

Problem 1-1. Weaknesses of CPA-secure cryptosystems

Let $F: \mathcal{K} \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ be a secure pseudorandom function. In class, we saw the CPA-secure encryption scheme (Enc, Dec) with keyspace $\mathcal{K}$, where

$$\text{Enc}(k, m) := \begin{cases} r \xleftarrow{\mathcal{R}} \{0, 1\}^n \\ c \leftarrow F(k, r) \oplus m \\ \text{output } (r, c) \end{cases} \quad \text{Dec}(k, (r, c)) := \begin{cases} m \leftarrow F(k, r) \oplus c \\ \text{output } m \end{cases}.$$

We will use a secure MAC scheme $\text{MAC}: \mathcal{K} \times \{0, 1\}^* \rightarrow \{0, 1\}^n$ over the same keyspace $\mathcal{K}$. MIT uses encryption to protect communications between department offices and its central payroll system. In particular, the EECS department office and MIT's payroll office share a secret encryption key $k_{\text{Enc}} \xleftarrow{\mathcal{R}} \mathcal{K}$ and a secret MAC key $k_{\text{MAC}} \xleftarrow{\mathcal{R}} \mathcal{K}$.

On each pay date, a machine in the EECS department sends a sequence of messages to the payroll office. Each plaintext message is a 54-byte ASCII-encoded string with format:

$m = \text{"date:NNNN-NN-NN, mit_id:NNNNNNNNN, amount:NNNNNNNNN.NN"}$,

where each N represents an ASCII-encoded decimal digit (i.e., ASCII values $0 \times 30 - 0 \times 39$). Unfortunately, the designers of the system make the wrong decision and use MAC-then-encrypt rather than encrypt-then-MAC. In particular, the EECS department encrypts each message m as:

$$\text{ct} = \text{Enc}(k_{\text{Enc}}, m \parallel \text{MAC}(k_{\text{MAC}}, m)).$$

The machine at the payroll office operates on each ciphertext ct as follows:

- Compute $(m' \parallel t') \leftarrow \text{Dec}(k_{\text{Enc}}, \text{ct})$.
- Check that the bytes of the string corresponding to the MIT ID number and the payment amount are valid ASCII decimal digits. If not, send a `FORMAT ERROR` message to the EECS machine.
- Check that the date corresponds to today's date. If not, send a `DATE ERROR` message to the EECS machine.
- Check that $t' = \text{MAC}(k_{\text{MAC}}, m')$. If not, send a `MAC ERROR` message to the EECS machine.
- Append the $(\text{ID}, \text{amount})$ pair to a log file for later processing and send an `OK` message to the EECS machine.

Explain how a network attacker that can intercept ciphertexts and interact with the payroll server can recover **all but one of the bits in each secret digit** in the plaintexts that the EECS department sends. Recovering these bits for a single plaintext using your attack should take no more than 10,000 interactions with the payroll server.

Problem 1-2. Encryption and RSA

- (a) Which of the following security goal(s) does encryption address :
(1) Confidentiality (2) Integrity (3) Sender authentication (4) Non-repudiation.
- (b) Suppose you obtain two ciphertexts C, C' encrypted using one-time pad, with key K and its bitwise complement $\bar{K} = K \oplus 1$ respectively. What can you infer about the corresponding plaintext messages?
- (c) Alice and Bob are using public keys $(e_1, N_1), (e_2, N_2)$ respectively. Suppose you are informed that their RSA moduli N_1, N_2 are not relatively prime. How would you break the security of their subsequent communication? It is sufficient to show that you can get $\phi(N_1)$ and $\phi(N_2)$.

Hint: Euclid's algorithm allows us to compute $\text{GCD}(x, y)$ efficiently.

- (d) Suppose that a system uses textbook RSA encryption. An attacker wants to decrypt a ciphertext c to obtain the corresponding confidential plaintext m . Assume that the victim system readily decrypts arbitrary ciphertexts that the attacker can choose, except for ciphertext c itself. Show that the attacker can obtain m from c even under this setting, i.e RSA is not CCA secure.