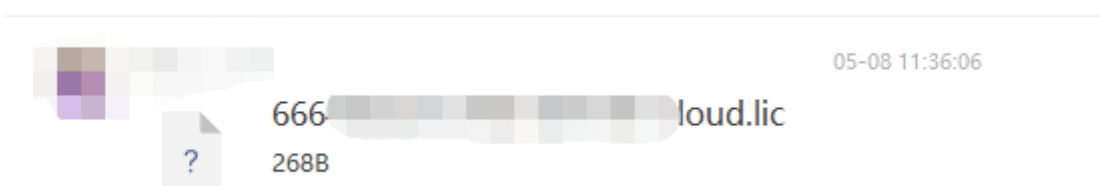
1. 使用 PCIdentifier 工具生成机器码

```
cd /deploy/AirCityCloud/
./PCIdentifier
```



2. 提供机器码给飞渡，申请授权文件

3. 拿到授权文件，重命名并复制到指定目录




```
# 假如授权文件名称为68887736D6C4901B9F99B-cloud.lic，并且已上传到/deploy目录

cd /deploy
```

```

# 重命名授权文件为License.lic
mv 68887736D6C4901B9F99B-cloud.lic License.lic

# 放置授权文件到指定目录
mv License.lic AirCityCloud/CloudServer/
```

五、拷贝依赖库到指定目录

拷贝依赖库到指定目录

X86架构

```
cd /deploy/
cd AirCityCloud/dep/dll/
cp lib* ../../CloudRenderer/AirCityExplorer/Binaries/Linux/
```

ARM64架构

```
cd /deploy/
cd AirCityCloud/dep/dll/
cp lib* ../../CloudRenderer/AirCityExplorer/Binaries/LinuxAArch64/
```

六、 执行脚本 startCloudServer.sh 启动集群管理服务 CloudServer

1. 修改配置文件 AirCityCloud/CloudServer.conf

1. 修改 serverIP 变量为本机 IP 地址

```
#!/usr/bin/env bash
# 集群管理服务配置文件

# 服务器IP地址
serverIP="192.168.20.145"
# 脚本所在目录，AircityCloud所在的目录
Aircity_Root=$PWD
```

2. 【可选】修改服务端口，默认 8087/tcp

```
# 网页与API访问端口
playerPort="8087"
```

2. 执行脚本，启动集群管理服务

```
cd /deploy/
cd AirCityCloud/

./startCloudServer.sh
```

```
[f                    r-01 AirCityCloud]$ ./startCloud.sh
正在强制停止cloudWeb进程
正在强制停止cloudDataserver进程
------------------等待web服务启动成功------------------
------------------5------------------
------------------4------------------
------------------3------------------
------------------2------------------
------------------1------------------
------------------0------------------
-------------------CloudWeb服务已启动，可以继续-------------------------
-------------------CloudDataserver服务已启动，可以继续-------------------------
^_^
^_^ 服务启动成功!
^_^ 请使用浏览器[推荐Chrome]打开网址:  http:/          087/samples/locale_zh/player.html 访问
```

3. 【可选】配置为systemd服务，并开机启动

```bash
cd /deploy/
cd AirCityCloud/

# 先停止脚本直接启动的服务
./stopCloudServer.sh

# 执行脚本自动配置服务
cd dep/
./makeService_CloudServer.sh

# 启动CloudServer服务
sudo systemctl start AirCityCloudServer.service

# 查看CloudServer服务状态
sudo systemctl status AirCityCloudServer.service
```

```
[freedo@ r-01 dep]$ sudo systemctl start AirCityCloudServer.service
[freedo ver-01 dep]$ sudo systemctl status AirCityCloudServer.service
● AirCityCloudServer.service - Freedo AirCityCloud Service
   Loaded: loaded (/etc/systemd/system/AirCityCloudServer.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2023-05-25 15:17:31 CST; 4s ago
  Process: 297931 ExecStart=/home/freedo/DTS/AirCityCloudPro/startCloudServer.sh (code=exited, status=0/SUCCESS)
 Main PID: 297945 (CloudServer_lin)
    Tasks: 10
   Memory: 26.9M
   CGroup: /system.slice/AirCityCloudServer.service
           └─297945 /home/freedo/DTS/AirCityCloudPro/CloudServer/CloudServer_lin --httpRoot="/home/freedo/DTS/AirCityCloudPro/SDK" --playerPort=8087 --license="/home/freedo/DTS/AirCityCloudPro/CloudServer/Lice>

5月 25 15:17:2  r-01 startCloudServer.sh[297931]: ------------------4------------------
5月 25 15:17:28  r-01 startCloudServer.sh[297931]: ------------------3------------------
5月 25 15:17:29  r-01 startCloudServer.sh[297931]: ------------------2------------------
5月 25 15:17:30  r-01 startCloudServer.sh[297931]: ------------------1------------------
5月 25 15:17:31  r-01 startCloudServer.sh[297931]: ------------------0------------------
5月 25 15:17:31  r-01 startCloudServer.sh[297931]: ------------------CloudServer已启动，可以继续------------------------
5月 25 15:17:31  r-01 startCloudServer.sh[297931]: ^_^
5月 25 15:17:31  r-01 startCloudServer.sh[297931]: ^_^ 服务启动成功！
5月 25 15:17:31  r-01 startCloudServer.sh[297931]: ^_^ 请使用浏览器[推荐Chrome]打开网址:  http://192.168.20.145:8087/samples/locale_zh/manage/ 管理节点
5月 25 15:17:31  r-01 systemd[1]: Started Freedo AirCityCloud Service.
```

3. 访问实例管理页面[无渲染实例]

七、 执行脚本 startNodeServcie.sh 启动单个渲染实例

> 子节点只需要启动本 NodeService 服务，不需要启动 CloudServer 以及中继服务

1. 修改配置文件 AirCityCloud/NodeService.conf，使渲染节点加入到集群管理服务中

> 1、 修改 serverIP 变量为集群管理服务的 IP 地址
> 2、 修改 serverPort 变量为集群管理服务的端口
> - 【默认8087端口；启用Https时，默认8089端口】

```
#!/usr/bin/env bash
# 渲染节点服务配置文件

# 服务器IP地址
serverIP="192.168.20.145"

# 服务器端口
serverPort="8087"

# 脚本所在目录，AircityCloud所在的目录
Aircity_Root=$PWD
```

2. 执行脚本，启动渲染实例

```
cd /deploy/
cd AirCityCloud/

./startNodeService.sh
```

CloudMaster

实例管理：（4个实例）

实例标识符	运行状态	运行时长	渲染节点	分辨率	帧率	模式	连接	次数	缩放	限制	显卡	锁定	工程	操作
2511176939984	Running	00:25:59	192.168.20.145	1920x937	25	1	0	3	Y	Y	0	N	/home/freedo/DTS/DTSProject/vtpkwjn/vtpkwjn.acp	设置参数 取消锁定 启动 停止
2511176939996	Running	00:25:52	192.168.20.145	1920x1080	25	1	0	0	Y	Y	1	N	/home/freedo/DTS/DTSProject/vtpkwjn/vtpkwjn.acp	设置参数 取消锁定 启动 停止
2511176940008	Running	00:25:46	192.168.20.145	1920x1080	25	1	0	0	Y	Y	0	N	/home/freedo/DTS/DTSProject/vtpkwjn/vtpkwjn.acp	设置参数 取消锁定 启动 停止
2511176940020	Running	00:25:39	192.168.20.145	1920x1080	25	1	0	0	Y	Y	1	N	/home/freedo/DTS/DTSProject/vtpkwjn/vtpkwjn.acp	设置参数 取消锁定 启动 停止

