Assignment 2 - Block/Stream

Cryptography

Robert Boerwinkle

Spring 2022-23

0 Introduction

This project involves decoding several (corny) messages and performing other basic cryptographic functions. These are implemented in Python, and the scripts can be found under `[heading number].py`. Unless otherwise noted, all strings are in encoded in ASCII.

1 XOR cipher

```
5132071002060A0B011005010B520C1F53051D0C5202170D10051C0A17520A0A530507081C01030301
1C1C0715520C02151E070413060C031D511C07061D450D53171A1B1F52110412055500015210020114
140D1310090953051A49131C1C031D14550C0A11001C075101011D01004C0418010152060D09530210
0A0017114C181F1A1E1E17010B16511B0C17160008530 51A49071C0903101A5500065C452507560649
1E1B0E095310551A171117090751160616174 94C1609160C020645181B1001491C1D07031708551C1C
16001E000514071601450507 5D55071D06450905141B49061A004C03141A191E17451B1B1E55041316
004C1A055B4B525F45391D1A1B06051C
```

This string of characters is binary data encoded in base16, because the raw data has no good representation with only printable characters. It has been encrypted with an xor cipher with the repeated key `squirrel`. This cipher repeats the key as many times as needed to reach the same amount of bytes as the text, and then bitwise xor's the message and the key. Because xor is reversible with the same key, encoding and decoding are the same operation. This message comes out to:

```
"Cryptography is the practice of transforming information into a form that is
unreadable to anyone except those with the secret knowledge needed to unlock it.
It's like a secret code, except that nobody understands it, not even the people
who made it." - Unknown
```

2 Seed Finding

```
import random

random.seed(xxxxx)
out=[]
for i in range(20):
        out.append(random.randint(1,8))
print(out)
```

The code presented was used to generate the following series of numbers: $7, 1, 5, 6, 4, 5, 7, 3, 6, 2, 7, 5, 3, 3, 6, 2, 7, 3, 3, 4$. This was put into a simple loop which varied the seed, and a seed value of 12345 was found. This produced the sequence: $7, 1, \cdots, 3, 4, 2, 4, 6, 6, 1$

3 DES in ECB Mode

This section deals with a custom encoding method which uses DES with 2 byte blocks in ECB mode. ECB mode operates on each block the same, regardless of the other blocks, creating what is effectively a substitution cipher, and when given part of the plaintext the rest can be found be using the substitutions already determined.

Encrypted message (hex):

```
d8d8 4783 f93e 4783 49aa 2dd5 06bc eec9 5c8e 5b76 b56d 9df6 1f8b 834d f93e 4783
7ed1 b269 8650 be31 78c2 7ed1 90e4 599c aaff 1368 834d 6f62 73ab 09a0 dad4 b804
b5b1 2422 ade5 4783 a0c0 ec9b 48e9 834d 09a0 1368 b804 b5b1 2422 78c2 7ed1 ade5
4783 d8d8 48e9
```

Partial plain text:

```
use the toilet paper roll  the otter is not a slob  respect should be earned
```

In this example, the bytes "d8d8"(hex) will always encrypt to "us". By following all of these substitutions, we can find the final sentence:

```
  ecb should not be used
```

4 Brute Forcing 2DES (ECB mode)

2DES is simply running DES twice with 2 different keys. In this case, the keys were restricted to 8-byte combinations of the characters 'a', 'b', and 'c'. Each combination was tried, and the resulting plain text was tested for printability.

Encrypted message (base64):

```
5gUYhK/XODuBERdDL43+CW1mihOtxbvTEgHrADRuiWVYw8CTN5kpL1mbpGNiE3a6W81cFZyz9DO3r4bVqK
UvFwqTU6Nsa+QOOyX7M3jBMVZ7xBwakuRMSYJ0a/EjTbI5GrRWkQLG5GNytHtiv5BU0bxdEbwoLxc2jk85
2c5l+8Bs9Y3Dobcxg+S1GeDgcJs797ZNYWsDbKe8XNoFxp9+SQ==
```

Plain text:

```
"Cryptography is the art of making it look like you're doing something really
clever, when in fact you're just making everyone's life more difficult." - Unknown
```

2048 key pairs were found (`keypairs.txt`). The first is the pair `bbaababa` and `bbabbbbb`. The significance of this number being a power of 2 is currently not understood.

5 Column Mixing

In order to achieve 'diffusion' in AES, an operation known as column mixing is performed. In this opperation, bytes are considered a polynomial where each term's coeffecient is the bit at the position given by the exponent (`0101` becomes $0x^3 + 1x^2 + 0x + 1$),then 4 bytes (in a 4x1 matrix) can be multiplied by a matrix that looks like this:

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

In this context, addition is replaced with the xor opperation. Multiplication is normal polynomial multiplication performed under a modulo of $x^8 + x^4 + x^3 + x + 1$. The resulting polynomials are then converted back into binary. This was used to convert the bytes `0xdeadbeef` to `0x1aa93eaf`.