山东大学网络空间安全学院

Python高级程序设计课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：202100460065 | 姓名：李昕 | | 班级：21级密码2班 |
| 实验题目：实验11. Python cryptography或pycryptodome模块使用 | | | |
| 实验学时：2 | | 实验日期：2022/12/13 | |
| 实验目的：熟悉模块的用法 | | | |
| 硬件环境：  AMD Ryzen 7 5800H with Radeon Graphics     3.20 GHz  机带 RAM 16.0 GB (13.9 GB 可用 | | | |
| 软件环境：  系统：Windows11  编译器：IDLE | | | |
| 实验步骤与内容：  本次实验包括两个内容：  1、使用cryptography模块，编写完整的AES-CBC加解密函数，函数接口为：  def encrypt\_CBC(key, plaintext, iv)、def decrypt\_CBC(key, ciphertext, iv)；  2、使用pycryptodome模块，编写程序，实现RSA-OAEP加解密；  【cryptography模块编写AES-CBC加解密函数】  1、在cryptography库中，对称加密算法的抽象是fernet模块，包括了对数据的加解密以及签名验证功能，以及密钥过期机制；AES加密数据块分组长度必须为128比特，密钥长度可以是128比特、192比特、256比特中的任意一个（如果数据块及密钥长度不足时，会补齐）。AES加密有很多轮的重复和变换。  2、输入的数据应当为字节串类型，在函数执行时，使用isinstance函数判断输入的数据的类型，如果不是指定类型则将其转换为字节串；  3、使用os.urandom随机生成密钥和CBC初始向量；  4、指定padding方式为PKCS7，把要加密的原始明文plaintext用padding方式补齐，加密算法使用AES，分组密码模式使用CBC；  5、AES加密过程是在一个4×4的字节矩阵上运作，这个矩阵又称为“状态（state）”，其初值就是一个明文区块（矩阵中一个元素大小就是明文区块中的一个Byte）  代码实现：   1. import os 2. import cryptography 3. import cryptography.hazmat.primitives.ciphers.aead 4. from cryptography.hazmat.primitives import padding 5. from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes 6. from cryptography.hazmat.backends import default\_backend 7. class AESCrypto(object): 8. @classmethod 9. def encrypt(cls, data, mode='cbc'): 10. func\_name = '{}\_encrypt'.format(mode) 11. func = getattr(cls, func\_name) 12. return func(data) 13. @classmethod 14. def decrypt(cls, data, mode='cbc'): 15. func\_name = '{}\_decrypt'.format(mode) 16. func = getattr(cls, func\_name) 17. return func(data) 18. @staticmethod 19. def pkcs7\_padding(data): 20. if not isinstance(data, bytes): 21. data = data.encode() 22. padder = padding.PKCS7(algorithms.AES.block\_size).padder() 23. padded\_data = padder.update(data) + padder.finalize() 24. return padded\_data 25. @classmethod 26. def encrypt\_CBC(cls,key,plaintext,iv): 27. if not isinstance(plaintext, bytes): *#判断数据是否是字节串类型* 28. plaintext = plaintext.encode() 29. cipher = Cipher(algorithms.AES(key), 30. modes.CBC(iv), 31. backend=default\_backend()) 32. encryptor = cipher.encryptor() 33. padded\_data = encryptor.update(cls.pkcs7\_padding(plaintext)) 34. return padded\_data 35. @classmethod 36. def decrypt\_CBC(cls, key,ciphertext,iv): 37. if not isinstance(ciphertext, bytes): 38. ciphertext = ciphertext.encode() 39. cipher = Cipher(algorithms.AES(key), 40. modes.CBC(iv), 41. backend=default\_backend()) 42. decryptor = cipher.decryptor() 43. uppaded\_data = cls.pkcs7\_unpadding(decryptor.update(ciphertext)) 44. uppaded\_data = uppaded\_data.decode() 45. return uppaded\_data 46. @staticmethod 47. def pkcs7\_unpadding(padded\_data): 48. unpadder = padding.PKCS7(algorithms.AES.block\_size).unpadder() 49. data = unpadder.update(padded\_data) 50. try: 51. uppadded\_data = data + unpadder.finalize() 52. except ValueError: 53. raise Exception('无效的加密信息!') 54. else: 55. return uppadded\_data 56. a=AESCrypto() 57. key=os.urandom(32) 58. iv=os.urandom(16) 59. plaintext=input("please enter the plaintext:") 60. print(a.encrypt\_CBC(key, plaintext, iv)) 61. ciphertext=a.encrypt\_CBC(key, plaintext, iv) 62. print(a.decrypt\_CBC(key, ciphertext, iv))   执行结果：    【pycryptodome模块，编写程序，实现RSA-OAEP加解密】  1、pycryptodom提供了各种加密方式对应的多种加密算法的实现，包括 单向加密、对称加密以及公钥加密和随机数操作；  2、RSA非对称加密：加密变换为：；解密变换为：。  3、使用Cryto.Random 产生随机密钥，Crypto.Cipher支持对称加密  代码实现：   1. from Crypto.PublicKey import RSA 2. from Crypto.Cipher import PKCS1\_OAEP 3. import time*#生密钥* 4. start = time.time() 5. for i in range(100): key = RSA.generate(1024) 6. end = time.time() 7. print('generatekey time:%d circles using %f seconds'%(100, end-start))*# 提取私钥并存入文件* 8. private\_key = key.export\_key() *#提取私钥* 9. file\_out = open("private\_key.pem", "wb") 10. file\_out.write(private\_key) 11. file\_out.close()*# 提取公钥存入文件* 12. public\_key = key.publickey().export\_key() *#提取公钥* 13. file\_out = open("public\_key.pem", "wb") 14. file\_out.write(public\_key) 15. file\_out.close() 16. *#加密* 17. data = b"123456" 18. public\_key = RSA.import\_key(open("public\_key.pem").read()) *#读取公钥* 19. cipher = PKCS1\_OAEP.new(public\_key) *# 实例化加密套件* 20. encrypted\_data = cipher.encrypt(data)*# 加密* 21. *# 将加密后的内容写入到文件* 22. file\_out = open("encrypted\_data.bin", "wb") 23. file\_out.write(encrypted\_data) 24. file\_out.close() 25. *#解密* 26. private\_key = RSA.import\_key(open("private\_key.pem", "rb").read()) *#读取私钥* 27. cipher = PKCS1\_OAEP.new(private\_key) *# 实例化加密套件* 28. encrypted\_data = open("encrypted\_data.bin", "rb").read() *# 读取加密内容* 29. data = cipher.decrypt(encrypted\_data)*# 解密* 30. print(data)   执行结果： | | | |
| 结论分析与体会：  通过本次练习，我熟悉了解了Python的密码学编程知识，包括对称密码AES非对称密码RSA的加密和解密过程。同时，我还深入了解了cryptography等模块，熟悉了这部分模块的部分用法。 | | | |