测试页面： 视频流测试 API示例 实例管理接口 实时运行状态 样例集锦 Hello World 二三维坐标转换

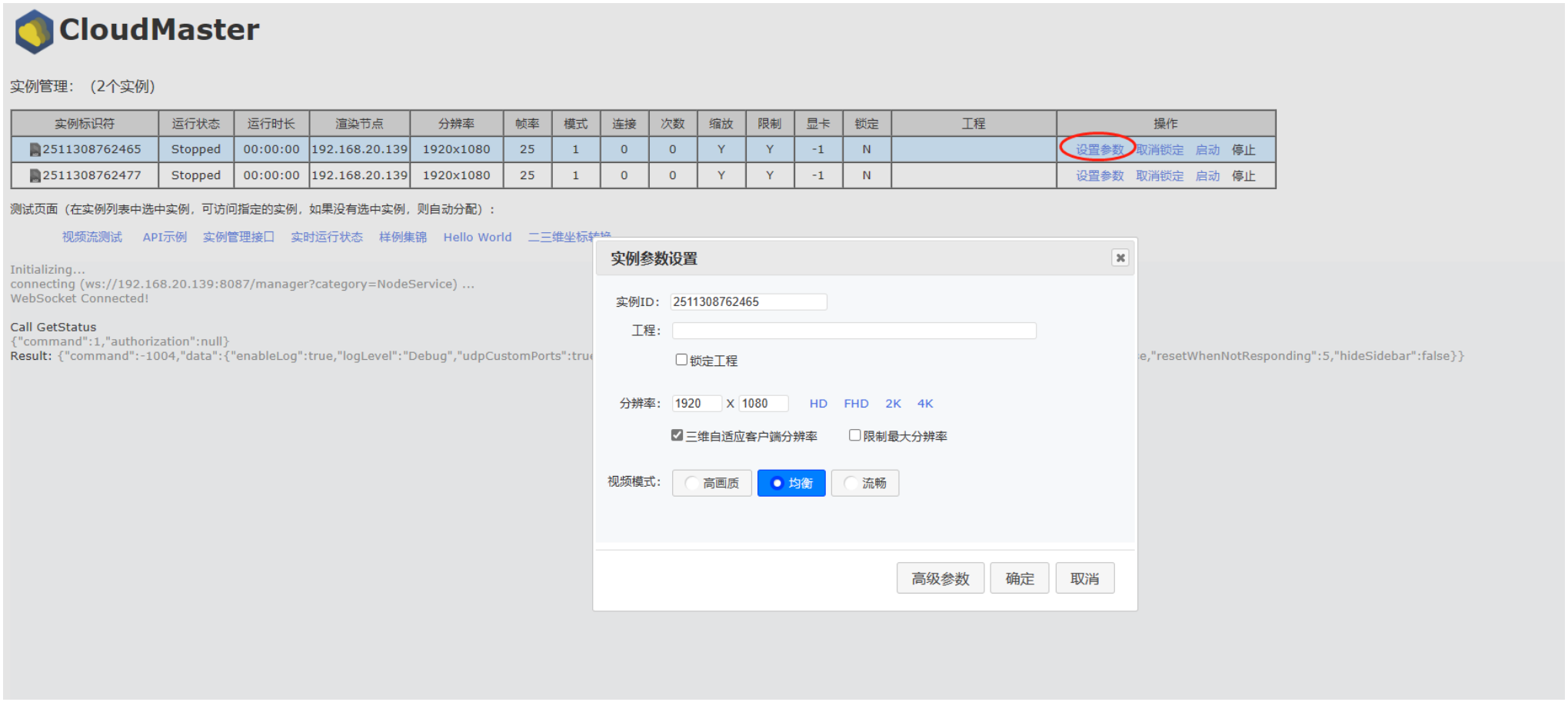
3. 【可选】配置为systemd服务，并开机启动

```
cd /deploy/
cd AirCityCloud/

# 先停止脚本直接启动的服务
./stopNodeService.sh

# 执行脚本自动配置服务
cd dep/
./makeService_NodeService.sh

# 启动NodeService服务
sudo systemctl start AirCityNodeService.service

# 查看NodeService服务运行状态
sudo systemctl status AirCityNodeService.service
```

```
[freedo[redacted]-01 AirCityCloudPro]$ sudo systemctl status AirCityNodeService.service
● AirCityNodeService.service - Freedo AirCityCloud Service
   Loaded: loaded (/etc/systemd/system/AirCityNodeService.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2023-05-25 15:13:48 CST; 8s ago
  Process: 296738 ExecStart=/home/freedo/DTS/AirCityCloudPro/startNodeService.sh (code=exited, status=0/SUCCESS)
 Main PID: 296761 (mono)
    Tasks: 439
   Memory: 2.3G
   CGroup: /system.slice/AirCityNodeService.service
           ├─296761 mono /home/freedo/DTS/AirCityCloudPro/CloudRenderer/RendererAgent/NodeService.exe --ip=192.168.20.145 --port=8089 --dataRoot="/home/freedo/DTS/AirCityCloudPro/CloudRenderer/RendererAgent/No
           ├─296782 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/AirCityExplorer/Binaries/Linux/AirCityExplorer-Linux-Shipping -id=2512210311074 -projectpath=/home/freedo/DTS/AirCityCloudPro/SDK/media/projec
           ├─296785 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/AirCityExplorer/Binaries/Linux/AirCityExplorer-Linux-Shipping -id=2512210311085 -projectpath=/home/freedo/DTS/AirCityCloudPro/SDK/media/projec
           ├─296788 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/AirCityExplorer/Binaries/Linux/AirCityExplorer-Linux-Shipping -id=2512210311096 -projectpath=/home/freedo/DTS/AirCityCloudPro/SDK/media/projec
           ├─296791 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/AirCityExplorer/Binaries/Linux/AirCityExplorer-Linux-Shipping -id=2512210311107 -projectpath=/home/freedo/DTS/AirCityCloudPro/SDK/media/projec
           ├─296933 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=zygote --no-zygote-sandbox --no-sandbox --locales-dir-path=/home/freedo/DTS/AirCityCloudPro/CloudR
           ├─296935 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=zygote --no-zygote-sandbox --no-sandbox --locales-dir-path=/home/freedo/DTS/AirCityCloudPro/CloudR
           ├─296941 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=zygote --no-sandbox --locales-dir-path=/home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binar
           ├─296942 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=zygote --no-sandbox --locales-dir-path=/home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binar
           ├─296945 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=zygote --no-zygote-sandbox --no-sandbox --locales-dir-path=/home/freedo/DTS/AirCityCloudPro/CloudR
           ├─296946 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=zygote --no-zygote-sandbox --no-sandbox --locales-dir-path=/home/freedo/DTS/AirCityCloudPro/CloudR
           ├─296947 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=zygote --no-sandbox --locales-dir-path=/home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binar
           ├─296948 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=zygote --no-sandbox --locales-dir-path=/home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binar
           ├─296973 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=gpu-process --field-trial-handle=11900335550845050427,18213647869669702984,131072 --no-sandbox --o
           ├─296974 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=gpu-process --field-trial-handle=1501349583957061839,5662202684715684194,131072 --no-sandbox --ozo
           ├─296999 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=gpu-process --field-trial-handle=1297291133929895741 5,7373245993841430247,131072 --no-sandbox --oz
           ├─297005 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=gpu-process --field-trial-handle=7149277571050211753,4606713364906240720,131072 --no-sandbox --ozo
           ├─297200 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=utility --utility-sub-type=network.mojom.NetworkService --field-trial-handle=11900335550845050427,
           ├─297201 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=utility --utility-sub-type=network.mojom.NetworkService --field-trial-handle=1501349583957061839,5
           ├─297230 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=utility --utility-sub-type=network.mojom.NetworkService --field-trial-handle=7149277571050211753,4
           └─297231 /home/freedo/DTS/AirCityCloudPro/CloudRenderer/Engine/Binaries/Linux/CloudWebHelper --type=utility --utility-sub-type=network.mojom.NetworkService --field-trial-handle=1297291133929895741 5,

5月 25 15:13:48 [redacted]Server-01 systemd[1]: Starting Freedo AirCityCloud Service...
5月 25 15:13:48 [redacted]Server-01 sudo[296739]:   freedo : TTY=unknown ; PWD=/home/freedo/DTS/AirCityCloudPro ; USER=root ; COMMAND=/usr/bin/chmod -R a+x ./CloudRenderer/RendererAgent/BatchFiles
5月 25 15:13:48 [redacted]-01 startNodeService.sh[296738]: ------------------Mono已安装------------------------
5月 25 15:13:48 kyli[redacted]-01 sudo[296755]:   freedo : TTY=unknown ; PWD=/home/freedo/DTS/AirCityCloudPro ; USER=root ; COMMAND=/usr/bin/ps -ef
5月 25 15:13:48 [redacted]-01 startNodeService.sh[296738]: 开始启动NodeService.
5月 25 15:13:48 [redacted]Server-01 sudo[296762]:   freedo : TTY=unknown ; PWD=/home/freedo/DTS/AirCityCloudPro ; USER=root ; COMMAND=/usr/bin/ps -ef
5月 25 15:13:48 [redacted]-01 startNodeService.sh[296738]: ------------------NodeService已启动------------------------
5月 25 15:13:48 [redacted]-01 systemd[1]: Started Freedo AirCityCloud Service.
[redacted] AirCityCloudPro]$
```

