# 案例一 安全邮件传送

**一、实验案例描述**

为了保护邮件在传输过程中的机密性（不被未授权者获取邮件内容），邮件收发方之间通过对邮件内容的加密实现邮件安全传送。

**二、实验目的**

1、了解OpenSSL 开源加密库；

2、了解 RSA签名算法

3、使用OpenSSL 开源库提供的算法接口实现简单的文件加密和解密。

**三、实验任务**

**（一）使用OpenSSL命令生成RSA 算法的公私钥对**

掌握OpenSSL生成RSA 算法公私钥对命令的使用，观察生成的公私钥文件格式，分析哪一部分是公私钥本身，并保存。

**（二）使用OpenSSL命令对文件进行加密**

掌握OpenSSL进行RSA加密的使用，使用第一步生成的公钥对简单的文件进行加密，使用python 通过QQ邮箱发送该文件。

**（三）使用OpenSSL命令对文件进行解密**

掌握OpenSSL进行RSA解密的使用，使用第一步生成的私钥对第二步邮箱收到的加密文件进行解密，观察返回信息，是否验证成功。

**四、实验环境与工具说明**

1、Ubuntu 20.04操作系统；

2、OpenSSL 1.1.1v。

**五、实验原理及技术**

**（一）OpenSSL开源库介绍**

OpenSSL是一个开源的软件库，主要用于提供安全通信过程中的加密功能。这个库广泛应用于互联网的各种应用中，特别是在实现HTTPS协议以及其他基于SSL（安全套接层）和TLS（传输层安全）协议的安全通信中。



**1、核心功能**

**加密和解密：**OpenSSL提供了多种加密算法的实现，包括对称加密算法（如AES和DES）和非对称加密算法（如RSA）。这使得它能够加密数据以保护其在不安全网络上的传输。

**哈希算法实现：**它还包括多种哈希函数（如SHA-1, SHA-256）的实现，这些用于验证数据的完整性和生成数字签名。

**数字签名和验证：**OpenSSL支持创建和验证数字签名，这对于确保数据的来源和完整性至关重要。

**2、SSL/TLS 协议**

**安全通信：**OpenSSL最为人所知的功能之一是实现SSL和TLS协议，为网络通信提供了加密层。这对于实现如HTTPS等安全协议至关重要。

**握手过程：**在SSL/TLS协议中，OpenSSL处理密钥交换、加密协商以及身份验证过程，这些都是建立安全通信通道所必需的。

**会话管理：**管理SSL/TLS会话，包括会话的建立、维护和结束。

**3、数字证书**

**证书生成：**OpenSSL能够生成公钥和私钥，以及使用这些密钥创建SSL/TLS数字证书。

**证书签名：**它可以用来签署证书请求，生成自签名证书或作为一个简单的证书颁发机构（CA）。

**证书验证：**它还提供了验证数字证书的功能，以确保证书的有效性和信任度。

**4、开源和社区**

**开源许可：**OpenSSL根据一个包容性的许可证发布，允许个人和组织广泛使用。

**社区驱动：**一个活跃的开发社区支持着OpenSSL，不断地改进代码、修复漏洞并添加新功能。

**5、安全性和合规性**

**漏洞管理：**OpenSSL团队积极响应安全漏洞，定期发布安全更新和补丁。

**合规性：**对于需要遵循特定安全标准和法规的组织，OpenSSL提供了一种可行的解决方案，尽管在某些情况下可能需要额外的配置或修改。

**6、使用场景和应用**

**网络浏览器和服务器：**几乎所有的现代网络浏览器和服务器都支持通过OpenSSL实现的SSL/TLS协议。

**邮件传输：**用于保护SMTP、POP3和IMAP等邮件传输协议。

**VPN和其他远程通信：**在VPN解决方案中常用于加密远程通信。

**物联网（IoT）设备：**在IoT设备中实现数据传输的安全。

**（二）RSA算法原理**

RSA（Rivest-Shamir-Adleman）算法是一种非对称加密算法，被广泛应用于加密、数字签名和密钥交换等领域。

下面是 RSA 算法的基本原理：

1. **密钥生成：**
2. 选择两个大素数 和 ，它们的乘积 就是 RSA 的模数。
3. 计算模数 的欧拉函数值 。
4. 选择一个整数 （通常选择素数），满足 ，且 与 互质。 称为公钥指数。
5. 计算 的模反元素 ，即 。 称为私钥指数。
6. **加密过程：**
7. 将明文消息 转换为整数 ，满足 。
8. 使用公钥，计算密文 。
9. **解密过程：**

