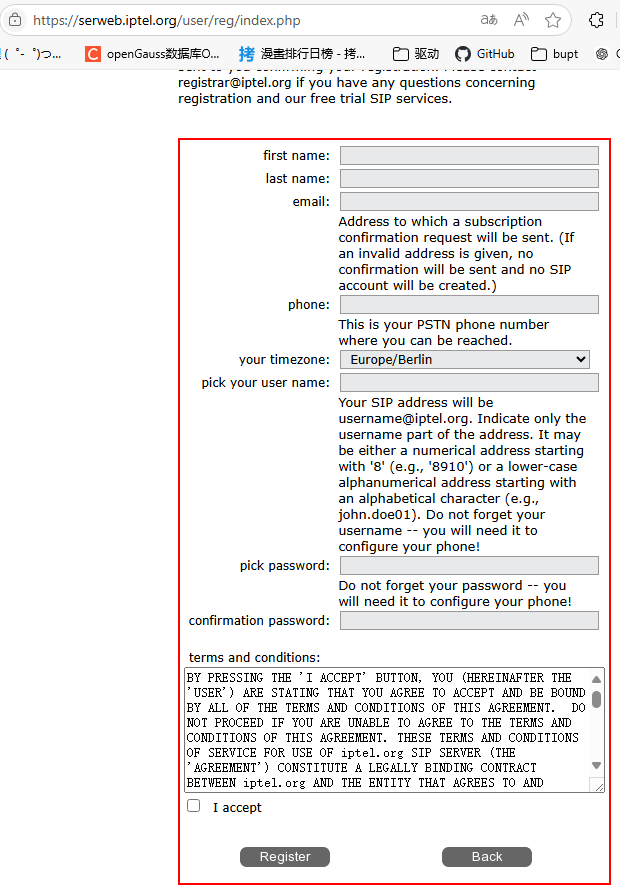
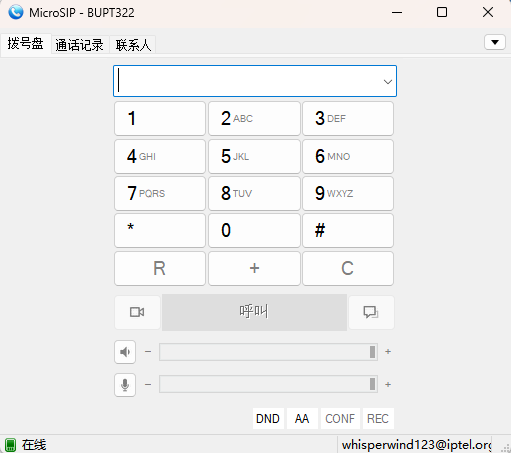
## 一、实验目的

本实验需要实验者通过将课堂中所讲的SIP信令工作流程，对SIP电话呼叫的信令进行抓包分析，理解VoIP呼叫中的会话信令、媒体协商信令的作用，从而加深对VoIP的理解。

