Github 账号：sptttt

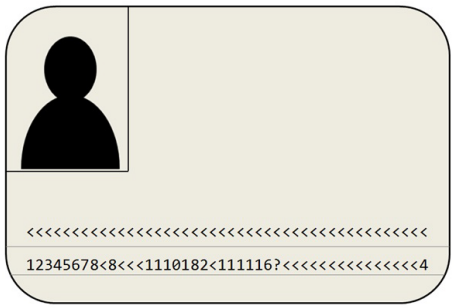
实验摘要：

根据题目描述的电子护照信息查找相关资料获取密钥计算方法，计算密钥，解密消息。锻炼代码能力、英文理解能力，了解密码学的一个实际应用。

实验题目

AES 密钥—在欧洲护照的机读区中使用

一个由 AES 加密过的消息已转发给您（CBC 加密模式，初始向量为 0，填充模式为 01-00）。此外，您还收到了对应的密钥信息，此密钥信息是以身份证明文件上使用的机器可读区域（MRZ）的形式给出的，例如欧洲使用的 ePassports。很遗憾，您收到的密钥信息不太完整。



目的是找到以下 base64 编码消息的明文。

9MgYwmuPrjiecPMx61O6zIuy3MtIXQQ0E59T3xB6u0Gyf1gYs2i3K9Jxaa0zj4gTMazJuApwd6+jdyeI5iGHvhQyDHGVlAuYTgJrbFDrfB22Fpil2NfNnWFBTXyf7SDI

为了加密，已生成并应用了基于基本访问控制协议（BAC）的密钥 K_{ENC}。为了解密，以下字符已经传送给您，从这些字符中可以导出 K_{ENC}，（参考文献[1]描述了这些字符的编码规则）：

12345678<8<<<1110182<111116?<<<<<<<<<<<<<<<4

不幸的是，在传输过程中丢失了一个字符，并用“？”突出显示了该字符。不过，您可以在[2]的帮助下计算出“？”。为了以后能够计算密钥 K_{ENC}，您可以在[3]，[4]和[5]中找到其使用的密钥生成协议。

AES 加密的原始消息包含要输入“密语”，那就是此题目的解答。

参考文献

可以在线获取以下文档：https://www.icao.int/

[1] ICAO MRTD DOC 9303 Part 1 Vol 1, p. IV-16 (Data structure of the lower machine readable line) and p. IV-42

[2] ICAO MRTD DOC 9303 Part 1 Vol 1, p. IV-24 to IV-26 (Check digits in the machine readable zone)

[3] ICAO MRTD DOC 9303 Part 1 Vol 2, p. IV-13 (MRTD Basic Access Control)

[4] ICAO MRTD DOC 9303 Part 1 Vol 2, p. IV-32
[5] ICAO MRTD DOC 9303 Part 1 Vol 2, p. IV-40 IV-41

实验内容

一 实验基本原理及步骤

1、资料获取

在国际民航组织官网上搜索 DOC 9303，下载中文版参考文档，方便阅读。

2、求未知字符“？”

阅读[1]可知，字符串‘12345678<8<<<1110182<111116?<<<<<<<<<<<<<<<4’中，

1 到 9 位为护照号码，第 10 位为其校验位；

14 到 19 位为出生日期，第 20 位为其校验位；

22 到 27 位为到期日，第 28 位为其校验位。

未知字符“？”是 28 位，即“？”是 22 到 27 位“111116”的校验位，文献[2]给出了校验位的计算方法，计算要点摘录如下：

校验位须在模数 10 的基础上进行计算，不断地重复 731 731……加权。

- 步骤 1 从左到右，相应顺序位置上的加权数乘相关数字数据要素的每一位数。
- 步骤 2 将每次乘法运算的乘积相加。
- 步骤 3 将得出的和除以 10（模数）。
- 步骤 4 余数即为校验位。

