

电子邮件接收客户机程序报告

学号：2120250723

姓名：于成俊

一、编程环境

- 操作系统：Windows 11
- Python版本：Python 3.7~Python 3.12均可
- 必要库（全部为Python 标准库，无需安装第三方）
 - socket：提供底层网络通信接口。本程序使用 `socket` 建立 POP3/TCP 连接，负责发送和接收 POP3 协议数据。
 - ssl：用于构建加密通信通道。在端口 995 (POP3S) 时，通过 `ssl.wrap_socket()` 创建安全的 SSL/TLS 连接，保证用户邮箱账号与密码不被泄露。
 - threading：用于多线程操作。本程序通过多线程避免 GUI 阻塞，例如登录、获取列表、下载邮件均放到线程中执行。
 - re：正则表达式库。用于清洗邮件内容，如删除多余空格、合并空行、解析 HTML 文本。
 - bs4 (BeautifulSoup)：用于解析 HTML 内容。当邮件正文是 HTML 时，通过 BeautifulSoup 将其转换为纯文本，便于在 GUI 中显示。
 - ctypes：调用 Windows 系统 API。使用 `ctypes.windll.shcore.SetProcessDpiAwareness(1)` 来解决 Windows 上的高 DPI 缩放问题，使 Tkinter GUI 在高分屏显示正常比例。
 - tkinter：Python 内置 GUI 库。用于创建主窗口、布局、文本框、按钮等界面元素。
 - tkinter.ttk：Tkinter 的主题控件库，可提供外观更好的 GUI 控件。
 - tkinter.scrolledtext：提供带滚动条的文本框控件。用于显示邮件正文、POP3 交互过程日志等。
 - tkinter.messagebox：提供标准弹窗（提示、警告、错误）。用于错误提示，如“登录失败”、“连接错误”。
 - tkinter.simpledialog：提供简单输入对话框。用于让用户输入要下载的邮件编号。
 - email.message_from_bytes：来自 `email` 标准库。用于解析 POP3 返回的 EML 原始邮件内容，支持多部分 MIME 邮件、HTML 邮件、附件等结构。

二、关键问题

(1) POP3 协议

POP3 使用客户端-服务器模型，通过明文或加密的通信通道传输邮件。它的核心功能是从服务器下载邮件。

它有两个端口：

- 110 (默认端口)：支持普通 TCP 连接

- 995：支持 **SSL 加密连接**（使用 `ssl.wrap_socket`），避免邮箱密码明文传输，提高安全性。

POP3 基础操作流程如下：

- 连接服务器
- 用户登录
- 获取邮件列表
- 下载邮件（EML 格式）
- 预览邮件内容
- 退出会话

连接服务器需要**邮箱的POP3服务器地址和端口号**，登录需要邮箱账号和授权密码，以南开大学邮箱为例，讲一下相关设置：

- 南开大学邮箱的POP3服务器地址为：`pop3.mail.nankai.edu.cn`
- 在南开大学邮箱的设置--->客户端设置中，需要开启客户端登陆（POP3）功能，并生成授权密码，这样才可以连接邮箱的POP3服务器：



知道以上基础知识和相关操作，才可以进行编程，POP3基础命令的实现已在第三节详细讲述。

(2) 多行响应的正确处理

POP3 的 `LIST` 和 `RETR` 命令返回多行数据，以 `.\r\n` 结尾，我们需要正确处理才能够完整接收多行内容，避免邮件内容截断。

代码实现如下：

```

def recv_multiline(self):
    """接收多行响应直到单独一行是 '.'"""
    data = ""
    while True:
        part = self.sock.recv(4096).decode(errors="ignore")
        data += part
        if "\r\n.\r\n" in data:
            break
    for line in data.splitlines():
        self.log("S: " + line)
    return data

