Stream-ciphers and Block-ciphers

Q1: Formally define IND-CPA security (CPA security via indistinguishability). Write down full mathematical details.

A1: Let us define the CPA indistinguishability experiment $\texttt{PrivK}^{CPA}_{\mathcal{A},\Pi}(n)$ for some security parameter n.

(1) A key k is generated by $Gen(1^n)$.
(2) The adversary $\mathcal{A}$ is given input 1^n and oracle access to $Enc_k(.)$. It can query this oracle many times, upper bounded by some polynomial $\texttt{poly}(n)$. After sometime, the adversary outputs a pair of messages (m_0, m_1), with the constraint $|m_0| = |m_1|$.
(3) A uniform bit $b \in \{0, 1\}$ is chosen, and the challenge ciphertext $c = Enc_k(m_b)$ is computed and given to $\mathcal{A}$.
(4) The adversary still has access to oracle and query it many times, but eventually it outputs a bit b'.
(5) If $b = b'$, the experiment outputs 1, 0 otherwise. We say $\mathcal{A}$ succeeds if the output is 1.

If for all PPT adversaries, there is a negligible function $\texttt{negl}(n)$ that:

$$Pr[\texttt{PrivK}^{CPA}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \texttt{negl}(n)$$

then the scheme Π is IND-CPA secure.

Q2: Why are deterministic encryption schemes insecure under CPA?

A2: Encrypting the same plaintext twice yields the same ciphertext in such a case. Given this fact, and that the adversary has $\texttt{poly}(n)$ many Oracle Accesses $\mathcal{O}(.)$, think about the $\texttt{PrivK}^{CPA}_{\mathcal{A},\Pi}$ experiment. In the experiment, the adversary can do as follows:

(1) Query the $\mathcal{O}$ with pair (m_0, m_0) and obtain c_0.
(2) Query the $\mathcal{O}$ with pair (m_1, m_1) and obtain c_1.
(3) Send $\texttt{Chal}$ the pair (m_0, m_1) and obtain c_b. Compare c_b to c_0 and c_1, find the one that is equal, and effectively learn which of the messages got encrypted.

As a result, $Pr[\texttt{PrivK}^{CPA}_{\mathcal{A},\Pi} = 1] = 1$, which is as worse as it could get in terms of security.

Q3: What is the difference between a nonce and a random number?

A3: A random number (in the context of our course) is often described as a value r chosen from a uniform distribution. As a result, it is possible (though with negligible probability for large distributions) that you may obtain the same value more than **once**. In contrast, a **nonce** value must only be used only once, never more than that.

Q4: What is the relationship between PRP/PRF and block ciphers?
A4: Block ciphers are "practical constructions" of pseudorandom permutations. Some block cipher operating modes only require PRF's, which do not require the bijection property and an efficient inversion of the PRF at hand. Thanks to these and the block ciphers, key re-use on multiple messages are possible.

Q5: Which block cipher modes may employ PRFs, which modes must employ PRPs? Why?
A5: ECB (Electronic Code Book), DETCTR (Deterministic Counter) and CBC (Cipher Block Chaining) modes strictly use PRPs. Note that since all PRP's are trivially PRF's, they are technically using PRF's too. The reason these modes require PRP's is that they require the inverse function during decryption, which is only guaranteed to be efficient and with respect to bijectivity, if the function is a PRP.
OFB (Output Feedback) and CTR (Counter) modes use PRFs. They do not use the inverse function in the decryption, therefore it is only sufficient that the function is a PRF.

Q6: Why is the ECB mode insecure? What about the deterministic counter mode?
A6: ECB is insecure because similar message blocks will have similar ciphertext blocks. In fact, if two different blocks are the same bit string, they have the same ciphertexts. This reveals a lot of information just by looking at the ciphertext alone. The main problem here is that the same F_k is used for all blocks. DETCTR (Deterministic Counter) mode combats this by introducing a counter variable, which is used by the function. In ECB: $Enc(k, m_i) = F_k(m_i) = c_i$ and $Dec(k, c_i) = F_k^{-1}(c_i) = m_i$, however in DETCTR: $Enc(k, m_i) = F_k(m_i, i) = c_i$ and $Dec(k, c_i) = F_k^{-1}(c_i, i) = m_i$ thereby allowing additional randomness on duplicate blocks.