使用私钥 ，计算密文 的解密结果 。

1. **安全性：**

RSA 的安全性基于大整数分解困难性，即给定一个大合数 ，找到其素因子 和 的乘积。即使已知公钥 和加密后的消息，也很难从中恢复出明文消息 ，除非已知私钥 。

**六、实验步骤**

RSA算法应用于于SMTP邮件发送场景：

（1）接收者生成两把密钥(公钥和私钥)。公钥是公开的，任何人都可以获得，私钥则是保密的。

（2）发送者获取接收者的公钥，然后用它对信息加密。

（3）接收者得到加密后的信息，用私钥解密。

**（一）生成RSA 公私钥对**

1. **使用命令生成私钥：**

openssl genpkey -algorithm RSA -out RSA\_private\_key.pem

**命令解释：**

使用 OpenSSL 生成一个 RSA 私钥，并将其保存到一个指定的文件中。我们逐个解释命令中的参数和它们的意义：

**openssl:**

这是 OpenSSL 工具的命令行执行程序。OpenSSL 是一个开源的密码学工具包，提供了许多加密和安全功能。

**genpkey:**

这是 OpenSSL 中的一个子命令，用于生成私钥。

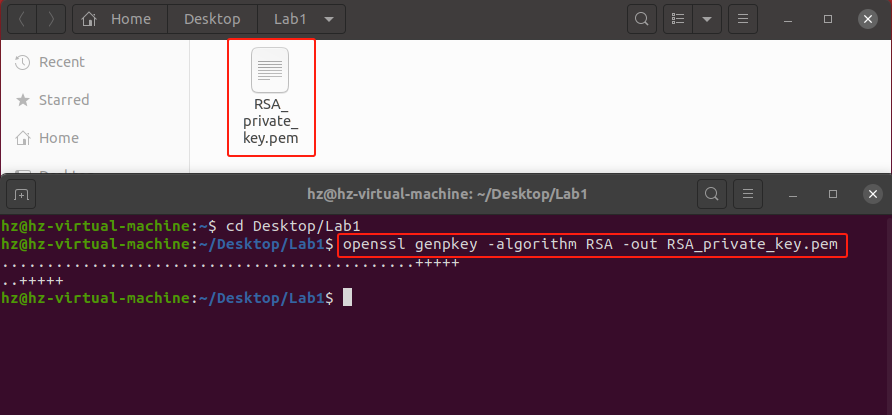
**-algorithm RSA:**

这个参数指定了要使用的算法，这里是 RSA。RSA 是一种非对称加密算法，通常用于生成密钥对，其中一个用作公钥，另一个用作私钥。

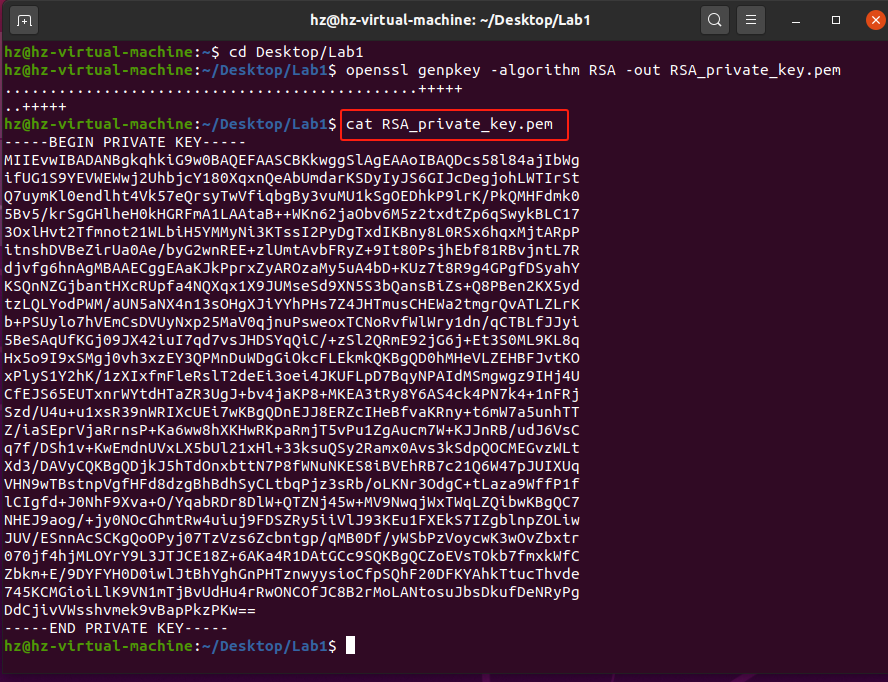
**-out RSA\_private\_key.pem:**

-out指定了生成的私钥文件的输出路径和文件名。在这种情况下，私钥将保存为名为 RSA\_private\_key.pem 的文件。

**运行截图：**



1. **使用 cat RSA\_private\_key.pem 查看生成的私钥内容**



1. **使用命令生成公钥：**

**openssl rsa -in RSA\_private\_key.pem -pubout -out RSA\_public\_key.pem**

**命令解释：**

从输入的 RSA 私钥文件中提取公钥，并将公钥保存到指定的文件中。这个公钥文件可以用于加密、验证签名等操作，而私钥文件用于解密和签名。让我们逐个解释命令中的参数及其意义：

