《现代交换原理》实验报告

实验名称 \_\_\_\_\_\_拨打 SIP 电话 \_\_\_\_\_\_

班 级 \_\_\_\_\_\_\_2017211305\_\_\_\_\_\_\_

姓 名 \_\_\_\_\_\_于海鑫\_田静悦\_\_\_\_\_\_

指导教师 \_\_\_\_\_\_\_\_\_丁玉荣\_\_\_\_\_\_\_\_\_

# 实验目的

结合课堂所讲的SIP信令工作流程，对软电话呼叫的信令进行抓包分析，理解VoIP呼叫中会话信令、媒体协商信令的作用，加深对VoIP的理解。

# 实验内容和实验步骤

## SIP 服务端软件的安装

本次我们选用的 SIP 服务端为 Flexisip，其提供了较为简单的搭建 SIP 服务器的方式，而且其配套客户端 UI 比较好看。

### 准备工作

本次实验的服务端我们选择在阿里云主机上进行搭建，操作系统为 Ubuntu 18.04 LTS。

### 软件安装

首先增加对应的软件源。

在 /etc/apt/source.list上增加如下代码：

# For Ubuntu 18.04 LTS

deb [arch=amd64] http://linphone.org/snapshots/ubuntu bionic stable beta # alpha

之后需要信任软件源，添加 PGP Key

wget https://www.linphone.org/snapshots/ubuntu/pubkey.gpg -O - | sudo apt-key add -

之后即可直接安装

sudo apt update

sudo apt install bc-flexisip

### 软件配置

**生成默认配置文件**

使用如下指令生成默认配置文件，以 root 权限执行如下指令：

/opt/belledonne-communications/bin/flexisip --dump-default all > /etc/flexisip/flexisip.conf

**编写配置文件**

之后我们打开 /etc/flexisip/flexisip.conf，删除全部内容，把如下配置复制到配置文件中：

[global]

debug=true

aliases=sip.name1e5s.fun

[module::Registrar]

enabled=true

reg-domains=sip.name1e5s.fun

db-implementation=internal

[module::Authentication]

enabled=true

auth-domains=sip.name1e5s.fun

db-implementation=file

datasource=/etc/flexisip/users.db.txt

这里面 sip.name1e5s.fun 为个人域名。

之后创建 /etc/flexisip/users.db.txt，输入账户配置：

version:1

test@sip.name1e5s.fun clrtxt:test ;

test2@sip.name1e5s.fun clrtxt:test ;