## 二、实验内容与实验步骤

### 下载软件并注册SIP账号



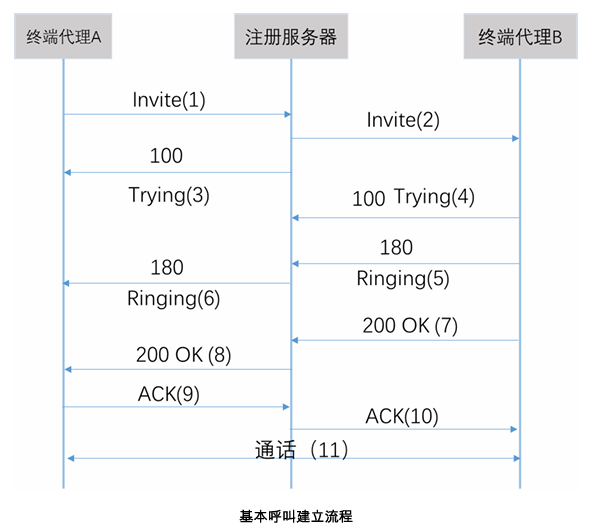
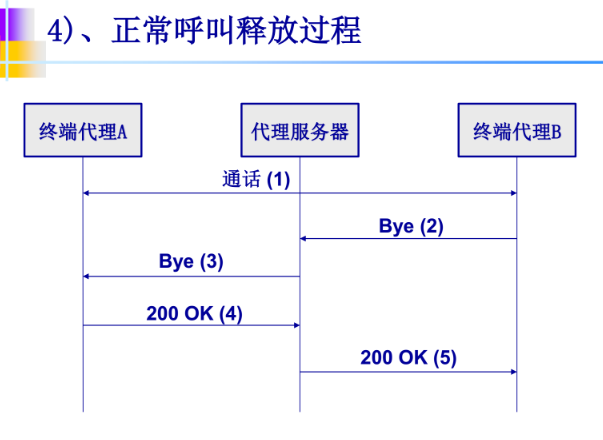
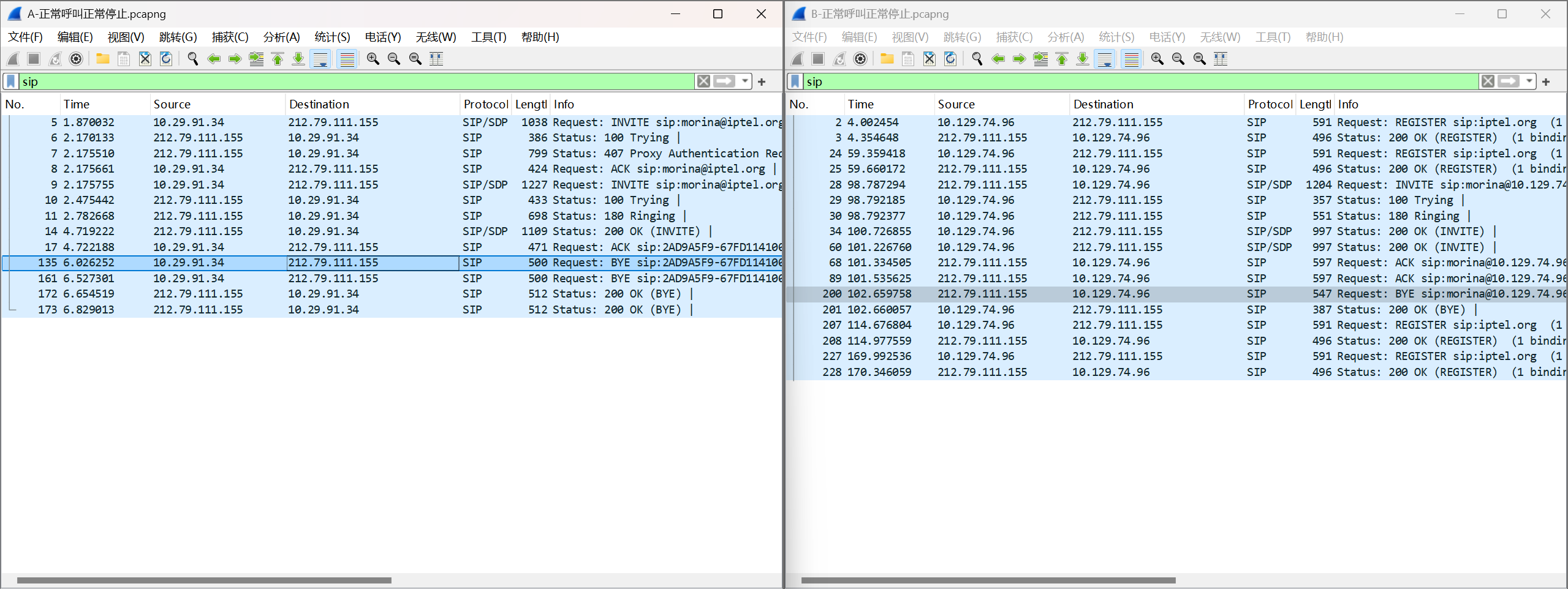
* 首先，我们进入网站<https://serweb.iptel.org/user/reg/index.php>注册一个SIP账号。  
  本次实验中，本组三名成员总共注册了。分别是：  
  [morina@iptel.org](mailto:morina@iptel.org) [whisperwind123@iptel.org](mailto:whisperwind123@iptel.org)
* 接着我们进行软件的下载，打开网站<https://www.microsip.org/downloads>进行软件下载，下载完成后单击可运行文件进行安装  
  
