Esercizio 1

Si consideri un cifrario con: $\mathcal{P} = \mathcal{C} = \mathbb{Z}_{21}$, $\mathcal{K} = \{(k_1, \rho, k_2) \mid k_1, k_2 \in \mathbb{Z}_{21}, \rho : \mathbb{Z}_{21} \to \mathbb{Z}_{21}$ biiettiva$\}$, e la funzione di cifratura definita come segue:

$$E_{(k_1,\rho,k_2)}(x) = \rho(x + k_1) - k_2 \bmod 21$$

1. Scrivere la funzione di decifratura e dimostrarne la correttezza.
2. Data la chiave $(3, \rho, 3)$ con ρ definita come

   ```
   A B C D E F G H I L M N O P Q R S T U V Z
   R T F C I M E Z U V Q P G O B D N L H S A
   ```

   mostrare la decifratura del testo cifrato HZPZ.
3. La cifratura con chiave $(3, \rho, 3)$ (con ρ definita come al punto 2) può essere ottenuta con un semplice cifrario a sostituzione con una particolare chiave ρ'. Esibire la chiave ρ' e spiegare il procedimento seguito per ottenerla.
4. Dimostrare che, se viene riutilizzata la stessa chiave per cifrare le diverse lettere di un testo in chiaro, il cifrario definito sopra non è perfetto.

Esercizio 2

Illustrare una modalità di cifratura che permetta di 'trasformare' un cifrario a blocchi in un cifrario a flusso operante su singoli byte. Discutere possibili applicazioni di tale modalità di cifratura.

Esercizio 3

Le implementazioni del cifrario RSA usano, spesso, esponenti di cifratura bassi per velocizzare le operazioni crittografiche. Generalmente, però, è sconsigliabile usare esponenti troppo piccoli come ad esempio il numero 3. Discutere, considerando come esempio la cifratura di una chiave DES tramite RSA.

Esercizio 4

Si consideri il protocollo:

$$\begin{array}{lcll} B & \to & A & : C, N_B \\ A & \to & B & : Sign_{SK_A}(C, N_B) \end{array}$$

dove SK_A è la chiave privata di Alice, N_B è un nonce generato da Bob e C è un contratto di acquisto di un bene su cui Bob chiede la firma di Alice. Modificare il protocollo in modo da permettere ad Alice di inviare, assieme alla firma del contratto, il numero di carta di credito per effettuare il pagamento del bene. Considerare eventuali attacchi su segretezza e autenticazione e discutere eventuali problemi implementativi.

Esercizio 5 (facoltativo)

Dimostrare che una funzione hash strong collision free con un digest di 64 bit **non** è resistente alle collisioni.