**实验标题: Python Socket编程与Wireshark分析**

**李振硕(2300093007)  
南宫圣光(2000094627)**

**1. 实验概述**

本实验使用Python Socket实现了一个简单的echo服务器-客户端通信，并通过Wireshark对网络数据包进行了分析。通过此实验，学习了客户端与服务器之间的通信过程，并理解了通信过程中产生的TCP数据包。

**2. 通信过程及程序工作原理**

使用Python的Socket编程实现了两台设备之间的消息收发。此代码中，一台设备充当服务器角色，另一台设备作为客户端角色，客户端向服务器发送消息，服务器接收并处理消息。

Socket通信的主要组成部分如下：

* **服务器端**：服务器使用bind()函数绑定到指定的IP地址和端口，并等待客户端连接。
* **客户端**：客户端通过connect()函数连接到服务器，然后将消息发送给服务器。
* **使用TCP**：本程序使用传输控制协议(TCP)来确保可靠的通信。

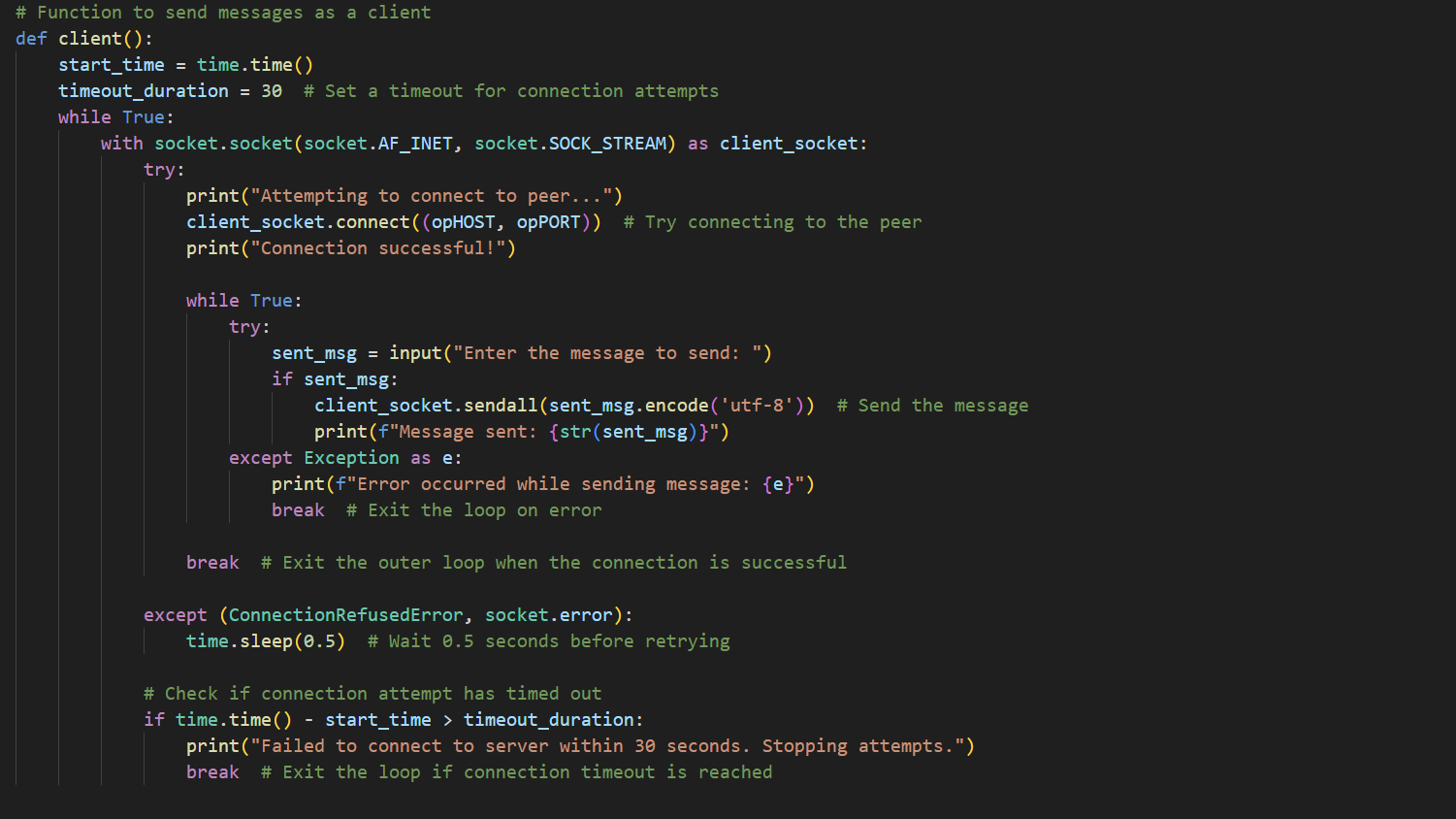
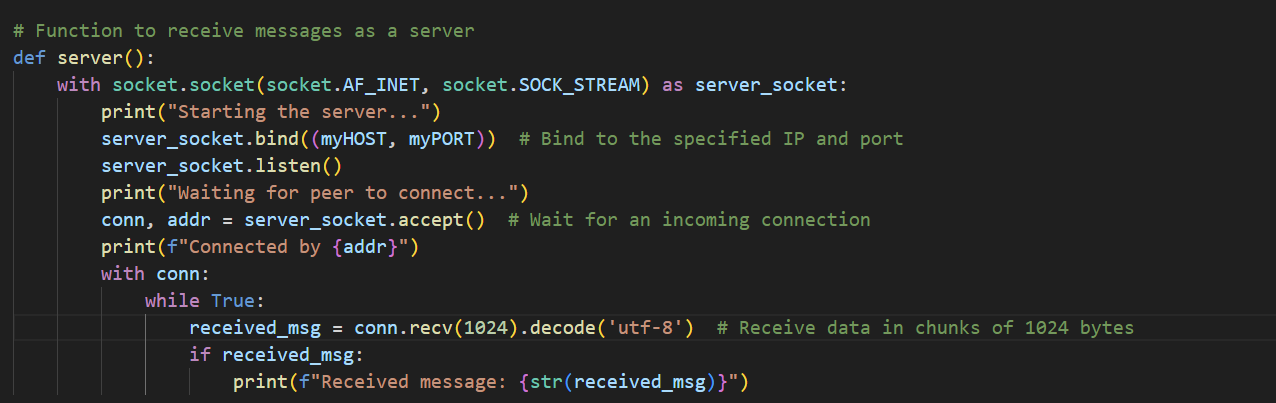
程序使用多线程技术，使得服务器和客户端可以异步进行通信。