* 安装完成后，单击添加账户，将注册的帐号添加至MicroSIP中。  
  
* 添加账户完成后，运行MicroSIP，即可开始进行实验。  
  

### 拨打电话并抓包分析

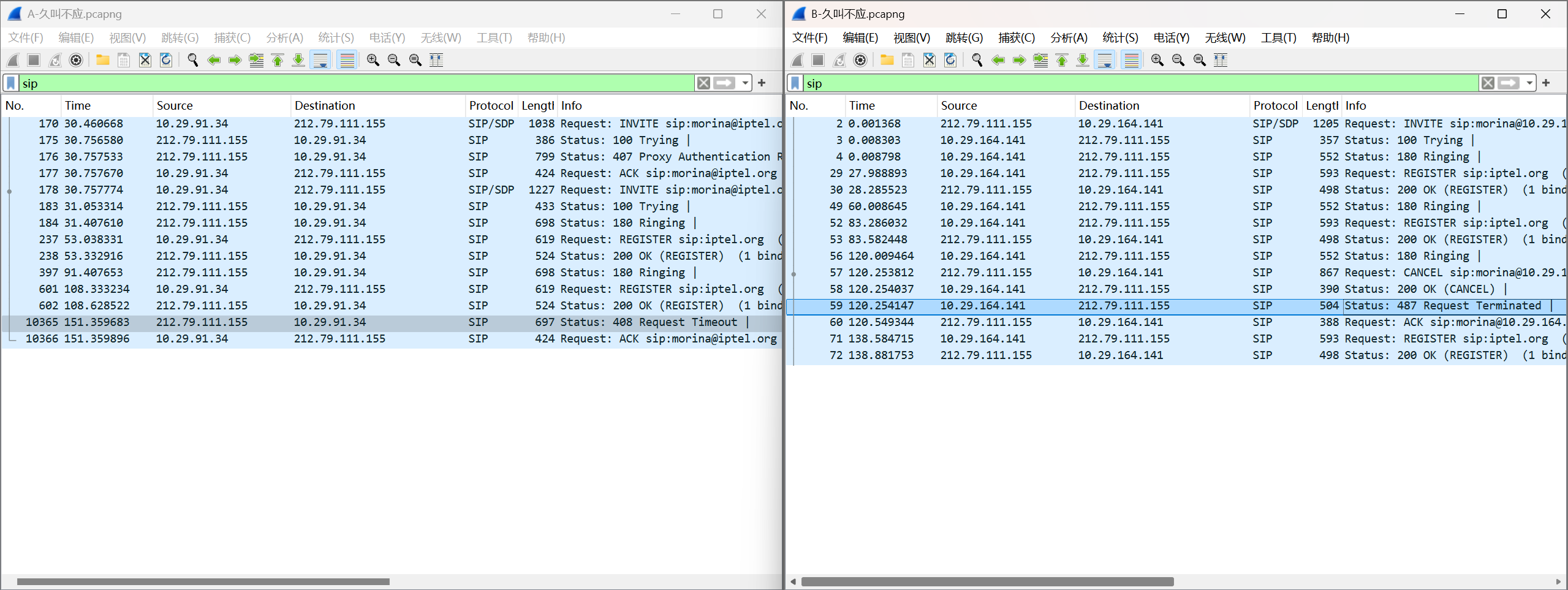
本实验中，我们主要对以下场景进行模拟并抓包，以进行分析：

* 正常呼叫SIP流
* 被叫久叫不应
* 主叫主动挂断
* 被叫超时无响应
* 被叫拒接

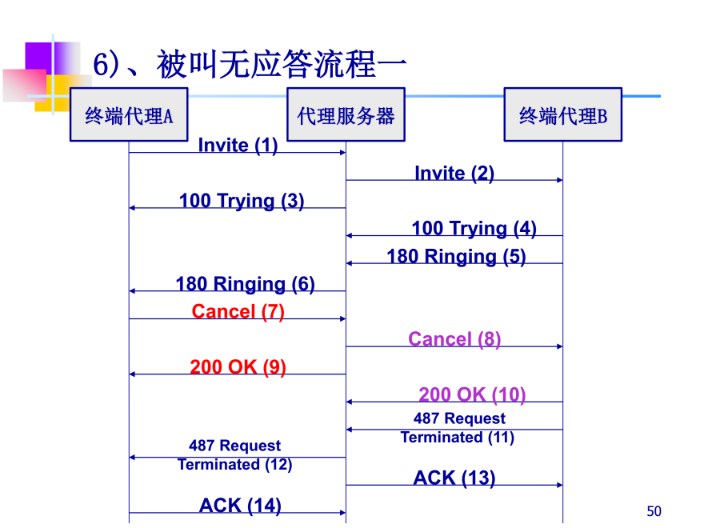
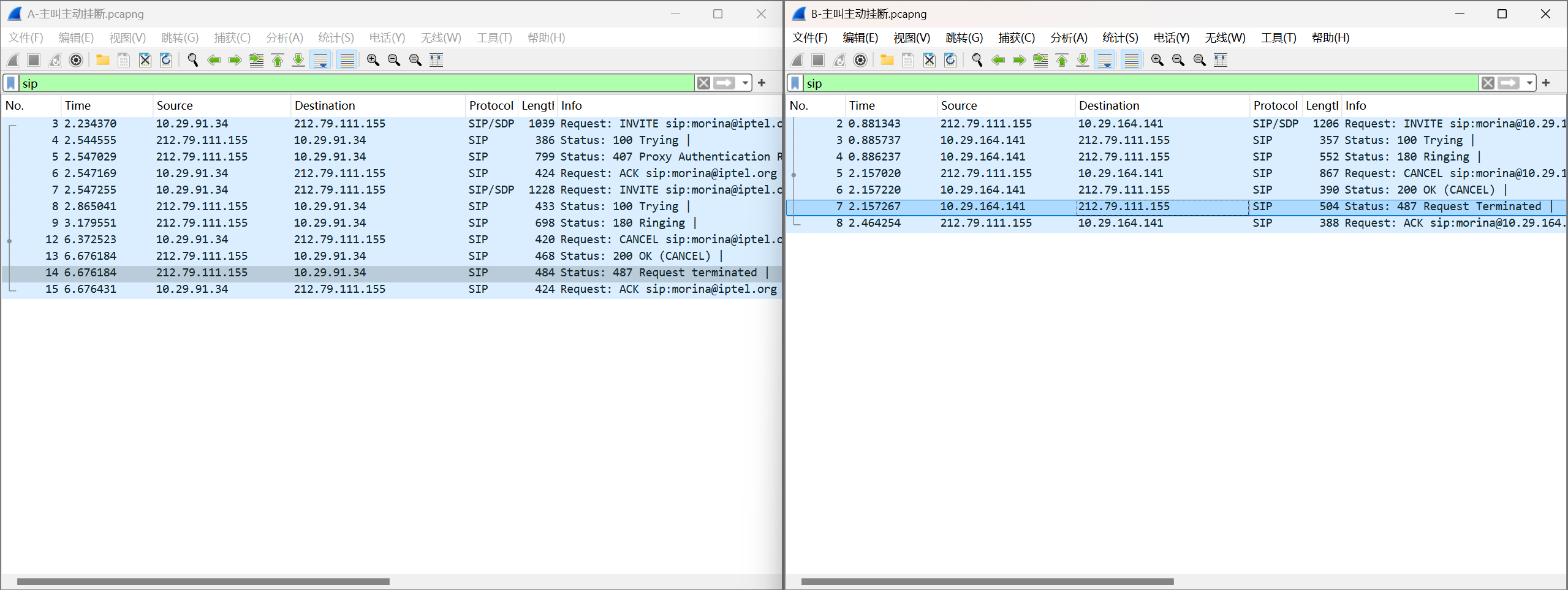
#### 正常呼叫SIP流