其中 test 为用户 ID， clrtxt: 后，空格前的为密码。

### 运行

以 root 权限执行如下指令：

/opt/belledonne-communications/bin/flexisip

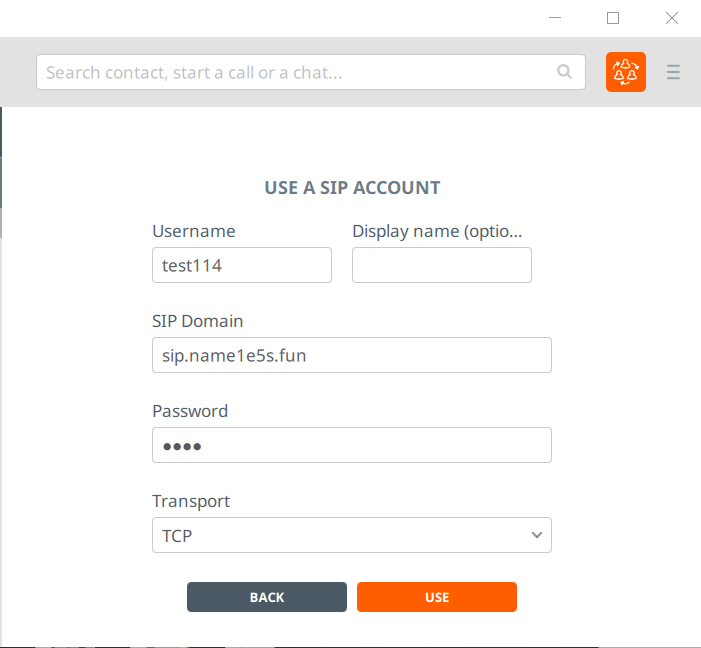
至此，服务端配置成功，可以开始进行通话。

## SIP 客户端软件的安装

### 软件安装

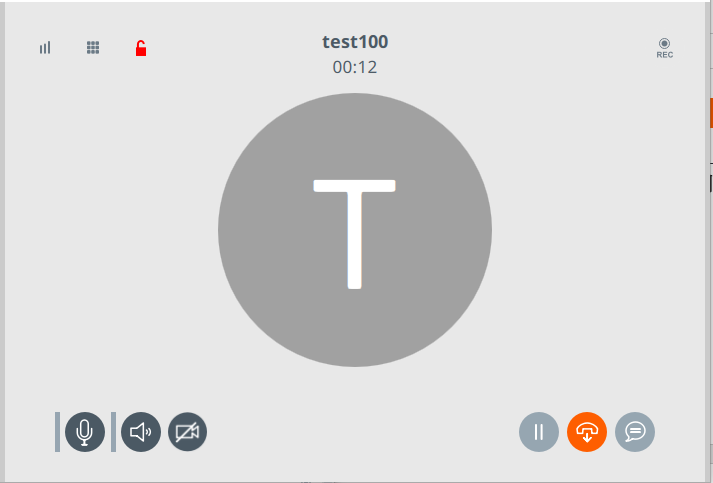
这里我们选用的 SIP 客户端为 Linphone，其特点为 UI 比较现代，而且为 GPLv3 授权下 的 Free Software。其安装过程十分简单，下载后一路下一步即可在此不再过多介绍。

### 用户注册

打开软件后，我们创建新的 SIP 账号，按照如下方式输入账号信息即可：

之后选择 “USE”，并在左上角选择这一用户。当注意到用户名前出现绿点时，表示与服务端链接成功，可以开始通话。

### 通话测试

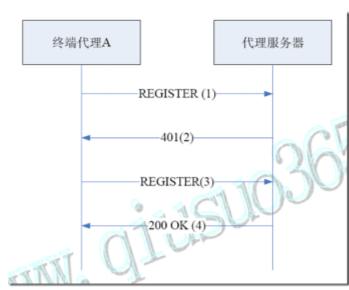
在最上方输入想要拨打的账号，即可开始拨打电话。通话页面如下：

Linphone 支持视频通话，但是碍于带宽等原因，我们没有进行测试。

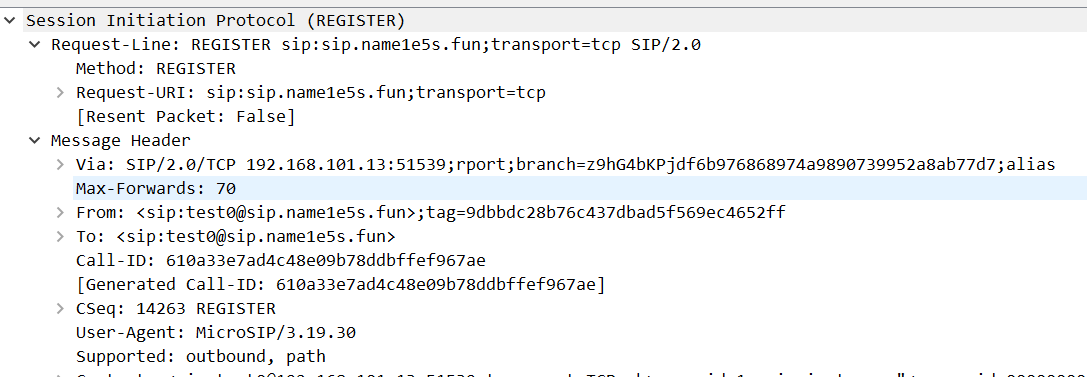
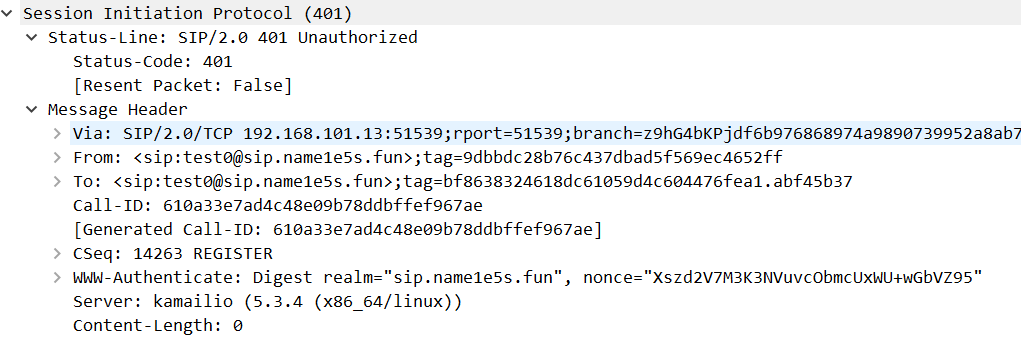
我们也使用 MicroSIP 以确认服务器的兼容性，其安装使用在此不表。