Q7: Which modes of operation should we use? Why? What would be the security guarantee such a mode will provide?
A7: Stream-cipher modes such as OFB and CTR seem to be the better ones, however there is a caveat. If an IV repeats in OFB or CTR, an attacker can obtain the content of encrypted messages just

by XOR'ing the ciphertexts, similar to what happens when one re-uses a key in one-time pad. Basically, the attacker can learn the entire content in case of such a vulnerability. However, CBC is stronger in this aspect, it would only allow the attacker to know about few blocks, and after those the block cipher will diverge and the attacker won't learn anything else. To combat IV misuse, one could have stateful OFB or CTR modes, but if stateful encryption is not possible, CBC must be used. The result would provide security against CPAs.

Q8: Which security definition cannot be achieved using the modes presented?

A8: The schemes and modes we have seen so far are not secure against chosen-ciphertext attacks (CCA) or message tampering, where the adversary is not only eavesdropping but also tampering with the ciphertext itself.

Q9: Consider a stateful variant of CBC-mode encryption where the sender simply increments the IV by 1 each time a message is encrypted (rather than choosing IV at random each time). Show that the resulting scheme is not CPA-secure.

A9: The key idea here is that CBC is not CPA-secure when the attacker can predict the IV. The scheme given in the question actually allows this. Here is what the attacker can do to break the system (in the experiment $\texttt{PrivK}^{CPA}_{\mathcal{A},\Pi}$):

(1) Query the oracle for a 1-block message such as $0 \in \mathcal{M}$.
(2) Oracle will respond with $c_1 = [IV_1, Enc(k, 0 \oplus IV_1)]$.
(3) The adversary can predict IV_2, which is pretty clear: $IV_2 = IV_1 + 1$ (assume that it does not overflow for now, for clarity, though it would not matter).
(4) Challenge the `chal` with the message pair (m_0, m_1) where $m_0 = IV_2 \oplus IV_1$ and $m_1 \neq m_0$.
(5) `chal` responds with c, that can be either of:
    (a) $c = [IV, Enc(k, IV_2 \oplus IV_1 \oplus IV_1)] = [IV, Enc(k, IV_1)]$
    (b) $c = [IV, Enc(k, IV_2 \oplus m_1)]$
(6) $b' = 0$ if $c[1] = c_1[1]$, as in, if the second blocks are same (which would be IV_1), then it knows the message was m_0. Otherwise, $b' = 1$.

The advantage of adversary here is 1, therefore completely breaking the system against CPA-security.

Q10: What is the effect of a dropped ciphertext block (e.g., if the transmitted ciphertext <c1, c2, c3, . . .> is received as <c1, c3, . . .>) when using the CBC, OFB, and CTR modes of operation?

A10:

- **CBC**: The decryption of c_i is done by $m_i = F_k^{-1}(c_i) \oplus c_{i-1}$. Suppose that c_{i-1} is dropped. The receiver cannot obtain it by doing $c_{i-1} = F_k(c_{i-2} \oplus m_{i-1})$ because it does not know m_{i-1}, and therefore the decryption can not happen for m_i. Since c_{i-1} is gone m_{i-1} is gone too. The rest of the blocks starting from c_i are gone too, because we would need c_i to be able to decrypt them eventually. In summary, all the blocks starting from and including the dropped block are gone.
- **OFB**: The decryption of c_i is done by $m_i = y_i \oplus c_i$. If c_i is dropped, m_i is gone. If the dropped block is anything other than IV there won't be a problem for the rest of the blocks. However, if IV is gone, all blocks are gone.
- **CTR**: Similar to OFB, if `ctr` block is gone, all messages are lost. Otherwise, only the dropped block is gone.