根据以上计算方法，可以手工计算“111116”的校验位，(1*7 + 1*3 + 1*1 + 1*7 + 1*3 + 6*1) % 10 = 7，即未知字符“？”为 7。

至此，恢复出了字符串‘12345678<8<<<1110182<1111167<<<<<<<<<<<<<<4’

3、计算 K_{ENC}

[3] [4] [5]描述了 K_{ENC} 的导出过程。

[3]描述了 MRZ_information 是字符串中证件号码、出生日期和到期日，并包括它们各自的校验位组合而成的。按照这些信息对应的位置提取子字符串，并重新组合，可得到 MRZ_information。

```
MRZ_line2 = b'12345678<8<<<1110182<1111167<<<<<<<<<<<<<<4'
Number_and_check = MRZ_line2[:10]
Birth_and_check = MRZ_line2[13:20]
Expiry_and_check = MRZ_line2[21:28]
mrz_information = Number_and_check + Birth_and_check + Expiry_and_check
```

另外，[3]描述了密钥种子 (K_{seed}) 是“MRZ_information”的 SHA-1 散列的前 16 字节。

```
K_seed = hashlib.sha1(mrz_information).hexdigest()[:32]
```

[4]描述了接下来的密钥生成方法，按照文件中的步骤操作即可。

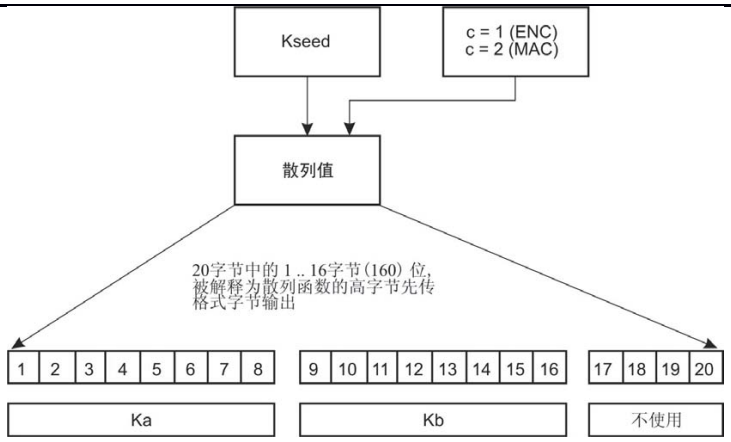


图 1 根据密钥种子计算密钥

1. D 为 K_{seed} 和 c 的拼接 ($D = K_{seed} \| c$)。（在加密中 c = 0x 00 00 00 01）
2. 计算 H = SHA-1(D)，即：H 为 D 的 SHA-1 散列。
3. H 的字节 1..8 形成密钥 Ka，H 的字节 9..16 形成密钥 Kb。
4. 调整密钥 Ka 和 Kb 的奇偶校验位形成正确的 DES 密钥。

其中，1 到 3 步的描述很明确，实现如下：

```
K_seed = hashlib.sha1(mrz_information).hexdigest()[:32]
D_hex = K_seed + '00000001'
Hash_D = hashlib.sha1(binascii.a2b_hex(D_hex)).digest()

K_a = Hash_D[:8]
K_b = Hash_D[8:16]
```

第 4 步的调整奇偶校验位不是很明确。可以阅读[5]的处理过程实例分析这一步的操作。

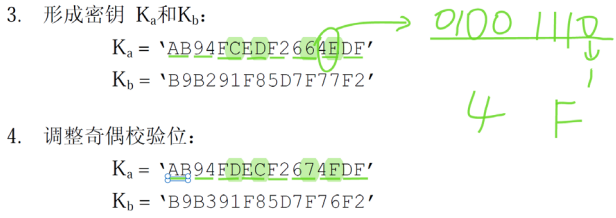


图 2 调整奇偶校验位实例

观察发现，字符串按字节进行奇偶校验位的调整，所谓奇偶校验位就是每个字节 8 位中的最后 1 位，需要改变此位使得每个字节的二进制表示中有奇数个“1”。了解了其调整过程，便可编程实现，计算调整后的 Ka 和 Kb，进而得到密钥 K。

```
def adjust_parity_bit(bytestocheck):
    res = b''
    for x in bytestocheck:
        if(bin(x).count('1') % 2 == 0):
            x = x ^ 1
        res += x.to_bytes(1, 'big')
    return res

K_a_adjust = adjust_parity_bit(K_a)
K_b_adjust = adjust_parity_bit(K_b)
key = K_a_adjust + K_b_adjust
```