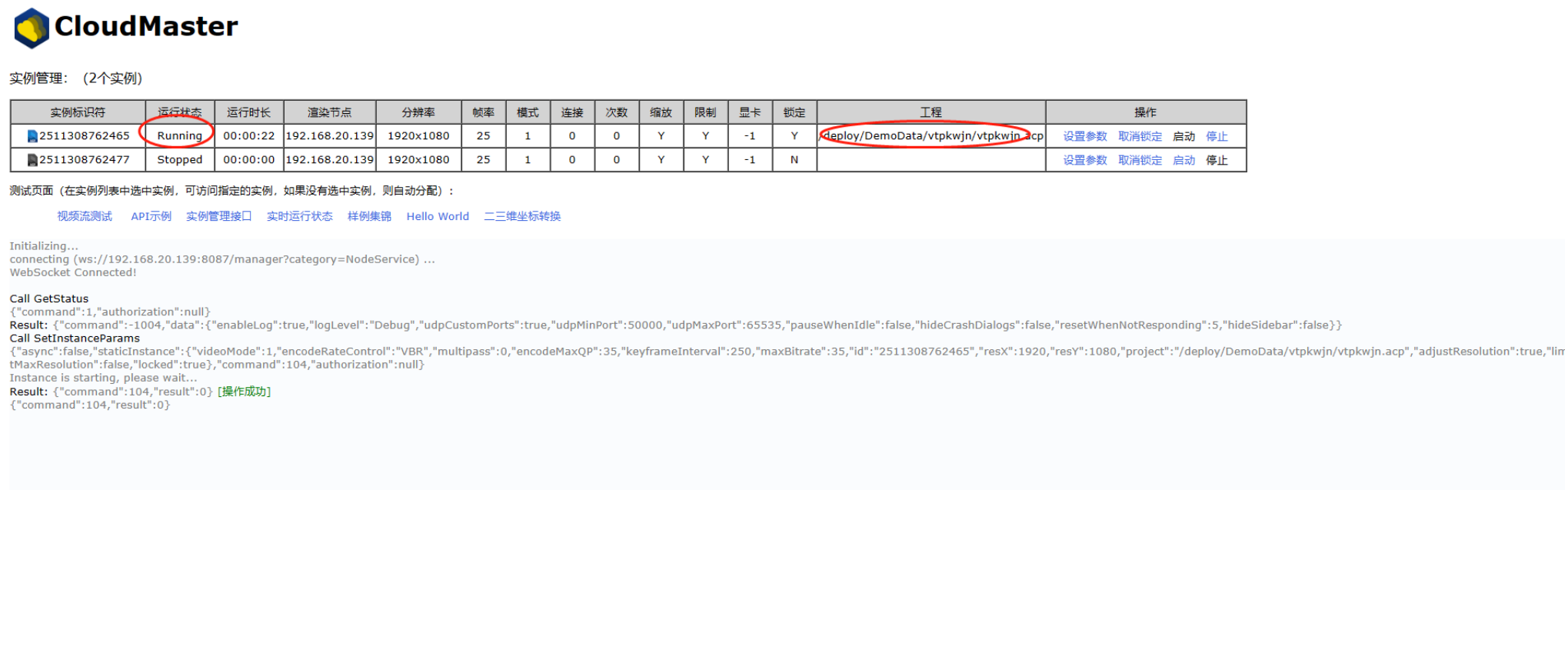
4. 添加工程数据并访问

1. 上传工程数据到服务器上
2. 点击"设置参数"，打开实例设置界面

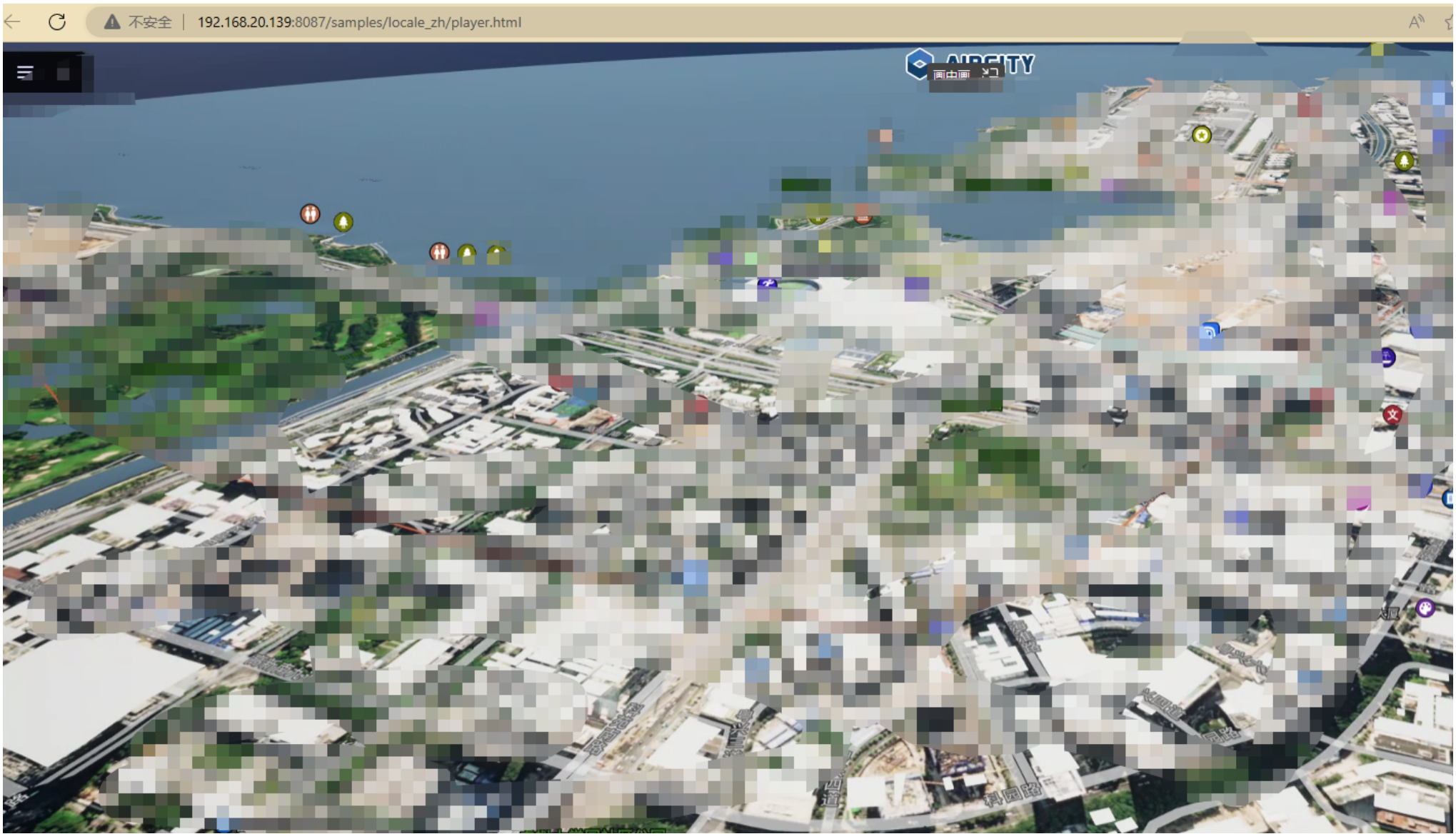


3. 复制粘贴之前上传的 acp 工程的绝对路径到工程空白框中

4. 点击确定，然后本实例就会自动启动



5. 点击"视频流测试"，然后就可以跳转到视频流页面



八、 【可选项】安装中继服务[在需要跨网段访问的 cloud 服务的环境下，就需要部署中继 coturn 服务]

coturn 是一个基于 TURN（Traversal Using Relay NAT）协议实现的用于在 NAT 网络下进行实时音视频通信的服务器软件。在网络层中，NAT 路由器会将源 IP 地址重写为自己的公网 IP 地址，而 TURN 服务器则可以通过中继数据包的方式，将经过 NAT 路由器的音视频数据转发到目标客户端，实现穿越 NAT 的效果，从而保证实时音视频通信的顺畅和稳定。

coturn提供了一个开源的免费实现，支持以下特性：

1. IPv4和IPv6网络的支持。
2. 高可用性和负载均衡的支持。
3. 证书验证和加密传输的支持。
4. 远程管理和监控的支持。
5. WebRTC（Web Real-Time Communications）的支持。
6. 简单可扩展的支持。

coturn适用于WebRTC、VoIP等网络通信应用，也可以作为一个代理服务器提供中间服务，增强网络连接稳定性与可靠性。

1. 安装 docker 服务

```
sudo dnf install docker
sudo systemctl enable docker
sudo systemctl start docker

# 使freedo用户可运行docker
sudo usermod -aG docker freedo
newgrp docker

# 设置内核支持ipv4转发
sudo sed -i "s+net.ipv4.ip_forward=0+net.ipv4.ip_forward=1+g" /etc/sysctl.conf
sudo sysctl -p
```

```
[ ]AirCityCloud]$ sudo dnf install docker
上次元数据过期检查：1:32:11 前，执行于 2023年05月09日 星期二 07时18分30秒。
依赖关系解决。
================================================================================================================================================================
 Package                                    Architecture                          Version                                  Repository                             Size
================================================================================================================================================================
安装:
 docker-engine                              x86_64                                18.09.0-202.ky10                         ks10-sp2-iso                           51 M

事务概要
================================================================================================================================================================
安装  1 软件包

总下载：51 M
安装大小：216 M
确定吗？[y/N]：  y
下载软件包：
docker-engine-18.09.0-202.ky10.x86_64.rpm                                                                                  68 MB/s |  51 MB     00:00
----------------------------------------------------------------------------------------------------------------------------------------------------------------
总计                                                                                                                       68 MB/s |  51 MB     00:00
运行事务检查
事务检查成功。
运行事务测试
事务测试成功。
运行事务
  准备中  :                                                                                                                                             1/1
  安装    : docker-engine-18.09.0-202.ky10.x86_64                                                                                                       1/1
  运行脚本: docker-engine-18.09.0-202.ky10.x86_64                                                                                                       1/1
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/system/docker.service.

/sbin/ldconfig: /usr/lib64/libLLVM-7.so 不是符号链接


  验证    : docker-engine-18.09.0-202.ky10.x86_64                                                                                                       1/1

已安装:
  docker-engine-18.09.0-202.ky10.x86_64

完毕！
```

