·实验教学·

基于 Scapy 的细粒度网络协议分析实验教学设计

杨永超

（池州学院 大数据与人工智能学院，池州 247000）

摘要：针对目前课程协议分析类实验仅从网络应用层面进行交互，导致分析粒度粗、教学效果差等问题，提出了一种基于 Scapy 的细粒度网络协议分析实验教学方法，由学生控制协议分组的构建及发送顺序，以分组为交互单元细化了协议分析的颗粒度。以 TCP 连接管理为例，详细描述了如何使用 Scapy 软件实施细粒度协议分析的实验教学。实践证明，将 Scapy 应用于协议分析的实验教学，可引发学生从收发双方分组的交互中深入思考协议执行的细节，获得显著的教学效果。

关　键　词：细粒度的；协议分析；实验教学；Scapy 软件

中图分类号：G424.31　　**文献标志码：**A　　DOI: 10.12179/1672-4550.20230313

Fine-grained Experimental Teaching Design for the Network Protocol Analysis Based on Scapy

YANG Yongchao

（School of Big Data and Artificial Intelligence, Chizhou University, Chizhou 247000, China）

Abstract: A new fine-grained protocol analysis experiment teaching method based on Scapy is proposed to address the problems of coarse analysis granularity and poor teaching effects due to the interoperation between network applications in traditional protocol analysis experiments. Using Scapy, students can control the construction and sending sequence of protocol data units, making it a fine-grained network protocol analysis method as the interoperation unit in a packet. Taking the experimental project “TCP connection management” as an example, this paper describes in detail how to implement the fine-grained protocol analysis experiment teaching with Scapy. The teaching practice demonstrates that applying Scapy to the experimental teaching of protocol analysis encourages students to think deeply about network protocols through the packet interaction of both sides, resulting in remarkable teaching effects.

Key words: fine-grained; protocol analysis; experimental teaching; Scapy software

作为计算机相关专业的骨干课程，计算机网络以概念抽象、难以理解等特点著称[1−2]。计算机网络的核心是协议，通过各层协议协同工作构成完整的网络体系，学生如果真正理解了协议，也就抓住了计算机网络的精髓。为此，教师在理论教学中穿插大量类比，并结合实验教学，梳理各协议间错综复杂的交织关系，使学生透彻理解各协议的工作原理，网络实验的重要性可见一斑。

目前高校计算机网络实验一般分为两类：网络仿真类和协议分析类。其中网络仿真类实验教学多选用 eNSP[3−5]、Cisco Packet Tracer[6−8] 或 GNS[9−10] 等模拟器，基本可达到真实网络设备实操的同等效果，同时不受教学时间和空间限制；协议分析类实验教学[10] 一般以 Wireshark 作为工具，捕获某网络通信过程的分组并直观地展示抽象协议的工作过程。Wireshark 开源免费、界面友好，但缺点是分析粒度较粗，仅能展示由网络应用进程自动产生的数据包，相关实验也以验证型为主，学生仅以“局外人”的身份查看各协议字段的语法语义，难以引发对协议本质的深入思考。针对此问题，文献 [11] 阐述了一种基于 C 语言和 OPNET 内核的细粒度实验仿真教学方法，引导

收稿日期：2023−06−24；**修回日期：**2023−10−23

基金项目：池州学院质量工程项目(2022XKCSZ03, 2022XJYXM16)。

作者简介：杨永超(1985−)，女，硕士，讲师，主要从事机器学习和网络安全方面的研究。E-mail: 330123414@qq.com

学生使用 C 语言实现协议，以“局内人”的身份，在与远程主机的数据交互中理解协议的本质，但 OPNET 实验内容更偏向于考虑协议的实现细节，对基础薄弱的应用型本科院校学生来说，仿真学习难度较大。

使用 Scapy[12] 同样可以达到深入理解协议的目的，且与基于 C 语言的 OPNET 相比，它基于 Python 语言实现，以面向对象思想将各协议封装成类，这样构建分组时可不必考虑协议的具体实现，从协议三要素[13] 出发，由学生控制协议分组的字段值（语法、语义）及分组交互顺序（时序），相较 Wireshark 仅能查看协议分组而言，是一种更细粒度的协议分析方法。因此教学团队尝试在网络协议分析实验教学中引入 Scapy，以一种细粒度方式进行协议数据单元的构建及分析，以期提升学生对各类网络协议的理解，优化实验教学效果。