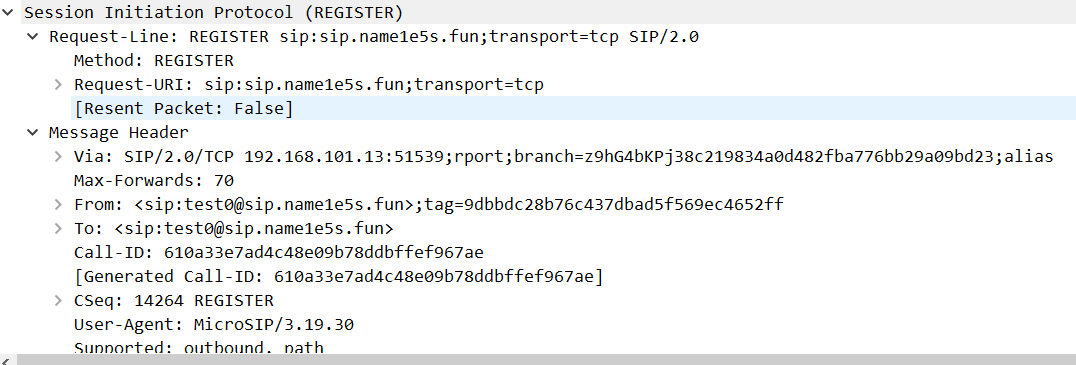
## 抓包分析 SIP 流程

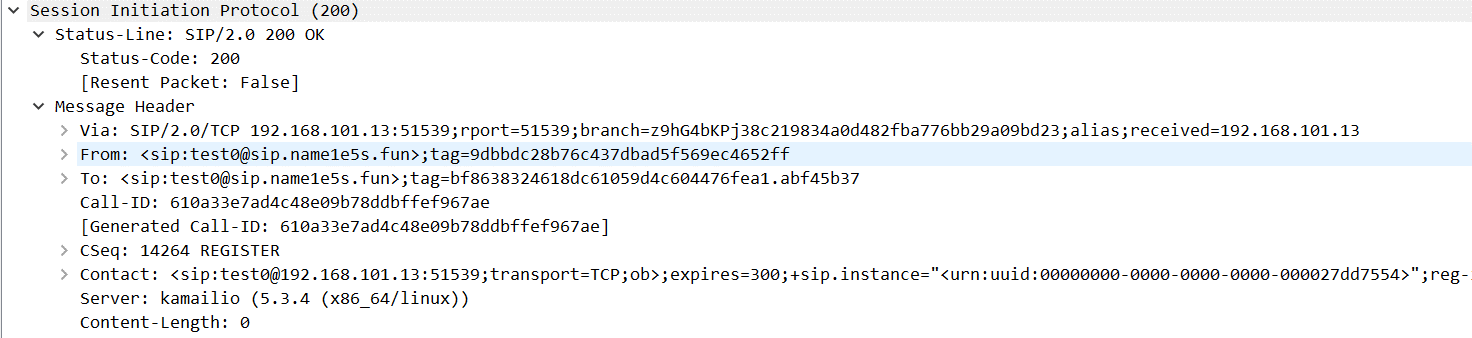
### 注册



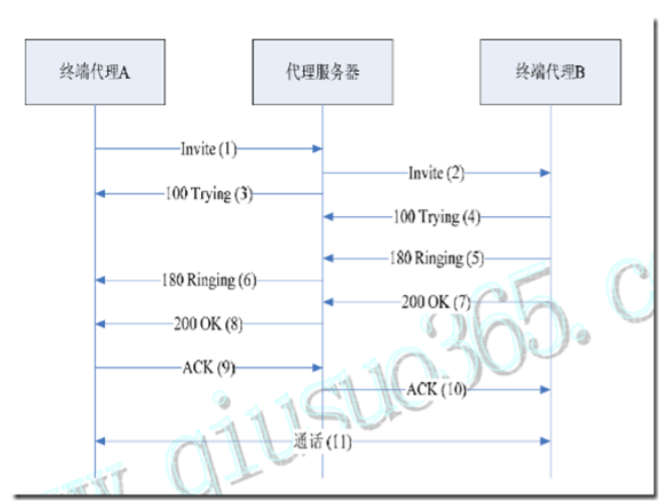
注册流程如下：

1. 终端向服务器发起 REGISTER 请求
2. 服务器返回 401，要求进行安全认证
3. 服务端按照要求加密用户信息，重新 REGISTER

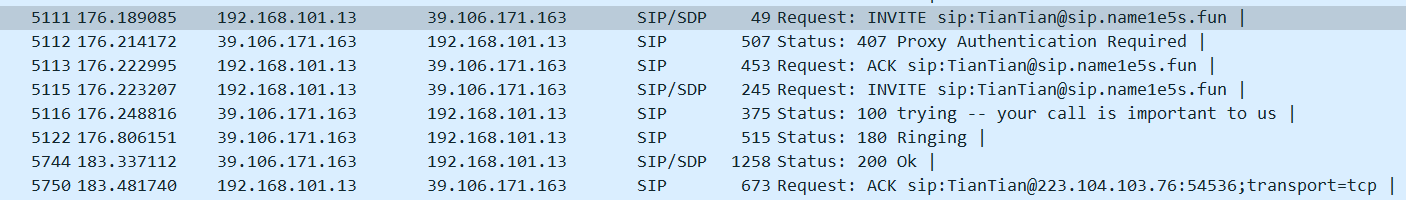


