tcpcopy测试布置——流程整理和问题小结

前期工作

1 步骤

1.1 知识储备

- tcp/ip协议【TCP/IP协议整理——报文/流程解析】
- websocket协议【websocket协议整理】
- tcpcopy功能及架构【*readme of tcpcopy*】

1.2 环境搭建准备

1.2.1 主机

- target server（流量拷贝目标）：10.12.7.9
- online server：10.12.8.30
- assistant server：10.12.7.8

1.2.2 工具

- tcpcopy version1.2.0
- intercept version1.0.0

1.2.3 服务

- 实时短语音私有云服务

服务布置

1 流程步骤

1.1 拉服务包

1. 服务打包在8.13机器上，7.9和8.30都需要布置服务；
2. 使用scp 拉取/推送服务包；
3. 解包。

scp command

```
pullscp (-P <PORT>) <USR>@<IP>:<remote filepath> <local dirpath>
scp -P 5837 xionghao.li@10.12.8.13:/home/xionghao.li/tcpcopy.zip /home


pushscp <local filepath> (-P <PORT>) <USR>@<IP>:<remote dst dirpath>
scp ./tcpcopy.zip -P5837 xionghao.li@101.12.8.13:/home/xionghao.li

-r:
usrroot
```

1.2 运行服务

1. 修改docker-compose.yaml中volumes参数和创建对应路径，保证资源和日志挂载成功；
2. 修改docker-compose.yaml中casr参数拉取所需服务镜像；
3. 修改docker-compose.yaml中ports参数绑定主机端口；（7.9上服务端口为58002，8.30上服务端口为38002）
4. 在docker-compose.yaml的目录下运行docker-compose up -d启动服务容器；
5. docker ps|grep <keyword>检查服务是否启动。

1.3 测试服务连通性

用任意主机跑测试脚本，观察容器内的服务日志，看看是否连通识别服务。

2 问题小结

2.1 在运行服务的过程中，7.9上的服务起不来【阻塞时长：0.5 hours】

原因：7.9这个主机的网络连通性较差，无法连接到镜像仓库服务器上，镜像拉取失败导致服务启动失败。

解决方法：

1. 找到同时连通仓库和7.9的服务器的主机（8.13），docker pull所需要镜像；
2. docker save <image ID> -o <filename>.tar将镜像保存为tar包；
3. 通过scp指令将镜像tar包传至7.9上；
4. docker load -i <image tar>将镜像加载，注意镜像的名字是<none>，与yaml文件内镜像不匹配
5. docker tag <image ID> <name:tag> 修改成对应镜像名
6. 拉起服务

tcpcopy工具布置

1 流程步骤

1.1 tcpcopy环境布置

tcpcopy布置到online server 10.12.8.30上

intercept布置到assistant server 10.12.7.8上

1.2 设置target server路由

set route

```
route add -net 10.12.7.0 netmask 255.255.255.0 -gw 10.12.7.8
```

将target server(7.9)上所有发送到7.0网段上的包都路由到assistan server(7.8)上

1.3 配置/编译/启动intercept

1. assistant server需要关闭路由转发功能，echo 0 > /proc/sys/net/ipv4/ip_forward；
2. 在intercept路径下,顺序执行./configure <option> →make→ make clean，编译intercept
3. route指令查看路由表，找target server的ip的对应项的网卡；<pcap filter statment> 填tcp and src port 58002即可抓取target server(7.9:58002)的tcp响应包

run intercept

```
./usr/local/intercept/sbin/intercept -i <> -F '<pcap filter statment>' -d
```

1.4 配置/编译/启动tcpcopy

1. 在tcpcopy路径下，按序执行./configure <option> →make→ make clean，编译tcpcopy；
2. 指定网卡同1.3.3，<transfer>格式为 localServerPort-targetServerIP:targetServerPort，拷贝得到的包源IP修改为<fakedIP>，以此设置target server的路由；
3. tcpcopy运行必须在intercept之后，否则一定失败

run tcpcopy

```
./usr/local/intercept/sbin/tcpcopy (-i <>) -x <transfer>  -s <assistantServerIP> -c <fakedIP> -d
```