1 Scapy 软件简介

Scapy 具备强大的数据包操纵功能，通过简单几行函数调用，即可实现协议分组的构建、发送、嗅探及分析[12,14]。Scapy 定义了互联网中所有的经典协议类，如 Ether、IP、ICMP、TCP、UDP、DNS 等，可通过实例化一个类对象，并指定参数值来构建该协议的分组，实现其语法及语义要素。如使用 p = IP(dst="192.168.1.1") /UDP(dport = 53)/ DNS(qd = DNSQR(qname = "baidu.com"))，可以构建一个 DNS 的查询请求报文，请求的域名服务器 IP 为 192.168.1.1，待解析的域名为 baidu.com。Scapy 还提供了分组发送的函数：send、sr、sendp 和 srp，其中 send 和 sendp 函数仅用于分组发送，sr 和 srp 函数可发送和接收响应，在需要接收响应分组时使用。如遵循协议规定的顺序发送对应的分组，即可实现协议的时序要素。

Scapy 支持 Windows、Linux、OSX 及大多数 Unix 操作系统平台，它提供丰富的组件支持可视化展示和二次开发，可用于实现多种经典的网络功能，如扫描、路由追踪、单元测试、攻击和网络发现等。

2 使用 Scapy 进行协议分析类实验设置的总体思路

考虑到计算机网络在不同专业的后续课程不同，在授课时需“瞻前顾后”合理设置实验内容。如学校计算机科学与技术、智能科学等专业后续没有其他网络类课程，因此 16 学时实验需要同时设置网络仿真类与协议分析类（占比约 25%）实验；网络工程专业后续有路由交换、综合布线等课程，则主要设置网络编程类与协议分析类（占比约 90%）实验。

由于 Scapy 的分组发送及响应均以文本形式呈现，结合 Wireshark 的图像化界面使用更为直观。以学校网络工程专业为例，基于 Wireshark+ Scapy 的实验内容设置如表 1 所示，从计算机网络五层体系结构出发，系统地设计基于 Scapy 的细粒度协议分析实验教学。在数据链路层从构建一个简单的以太网帧开始，设计构建一个类型值为学号的以太网帧发送出去，并使用 Wireshark 直观展示，熟悉 Scapy 及 Wireshark 的使用，在此基础上，完成局域网内 ARP 协议分组交互。在网络层使用 Scapy 完成 IP 分组的发送，并实现 ICMP 协议的 Ping 及 Tracert 功能。低层协议偏向验证型，而运输层协议则偏重设计型。使用 Python Socket 模块完成服务器设计后，客户端使用 Scapy 构建并按序发送 TCP 报文段，进而掌握 TCP 的 3 次报文握手建立连接和 4 次报文挥手释放连接[13, 15] 的过程。应用层设计综合型实验，完成 Web 服务器和 DNS 服务器搭建后，在客户端使用 Scapy 构建 DNS 协议分组，收到服务器的响应报文后，按顺序依次构建 TCP 第 1 次握手、第 3 次握手、HTTP 请求报文，与服务器进行交互，整个过程需综合考虑网络层、运输层和应用层的核心协议及交互顺序。

3 基于 Scapy 的细粒度协议分析实验设计及实施示例

接下来，以实验项目“基于 Scapy 的 TCP 连接管理”为例，介绍如何实施细粒度的协议分析实验教学。

3.1 实验目的

熟悉基于 Python Socket 编程的一般步骤；掌握 Python Scapy 构建 TCP 报文段的方法；掌握 TCP 的连接管理过程，理解 TCP 序号、确认号的变化机制及其可靠传输机制。

3.2 实验环境

在局域网中任选两台 Windows 7 以上操作系

统的主机，分别充当客户端及服务器角色。两机均需安装 Python3，客户端另需安装 Scapy、Npcap 及 Wireshark。为防止 Windows 防火墙屏蔽两机的通信数据，两台主机均添加入站规则：允许局域网网段所有通信数据入站，将所在局域网网段添加到作用域。