1. 服务器进行认证，返回 200

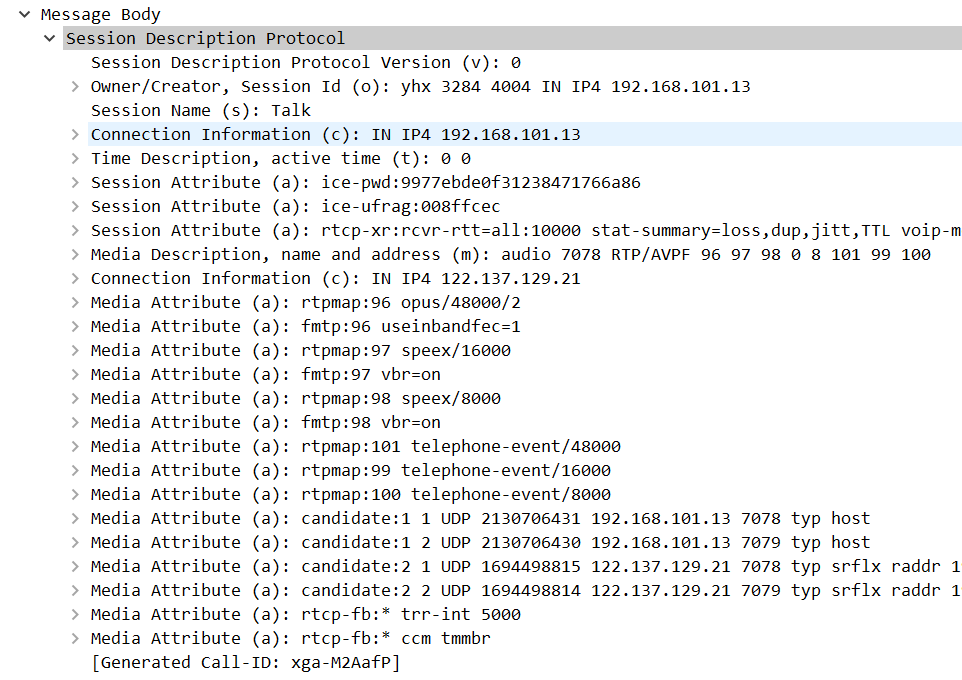
### 通话



#### A用户视角

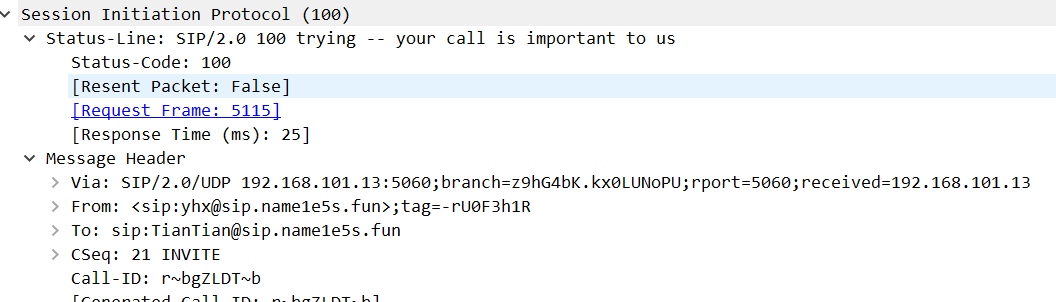


1. 发起 INVITE 信息

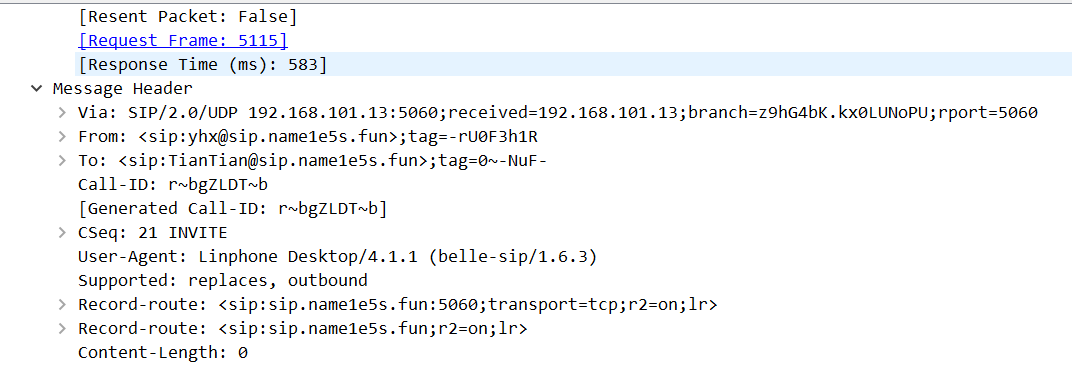


INVITE 信息中包含了本次通话的详细信息，如上图所示。由此可知我们的通话为语音通话，协议为 RTP。

1. 服务端回复 TRYING



1. 服务端回复 RINGING