```

(3) 邮件解析和纯文本/HTML内容提取

在本程序中，实现了以下功能：

- 捕获邮件列表
- 双击邮件列表，以预览邮件内容

通过 `RETR` 命令捕获的邮件是EML格式，我们需要进行相关处理，才可以将正文内容提取出来。

核心代码如下：

```

def preview_mail(self, event):
    index = self.mail_list.curselection()
    if not index:
        return

    msg_no = self.mail_list.get(index).split()[0]

    # 发送 RETR 指令
    self.send_cmd(f"RETR {msg_no}")
    resp = self.recv_line()
    if not resp.startswith("+OK"):
        self.log("获取邮件失败")
        return

    # 获取完整邮件 (EML 原文)
    data = self.recv_multiline()

    try:
        msg = message_from_bytes(data.encode() if isinstance(data, str) else
data)
        body = ""

        if msg.is_multipart():
            # 优先提取 text/plain
            for part in msg.walk():
                content_type = part.get_content_type()
                disp = str(part.get("Content-Disposition") or "")
                charset = part.get_content_charset() or "utf-8"

                if content_type == "text/plain" and "attachment" not in disp:
                    text = part.get_payload(decode=True).decode(charset,
errors="ignore")

```

```

# 美化 text/plain: 合并多空行, 去掉多空格
text = re.sub(r'[\t]+', ' ', text)
text = re.sub(r'\n\s*\n', '\n\n', text)
body += text

# 如果 text/plain 为空, 则尝试 text/html
if not body:
    for part in msg.walk():
        content_type = part.get_content_type()
        disp = str(part.get("Content-Disposition") or "")
        charset = part.get_content_charset() or "utf-8"

        if content_type == "text/html" and "attachment" not in disp:
            html = part.get_payload(decode=True).decode(charset,
                errors="ignore")
            # HTML → 文本
            soup = BeautifulSoup(html, "html.parser")
            text = soup.get_text()
            # 美化格式: 去掉多空行和多空格
            text = "\n".join([line.strip() for line in
text.splitlines() if line.strip()])
            text = re.sub(r'[\t]+', ' ', text)
            body = text
            break
    else:
        charset = msg.get_content_charset() or "utf-8"
        payload = msg.get_payload(decode=True)
        if payload:
            if msg.get_content_type() == "text/plain":
                text = payload.decode(charset, errors="ignore")
                text = re.sub(r'[\t]+', ' ', text)
                text = re.sub(r'\n\s*\n', '\n\n', text)
                body = text
            elif msg.get_content_type() == "text/html":
                html = payload.decode(charset, errors="ignore")
                soup = BeautifulSoup(html, "html.parser")
                text = soup.get_text()
                text = "\n".join([line.strip() for line in
text.splitlines() if line.strip()])
                text = re.sub(r'[\t]+', ' ', text)
                body = text

            if body.strip() == "":
                body = data

except Exception as e:
    body = f"邮件解析失败: {e}\n\n原始内容: \n{data}"

# 显示到预览窗口
self.preview.delete("1.0", tk.END)
self.preview.insert(tk.END, body)