表 1　实验内容设置

实验名称	学时	实验内容摘要
双绞线制作/常用网络命令	2	① 两人一组，分别制作直通及交叉线 ② 在Windows命令行熟悉常用的网络命令：ping、nslookup、tracert、ipconfig等
低层协议分析	4	① 两人一组，熟悉 Wireshark 及 Scapy 环境 ② 使用 Scapy 构建 ARP 请求帧，并发送到局域网的活动主机，期间使用 Wireshark 直观显示交互的帧内容。拓展：使用 Scapy 在局域网内实施 ARP 欺骗 ③ 使用 Scapy 构建 IP 分组及 ICMP 分组，查询与目的主机的连通性及路由追踪，实现 ping 及 tracert 功能。拓展：使用 Scapy 在局域网内实施 Ping of Death
基于Scapy的TCP连接管理	4	① 两人一组，熟悉 Python Socket 编程的一般步骤 ② 利用 Socket 函数实现基于 TCP 的两机通信。拓展：实现基于 UDP 的两机通信；实现多线程的两机通信 ③ 服务器继续运行，客户端使用 Scapy 构建 TCP 的 3 次握手报文建立连接，数据通信完毕后，使用 Scapy 构建 TCP 的 4 次挥手报文释放连接，期间使用 Wireshark 直观显示交互的报文段内容。拓展：实施 SYN-Flood 攻击
服务器搭建及高层协议分析	6	① 两人一组，利用 VMware WorkStation Pro 搭建虚拟网络，在 Winserver 2008 操作系统上安装 DNS 服务器及 HTTP 服务器，客户端可通过域名访问 HTTP 服务器 ② 在客户端使用 Scapy 构建 HTTP 请求报文发送到服务器，期间使用 Wireshark 直观显示交互的报文内容 ③ 使用 Scapy 构建 DNS 请求报文，期间使用 Wireshark 直观显示交互的报文内容。拓展：实施对虚拟机 DNS 服务器的攻击

3.3　实验内容

1）使用 Python Socket 搭建基于 TCP 的服务器，实现从客户端接收数据并显示的功能；

2）使用 Scapy 命令行在客户端构建 TCP 报文段，并按协议实现顺序与服务器交互，实现 TCP 连接管理过程，或通过 import scapy 模块，将代码保存在.py 文件中运行。具体实验过程如图 1 所示；

3）使用 Wireshark 展示交互过程中产生的报文段，并由此直观判断构建报文段是否有误。

3.4　实验过程

3.4.1　搭建服务器

服务器使用 Python 的 Socket 模块实现。

1）创建监听套接字 listen_socket，绑定本机缺省 IP 及端口 16668，监听并处理客户端发来的连接请求，实现代码如下：

```
listen_socket=socket.socket(socket.AF_INET,socket. SOCK_STREAM)
listen_socket.bind(('', 16668))#将 socket 绑定到本机的 16668 端口
print('listen_socket is listening on 16668.....')
listen_socket.listen(10)
```

2）如果有客户端连接请求到达，则接受连接请求并创建连接套接字 conn_socket，接着调用 recv 函数阻塞接收数据并显示，之后调用 close 函数关闭连接套接字。