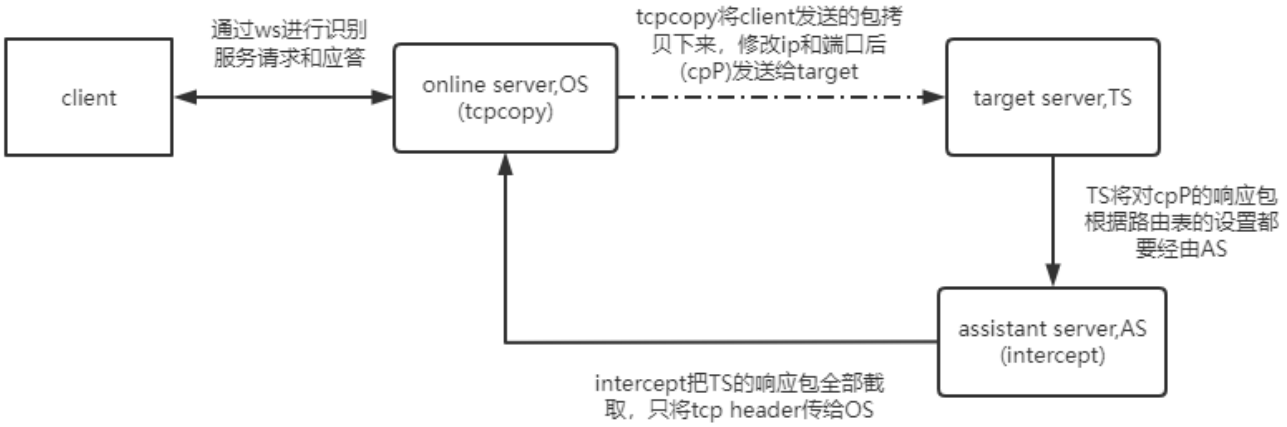
1.5 测试tcpcopy是否生效

1. 进入7.9和8.30的服务容器，追踪日志
2. 执行脚本向online server(8.30)申请识别服务
3. 如果online server 和target server都回应了识别服务请求说明tcpcopy工具生效

2 问题小结

2.1 测试tcpcopy生效时，发现没有生效，target server没有识别脚本进行服务【阻塞时长：3 days】

排查过程：



- ping <IP>确认连通性，发现整个链路都已连通；（IP链路正常）
- nc -vz <IP:port>，ssh -v -p <port> <usr>@<IP>指令，检测端口8.30/38002，7.9/58002，发现都能正常接收数据；（端口正常）
- 布置websocket通讯项目到OS和TS上，发现tcpcopy能正常使用，websocket协议也能够兼容，只是在服务上不能正常使用；（tcpcopy正常）
- 执行脚本，TS和OS的服务都能正常运行；（服务正常）
- 最后通过tcpdump抓包发现，tcpdump -i <dev> port <PORT> 抓网口能抓到client➡OS的数据包，但是只抓port时抓不到。tcpcopy默认方法里不要求提供网卡信息，即不抓网卡，由此推测是由于某原因导致从tcpcopy无法从IP层抓到client➡OS的casr服务的数据包。（特定环境下，抓包异常）
- 调整编译运行参数（1.4.2中的-i参数，默认是关闭的，需要安装pcap的资源库，通过./config --pcap-capture激活），让tcpcopy能够在数据链路层抓包，发现问题解决。

原因：由于某原因导致从tcpcopy无法从IP层抓到client➡OS的casr服务的数据包，猜测是虚拟主机的环境影响

解决方法：激活tcpcopy的链路层抓包功能（1.4.2中的-i参数，默认关闭，需要安装pcap的资源库，通过./config --pcap-capture激活）、

反思：

- 不能够结合实际灵活运用网络协议知识，无法规划出具体的排查流程；
- 对网络工具（特别是tcpdump）不熟悉，每个疑点的排查进度缓慢；
- 过分相信日志，定位问题时出错（认为intercept失效）。