**3. 客户端与服务器之间的通信方式**

客户端和服务器之间的通信按以下步骤进行：

1. **服务器**通过socket()和bind()函数在特定端口上监听，并等待客户端连接。
2. **客户端**通过connect()尝试连接服务器，连接成功后，客户端发送消息。
3. 服务器接收到客户端的消息后，在服务器端显示收到的消息，而客户端则显示消息发送成功。

**4. Echo代码的主要修改与功能**

我们对基础的echo代码进行了修改，实现了双向通信功能。客户端发送消息给服务器后，服务器处理并在其控制台上显示接收到的消息。本过程使用了recv()和send()函数来实现数据的接收与发送。

**5. 开发中遇到的问题及解决过程**

* **问题**：有时客户端无法连接到服务器。
* **解决**：设置了30秒的连接超时时间，若在规定时间内无法连接，则程序会自动重试连接。

**6. 使用Wireshark分析通信过程**

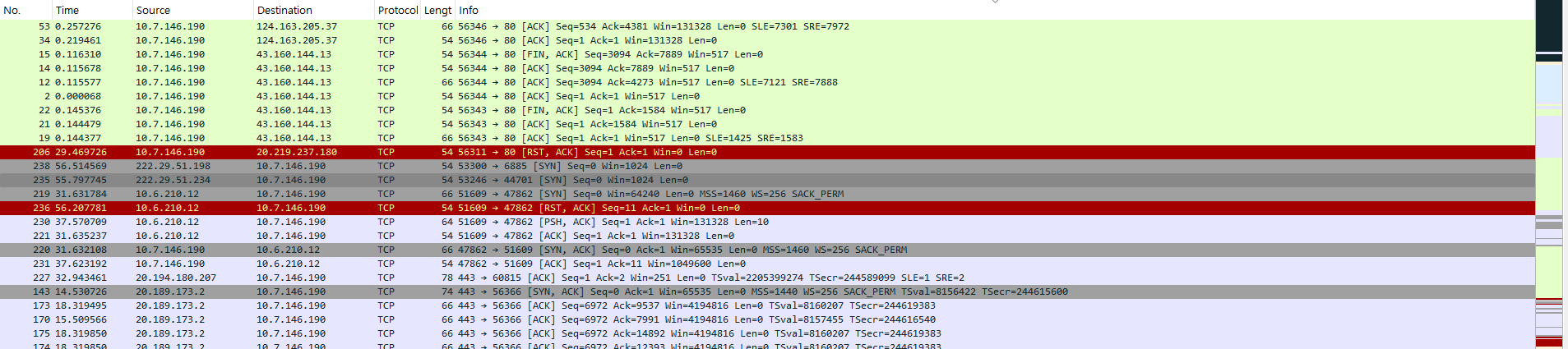
通过使用Wireshark捕获的TCP通信流，我们分析了客户端与服务器之间的交互过程，具体如下：

**1) TCP三次握手 (3-way Handshake)**

TCP三次握手是客户端向服务器请求连接时执行的过程，按以下步骤进行：

* **SYN**：客户端向服务器发送带有SYN标志的包，请求连接。
* **SYN-ACK**：服务器接收到客户端的请求后，发送带有SYN和ACK标志的包，表示接受连接请求。
* **ACK**：客户端接收到服务器的响应后，发送带有ACK标志的包，确认连接成功。