```

(4) 图形界面实现可视化操作

本程序基于 Tkinter 来实现图形界面，代码实现如下：

```
def __init__(self, master):
    self.master = master
    master.title("POP3 邮件接收客户端")
    master.geometry("1000x1000")

    self.sock = None

    # ----- 样式 -----
    style = ttk.Style()
    style.theme_use('clam')
    style.configure("TButton", padding=6, font=('Arial', 11))
    style.configure("TLabel", font=('Arial', 11))
    style.configure("Header.TLabel", font=('Arial', 13, 'bold'))

    # ----- 输入区 -----
    frame_top = ttk.Frame(master, padding=10)
    frame_top.pack(fill='x')

    ttk.Label(frame_top, text="POP3 服务器:",
    style="Header.TLabel").grid(row=0, column=0, sticky='e')
    ttk.Label(frame_top, text="账号:", style="Header.TLabel").grid(row=1,
    column=0, sticky='e')
    ttk.Label(frame_top, text="密码:", style="Header.TLabel").grid(row=2,
    column=0, sticky='e')
    ttk.Label(frame_top, text="端口:", style="Header.TLabel").grid(row=3,
    column=0, sticky='e')

    self.server_entry = ttk.Entry(frame_top, width=35)
    self.user_entry = ttk.Entry(frame_top, width=35)
    self.pass_entry = ttk.Entry(frame_top, width=35, show="*")
    self.port_entry = ttk.Entry(frame_top, width=10)

    self.server_entry.grid(row=0, column=1, pady=3)
    self.server_entry.insert(0, "pop3.mail.nankai.edu.cn")
    self.user_entry.grid(row=1, column=1, pady=3)
    self.user_entry.insert(0, "2120250723@mail.nankai.edu.cn")
    self.pass_entry.grid(row=2, column=1, pady=3)
    self.pass_entry.insert(0, "fFh1U4K7d8x$KfrQ")
    self.port_entry.grid(row=3, column=1, pady=3)
    self.port_entry.insert(0, "995")

    # ----- 按钮 -----
    frame_btn = ttk.Frame(master, padding=5)
    frame_btn.pack(fill='x')

    ttk.Button(frame_btn, text="连接",
    command=self.thread_connect).pack(side='left', padx=3)
    ttk.Button(frame_btn, text="登录",
    command=self.thread_login).pack(side='left', padx=3)
    ttk.Button(frame_btn, text="获取列表",
    command=self.thread_get_list).pack(side='left', padx=3)
```

```

        ttk.Button(frame_btn, text="下载邮件",
command=self.thread_download_mail).pack(side='left', padx=3)
        ttk.Button(frame_btn, text="退出 QUIT",
command=self.thread_quit).pack(side='left', padx=3)

# ----- 主体 -----
frame_main = ttk.Frame(master)
frame_main.pack(fill='both', expand=True)

self.mail_list = tk.Listbox(frame_main, width=40)
self.mail_list.pack(side='left', fill='y')
self.mail_list.bind("<Double-Button-1>", self.preview_mail)

self.preview = scrolledtext.ScrolledText(frame_main, font=('Consolas',
11))
self.preview.pack(side='right', fill='both', expand=True)

ttk.Label(master, text="交互过程: ",
style="Header.TLabel").pack(anchor='w')
self.output = scrolledtext.ScrolledText(master, height=7, font=
('Consolas', 10))
self.output.pack(fill='x', padx=10, pady=5)

```

(5) 使用线程处理，避免 GUI 卡死

所有网络操作通过 `threading.Thread` 异步执行，确保界面不卡顿。

```

# ----- 线程 -----
def thread_connect(self):
    threading.Thread(target=self.connect, daemon=True).start()

def thread_login(self):
    threading.Thread(target=self.login, daemon=True).start()

def thread_get_list(self):
    threading.Thread(target=self.get_list, daemon=True).start()

def thread_download_mail(self):
    self.master.after(0, self.download_mail_prompt)

def download_mail_prompt(self):
    msg_id = simpledialog.askinteger("下载邮件", "输入邮件编号: ",
parent=self.master)
    if msg_id is None:
        return
    threading.Thread(target=self.download_mail, args=(msg_id,), ,
daemon=True).start()

def thread_quit(self):
    threading.Thread(target=self.quit_pop3, daemon=True).start()

```

三、主要POP3命令实现

(1) USER、PASS、STAT命令实现

- USER - 指定用户名，语法为 `USER <username>`。作为登录的第一步，用于向 POP3 服务器提交邮箱账号。
- PASS - 提交密码完成身份验证，语法为 `PASS <password>`。与 USER 配合完成用户身份认证，若认证成功，服务器进入 **事务状态 (TRANSACTION)**。
- STAT - 查看邮箱状态，语法为 `STAT`，该命令会返回如下内容：

```
+OK <邮件数量> <总字节数>
```

其作用是查询邮箱中邮件的数量与总大小。

本程序在登录过程中实现了这三个命令，代码如下：

```
def login(self):  
    try:  
        self.send_cmd("USER " + self.user_entry.get())  
        resp1 = self.recv_line()  
        self.send_cmd("PASS " + self.pass_entry.get())  
        resp2 = self.recv_line()  
  
        # 登录成功后调用 STAT 获取邮箱状态  
        if resp1.startswith("+OK") and resp2.startswith("+OK"):  
            self.get_stat()  
        else:  
            self.log("登录失败，请检查账号或密码")  
    except:  
        messagebox.showerror("登录失败", "账号或密码错误")
```

