6.857 — Problem Set 1 — Problem 2

Part (a)

We are given two ciphertexts resulting from encrypting two different 8-character English words using the same one-time pad. Let the plaintext messages be M_1 and M_2, the shared pad be P, and the two ciphertexts be C_1 and C_2. Since one-time pad encryption is defined as

$$C_i = M_i \oplus P,$$

we can eliminate the pad by XORing the ciphertexts:

$$C_1 \oplus C_2 = (M_1 \oplus P) \oplus (M_2 \oplus P) = M_1 \oplus M_2,$$

using the fact that $P \oplus P = 0$.

The ciphertexts provided are:

$$C_1 = \texttt{e9 3a e9 c5 fc 73 55 d5}, \qquad C_2 = \texttt{f4 3a fe c7 e1 68 4a df}$$

Computing the XOR byte-by-byte gives:

$$C_1 \oplus C_2 = \texttt{1d 00 17 02 1d 1b 1f 0a}$$

The resulting bytes fall within the range `0x00--0x1f`, which is consistent with the XOR of two lowercase ASCII letters. Since each ciphertext is 8 bytes long, both plaintexts must be 8-letter English words.

By testing candidate 8-letter English words and checking which pairs satisfy

$$M_2 = M_1 \oplus (C_1 \oplus C_2),$$

we find that the only valid pair is:

`networks` and `security`.

Thus, the two plaintext words are:

$$\boxed{\text{security and networks}}$$

The ordering cannot be determined from the ciphertexts alone.

Part (b)

Ben Bitdiddle proposed a modification to the one-time pad where each ciphertext byte depends on the previous ciphertext byte. His scheme computes ciphertext as:

$$c_i = m_i \oplus ((p_i + c_{i-1}) \bmod 256), \qquad c_0 = 0.$$

Defining the intermediate value

$$k_i = (p_i + c_{i-1}) \bmod 256,$$

reduces the construction to standard stream-cipher form:

$$c_i = m_i \oplus k_i.$$

Thus, if any plaintext byte m_i is recovered, the corresponding keystream byte k_i follows from $k_i = c_i \oplus m_i$, and the pad value is then recovered through

$$p_i = (k_i - c_{i-1}) \bmod 256.$$

Since all ten ciphertexts are valid English, we exploited predictable English structure. We guessed likely positions for the space character (ASCII 0x20), computed candidate keystream bytes $k_i = c_i \oplus 0x20$, and retained only those that produced readable English when applied to all ten ciphertexts. Iteratively expanding confirmed pad values allowed partial plaintext recovery, which in turn revealed further pad bytes until all 60 were determined.

Recovered Messages

```
We stand today on the brink of a revolution in cryptography.
Probabilistic encryption is the use of randomness in an encr
Secure Sockets Layer (SSL), are cryptographic protocols that
This document will detail a vulnerability in the ssh cryptog
MIT developed Kerberos to protect network services provided
NIST announced a competition to develop a new cryptographic
Diffie-Hellman establishes a shared secret that can be used
Public-key cryptography refers to a cryptographic system req
The keys used to sign the certificates had been stolen from
We hope this inspires others to work in this fascinating fie
```

Recovered Pad

```
[119, 75, 116, 51, 85, 113, 72, 105, 76, 78, 114, 79, 84, 49, 71, 101,
71, 88, 116, 78, 113, 102, 113, 87, 84, 65, 51, 55, 99, 56, 107, 69,
116, 105, 110, 109, 97, 113, 79, 106, 122, 68, 66, 98, 77, 72, 112,
```

```
72, 55, 53, 104, 54, 99, 71, 87, 97, 68, 98, 112, 49]
```

Conclusion

Although Ben introduced feedback hoping to avoid pad cancellation, reusing the pad still exposes the entire keystream. Once any plaintext is found, both the pad and all encrypted messages become recoverable.