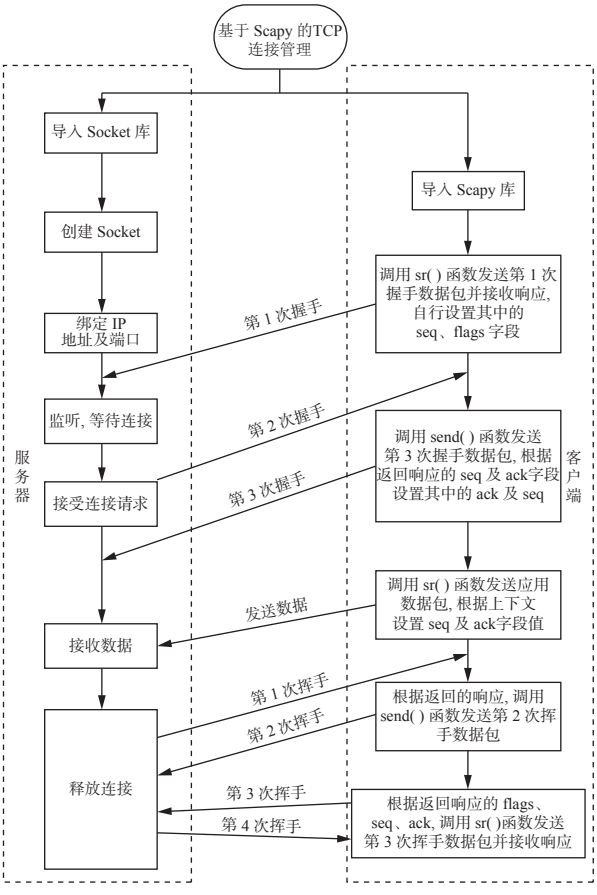

图 1　基于 Scapy 的 TCP 连接管理实验流程

```
while True:
conn_socket, addr = listen_socket.accept()
#接受连接，创建连接套接字 conn_socket
data = conn_socket.recv(1024).decode()#从连接套接字接收数据
print(data)
conn_socket.close()
```

服务器搭建完毕后，即启动运行。

3.4.2 启动 Wireshark 监听

在客户端启动 Wireshark 监听连接到局域网的网卡，并在过滤框输入规则：tcp.port==16668，过滤出与服务器 16668 端口交互的报文段。

3.4.3 客户端实现

客户端使用 Scapy 交互界面构建并发送 TCP 报文段，由图 1 可知，整个过程客户端共需发送 5 个 TCP 报文段。在这期间，学生需认真思考 TCP 连接不同阶段分组首部字段的变化，在构建 TCP 报文段时正确设置首部字段值并严格遵守 TCP 时序发送，否则会引起 TCP 连接异常。在客户端的 cmd 命令行输入 scapy 启动 Scapy，进入交互界面。

1) 连接建立阶段

TCP 连接建立以 3 次握手报文实现，其中第 1 次和第 3 次握手由客户端构建，第 2 次握手由服务器构建。第 1 次握手的特点是标志位 SYN 置 1，因此设置 $flags_1$="S"，$dport_1$ 为服务器监听端口 16668，seq_1 和 $sport_1$ 可在字段值的允许范围内随机生成，其他字段使用协议的缺省值。考虑到 TCP 第 1 次握手后，需接收由服务器返回第 2 次握手，因此使用 sr()函数发送第 1 次握手。

第 3 次握手首部字段（$dport_3$, $sport_3$, seq_3, ack_3, $flags_3$）根据返回的第 2 次握手首部构建，seq_3=ack_2，ack_3=seq_2+1，$flags_3$='A'，由于第 3 次握手后即发送应用数据，无需服务器回复，因此使用 send()函数发送。连接建立阶段的实现代码如下：

```
#发送第 1 次握手并接收第 2 次握手
>>>result_synack = sr(IP(dst= "192.168.1.4" )/TCP( dport= 16668, sport =random.randint（49152, 65535）, flags='S', seq= random.randint( 0, 2**32-1)))
#返回的响应存储在 result_synack 中
>>>result_synack_list = result_synack[0].res
>>>tcpfields_synack=result_synack_list[0][1][1].fields
#根据第 2 次握手的 TCP 首部设置第 3 次握手首部
>>>sc_sn = tcpfields_synack['seq'] + 1
#ack3=seq2+1
>>>cs_sn = tcpfields_synack['ack']
#seq3=ack2
>>>sportid = tcpfields_synack['dport']
#发送第 3 次握手
>>>send( IP( dst="192.168.1.4" ) /TCP( dport= 16668,sport= sportid, flags= 'A', seq= cs_sn, ack = sc_sn))
```

