Problem 1: MACs and encryption

Consider the following symmetric encryption scheme \((KG, E, D)\). KG chooses an AES key. \(E(k, m) := E_{AES}(k, m) || 0^{32}\). \((0^{32}\) stands for a string consisting of 32 zeros.) And the decryption \(D(k, c)\) does the following: Let \(c' || p := c\) where \(p\) has length 32 bit and \(c'\) is all but the last 32 bits of \(c\). \(m := D_{AES}(k, c')\). If \(p = 0^{32}\), then \(D(k, c)\) returns \(m\). If \(p \neq 0^{32}\) and \(k_p = 0\) (here \(k_p\) is the \(p\)-th bit of the key \(k\)), then \(D(k, c)\) returns \(m\). If \(p \neq 0^{32}\) and \(k_p = 1\), then \(D(k, c)\) aborts.

(a) Show that \((KG, E, D)\) can be totally broken using a chosen ciphertext attack\(^1\). That is, show that it is possible to recover the key \(k\) using a chosen ciphertext attack.

(b) To avoid the issue, we try to use authentication: Let \(MAC\) be an EF-CMA secure MAC. We construct a new encryption scheme \(E'\). The key of this scheme consists of an AES key \(k_1\) and a MAC-key \(k_2\). Encryption is as follows: \(E'(k_1k_2, m) := E(k_1, (MAC(k_2, m), m))\). Decryption \(D'\) checks the tag \(MAC(k_2, m)\) and aborts if it is incorrect\(^2\) (This is called MAC-then-encrypt.)

Does \(E'\) withstand chosen ciphertext attacks that reveal the whole key \(k_1\)? If yes, explain why (without proof). If no, how to attack?

(c) We try to use authentication in another way: Let \(MAC\) be an EF-CMA secure MAC. We construct a new encryption scheme \(E''\). The key of this scheme consists of an AES key \(k_1\) and a MAC-key \(k_2\). Encryption is as follows: \(E''(k_1k_2, m) := MAC(k_2, c) || c\) with \(c := E(k_1, m)\). Decryption \(D'\) checks the tag \(MAC(k_2, c)\) and aborts if it is incorrect\(^3\) (This is called encrypt-then-MAC.)

Does \(E''\) withstand chosen ciphertext attacks that reveal the whole key \(k_1\)? If yes, explain why (without proof). If no, how to attack?

\textbf{Hint:} One of \((b), (c)\) is secure, the other is insecure.

---

\(^1\)In a chosen ciphertext attack, the adversary is also allowed to submit plaintexts for encryption, not only ciphertexts for decryption.

\(^2\)We assume that you cannot distinguish between an abort due to a wrong tag or an abort of the underlying algorithm \(D\).

\(^3\)We assume that you cannot distinguish between an abort due to a wrong tag or an abort of the underlying algorithm \(D\).