Exercise on polynomial-evaluation MACs

Part of this exercise will require a computer.

In this exercise, we shall study the message authentication code discussed in the lecture. The finite field with 456979 elements is denoted by $\mathbf{F}$. Let $\mathcal{X} = \cup_{k=1}^{\infty} \mathbf{F}^k$ (that is, the set of all tuples of field elements). The MAC is based on the function $f : \mathbf{F}^2 \times \mathcal{X} \to \mathbf{F}$ given by

$$f(k_1, k_2, (m_0, m_1, m_2, \ldots, m_{L-1})) = k_2 + \sum_{i=1}^{L} m_{i-1} k_1^i.$$

We encode messages with letters from a-z by mapping letters to numbers from 0 to 25 in the usual way. We also encode a full stop ., a comma , and a space ␣ to the numbers 26, 27 and 28, respectively.

We then encode strings of letters as tuples of field elements in the obvious way.

The MAC of a string of letters is computed by first turning the letters into a tuple of field elements, then applying the f function.

a. Alice the rating agency wants to send the rating eve inc has rating a to Bob the stock broker. The network provider Eve Inc. wants Bob to believe that it has a triple-A rating, that is, Bob should receive and accept the message eve inc has rating aaa. Explain why Eve Inc. can change Alice's message without Bob noticing.

Explain why the arguments for security we saw in class fail here.

b. Padding must be used to solve this problem. Consider the alternative MAC given by

$$f'(k_1, k_2, (m_0, m_1, m_2, \ldots, m_{L-1})) = k_2 + k_1^{L+1} + \sum_{i=1}^{L} m_{i-1} k_1^i.$$

Explain why the attack above no longer works.

c. Write a computer implementation of the MAC.

The following code may be useful:

```
alphabet = [
        'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j',
        'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't',
        'u', 'v', 'w', 'x', 'y', 'z', '.', ',', '␣' ]

encoding = dict()

for i in range(len(alphabet)):
        encoding[alphabet[i]] = i

def encode_letter(l):
        if l in encoding:
                return encoding[l]
        else:
                return '␣'

def text2numbers(s):
        return [ encode_letter(l) for l in s.lower() ]
```

d. You have received three messages, all claiming to be from Alice. You share the key $(12345, 54321)$ with Alice. Decide which message to accept:

- (buy enron, 230887)
- (invest in penny stocks, 230887)
- (sell everything and the box it was in, 230887)