此时切到 Wireshark，发现捕获了 TCP 连接建立的 3 次握手报文，如图 2 所示。

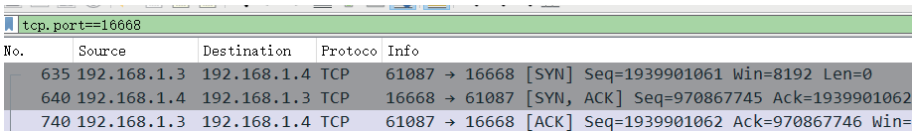

图 2 Wireshark 捕获 TCP 连接建立阶段截图

2) 数据传输阶段

由于第 3 次握手并未发送数据载荷，服务器也未返回响应，因此 seq_d 与 ack_d 应与第 3 次握手一致。为避免 TCP 协议使用 Nagle 算法，还应将应用数据尽快推送到应用层，即将 Push 标志位置 1。由图 1 所示，客户端发送数据后即由服务器端释放连接，因此选用 sr()函数发送。实现代码如下：

```
#构建应用数据，并接收服务器返回第 1 次挥手报文
>>>result_raw_msg = sr(IP(dst ="192.168.1.4")/TCP( dport = 16668,sport = sportid, flags = 'PA', seq= cs_sn , ack= sc_sn)/"welcome to czu")
```

切到 Wireshark，发现捕获了应用数据及第 1 次挥手的报文段，如图 3 和图 4 所示。

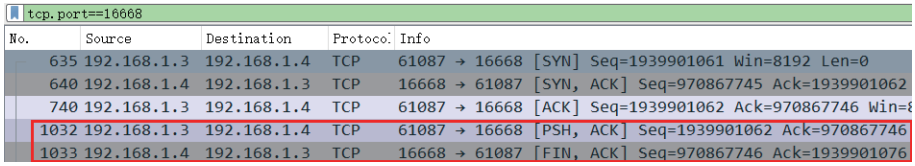

图 3　Wireshark 捕获应用数据及第一次挥手

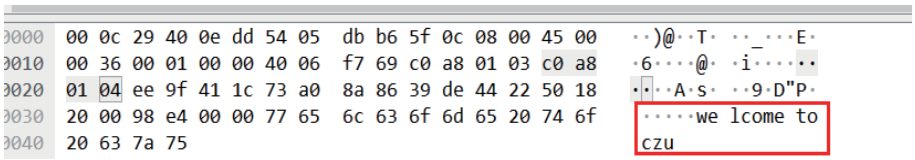

图 4　应用数据原始 ASCII 码编码

3）连接释放阶段

根据 TCP 时序，第 1 次和第 4 次挥手由服务器发送，第 2 次和第 3 次挥手由客户端发送。可根据第 1 次挥手设置第 2 次挥手首部，$ack_{f2}=seq_{f1}+1$，$seq_{f2}=ack_{f1}$，$flags_{f2}$='A'，由于无需服务器返回响应，故选用 send()函数发送。由于第 2 次挥手未携带数据，第 3 次挥手首部序号和确认号与第 2 次一致，$ack_{f3}=ack_{f2}$，$seq_{f3}=seq_{f2}$，$flags_{f3}$='FA'，并且需接收服务器返回的第 4 次挥手，故选用 sr()函数发送。实现代码如下：

```
>>>rec_msg_ack = result_raw_msg[0].res
>>>tcp_msg = rec_msg_ack[0][1][1].fields
#将第一次挥手的首部赋给变量 tcp_msg
>>>cs_sn = tcp_msg['ack']# seq_f2=ack_f1
>>>sc_sn = tcp_msg['seq'] + 1# ack_f2=seq_f1+1
>>>send( IP( dst="192.168.1.4") /TCP( dport=16668,sport=sportid,flags='A',seq=cs_sn,ack=sc_sn) , verbose = False)
#构建第 2 次挥手报文段并发送
>>>recv=sr(IP(dst="192.168.1.4")/TCP(dport=16668, sport=sportid, flags='FA', seq=cs_sn, ack=sc_sn), verbose=False)
#发送第 3 次挥手并将收到的第 4 次挥手存储在 recv 中
```