* 基本呼叫建立过程
  + 建立过程中的SIP包传输流程如下图所示：  
    
  + 其中，终端代理A（以下简称**主叫**），向代理服务器（以下简称**服务器**）首先发送呼叫建立请求（INVITE），然后服务器同时向主叫发送100 Trying包与407 Proxy Authentication Required，要求主叫进行安全认证，主叫发回ACK进行确认。接着，主叫向服务器发送加上终端代力安全认证信息的呼叫建立请求（INVITE），服务器接收后向终端代理B（以下简称为**被叫**）发送呼叫建立请求，并给主叫发回100 Trying表示正在尝试建立呼叫连接。被叫接收到服务器的呼叫请求后，先后向其发送100 Trying与180 Ringing，表示尝试建立连接并送回铃声。服务器接收到后，将180 Ringing发给主叫送回铃声。当被叫接受通话后，向服务器发送200 OK包表示请求成功，服务器接收到后，向主叫发送200 OK表示请求成功，然后主叫发回ACK进行确认，服务器接收到后向被叫发送ACK进行确认，此时通话建立成功。
* 正常呼叫释放过程
  + 释放过程中的SIP包流程图如下所示：  
    
  + 首先，终端代理B（以下简称为**挂断方**）向服务器发出Bye包，服务器接收到后，向终端代理A（以下简称为**被挂断方**）发出SIP包头中From字段与To字段相同的Bye包提出结束通话请求。被挂断方接收到后，向服务器发送 200 OK表示释放成功，服务器接收到后，向挂断方发送 200 OK进行通话释放。
* wireshark抓包：  
  