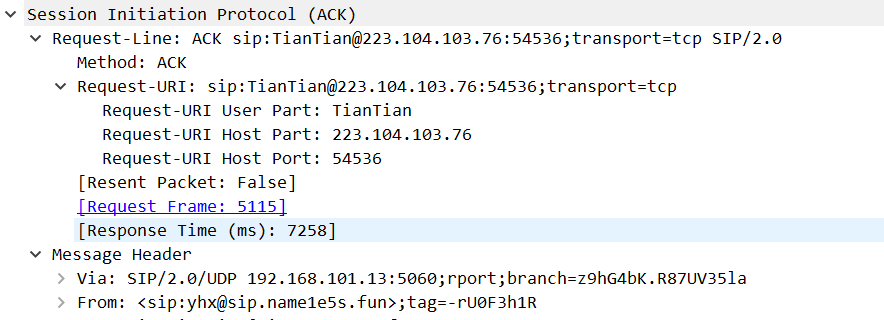


此时，我们得知对面的客户端开始响铃。

1. 服务端回复 OK

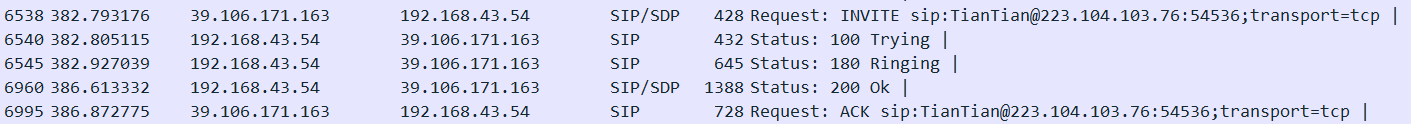
此时，对面接起电话。

1. 回应 ACK

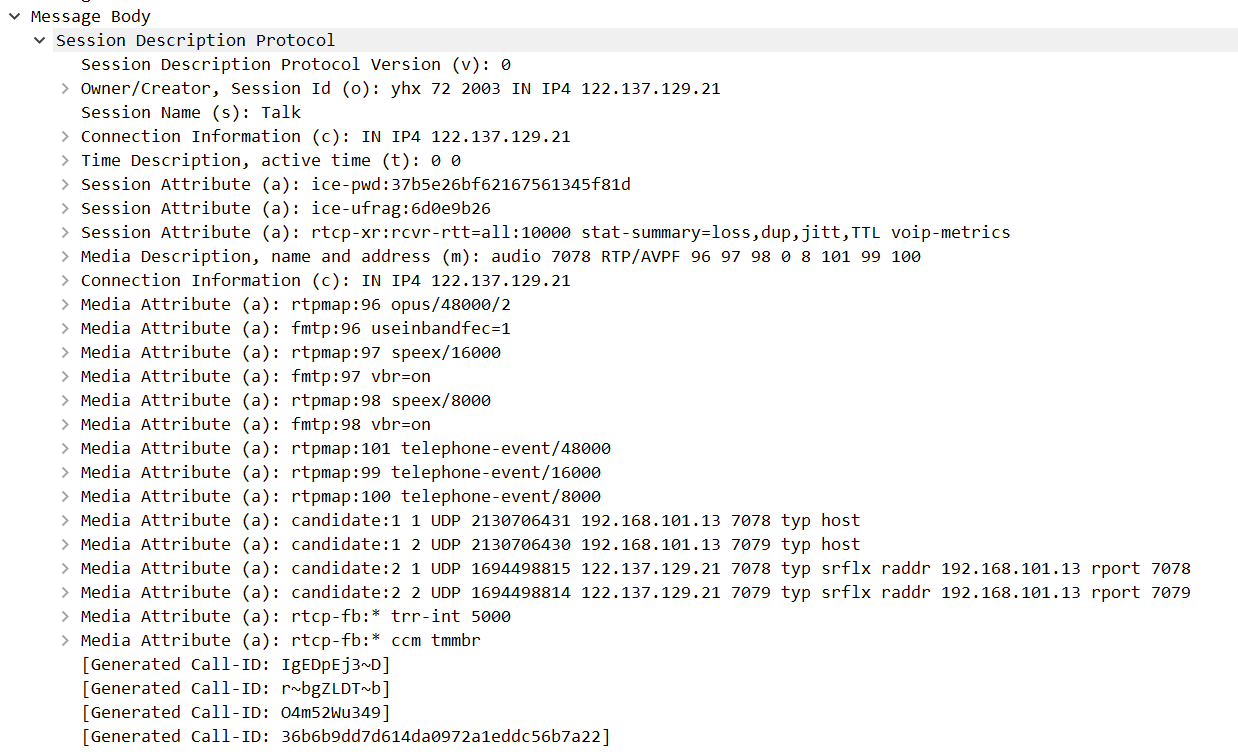


此时，我们确认了我们已经得知消息。

#### B 用户视角

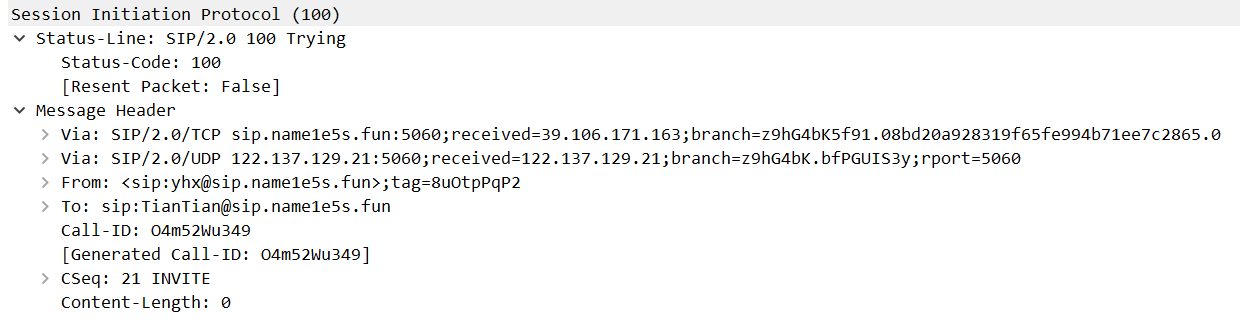


1. 收到 INVITE 信息



如图，我们收到了A发来的请求，包含了与A相同的通话信息。