切到 Wireshark，发现捕获了连接释放过程的数据包，如图 5 所示。

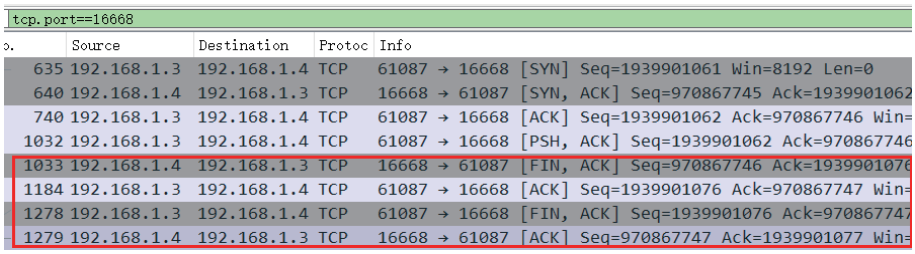

图 5　Wireshark 捕获第 3 次与第 4 次挥手

3.5　教学小结

学生在实操过程中，如所有的分组都构建正确，则 TCP 连接管理过程符合标准的 3 次握手和 4 次挥手形式，如图 5 所示，否则会出现异常。如图 6 所示为错误设置了第 2 次挥手 ack 后(将 sc_sn = tcp_msg['seq'] + 1 错置为 sc_sn = tcp_msg ['seq'])，Wireshark 捕获的数据包列表。由于第 1 次挥手(No.5)的 FIN 置 1 需消耗一个序号，此时第 2 次挥手(No.6)ack 未加 1，服务器会错误认为第 1 次挥手报文丢失，则连续重传 5 次(No.9~13)，后因仍未收到客户端响应，重置连接(No.14，此时 RST=1)。

在实际的教学组织中，教师可演示 3 次握手过程，由学生独立设计并完成 4 次挥手过程，提高学生实践动手能力。对于学有余力的同学，还可增加利用 Scapy 实现对服务器的 SYN- Flood 攻击，体现实验挑战度。

基于 Scapy 的实验教学过程中，由学生控制协议分组的控制字段及发送顺序，除协议三要素的执行细节之外，还需考虑 TCP 的 Nagle 算法及如何实现可靠传输，与传统基于 Wireshark 的协议分析实验相比，可促使学生对协议更深层次的思考。

No.	Source	Destination	Protocol	Info
1	192.168.1.3	192.168.1.4	TCP	50190 → 16668 [SYN] Seq=2922045849 Win=8192 Len=0
2	192.168.1.4	192.168.1.3	TCP	16668 → 50190 [SYN, ACK] Seq=972825082 Ack=2922045850 Win=8192 Len=
3	192.168.1.3	192.168.1.4	TCP	50190 → 16668 [ACK] Seq=2922045850 Ack=972825083 Win=8192 Len=0
4	192.168.1.3	192.168.1.4	TCP	50190 → 16668 [PSH, ACK] Seq=2922045850 Ack=972825083 Win=8192 Len=
5	192.168.1.4	192.168.1.3	TCP	16668 → 50190 [FIN, ACK] Seq=972825083 Ack=2922045864 Win=65392 Len
6	192.168.1.3	192.168.1.4	TCP	[TCP Dup ACK 3#1] 50190 → 16668 [ACK] Seq=2922045864 Ack=972825083
7	192.168.1.3	192.168.1.4	TCP	50190 → 16668 [FIN, ACK] Seq=2922045864 Ack=972825083 Win=8192 Len=
8	192.168.1.4	192.168.1.3	TCP	16668 → 50190 [ACK] Seq=972825084 Ack=2922045865 Win=65392 Len=0
9	192.168.1.4	192.168.1.3	TCP	[TCP Retransmission] 16668 → 50190 [FIN, ACK] Seq=972825083 Ack=292
10	192.168.1.4	192.168.1.3	TCP	[TCP Retransmission] 16668 → 50190 [FIN, ACK] Seq=972825083 Ack=292
11	192.168.1.4	192.168.1.3	TCP	[TCP Retransmission] 16668 → 50190 [FIN, ACK] Seq=972825083 Ack=292
12	192.168.1.4	192.168.1.3	TCP	[TCP Retransmission] 16668 → 50190 [FIN, ACK] Seq=972825083 Ack=292
13	192.168.1.4	192.168.1.3	TCP	[TCP Retransmission] 16668 → 50190 [FIN, ACK] Seq=972825083 Ack=292
14	192.168.1.4	192.168.1.3	TCP	16668 → 50190 [RST, ACK] Seq=972825084 Ack=2922045865 Win=0 Len=0