2. 上传镜像到服务器

x86 架构镜像： https://installpackages.gbim.vip/docker_update/coturn-latest-amd64-20230510.tar.gz
aarch64 架构镜像： https://installpackages.gbim.vip/docker_update/coturn-latest-arm64-20230511.tar.gz

3. 挂载镜像

```
tar zxvf coturn-latest-amd64-20230510.tar.gz
docker load < coturn-latest-amd64-20230510.tar
```

```
[ ]ver-01 coturn]$ ls
coturn-latest-20230213.tar.gz
[ ]er-01 coturn]$ tar zxvf coturn-latest-20230213.tar.gz
coturn-latest-20230213.tar
[ ]ver-01 coturn]$ sudo docker load < coturn-latest-20230213.tar
4695cdfb426a: Loading layer [==================================================>]     84MB/84MB
46e2019b70bf: Loading layer [==================================================>]  49.25MB/49.25MB
6b66680faf51: Loading layer [==================================================>]  4.193MB/4.193MB
fe03f45141bf: Loading layer [==================================================>]  1.591MB/1.591MB
Loaded image: coturn/coturn:latest
```

4. 编辑启动脚本，然后启动服务

默认使用3478/tcp 和 3478/udp 端口
如果要修改中继服务为其他端口，比如9999，替换下面脚本中的
`-p 3478:3478 \`
`-p 3478:3478/udp \`
为
`-p 9999:3478 \`
`-p 9999:3478/udp \`

```
cd /deploy
mkdir coturn
cat > start_coturn.sh << "EOF"
docker run -d \
--name freedo-coturn-01 \
--restart=always \
-e DETECT_EXTERNAL_IP=yes \
-e DETECT_RELAY_IP=yes \
-p 3478:3478 \
-p 3478:3478/udp \
-p 5349:5349 \
-p 5349:5349/udp \
-p 48100-48200:48100-48200/udp \
coturn/coturn:latest-amd64 \
-n --log-file=stdout \
--min-port=48100 \
--max-port=48200 \
--user=freedo:freedo \
```

```
--realm=feidu \
--lt-cred-mech
EOF

# 授予脚本执行权限
chmod +x start_coturn.sh

# 启动脚本
./start_coturn.sh

# 查看服务是否正常
sudo docker ps
```

> 若架构为ARM64,修改start_coturn.sh中的`coturn/coturn:latest-amd64`为`coturn/coturn:latest-arm64`,然后执行命令`docker rm -f freedo-coturn-01`，重新执行脚本`./start_coturn.sh`

```
[f...@...ver-01 coturn]$ cat >> start_coturn.sh << EOF
> sudo docker run -d \
> --name freedo-coturn-01 \
> --restart=always \
> -p 3478:3478 \
> -p 3478:3478/udp \
> -p 5349:5349 \
> -p 5349:5349/udp \
> -p 48100-48200:48100-48200/udp \
> coturn/coturn:latest \
> -n --log-file=stdout \
> --min-port=48100 \
> --max-port=48200
> EOF
[...r-01 coturn]$ chmod +x start_coturn.sh
[...01 coturn]$ ./start_coturn.sh
7f6cb19f68481951234a015d68d53689bd59c07e13e50b58a6fc5c3767a9e4de
[...01 coturn]$ sudo docker ps
CONTAINER ID        IMAGE                  COMMAND                  CREATED             STATUS              PORTS
                    NAMES
7f6cb19f6848        coturn/coturn:latest   "docker-entrypoint.s…"   9 seconds ago       Up 6 seconds        0.0.0.0:3478->3478/tcp, 0.0.0.0:3478->3478/udp, 0.0.0.0:5349->5349/udp, 0.0.0.0:5349->5349/tcp, 0.0.0.0:48100-48200->48100-48200/udp   freedo-coturn-01
```

5. 配置 AirCityCloud/Config/TurnServer.conf 配置文件，启用中继服务

```
#!/usr/bin/env bash
# 中继服务配置文件

# 是否使用中继服务连接配置，默认不使用
useTurnServer="yes"

# 中继服务访问端口
turnServerPort="3478"

# 当中继服务无法使用udp端口时，使用tcp端口，默认使用udp
useTCP="no"

# 中继服务连接配置
stunServer="stun:$serverIP:$turnServerPort"
if [ "$useTCP" = "yes" ]; then
    turnProtocol="tcp"
    turnServer="turn:$serverIP:$turnServerPort?transport=$turnProtocol"
else
    turnServer="turn:$serverIP:$turnServerPort"
fi
turnUsername="freedo"
turnPassword="freedo"

# 视频流中继方式
iceTransportPolicy="relay"
```

1. 修改turnServerPort变量为上一步启动的中继服务的端口，默认3478
2. 修改useTurnServer变量，yes表示启用中继服务，no表示不启用
3. 修改useTCP变量，默认使用udp协议进行数据转发,yes表示使用TCP协议

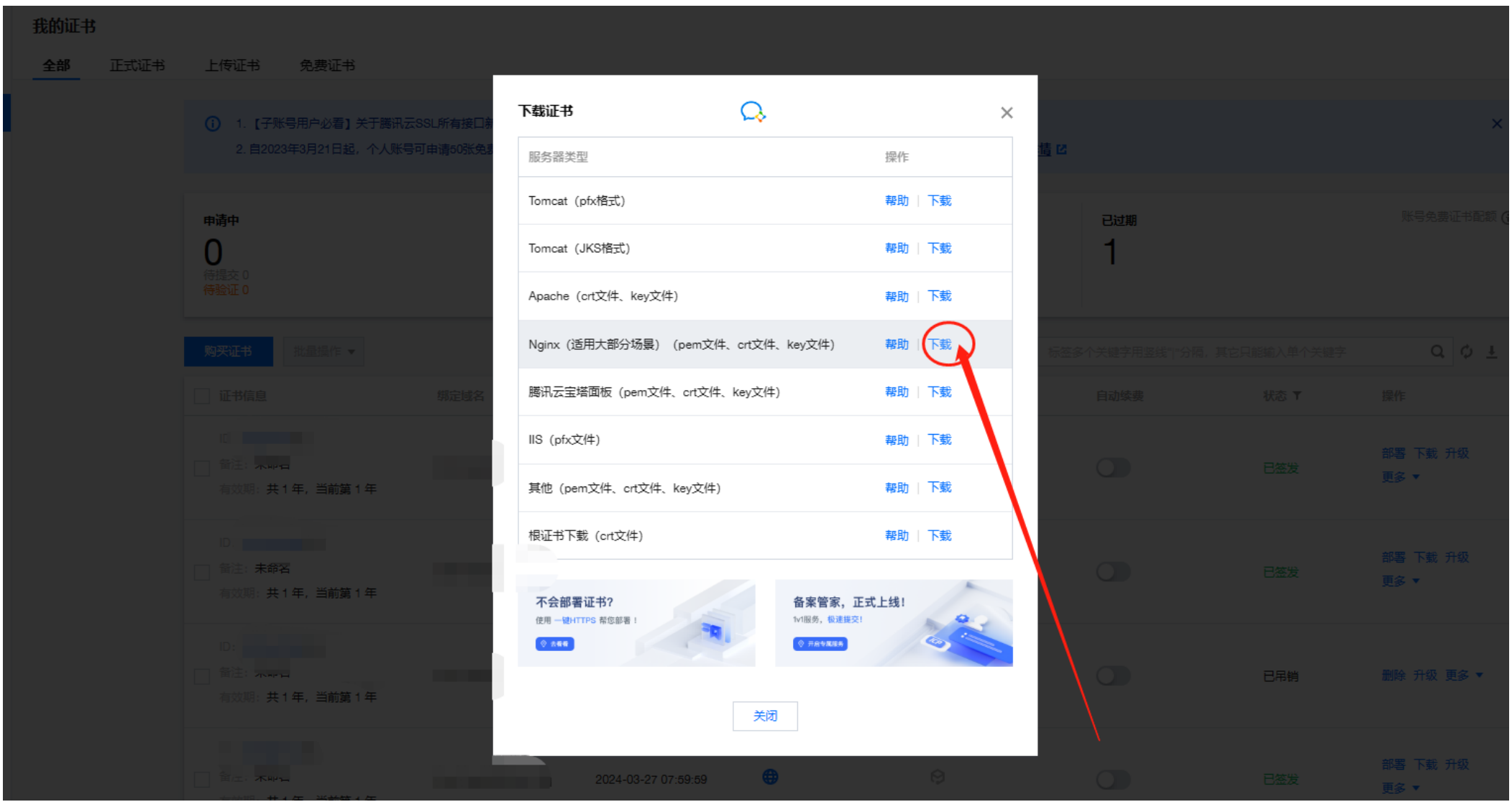
6. 重启服务使中继服务配置生效

```
sudo systemctl restart AirCityCloudServer.service
```

九、【可选】配置Https访问

1. 申请证书，推荐腾讯云免费证书

2. 申请完成之后，下载nginx证书



3. 上传下载的证书到服务器任意目录，并解压

```
[free                    r-01 cert]$ tree -L 2 .
.
├── b                        _nginx
│   ├──                      com_bundle.crt
│   ├── b                    l.com_bundle.pem
│   ├── b                    .com.csr
│   └── b                    3d.com.key
└── b                        .com_nginx.zip

1 directory, 5 files
```