**openssl:**

这是 OpenSSL 工具的命令行执行程序。

**rsa:**

这是 OpenSSL 中的一个子命令，用于执行 RSA 相关操作。

**-in RSA\_private\_key.pem:**

这个参数指定了要输入的私钥文件的路径和文件名。在这种情况下，输入的私钥文件是名为 RSA\_private\_key.pem 的文件。

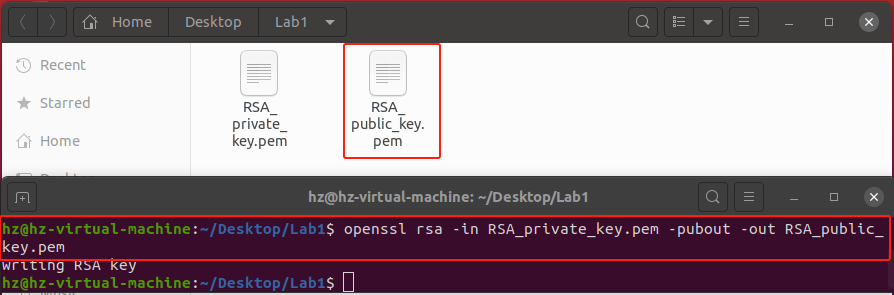
**-pubout:**

这个参数指示 OpenSSL 将从输入的私钥中提取公钥，并输出为一个新的文件。这个参数告诉 OpenSSL 生成公钥。

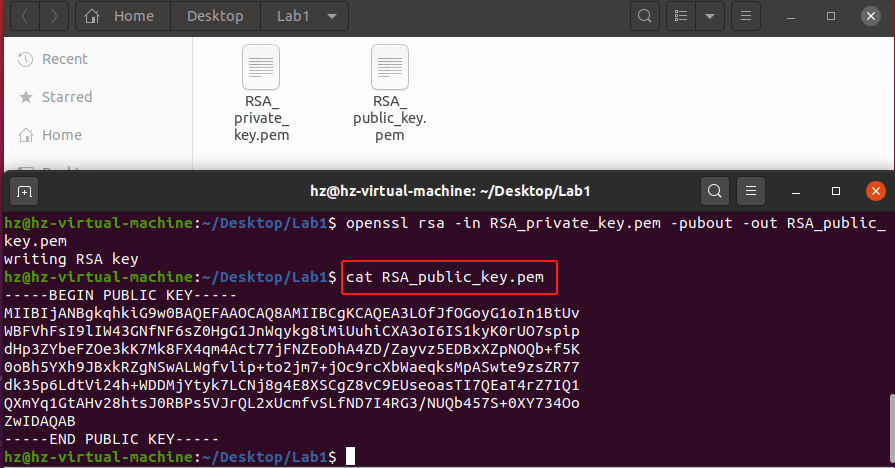
**-out RSA\_public\_key.pem:**

这个参数指定了生成的公钥文件的输出路径和文件名。在这种情况下，公钥将保存为名为 RSA\_public\_key.pem 的文件。

运行截图：



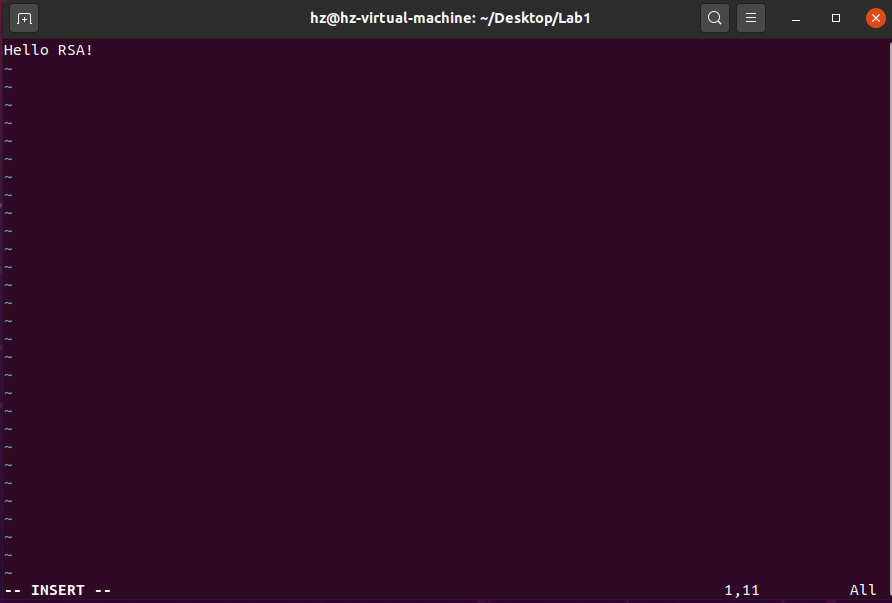
1. **使用 cat RSA\_public\_key.pem 查看生成的公钥内容：**



**（二）使用私钥对文件进行加密**

1. **创建样例文件：**

使用命令**vim plaintext\_email.txt**，在进入的编辑界面中按下**i**键，随后输入 **Hello RSA!**，输入完成后按**ESC**键，键入**:wq**退出vim编辑器。



1. **使用命令进行私钥加密：**

**openssl rsautl -encrypt -in plaintext\_email.txt -out** **encrypted\_email.txt -pubin -inkey RSA\_public\_key.pem**

**命令解释：**

整个命令的作用是使用指定的 RSA 公钥对名为 plaintext\_email.txt 的文本数据进行加密，并将加密后的数据保存为名为 encrypted\_email.txt 的文件中。下面是对命令中各个参数的详细解释：