在Wireshark中捕获的相关数据包如下：

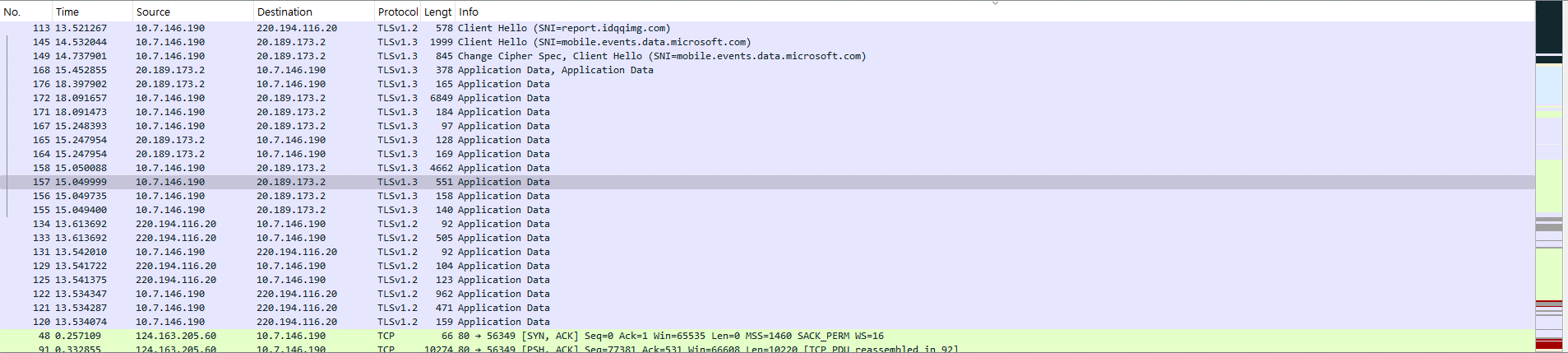
* **SYN包**：客户端向服务器发送的首次连接请求包。
* **SYN-ACK包**：服务器接受客户端请求并回复的包。
* **ACK包**：客户端确认连接建立后发送的包。

**2) 数据传输**

在完成三次握手后，客户端开始向服务器发送数据。此时，带有**PSH**（Push）标志的包用于传输数据，服务器接收数据后则回复**ACK**包，确认收到数据。

在Wireshark中捕获的数据传输包如下：

* **PSH-ACK包**：客户端向服务器发送数据时，包含PSH标志的包。
* **ACK包**：服务器接收到数据后确认成功接收的ACK包。



**3) TCP连接终止 (FIN)**

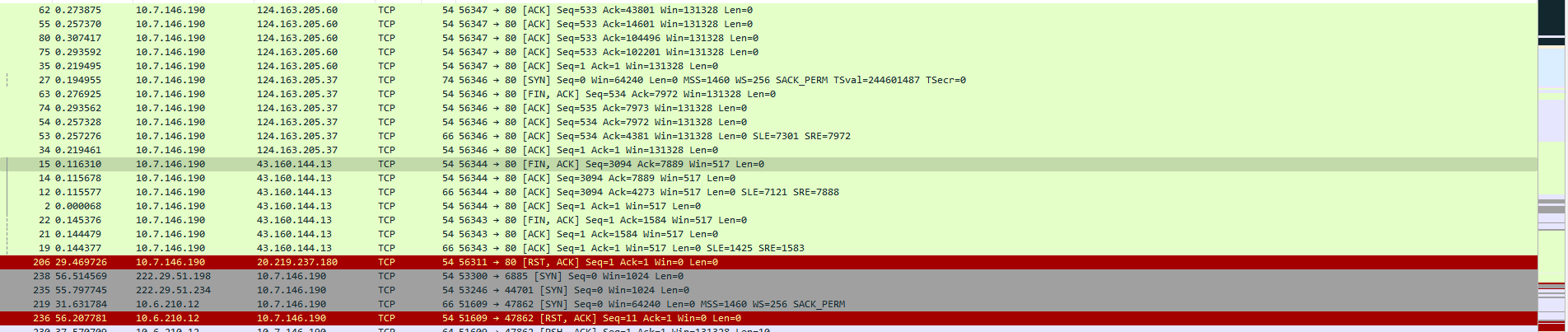
在客户端和服务器完成数据传输后，TCP连接通过发送**FIN**标志的数据包来终止。此过程包括以下步骤：

* **FIN**：客户端或服务器发送带有FIN标志的包，表示结束连接。
* **ACK**：另一方接收FIN请求后，回复ACK包。
* **FIN-ACK**：另一方也发送FIN标志的包，并通过ACK包来完成连接的关闭。

在Wireshark中捕获的连接终止包如下：

* **FIN包**：由客户端或服务器发送的终止连接请求包。
* **FIN-ACK包**：双方交换的结束连接确认包。

通过此数据包的流向分析，我们能够确认客户端与服务器之间的通信是通过TCP协议安全进行的。



**7. 结论**

通过本次实验，我们学习了Python Socket编程的基本工作原理，并理解了基于TCP的通信流程。特别是通过Wireshark捕获的数据包，我们详细观察到了TCP通信的过程（如SYN、ACK、FIN数据包的交换）。实验中开发的代码成功实现了双向通信功能，同时也解决了通信过程中遇到的各种问题，顺利完成了实验。