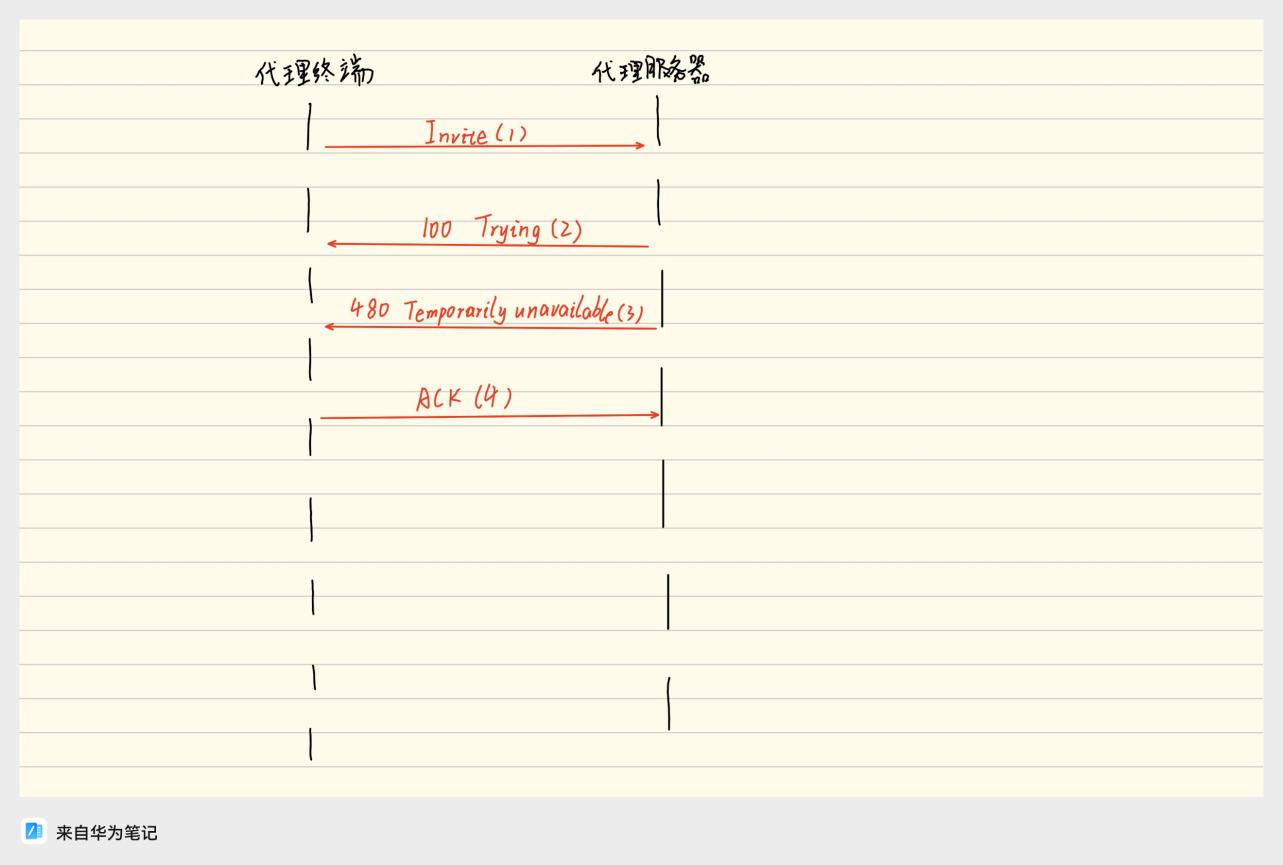
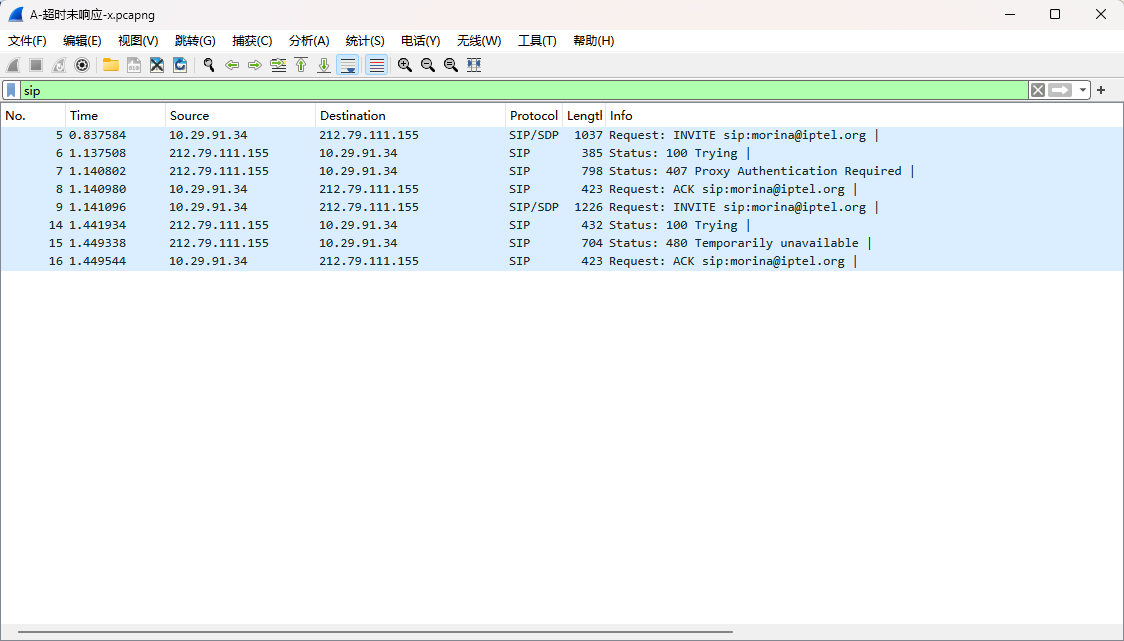
#### 被叫久叫不应

* SIP包流程图如下：  
  
* 根据图片中的流程描述，**主叫**（终端代理A）首先向**服务器**（代理服务器）发送呼叫建立请求INVITE。随后，服务器同时向主叫发送100 Trying和180 Ringing，表示正在尝试建立连接并送回铃声。经过等待时间后，等待时间计时器超时，服务器先向被叫发送Cancel请求，然后向主叫发出408 Request timeout结束呼叫请求。被叫接收到Cancel包后向服务器发送200 OK包与487 Request Canceled结束通话，服务器向其发送ACK确认收到。然后主叫发送ACK确认收到超时通知。
* wireshark中抓到的包如下所示：  
  