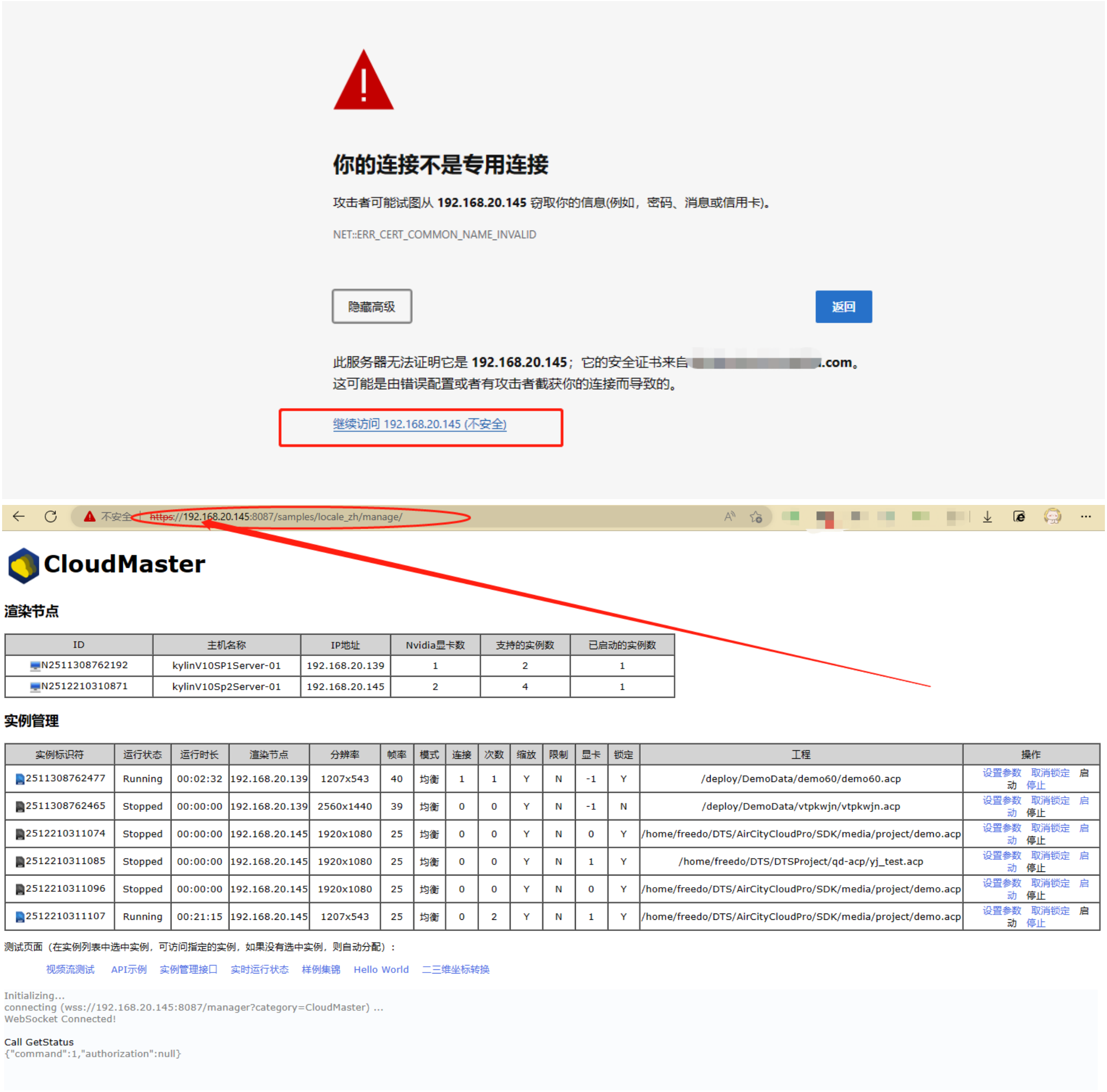
4. 配置AirCityCloud/Config/Https.conf配置文件，启用Https服务

```
#!/usr/bin/env bash
# HTTPS配置文件

# 是否使用HTTPS访问，默认不使用，如果使用HTTPS访问请对此配置中的各项指定正确的参数
useHttps="yes"

# 指定渲染程序连接的端口
streamerPort="8088"
# 指定实例管理服务使用的端口
managerPort="8089"

#证书，如果提供了有效的证书参数，则提供https服务以代替http
ssl_cert="/home/freedo/DTS/AirCityCloudPro/Config/cert/bi                    m_nginx/l                    om_bundle.pem"

#私钥文件
ssl_key="/home/freedo/DTS/AirCityCloudPro/Config/cert/b                    .com_nginx/                    .com.key"

#证书密码（可选，创建证书时设置了密码则此处就要设置对应密码）
ssl_passphrase=""
```

5. 重启服务使Https配置生效

```
sudo systemctl restart AirCityCloudServer.service
```

6. 使用https访问集群管理页面



7. NodeService服务配置修改【所有节点服务配置都要修改】

> 当启用Https的时候，实例管理服务的端口为8089端口，所以需要修改配置文件中的端口

```
#!/usr/bin/env bash
# 渲染节点服务配置文件

# CloudServer服务器IP地址
serverIP="192.168.20.145"

# 服务器端口
managerPort="8089"

# 脚本所在目录，AircityCloud所在的目录
Aircity_Root=$PWD

# dataRoot路径
NodeServiceDataRoot="$Aircity_Root/CloudRenderer/RendererAgent/NodeServiceDataRoot"

# 是否后台运行，默认后台运行，只有运行出错的情况下才关闭后台运行便于观察错误
BackgroundProcess="yes"
```

8. 重启NodeService服务

```
sudo systemctl restart AirCityNodeService.service
```

十、 公网端口映射配置

1. 在出口路由器上映射 `8087/tcp` 和 `3478/udp` 端口到公网

> 8087/tcp 为 player页面访问端口，保证页面和接口的正常访问
> 3478/udp 为 coturn中继服务访问端口，保证视频流正常传输

本文档测试环境映射配置

内网IP	内网端口	映射IP	映射端口
192.168.20.145	8087/tcp	43.227.255.154	20535/tcp
192.168.20.145	3478/udp	43.227.255.154	13489/udp

2. 根据端口映射配置，修改AirCityCloud/Config/TurnServer.conf

```
useTCP="no"

# 中继服务连接配置
stunServer="stun:$serverIP:$turnServerPort"
if [ "$useTCP" = "yes" ]; then
    turnProtocol="tcp"
    turnServer="turn:$serverIP:$turnServerPort?transport=$turnProtocol"
else
    turnServer="turn:$serverIP:$turnServerPort"
fi
turnUsername="freedo"
turnPassword="freedo"

# 视频流中继方式
iceTransportPolicy="relay"

# 域名和ip映射
    # 只有在开启中继服务之后，在需要的情况下才设置useMappingServer
    useMappingServer="yes"

    # 指定端口映射服务器的IP地址或域名
    mappingServerIP="43.227.255.154"

    # 指定中继服务的映射端口
    portTurnMapping="13489"

    # playerPort的映射端口
    playerPortMapping="20535"


changeConf_MappingServer(){
    # $1 传进来的ac_conf.js的路径
    if [ "$useTurnServer" = "yes" -a "$useMappingServer" = "yes" -a -n "$mappingServerIP" ]; then
        sed -i "s+\"Path\"+\"PlayerMapping\":\ \"$mappingServerIP:$playerPortMapping\",\n\ \ \"Path\"+g" $1
    fi
}
```

1. 修改 `useMappingServer` 为 `yes`
2. 修改 `mappingServerIP` 为映射后的IP
3. 修改 `portTurnMapping` 为 `3478/udp` 映射后的端口
4. 修改 `playerPortMapping` 为 `8087/tcp` 映射后的端口

3. 重启CloudServer服务使端口映射配置生效

```
sudo systemctl restart AirCityCloudServer.service
```

十一、【可选】KylinV10 系统配置 nfs

本文档中的用户的 uid 为 9988 gid 为 9988
共享目录和挂载目录均为 /deploy/shareData
KylinV10Server 默认已安装 nfs-utils 和 rpcbind

以上参数根据系统实际情况进行灵活修改

1、环境准备

IP	功能
192.168.20.145	NFS 服务端
192.168.20.139	NFS 客户端

2、启动 NFS 服务

1. 配置防火墙【若 firewalld 已禁用，跳过本步骤】

```
sudo firewall-cmd --permanent --add-service=nfs
sudo firewall-cmd --permanent --add-service=mountd
sudo firewall-cmd --permanent --add-service=rpc-bind
sudo firewall-cmd --reload
```

2. 创建共享目录并授权

```
mkdir /deploy/shareData
chmod 777 /deploy/shareData
```

3. 修改 nfs 配置文件

```
# anonuid=9988,anongid=9988 若客户端用户的uid/gid有变化，请自行修改
sudo bash -c "cat >> /etc/exports" << "JIESHU"
/deploy/shareData 192.168.20.0/24(rw,sync,insecure,no_subtree_check,no_root_squash,anonuid=9988,anongid=9988)
JIESHU
```