图 6　错误构建 TCP 报文段的异常显示

4　教学效果分析

目前各高校积极开展工程教育认证，课程目标的达成度是衡量教学质量、推动教学效果持续改进的重要依据，是工程教育认证的重要一环。协议分析部分支撑计算机网络课程目标 CO2：能够对实际网络应用开发、协议、网络规划及运维中存在的问题进行分析。具备基本的网络协议分析及设计能力，以及分析解决现代网络中具体问题的创新意识和能力。课程考核评价中各环节占比如表 2 所示。

如图 7 所示为教改前后课程目标 CO2 的达成度变化，教改前课程目标 CO2 的达成度仅为 0.67，基本达成。但教改后达成度达 0.75，有明显提升。

表 2　课程考核评价表

课程目标	评价依据及比例/%				权重/%
	课堂表现	实验	作业	考试	
CO1	10.0	5	12	30	57.0
CO2	2.5	10	1	10	23.5
CO3	2.5	5	2	10	19.5
合计	15.0	20	15	50	100.0

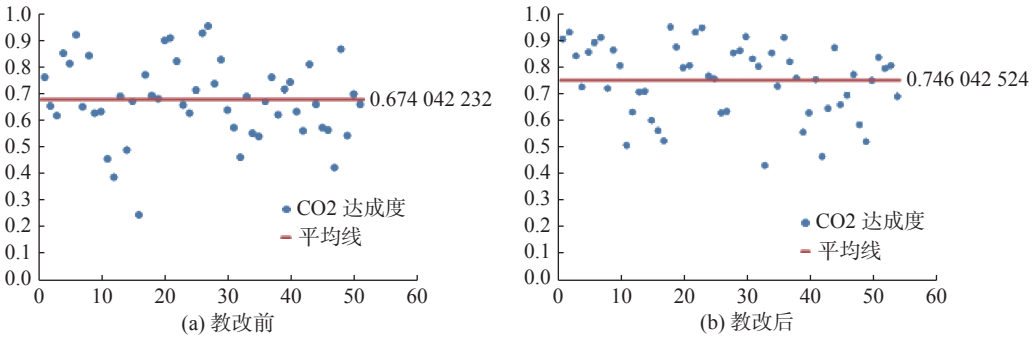

图 7　教改前后课程目标 CO2 达成度对比

在课程的满意度调查中，教学评价反馈良好，教改前后学生对课程的满意度评价如图 8 所示，可见理论掌握度及实验效果评价有明显的提升。

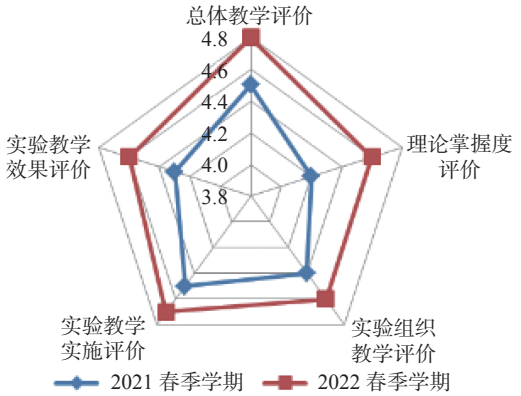

图 8　学生满意度调查对比

5　结束语

网络协议概念抽象且数量众多，只有理解了协议的本质，才能真正理解互联网的工作原理。Scapy 具有功能强大、操作简单等优点。实践证明，将其作为辅助工具应用于网络课程的实验教学中，可将实验过程主线变“观察”为“探索”，将协议分析细粒度化，促进了学生对协议本质的深入思考，收获了良好的教学效果，且易于提升课程的挑战度，为网络协议分析实验教学提供了一种新的参考方案。

