1. A nonce-based symmetric encryption scheme is a three-tuple of algorithms $\Pi = (K, E, D)$ that is like the encryption schemes we have defined before except that $E$ is now deterministic and stateless (as is $D$), and $E$ and $D$ now take in an additional argument $N \in \mathbb{N} \subseteq \{0, 1\}^*$, the nonce. When encrypting, a party is required to select a new nonce $N$ to go with each message that is encrypted. As long as he does this, privacy should be assured. The nonce could be a counter, for example, or a long enough random string.

(a) Carefully formalize a notion of ind$^-$-security for a nonce-based symmetric encryption scheme.

(b) Describe a blockcipher-based scheme $\Pi$ that achieves your notion of security from (a), assuming that the blockcipher $E : K \times \{0, 1\}^n \rightarrow \{0, 1\}^n$ from which $\Pi$ is defined is secure as a PRP.

(c) Do you see any advantages of the nonce-based notion? Any disadvantages? Briefly discuss.

2. Suppose there exists a public-key encryption scheme that is IND-CPA secure. Show that there is a public-key encryption scheme that is IND-CPA secure but that is not IND-CCA secure.

3. Suppose you have a fast deterministic algorithm $I$ that inverts $f(x) = x^e \mod N$ on 1% of all inputs—the inputs in $\mathbb{Z}_N^*$ that your algorithm likes. Construct a usually-fast probabilistic algorithm $J$ that inverts $f(x) = x^e \mod N$ on every point in $\mathbb{Z}_N^*$. Analyze the efficiency of your algorithm: what is the expected running time of $J$? Your algorithm should be of the “Las Vegas” variety: it is always correct, and on every input it is usually fast. Analyze the efficiency of your algorithm.