4. 启动 nfs 服务，并配置开机启动

```
sudo systemctl start nfs.service
sudo systemctl enable nfs.service
```

3、客户端挂载 NFS 共享目录，并配置开机自动挂载

1. 确认可正常访问共享

```
showmount -e 192.168.20.145
```

2. 配置开机自动挂载

```
mkdir /deploy/shareData

sudo bash -c "cat >> /etc/fstab" << "JIESHU"
192.168.20.145:/deploy/shareData /deploy/shareData nfs4 defaults 0 0
JIESHU
```

3. 挂载目录

```
sudo mount -a
sudo df -hT
```

4. 测试

```
# 服务端
cd /deploy/shareData
touch ThisIsServer

# 客户端
cd /deploy/shareData
touch ThisIsClient
```

十二、常见问题

1. 实例无法启动

1. 检查是否已申请授权，并重命名为 License.lic，并放置到指定目录
2. 检查是否使用拥有 sudo 权限的普通用户运行 CloudServer 和 NodeService
3. 检查是否安装 vulkan 相关 rpm 包
4. 集群环境下，若实例一直处于 starting 状态，尝试先停止实例，再启动实例

2. 无法访问页面

1. 检查服务是否正常启动 检查命令 `ps -ef|grep AirCityCloud`

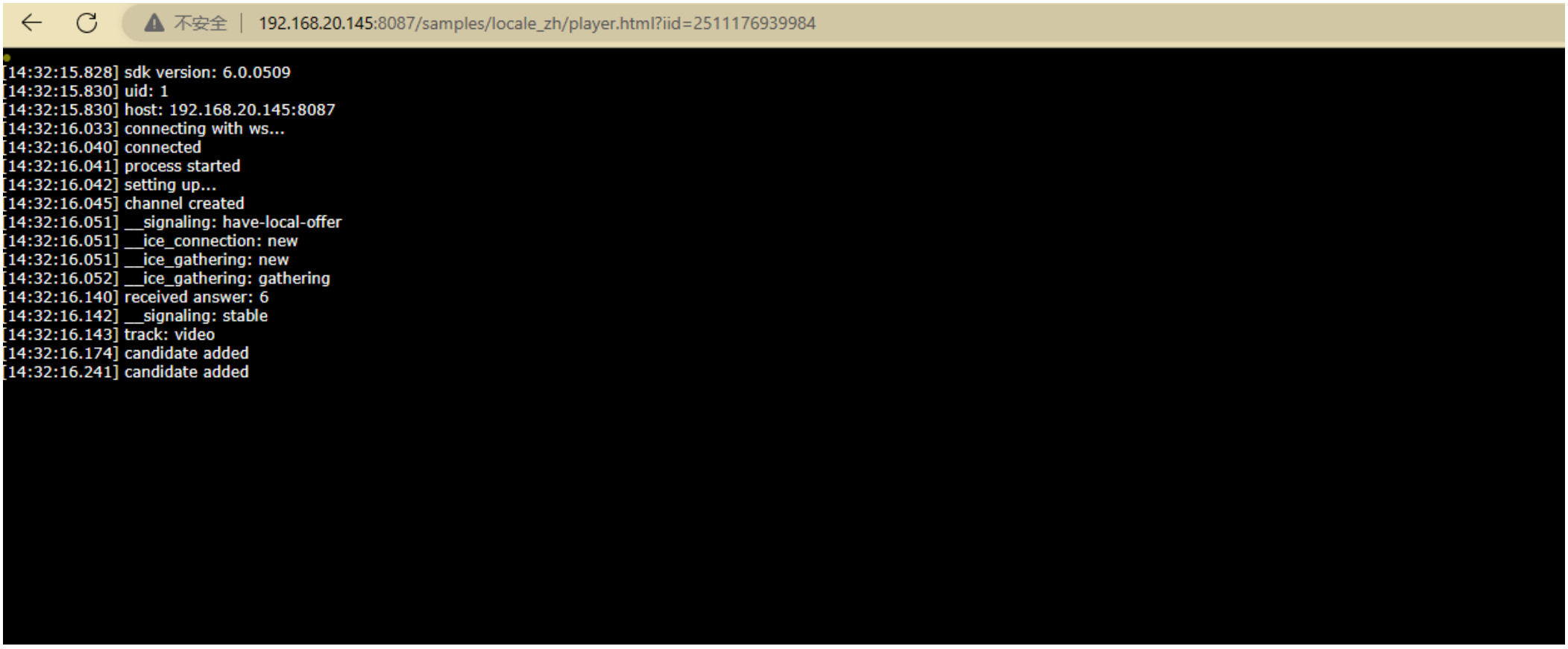
```
[...01 AirCityCloudPro]$ ps -ef|grep AirCityCloudPro
freedo     59194       1  0 13:59 pts/1    00:00:03 /home/freedo/DTS/AirCityCloudPro/AirCityCloudServer/CloudServer_lin --httpRoot=/home/freedo/DTS/AirCityCloudPro/AirCityCloudSDK/JS --playerPort=8087 --fps=25
--license=/home/freedo/DTS/AirCityCloudPro/AirCityCloudServer/License.lic --enableFileLog=1 --logLevel=Debug --logSubDir=v6.0 --stun=stun:192.168.20.145:3478 --turn=turn:192.168.20.145:3478?transport=udp --turn
Username=freedo --turnPassword=freedo --iceTransportPolicy=relay
freedo     59386       1  0 14:01 pts/2    00:00:00 mono /home/freedo/DTS/AirCityCloudPro/NodeService/NodeService.exe --ip=192.168.20.145 --port=8087 --dataRoot=/home/freedo/DTS/AirCityCloudPro/NodeService/Node
ServiceDataRoot
freedo     59698   48394  0 14:08 pts/1    00:00:00 grep AirCityCloudPro
```