4、解密

初始的密文需要先进行 base64 解密得到密文的字节码，加密模式为 CBC，初始向量为 0，调用 AES 解密可得到解密结果。

```
1. cipher_base64 = '9MgYwmuPrjiecPMx61O6zIuy3MtIXQQ0E59T3xB6u0Gyf1gYs2i3K9Jxaa0zj4gTMazJuApwd
   6+jdyeI5iGHvhQyDHGVlAuYTgJrbFDrfB22Fpil2NfNnWFBTXyf7SDI'
2. cipher_bytes = base64.b64decode(cipher_base64)
3. IV = b'\x00'*16
4. m = AES.new(key, AES.MODE_CBC, IV).decrypt(cipher_bytes)
```

解密结果为：b'Herzlichen Glueckwunsch. Sie haben die Nuss geknackt. Das Codewort lautet: Kryptographie!\x01\x00\x00\x00\x00\x00\x00\x00'

手动移除 01-00 填充，可得到明文 Herzlichen Glueckwunsch. Sie haben die Nuss geknackt. Das Codewort lautet: Kryptographie!

二 实验结果

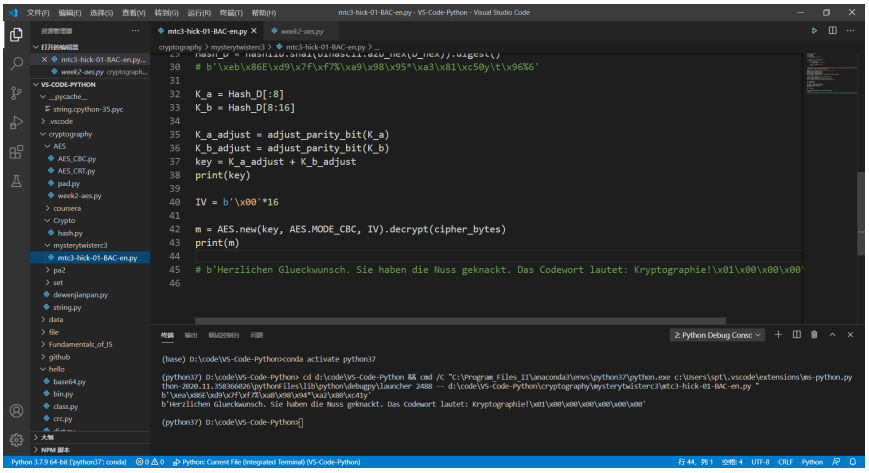


图 3 实验结果

三 实验结果的分析

尝试翻译该明文，发现翻译软件提示其为德文。大意为：恭喜你，你破解了难题。该密语是：密码学！

实验总结

1、英文的题目描述不易明白，需要利用软件翻译、查单词等手段帮助理解。认真阅读，理解题意。

2、题目参考文档的内容很多，需要只看有用的部分节约时间。

参考文献

[6] Koz_0. mysterytwisterc3-challenge-AES key — encoded in the machine readable zone of a European ePassport. [EB/OL]. (2020-11-07)[2020-11-12]. https://blog.csdn.net/Koz_0/article/details/109540921

[7] Joel-Q-Xu. MT3-mysterytwisterc3. [DB/OL]. (2019-05-10)[2020-11-12]. https://github.com/Joel-Q-Xu/MT3-mysterytwisterc3