1. 服务端回复 TRYING



1. 服务端回复 RINGING



开始响铃。

1. 服务端回复 OK



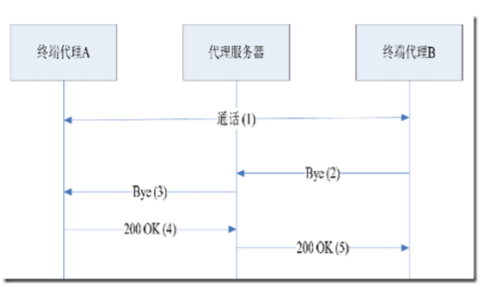
接起电话。

1. 回应 ACK



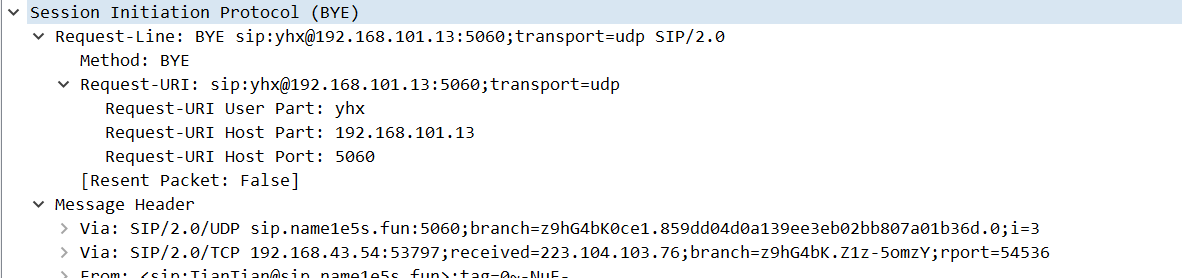
此时，我们收到ACK,开始通话。

### 挂断

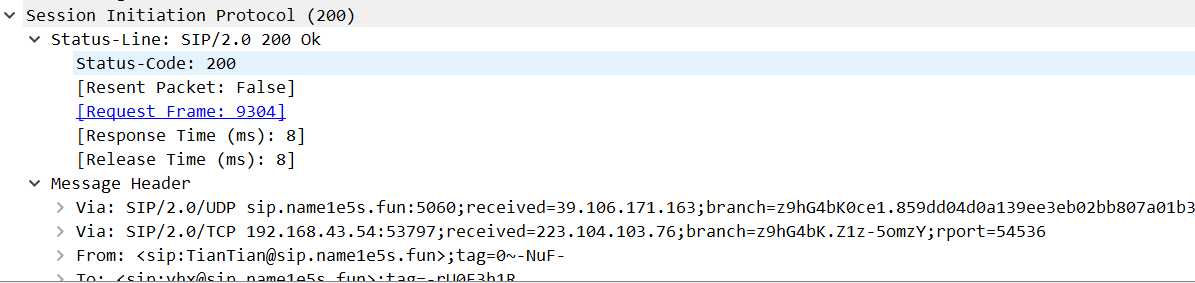


#### A 用户视角

### 收到挂断要求



### 回复 OK

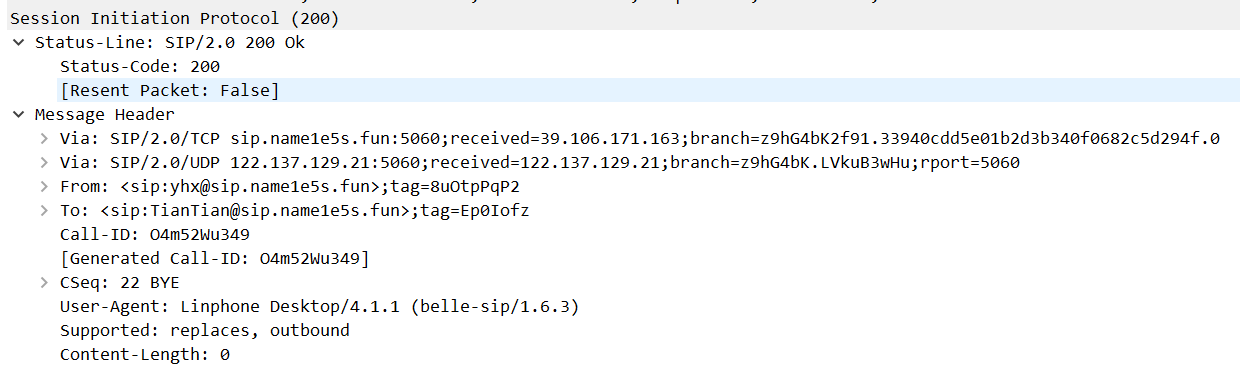