**openssl:**

这是 OpenSSL 工具的命令行执行程序。

**rsautl:**

这是 OpenSSL 中用于 RSA 加密、解密和签名的实用工具。

**-encrypt:**

这个参数指示 OpenSSL 执行加密操作。

**-in plaintext\_email.txt:**

这个参数指定了要加密的原始文本数据的路径和文件名。在这种情况下，要加密的数据是名为 plaintext\_email.txt 的文件中的文本内容。

**-out encrypted\_email.txt:**

这个参数指定了加密后的数据的输出路径和文件名。在这种情况下，加密后的数据将保存为名为 encrypted\_email.txt 的文件。

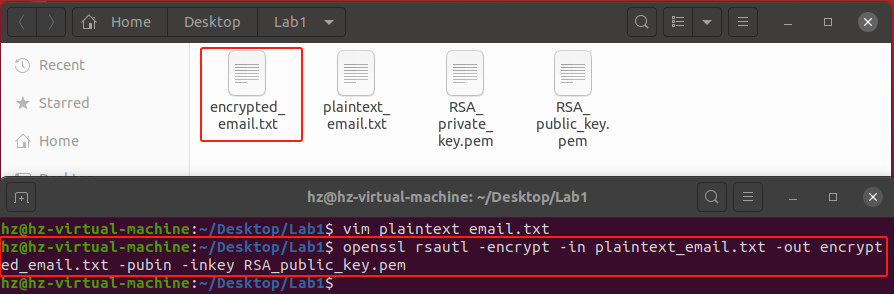
**-pubin:**

这个参数指示 OpenSSL 读取输入的密钥文件为公钥。在这个命令中，RSA\_public\_key.pem 是一个公钥文件。

**-inkey RSA\_public\_key.pem:**

这个参数指定了用于加密操作的公钥文件的路径和文件名。在这种情况下，使用的公钥是名为 RSA\_public\_key.pem 的文件中的公钥。

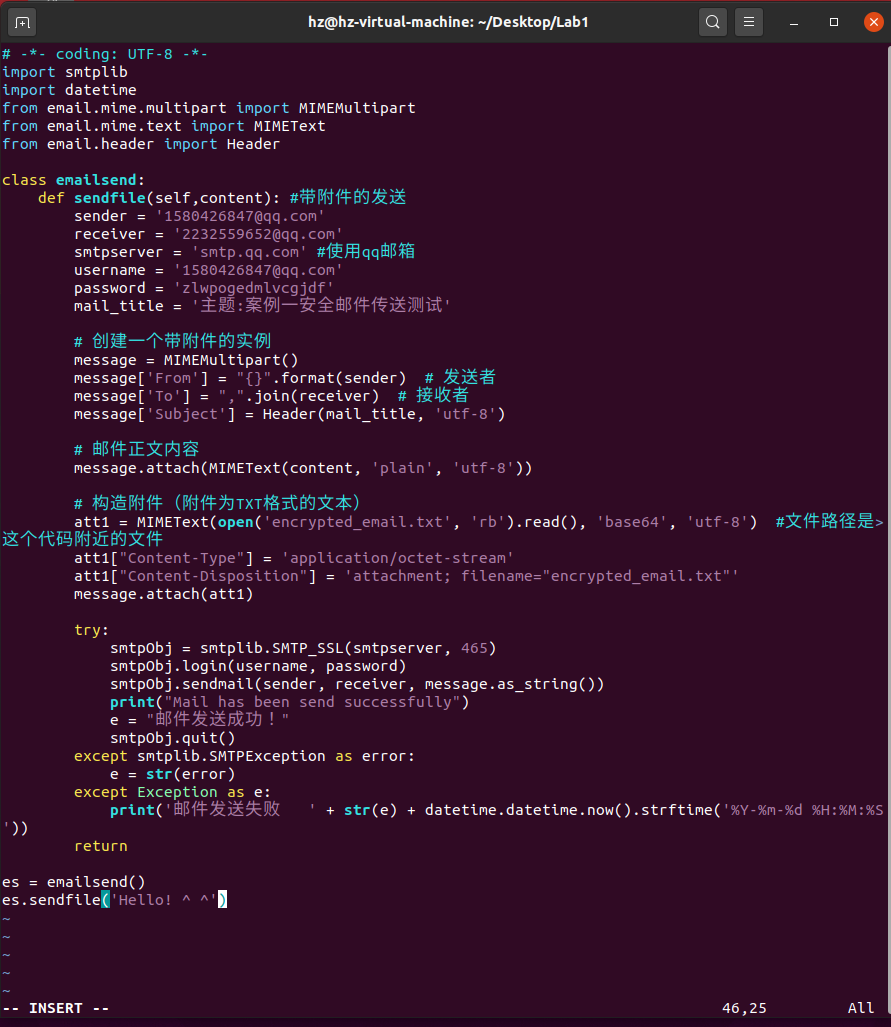
**运行截图：**



1. **编写python文件，功能是发送密文文件：**

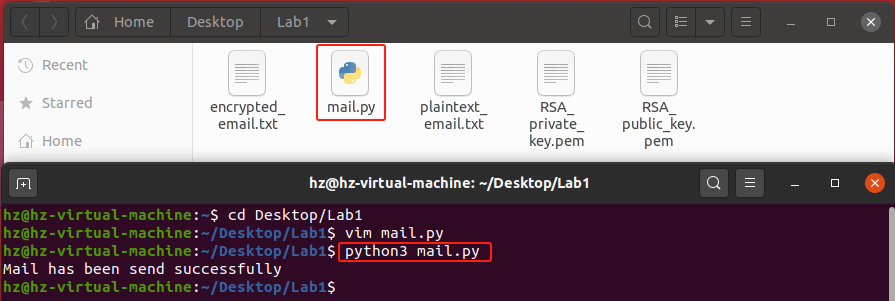
使用命令**vim mail.py**，在进入的编辑界面中按下**i**键，随后输入以下内容，输入完成后按**ESC**键，键入**:wq**退出vim编辑器。

|  |
| --- |
| # -\*- coding: UTF-8 -\*-  import smtplib  import datetime  from email.mime.multipart import MIMEMultipart  from email.mime.text import MIMEText  from email.header import Header  class emailsend:  def sendfile(self,content): #带附件的发送  sender = '1580426847@qq.com'  receiver = '2232559652@qq.com'  