参 考 文 献

[1]　韩慧妍, 郁晓庆, 樊彩霞, 等. “计算机网络基础”课程

“金课”建设初探[J]. 计算机时代, 2021(5): 73−76.
[2] 王平, 丁小妹. 应用技术型大学的《计算机网络》课程教学探索与改革[J]. 现代计算机, 2020(12): 106−109.
[3] 崔晓龙, 刘欣, 王建萍, 等. 计算机网络“三层次渐进式”实践教学设计[J]. 实验室研究与探索, 2022, 41(4): 163−169.
[4] 梁海华, 盘丽娜, 钱振江. 基于eNSP的网络实验综合设计与渐进实践[J]. 实验室研究与探索, 2022, 41(9): 140−145.
[5] 王菲菲, 蒋丽平. eNSP虚拟仿真平台在计算机网络实验教学中的应用探究: 以昌吉学院为例[J]. 信息与电脑(理论版), 2022, 34(5): 222−224.
[6] 廖伯勋. 新工科背景下基于Packet Tracer的计算机通信与网络课程实践教学方法探索[J]. 数字技术与应用, 2022, 40(12): 56−58.
[7] 黄继文, 邓志龙, 李良良, 等. 综合应用多平台“团队网课”在线课程教学的探索与实践: 以《数据网组建与维护》在线课程教学为例[J]. 襄阳职业技术学院学报, 2023, 22(3): 84−87.
[8] 孙良旭, 李林林. 网络工程专业虚拟仿真实验平台研究[J]. 实验科学与技术, 2023, 21(2): 34−38.
[9] 孙光懿, 缴健. 基于GNS3的NAT技术仿真设计与实现[J]. 西北民族大学学报(自然科学版), 2021, 42(2): 18−26.
[10] 彭玉兰, 代琪怡, 李佳芮, 等. 基于GNS3+Wireshark的网络协议分析实验教学改革[J]. 现代信息科技, 2022, 6(18): 185−187.
[11] 刘宴涛, 秦娜. “计算机网络”课程实验仿真教学的路径探索[J]. 电气电子教学学报, 2021, 43(5): 174−182.
[12] PHILIPPE. Scapy online document[EB/OL]. (2021-12-21)[2023-3-15]. https://scapy.readthedocs.io/en/latest/introduction.html.
[13] 谢希仁. 计算机网络[M]. 8版. 北京: 电子工业出版社, 2021: 246−250.
[14] MAJUMDAR A, RAJ S, SUBBULAKSHMI T. ARP poisoning detection and prevention using Scapy[J]. Journal of Physics: Conference Series, 2021, 1911(1): 1−10.
[15] KOUROSE J F, ROSS K W. 计算机网络: 自顶向下方法[M]. 8版. 陈鸣, 译. 北京: 机械工业出版社, 2022.

编辑　葛晋

(上接第 86 页)

[9] SUSMAN M D, PHAM H N, DATYE A K, et al. Factors governing MgO(111) faceting in the thermal decomposition of oxide precursors[J]. Chemistry of Materials, 2018, 30(8): 2641−2650.
[10] 黄建翠, 凌观爽, 宗俊. 水菱镁矿制备不同形貌纳米氧化镁的研究[J]. 盐科学与化工, 2020, 49(8): 11−18.
[11] 程敬泉. 无机阻燃剂纳米氧化镁的制备[J]. 广东化工, 2018, 45(22): 13−14.
[12] 房川琳, 李俊玲, 齐悦, 等. 微波法制备磁性纳米粒子Fe_3O_4探究[J]. 实验技术与管理, 2018, 35(5): 64−67.
[13] 王连连, 田磊, 王明, 等. 立方纳米氧化镁的制备及形貌表征[J]. 中国粉体技术, 2019, 25(6): 50−55.
[14] 哈恩娜, 吴江红, 胡鑫, 等. 基于“纳米技术”科学实践构建生物医学工程专业“大学化学”课程体系[J]. 教育教学论坛, 2020, 23: 327−328.
[15] 王吕阳, 胡俊青, 哈恩娜, 等. 表面化学技术引入“大学化学”教学改革初探[J]. 教育教学论坛, 2022, 44: 58−61.
[16] 贾春晓, 秦肖云, 杨鹏飞, 等. 基于成果导向教育理念的近代测试技术课程教学改革[J]. 化工教育, 2021, 38(6): 59−61.

编辑　王燕