其中，STAT命令的具体实现函数为 `get_stat()`，具体代码如下：

```
def get_stat(self):  
    """获取邮箱状态：总邮件数和总大小"""  
    try:  
        self.send_cmd("STAT")  
        resp = self.recv_line()  
        if resp.startswith("+OK"):  
            parts = resp.split()  
            if len(parts) >= 3:  
                num_messages = parts[1]  
                total_size = int(parts[2])  
                if total_size > 1024 * 1024:  
                    size_str = f"{total_size / 1024 / 1024:.2f} MB"  
                elif total_size > 1024:  
                    size_str = f"{total_size / 1024:.1f} KB"  
                else:  
                    size_str = f"{total_size} B"  
                self.log(f"邮箱状态：共 {num_messages} 封邮件，总大小  
{size_str}")
```

```

        else:
            self.log("STAT 响应格式异常")
    else:
        self.log("STAT 命令失败")
except Exception as e:
    messagebox.showerror("错误", str(e))

```

(2) LIST命令实现

- LIST - 获取邮件编号与大小列表，语法为 LIST，服务器返回内容如：

```

+OK
1 1200
2 2500
.
.
```

其作用：

- 返回所有邮件的编号和大小（字节）
- 支持选择单个邮件编号查询其大小

代码实现如下：

```

def get_list(self):
    try:
        self.send_cmd("LIST")
        resp = self.recv_line()
        if not resp.startswith("+OK"):
            self.log("获取列表失败")
            return

        data = self.recv_multiline()
        mails = []
        for line in data.splitlines():
            line = line.strip()
            if line == "" or line == ".":
                continue
            parts = line.split()
            if len(parts) != 2:
                continue
            num, size = parts
            size_int = int(size)
            if size_int > 1024 * 1024:
                size_str = f"{size_int / 1024 / 1024:.2f} MB"
            elif size_int > 1024:
                size_str = f"{size_int / 1024:.1f} KB"
            else:
                size_str = f"{size_int} B"
            mails.append(f"{num} {size_str}")

        self.master.after(0, lambda: self.update_mail_list(mails))
        self.log(f"邮件列表获取完成, 共 {len(mails)} 封邮件")
    except Exception as e:

```

```
messagebox.showerror("错误", str(e))
```

(3) RETR命令实现

- RETR - 下载/获取邮件内容, 语法为 `RETR <邮件编号>`, 其作用:
 - 从服务器下载该邮件的完整内容 (包括头部、正文、附件编码)
 - 返回 **multiline response**, 以 `.\r\n` 结束

代码实现如下:

```
def download_mail(self, msg_id):  
    try:  
        self.send_cmd(f"RETR {msg_id}")  
        resp = self.recv_line()  
        if not resp.startswith("+OK"):  
            self.log("获取邮件失败")  
            return  
        data = self.recv_multiline()  
        path = f"mail_{msg_id}.eml"  
        with open(path, "wb") as f:  
            f.write(data.encode())  
        self.log(f"邮件已保存为: {path}")  
    except Exception as e:  
        messagebox.showerror("错误", str(e))
```