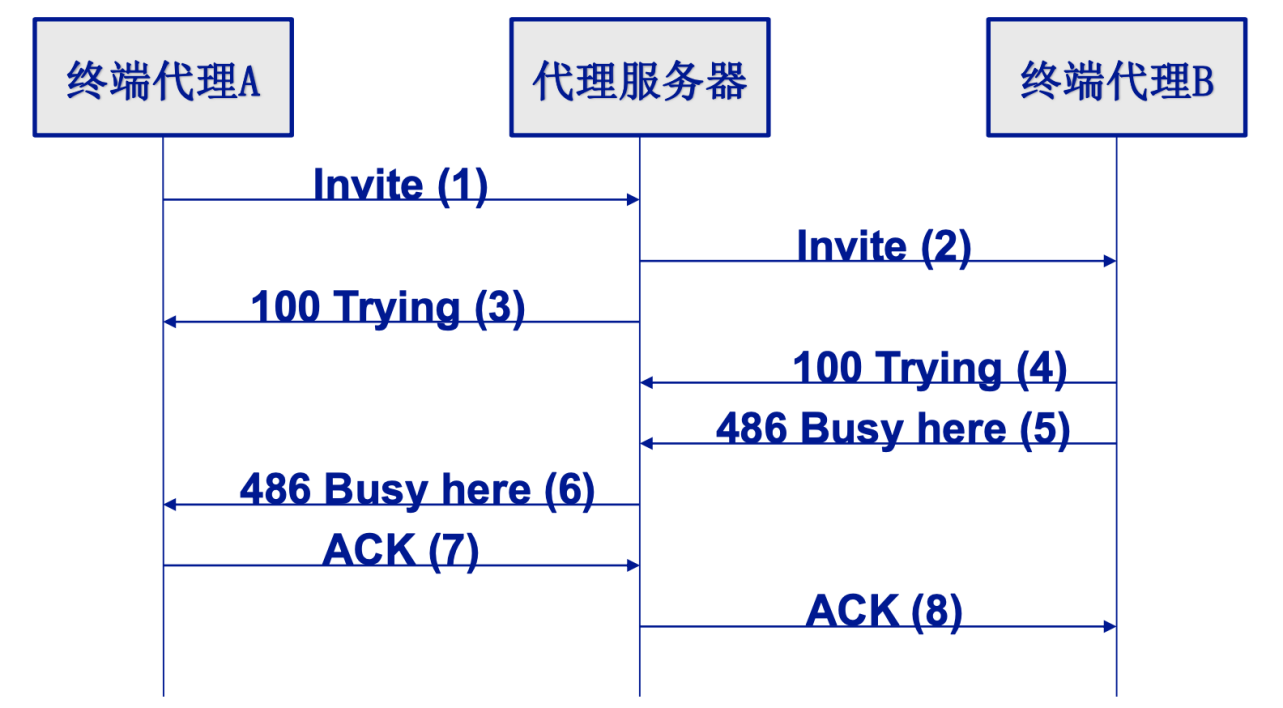
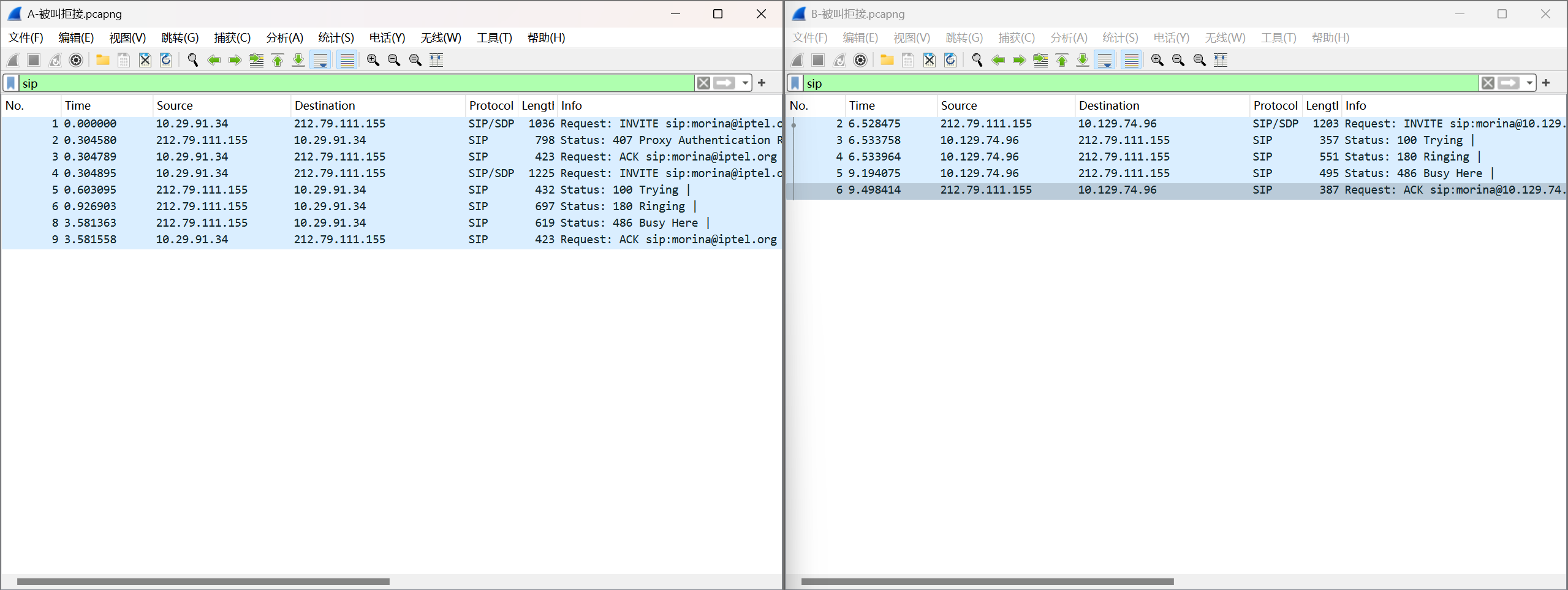
#### 主叫主动挂断