smtpserver = 'smtp.qq.com' #使用qq邮箱  username = '1580426847@qq.com'  password = 'zlwpogedmlvcgjdf'  mail\_title = '主题:案例一安全邮件传送测试'  # 创建一个带附件的实例  message = MIMEMultipart()  message['From'] = "{}".format(sender) # 发送者  message['To'] = ",".join(receiver) # 接收者  message['Subject'] = Header(mail\_title, 'utf-8')  # 邮件正文内容  message.attach(MIMEText(content, 'plain', 'utf-8'))  # 构造附件（附件为TXT格式的文本）  att1 = MIMEText(open('encrypted\_email.txt', 'rb').read(), 'base64', 'utf-8') #文件路径是这个代码附近的文件  att1["Content-Type"] = 'application/octet-stream'  att1["Content-Disposition"] = 'attachment; filename="encrypted\_email.txt"'  message.attach(att1)  try:  smtpObj = smtplib.SMTP\_SSL(smtpserver, 465)  smtpObj.login(username, password)  smtpObj.sendmail(sender, receiver, message.as\_string())  print("Mail has been send successfully")  e = "邮件发送成功！"  smtpObj.quit()  except smtplib.SMTPException as error:  e = str(error)  except Exception as e:  print('邮件发送失败 ' + str(e) + datetime.datetime.now().strftime('%Y-%m-%d %H:%M:%S'))  return  es = emailsend()  es.sendfile('Hello! ^ ^') |