程序会将邮件保存为 `mail_xx.eml`, 并在预览界面展示解析后的正文内容。

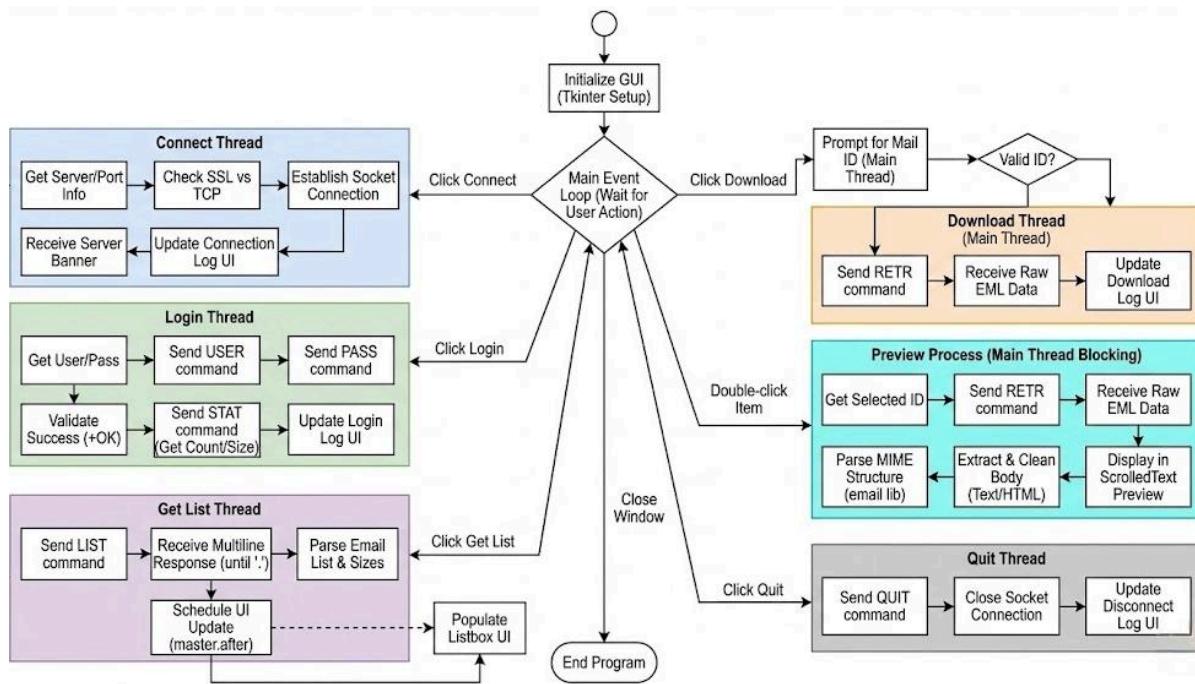
(4) QUIT命令实现

- QUIT - 结束会话并断开连接, 语法为 `QUIT`, 其作用:
 - 服务器返回 `+OK` 并关闭会话

代码实现如下:

```
def quit_pop3(self):  
    if self.sock:  
        self.send_cmd("QUIT")  
        self.recv_line()  
        self.sock.close()  
        self.sock = None  
        self.log("连接已关闭")
```

四、程序流程图



五、测试截图

- 初始时，输入POP3服务器地址、账号、密码、端口，界面如下：



- 点击连接和登陆按钮后，下方会显示相关交互过程：

- 连接种类
- 邮箱地址和密码
- 当前邮箱状态（邮件个数和总大小）

界面如下：

交互过程：

```
使用 SSL 连接...
S: +OK POP3 ready
C: USER 2120250723@mail.nankai.edu.cn
S: +OK send PASS
C: PASS fFh1U4K7d8x$KfrQ
S: +OK 6 message(s) [58897 byte(s)]
C: STAT
S: +OK 6 58897
邮箱状态：共 6 封邮件，总大小 57.5 KB
```

- 点击获取列表按钮后，下方依然会显示相关交互过程，并在左侧还会显示所获取的邮件列表（邮件编号和大小）：

1	16.6 KB
2	5.5 KB
3	1.5 KB
4	13.3 KB
5	3.0 KB
6	17.6 KB

交互过程：

```
C: LIST
S: +OK 6 messages
S: 1 17025
S: 2 5640
S: 3 1490
S: 4 13652
S: 5 3051
S: 6 18039
S: .
```

邮件列表获取完成，共 6 封邮件

- 双击左侧邮件列表中的邮件，可以在右侧预览邮件的正文内容，在下方依然显示相关交互过程

连接 登录 获取列表 下载邮件 退出 QUIT

1 16.6 KB	尊敬的用户：
2 5.5 KB	截止至 2025-12-02 00:00:00，您的邮箱账号（2120250723@mail.nankai.edu.cn）共收到异常邮件 1 封，其中：
3 1.0 KB	垃圾邮件 1 封
4 13.3 KB	系统隔离区 1 封
5 3.0 KB	广告文件夹 0 封
6 17.6 KB	病毒邮件 0 封

