CS 4823: Homework #8

Due on March 30, 2018

Christopher Tse

Problem 1

Suppose that a Hill cipher with alphabet 0,1 and block length 3 is used to encrypt messages. And suppose that we discover three plaintext-ciphtertext pairs:

$$(100) \rightarrow (101),\ (110) \rightarrow (110),\ (111) \rightarrow (001)$$

Recover the encryption key.

Solution

Since Hill Ciphers are in the form $C = KP$ where C is ciphertext, K is the key, and P is the plaintext, we can set up a matrix equation with the given pairs as such:

$$K \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

To find K, we must first ensure P is invertible. Since det(P) is not 0, it is invertible.

We then determine the inverse of P, P^{-1}, which gives:

$$P^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

Since KP = C, then $CP^{-1} = K$

$$\begin{pmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

Problem 2

Explain why in the AES S-box, the hexadecimal number `0x93` is substituted by `0xdc`. Please show step-by-step calculations

Solution

To find the S-box substitutions, we first find the inverse of our desired value over GF(2) and find the binary representation.

$$\texttt{0x93} \rightarrow \texttt{0x6D} = \texttt{0b01101101}$$

We then perform logical AND on this binary representation with the affine transformation matrix as follows:

```
Input  1 0 1 1 0 1 1 0 (LSB First)
Row 0  1 0 0 0 1 1 1 1
Bit 0  1 0 0 0 0 1 1 0 = 1

Row 1  1 1 0 0 0 1 1 1
Bit 1  1 0 0 0 0 1 1 0 = 1

Row 2  1 1 1 0 0 0 1 1
Bit 2  1 0 1 0 0 0 1 0 = 1

Row 3  1 1 1 1 0 0 0 1
Bit 3  1 0 1 1 0 0 0 0 = 1

Row 4  1 1 1 1 1 0 0 0
Bit 4  1 0 1 1 0 0 0 0 = 1

Row 5  0 1 1 1 1 1 0 0
Bit 5  0 0 1 1 0 1 0 0 = 1

Row 6  0 0 1 1 1 1 1 0
Bit 6  0 0 1 1 0 1 1 0 = 0

Row 7  0 0 0 1 1 1 1 1
Bit 7  0 0 0 1 0 1 1 0 = 1
```

Writing the binary result with MSB First, we get `0b10111111`, or `0xbf`. Finally, we XOR this result with `0x63` to get the s-box substitution value:

$$\texttt{0xbf} \oplus \texttt{0x63} = \texttt{0xdc}$$

Problem 3

Suppose the current state matrix before the AES MixColumns transformation is

$$\begin{pmatrix} O & K & L & A \\ H & O & M & A \\ I & L & L & I \\ N & O & I & S \end{pmatrix}$$

(each letter is encoded as a byte according to the ASCII table), write a program to calculate the output state after the MixColumns transformation.

Solution

Solution begins on next page.

Problem 3 continued on next page. . .

```
// mixcolumns.sage
from sage.crypto.mq.rijndael_gf import RijndaelGF
from sage.crypto.util import bin_to_ascii

def my_mix_columns(string):
    """
    Takes an input string and performs AES
    MixColumns transform on it

    Arguments:
        string {str} -- Input string

    Throws:
        ValueError -- Input string must be 16 characters long
    """
    if len(string) != 16:
        raise ValueError("Input string must be 16 characters long")
        return

    rgf = RijndaelGF(4, 4)

    s = HexadecimalStrings().encoding(string)

    state = rgf._hex_to_GF(str(s))
    result = rgf.mix_columns(state)
    line = rgf._GF_to_hex(result)

    n = 2
    split = [line[i:i+n] for i in range(0, len(line), n)]

    def parse_and_mod(n):
        n = int(n, base=16)
        hexval = str(hex(n))
        return bin_to_ascii(Integer(int(hexval, base=16)).binary().zfill(8))



    encoded = "".join(map(parse_and_mod, split))
    print encoded
```

```
sage: my_mix_columns("OKLAHOMAILLINOIS")
NL_TM@^XCLFIWXfr
```

Therefore the resulting matrix would be

$$\begin{pmatrix} N & L & \_ & T \\ M & @ & \hat{} & X \\ C & L & F & I \\ W & X & f & r \end{pmatrix}$$