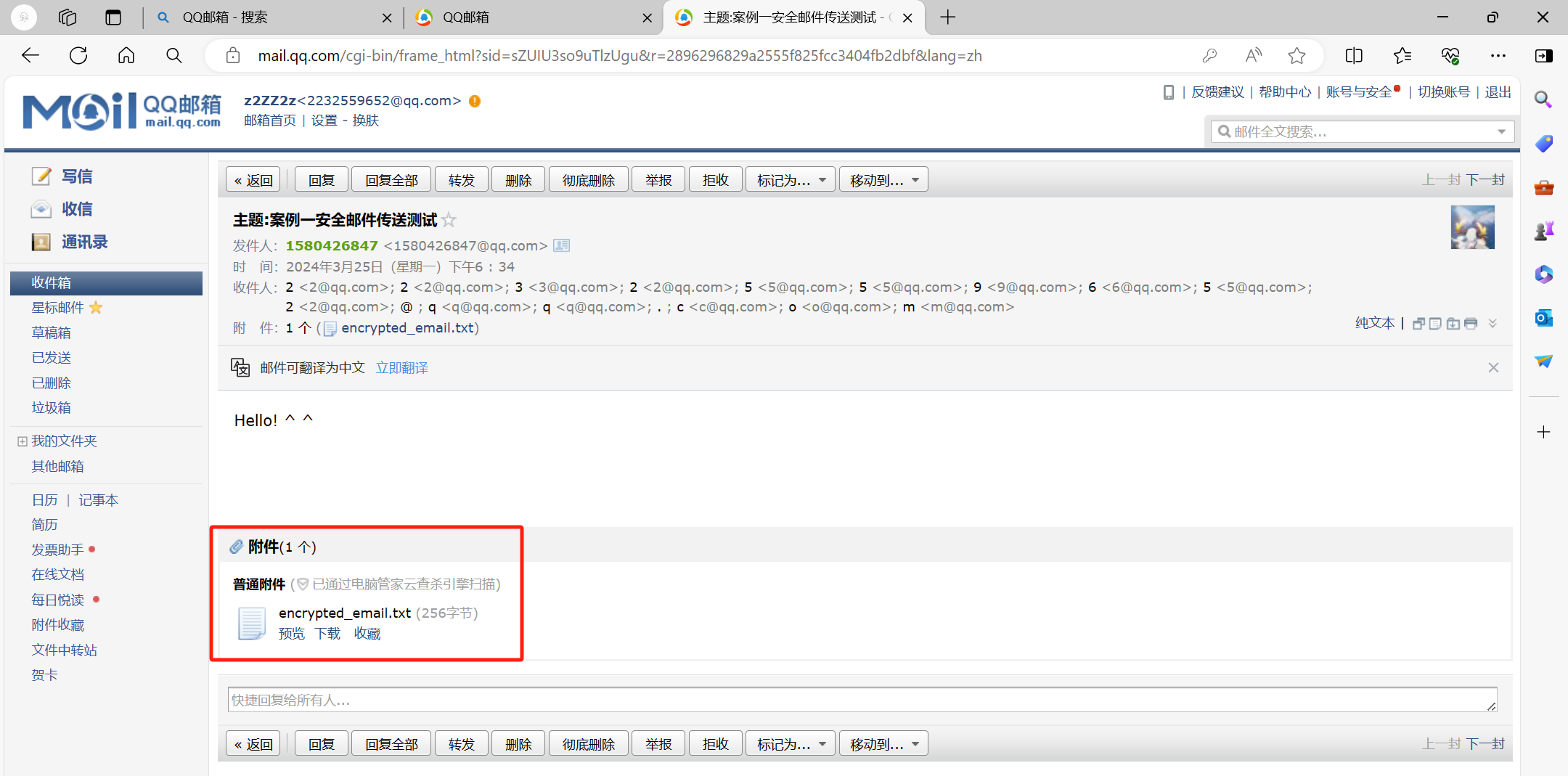
1. **输入python3 mail.py进行编译**

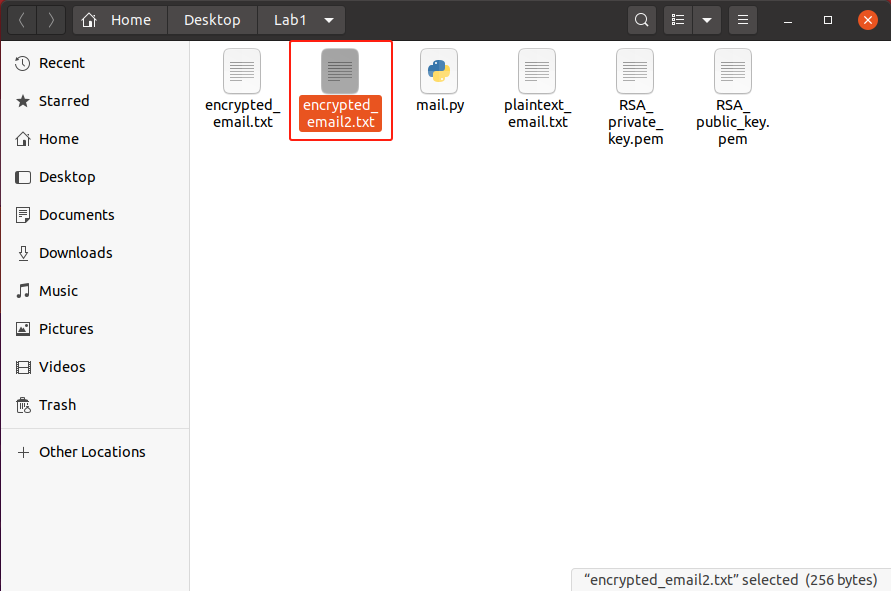
显示“Mail has been send successfully”，表示邮件成功发送。



**（三）使用公钥对文件进行解密**

1. **接受方收到邮件，将附件重命名后下载到ubuntu：**





1. **接收方用私钥对附件进行解密**

**使用命令：**

**openssl rsautl -decrypt -in encrypted\_email2.txt -out decrypted\_email.txt -inkey RSA\_private\_key.pem**

**命令解释：**

整个命令的作用是使用指定的 RSA 私钥对名为 encrypted\_email2.txt 的数据进行解密，并将解密后的数据保存为名为 decrypted\_email.txt 的文件中。命令中的每个参数具有特定的意义和作用：

**openssl:**

这是 OpenSSL 工具的命令行执行程序。