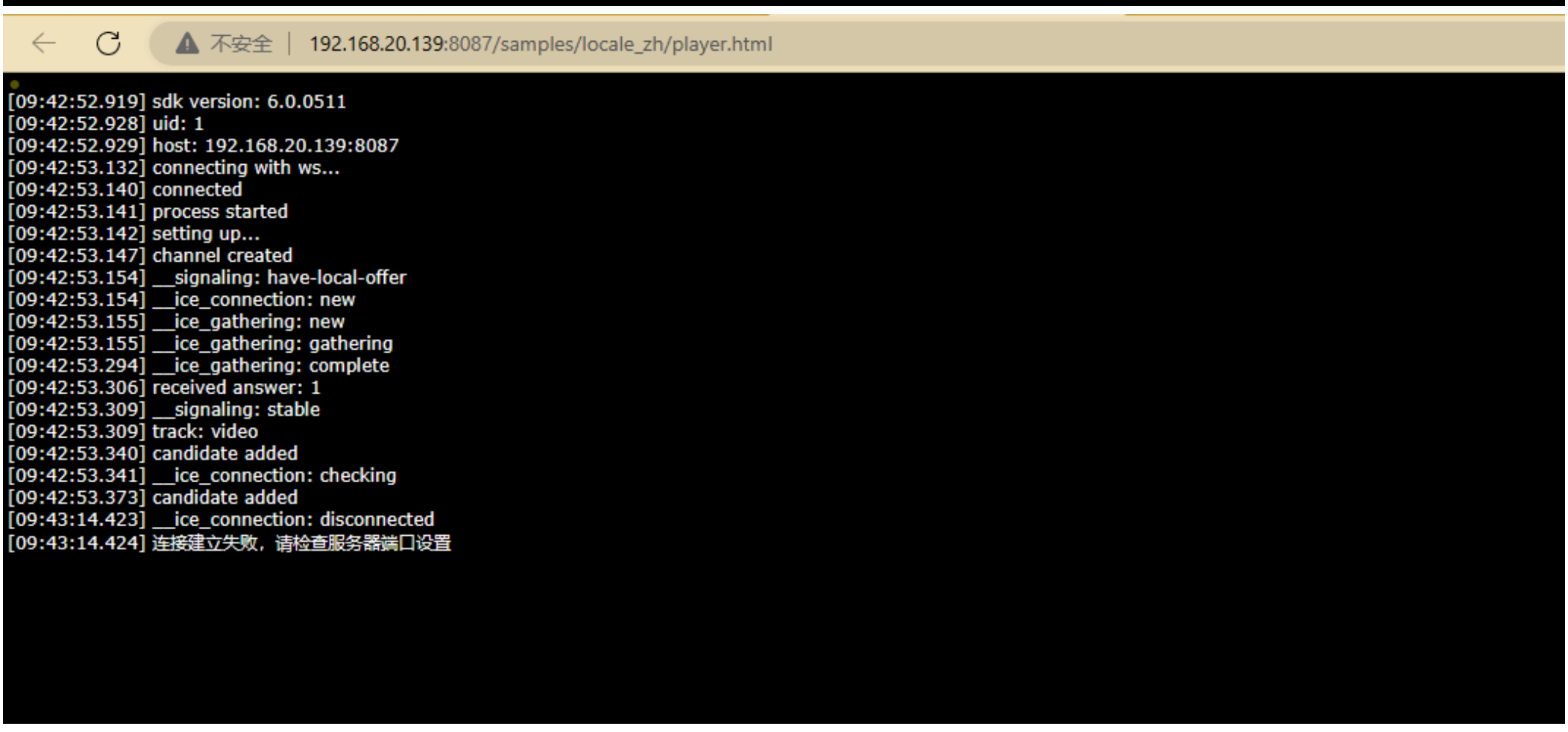
2. 检查系统防火墙是否关闭，或者放行 cloud api 的端口，默认 8087/tcp

```
[...01 AirCityCloudPro]$ sudo systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
   Active: active (running) since Tue 2023-05-09 09:46:41 CST; 2 days ago
     Docs: man:firewalld(1)
 Main PID: 34937 (firewalld)
    Tasks: 2
   Memory: 24.1M
   CGroup: /system.slice/firewalld.service
           └─34937 /usr/bin/python3 /usr/sbin/firewalld --nofork --nopid

5月 09 09:46:48 kylinV10Sp2Server-01 firewalld[34937]: WARNING: COMMAND_FAILED: '/usr/sbin/iptables -w10 -t filter -X DOCKER' failed: iptables: No chain/target/match by that name.
5月 09 09:46:48 kylinV10Sp2Server-01 firewalld[34937]: WARNING: COMMAND_FAILED: '/usr/sbin/iptables -w10 -t filter -F DOCKER-ISOLATION-STAGE-1' failed: iptables: No chain/target/match by that name.
5月 09 09:46:48 kylinV10Sp2Server-01 firewalld[34937]: WARNING: COMMAND_FAILED: '/usr/sbin/iptables -w10 -t filter -X DOCKER-ISOLATION-STAGE-1' failed: iptables: No chain/target/match by that name.
5月 09 09:46:48 kylinV10Sp2Server-01 firewalld[34937]: WARNING: COMMAND_FAILED: '/usr/sbin/iptables -w10 -t filter -F DOCKER-ISOLATION-STAGE-2' failed: iptables: No chain/target/match by that name.
5月 09 09:46:48 kylinV10Sp2Server-01 firewalld[34937]: WARNING: COMMAND_FAILED: '/usr/sbin/iptables -w10 -t filter -X DOCKER-ISOLATION-STAGE-2' failed: iptables: No chain/target/match by that name.
5月 09 09:46:48 kylinV10Sp2Server-01 firewalld[34937]: WARNING: COMMAND_FAILED: '/usr/sbin/iptables -w10 -t filter -F DOCKER-ISOLATION' failed: iptables: No chain/target/match by that name.
5月 09 09:46:48 kylinV10Sp2Server-01 firewalld[34937]: WARNING: COMMAND_FAILED: '/usr/sbin/iptables -w10 -t filter -X DOCKER-ISOLATION' failed: iptables: No chain/target/match by that name.
5月 09 09:46:48 kylinV10Sp2Server-01 firewalld[34937]: WARNING: COMMAND_FAILED: '/usr/sbin/iptables -w10 -D FORWARD -i docker0 -o docker0 -j DROP' failed: iptables: Bad rule (does a matching rule exist in that>
5月 09 10:46:18 kylinV10Sp2Server-01 firewalld[34937]: WARNING: COMMAND_FAILED: '/usr/sbin/iptables -w10 -D FORWARD -i docker0 -o docker0 -j DROP' failed: iptables: Bad rule (does a matching rule exist in that>
5月 09 10:48:19 kylinV10Sp2Server-01 firewalld[34937]: WARNING: COMMAND_FAILED: '/usr/sbin/iptables -w10 -D FORWARD -i docker0 -o docker0 -j DROP' failed: iptables: Bad rule (does a matching rule exist in that>
[...01 AirCityCloudPro]$ sudo firewall-cmd --list-ports
5900-5902/tcp 8087/tcp 3478/tcp 3478/udp
```

3. player **页面正常的情况下无法获取视频流**

问题描述： 卡在以下画面无法获取视频流

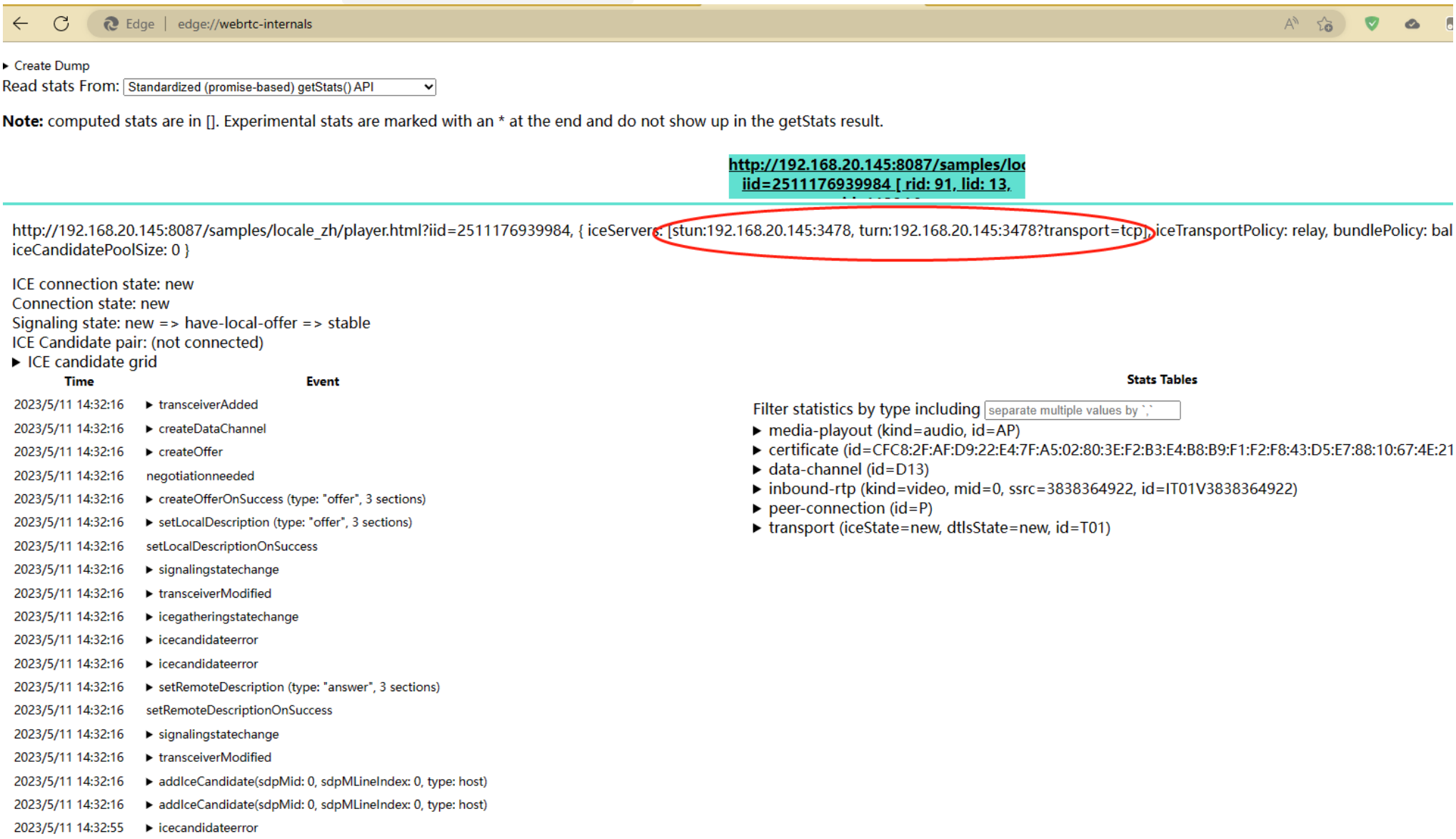




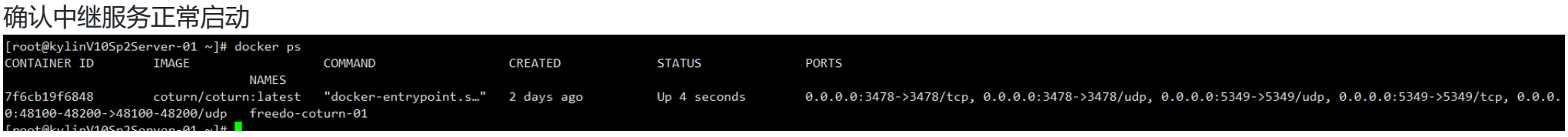
排查步骤：

1. 未使用中继服务的情况下，请确认已完成系统防火墙的配置

2. 通过浏览器打开 webrtc 调试地址 `chrome://webrtc-internals/`，获取 webrtc 调试信息，确认访问中继服务的地址是否正确