#### B 用户视角

### 发起挂断要求

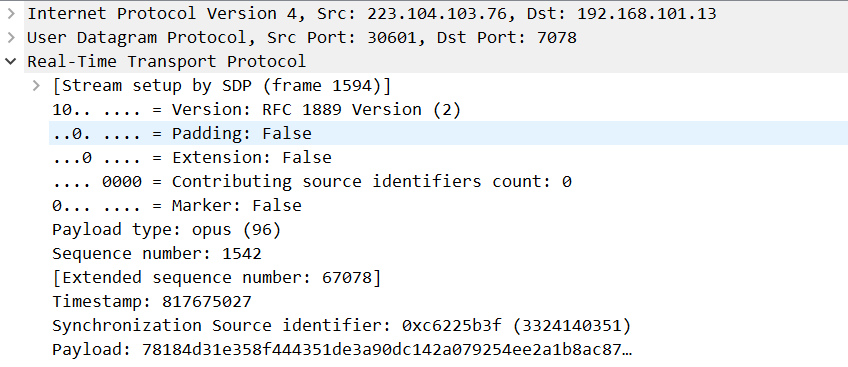


### 收到 OK



此时，挂断成功。

## 通话过程中

通话使用 RTP 协议，其中一个包如下：

可见其为客户端间的直接通信，与服务端没有关系。

# 实验总结与心得

#### 于海鑫

本次实验我们把大部分时间花在了搭建 SIP 服务器上，因为之前分别尝试了 GNU SIP Witch 等软件，但是经过测试都存在各种各样的问题，最后选择了一个比较完善的 Flexisip。我们在实际部署时候仍然发现了一些心得问题，后续有时间将尝试看能否帮助开源社区修复掉我们遇到的问题。

通过本次实验，我对 SIP 协议的通信方式有了较为深入的了解。

#### 田静悦

本次实验过程中，遇到了拨打电话但是听不到声音的情况，经过排查，我们怀疑是协议不统一的原因，我使用了TCP,对端使用了UDP,在统一协议之后，通话正常进行。此次实验通过抓包分析，我深入理解了拨打SIP电话的过程，看到了过程中使用的各类数据包，明白了包的内部参数及其意义。实验过程中也体会到了小组协作的快乐。通过本次实验对SIP协议有了更深入的认识。