**rsautl:**

这是 OpenSSL 中用于 RSA 加密、解密和签名的实用工具。

**-decrypt:**

这个参数指示 OpenSSL 执行解密操作。

**-in encrypted\_email2.txt:**

这个参数指定了要解密的已加密数据的路径和文件名。在这种情况下，要解密的数据是名为 encrypted\_email2.txt 的文件中的内容。

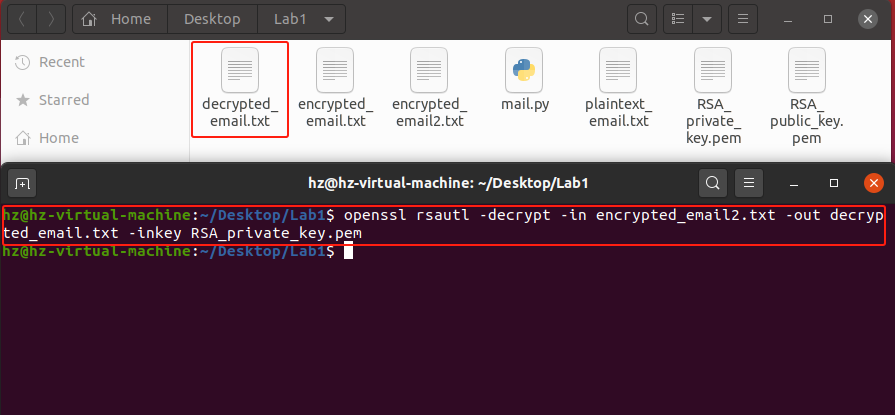
**-out decrypted\_email.txt:**

这个参数指定了解密后的数据的输出路径和文件名。在这种情况下，解密后的数据将保存为名为 decrypted\_email.txt 的文件。

**-inkey RSA\_private\_key.pem:**

这个参数指定了用于解密操作的私钥文件的路径和文件名。在这个命令中，RSA\_private\_key.pem 是一个私钥文件。

**运行截图：**



打开解密后的文件decrypted\_email.txt，内容与明文plaintext\_email.txt一致。