* SIP包流程图如下：  
  
* **主叫**（终端代理A）首先向代理服务器发送INVITE请求发起呼叫，服务器收到后向主叫回复100 Trying响应并转发INVITE至**被叫**（终端代理B）。被叫返回100 Trying和180 Ringing响应（经服务器转发给主叫）。由于被叫无应答，主叫发送CANCEL请求，服务器回复200 OK确认并向被叫转发CANCEL。被叫终止呼叫后返回487 Request Terminated和200 OK（经服务器转发），最终各方通过ACK完成确认。
* wireshark抓包如下：  
  

#### 被叫超时无响应

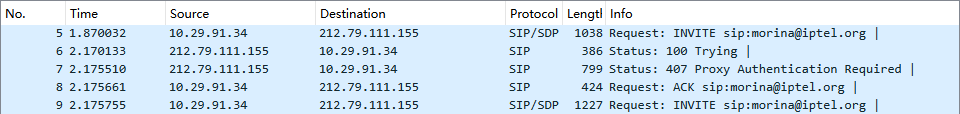
* SIP包流程图如下：  
  
* **主叫**（代理终端）向代理服务器发送INVITE请求发起呼叫，服务器收到后立即返回100 Trying响应表示正在处理。随后服务器判定呼叫暂时不可达，向主叫终端发送480 Temporarily Unavailable响应，主叫终端收到后回复ACK进行确认，最终终止此次呼叫请求。
* wireshark抓包如下（我们通过设置请勿打扰状态模拟繁忙无响应）：  
  

#### 被叫拒接

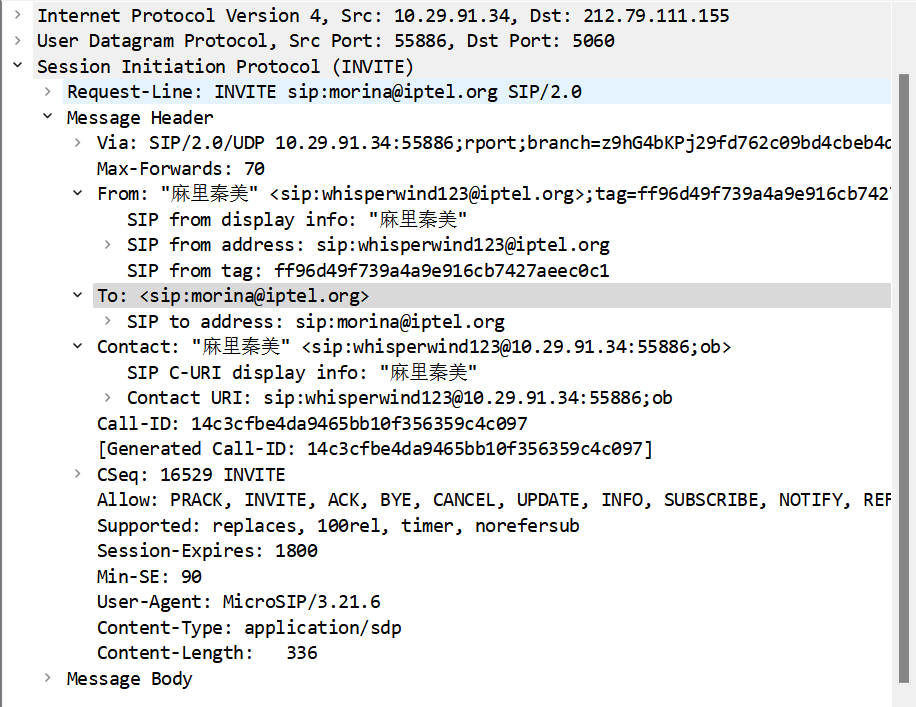
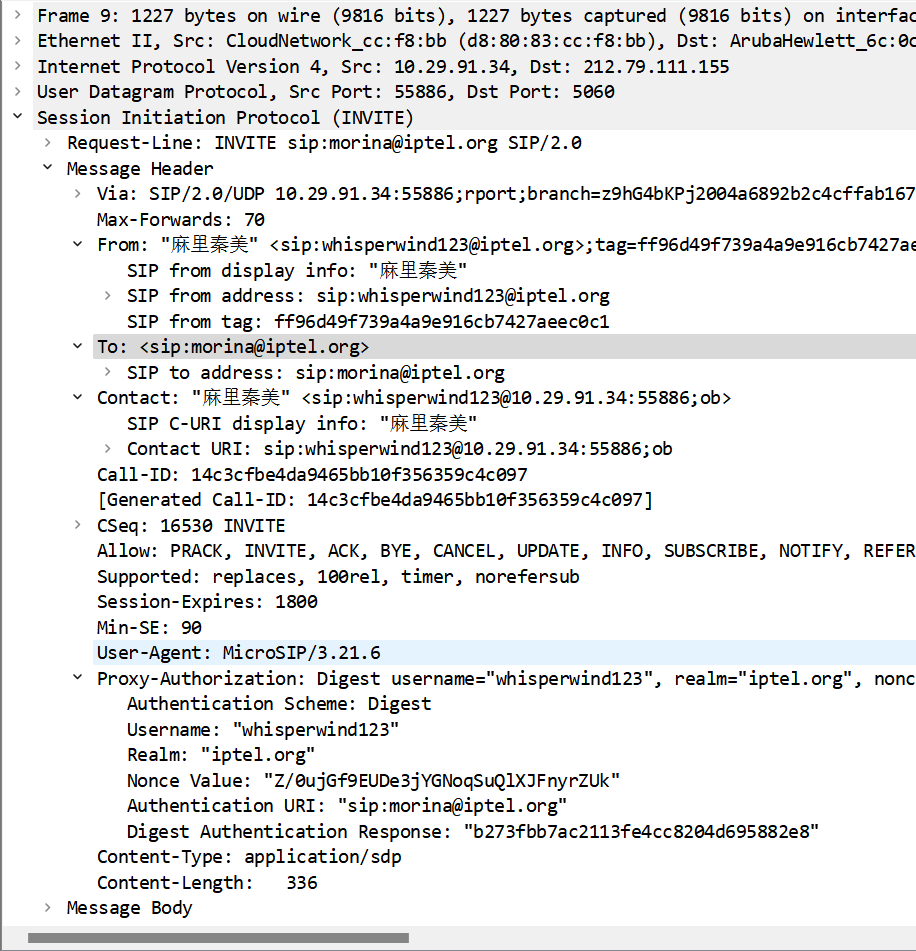
* SIP包流程图如下：  
  