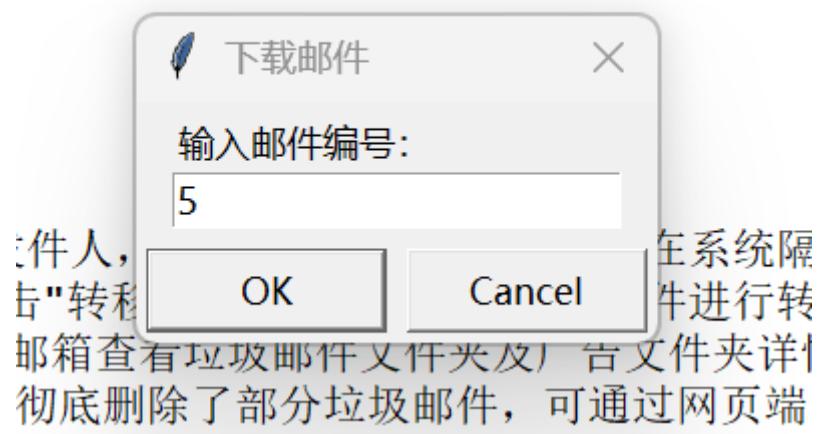
详情如下：

收件人：同
邮件主题：
发件人：
所在区域：
操作时间：(1) 在线操作的 48 小时内可操作
2025-12-01 21:37:19
关于向驾校考生开放AI学习平台高级权益的通知
admin@baldukar.com
系统隔离区
禁转至收件箱
提示：
(1) 被隔离邮件一般为系统认为垃圾邮件的邮件，会将相关邮件退回发件人。且由于安全考虑一般存放在系统隔离区，而非您的个人邮箱中，若需查看，请将此类邮件转移至收件箱。操作方法：登录网页版邮箱/自助查阅/微信查看/点击“转到”按钮后在右侧“操作”栏中选择“转到”→弹出的操作中选择“转到收件箱”进行转移。
(2) 若您是通过 Outlook 等客户端软件接收邮件，请登录网页邮箱后查看垃圾邮件是否已被转移。
(3) 本邮件中的数据均来自系统日志。如果您收到本邮件前已经彻底删除了部分垃圾邮件，可通过网页版邮箱/自助查询的“断信查询”功能对信件进行核对。
(4) 本邮件为系统邮件，无需回复。如果不再希望接收此邮件，可修改网页端邮箱“垃圾过滤设置”中的“垃圾过滤规则”为“不提防”。
网易企业服务
Dear User,
Until 2025-12-02 00:00:00, your email account (2120250723@mail.nankai.edu.cn) has received 1 abnormal email(s), among which:
0 in spam folder
1 in system quarantine area
0 in advertisement folder
0 via mail(s)
For more details:

```
C: RETR 6
S: +OK message follows
S: Received: from mail-118299.nginx.iteemail.com (unknown [115.236.118.209])
S:   by smtp.qiye.163.com (Hmail) with ESMTP id 27633e240
S:     for <2120250723@mail.nankai.edu.cn>; Tue, 2 Dec 2025 03:31:21 +0800 (GMT+08:00)
S: Received: from qlyesender-hz-4-entmail-virt173.qy.nts (qlyesender-hz-4-entmail-virt173.qy.nts [10.58.6.179])
S:   by qlyesender-hz-4-entmail-virt173.qy.nts (Postfix) with ESMTP id 8C203274002D
S:     for <2120250723@mail.nankai.edu.cn>; Tue, 2 Dec 2025 03:31:20 +0800 (GMT+08:00)
S: Date: Tue, 2 Dec 2025 03:31:20 +0800 (GMT+08:00)
S: From: notice@qiye.163.com
```