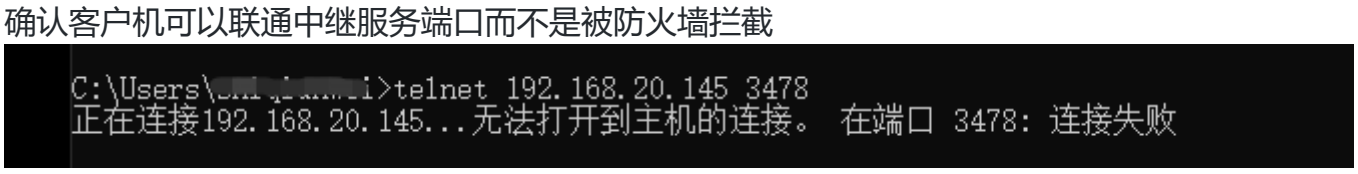


3. 确认中继服务正常启动



4. 确认从服务器本地可以访问映射之后的中继服务端口

5. 确认客户机可以联通中继服务端口而不是被防火墙拦截

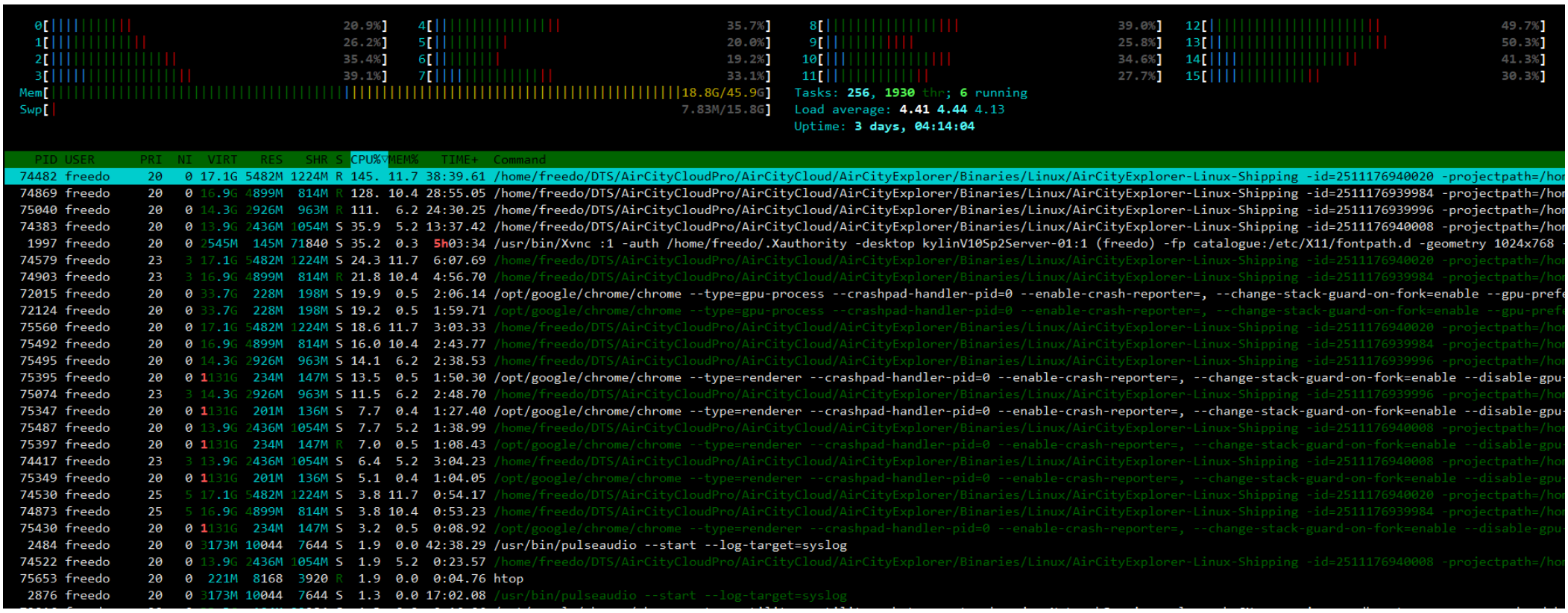


4. 实例卡死、性能问题

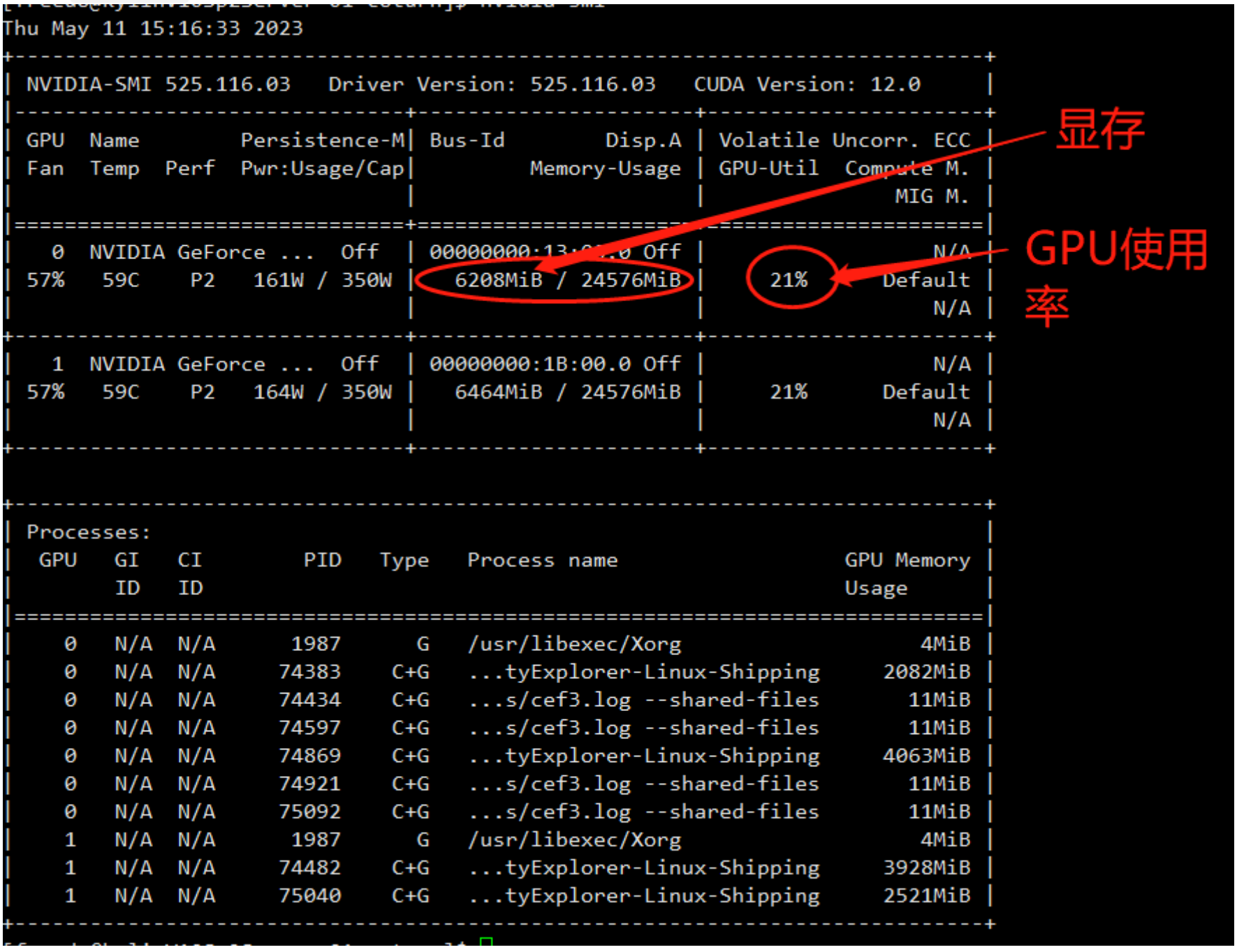
1. 使用`htop`命令排查 cpu，查看各个 cpu 核心使用率

> htop 命令系统没有自带，请单独安装。

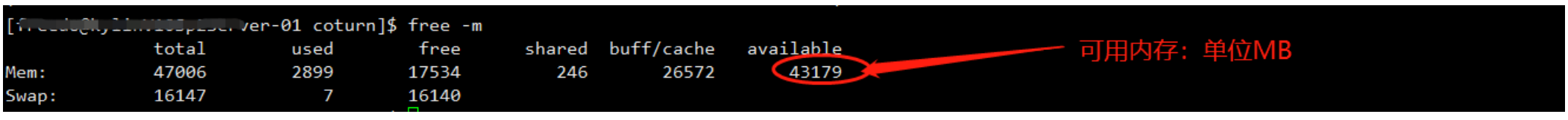
```
wget https://update.cs2c.com.cn/NS/V10/V10SP2/os/adv/lic/base/x86_64/Packages/htop-3.0.5-4.ky10.x86_64.rpm
sudo dnf localinstall ./htop-3.0.5-4.ky10.x86_64.rpm htop
```



2. 使用 `nvidia-smi` 命令查看显卡运行状态



3. 使用 `free -m` 命令查看内存使用



5. 中继服务无法访问问题

> 有些服务器修改了内核参数 net.ipv4.ip_forward=0，关闭了IPv4转发，导致无法访问容器端口，从而无法访问容器部署的中继服务

- 修复方式

1. 修改/etc/sysctl.conf

```
sudo vi /etc/sysctl.conf

net.ipv4.ip_forward=1
```

2. 执行命令重载内核参数

```
sudo sysctl -p
```

3. 重启docker服务

```
sudo systemctl restart docker
```