* **主叫**（终端代理A）向代理服务器发送INVITE请求发起呼叫，服务器收到后转发给**被叫**（终端代理B），同时向主叫返回100 Trying响应。被叫收到INVITE后也返回100 Trying，但因线路忙直接回复486 Busy here，服务器将该响应转发给主叫。最后，主叫和服务器分别发送ACK确认结束流程，完成呼叫终止。
* wireshark抓包如下：  
  

## 三、结果分析

### 407 安全协议授权

通过分析，我们可以看到主叫方每次与服务器建立连接前均存在以下过程：  


服务器会返回一个407 Proxy Authentication Required包请求安全授权信息，下面我们对比分析创建连接的两个INVITE包

* 初始INVITE包  
  
* 第二个INVITE包  
  

通过对比内容，我们可以发现第二个包中在SIP头中增加了相对应的Proxy-Authorization字段。通过查阅，我们了解到该包用于进行鉴权，验证主叫的身份。其流程如下：

* 用户代理客户端将SIP消息发送到用户代理服务器
* 用户代理服务器以4xx质询回应
* 用户代理客户端使用质询包中的数据来加密身份凭证，然后将加密后的凭证加入SIP头中作为Proxy-Authorization字段发送给用户代理服务器

观察Proxy-Authorization字段，我们可以发现Nonce Value参数，代表密码通信中仅使用一次的任意数字。主叫使用它作为加密凭证，加密方法为单向不对称的SHA2加密算法。

## 实验心得

这次实验真的让我对SIP协议和VoIP通话有了更直观的理解！刚开始注册账号和配置MicroSIP的时候还觉得挺简单的，但真正开始抓包分析时才发现原来一个简单的电话呼叫背后有这么多信令交互。

最让我印象深刻的是那个407安全认证的过程。原来我们每次打电话前，系统都会先"验明正身"，通过Nonce和加密算法确保通话安全。看着wireshark里那些带Proxy-Authorization字段的数据包，感觉就像在看一场加密对话，特别有意思。

在模拟各种异常场景时也遇到不少状况。比如测试"被叫久叫不应"时，刚开始总是抓不到408超时包，后来才发现是计时器设置问题。还有一次误操作把ACK包发错了顺序，直接导致整个通话建立失败，这才体会到SIP协议对交互顺序的严格要求。

通过这次实验，我不仅学会了使用wireshark抓包分析，更重要的是理解了VoIP通话的完整生命周期。从呼叫建立、媒体协商到通话释放，每个环节都有严谨的信令控制。特别是看到486 Busy和480 Unavailable这些状态码时，终于明白平时打电话听到的"您拨打的电话正在通话中"是怎么实现的了。

唯一的小遗憾是由于网络环境限制，没能深入分析媒体流（RTP）的具体内容。不过整体来说，这次实验收获很大，让我对现代通信协议的设计理念有了更深的体会。以后接打电话时，脑海里都会自动浮现出那些INVITE、100 Trying、180 Ringing的数据包了！