- 点击 下载邮件 按钮，会弹出窗口，输入邮件编号后，程序会将指定邮件保存为eml格式，默认保存在和程序相同的目录下：

- 弹出窗口并输入编号：



- 邮件保存为eml格式：

Project main.py mail_5.eml

```
Received: from njjs-sys-mailin03.njjs.baidu.com (unknown [180.101.52.76])
by smtp.qiye.163.com (Hmail) with ESMTP id 27633e240
for <2120250723@mail.nankai.edu.cn>; Tue, 28 Oct 2025 08:07:55 +0800 (GMT+08:00)
Received: from localhost (bddwd-feed-173555750.bddwd.baidu.com [10.88.64.38])
by njjs-sys-mailin03.njjs.baidu.com (Postfix) with ESMTP id 8C203274002D
for <2120250723@mail.nankai.edu.cn>; Tue, 28 Oct 2025 08:07:55 +0800 (CST)
Subject: =?UTF-8?B?55m+5bqm5a2m55SF6K6k6K+B?=
From: "百度认证" <certification@baidu.com>
To: 2120250723@mail.nankai.edu.cn
Cc:
Reply-To:
MIME-Version: 1.0
Content-Type: text/html; charset=UTF-8
Content-Transfer-Encoding: base64
Message-ID: <0251028080755.27633e240@smtp.qiye.163.com>
Date: Tue, 28 Oct 2025 08:07:55 +0800 (GMT+08:00)
X-HM-Tid: 0a9a28248ea90377kumm9ae2656116dfe2
X-HM-MType: 1
Received-SPE:pass (snfCheck: domain of baidu.com designates 180.101.52.76 as permitted sender)
```

- 交互过程也显示了命令 RETR 5：

交互过程：

```
C: RETR 5
S: +OK message follows
S: Received: from njjs-sys-mailin03.njjs.baidu.com (unknown [180.101.52.76])
S:   by smtp.qiye.163.com (Hmail) with ESMTP id 27633e240
S:     for <2120250723@mail.nankai.edu.cn>; Tue, 28 Oct 2025 08:07:55 +0800 (GMT+08:00)
S: Received: from localhost (bddwd-feed-173555750.bddwd.baidu.com [10.88.64.38])
S:   by njjs-sys-mailin03.njjs.baidu.com (Postfix) with ESMTP id 8C203274002D
S:     for <2120250723@mail.nankai.edu.cn>; Tue, 28 Oct 2025 08:07:55 +0800 (CST)
S: Subject: =?UTF-8?B?55m+5bqm5a2m55SF6K6k6K+B?=
S: From: "百度认证" <certification@baidu.com>
```

- 交互过程也显示了成功保存:

交互过程 :

```
S: ICAgIAogICAgICAgICAgICA8L3A+CgogICAgICAgICAgICA8cCBzdHlsZT0iY29sb3I6IGdyZXk7
S: Ij4KICAgICAgICAgICAgIOivpemCruS7tueUseeu+e7n+WPkemAge+8j0ivt+WLv+lbnuWk
S: je+8gQogICAgICAgICAgICA8L3A+CiAgICAgICAgPC9kaXY+CiAgICAgICAgPGRpdiBzdHlsZT0i
S: bWFyZ2luLXRvcDogNTBweDsipgogICAgICAgICAg55m+5bqm5a2m55sf6K6k6K+BPgJy
S: PgogICAgICAgICAgICAgMjAyNeW5tDEw5pyIMjjml6UKCiAgICAgPC9kaXY+CiAgICAg
S: L2JvZHk+CjwvaHRtbD4K
S:
S: .
邮件已保存为: mail_5.eml
```

- 点击退出 QUIT 按钮，则断开连接，交互过程显示如下:

交互过程 :

```
S: bWFyZ2luLXRvcDogNTBweDsipgogICAgICAgICAgICAg55m+5bqm5a2m55sf6K6k6K+BPgJy
S: PgogICAgICAgICAgICAgMjAyNeW5tDEw5pyIMjjml6UKCiAgICAgPC9kaXY+CiAgICAg
S: L2JvZHk+CjwvaHRtbD4K
S:
S: .
邮件已保存为: mail_5.eml
C: QUIT
S: +OK GoodBye
连接已关闭
```