Homework 3

Posted: Wednesday, October 18, 2023 – 11:59pm
Due: Wednesday, October 25, 2023 – 11:59pm

Task 1 – Key Recovery (5 + 5 points)

Let $\Pi = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec})$ be a symmetric encryption scheme with plaintext space $\mathcal{M} = \{0,1\}^\ell$. Consider the following oracle (see Section 3 for details about the syntax):

```
oracle KR[Π]:

private procedure Init():
K ←$ Kg()

public procedure Encrypt(M):
C ←$ Enc(K, M)
return C

private procedure Fin(K′):
if K = K′ return 1 else return 0
```

We define the *key recovery* (kr) advantage of an adversary A against Π as

$$\mathsf{Adv}^{\mathrm{kr}}_{\Pi}(A) = \Pr[A^{\mathsf{KR}[\Pi]} \Rightarrow 1] \ ,$$

a) [5 points] Let A be an adversary such that $\mathsf{Adv}^{\mathrm{kr}}_{\Pi}(A) = 1$. Use A to build a distinguisher D that is roughly as efficient as A with $\mathsf{Adv}^{\text{ind-cpa}}_{\Pi}(D) = 1$.

Hint: Give an oracle O such that $\Pr\left[A^{\mathsf{O}^{\mathsf{LR}_0[\Pi]}} \Rightarrow 1\right] = 1$ and $\Pr\left[A^{\mathsf{O}^{\mathsf{LR}_1[\Pi]}} \Rightarrow 1\right] = 0$ whenever A has kr advantage one.

b) [5 bonus points] We say that Π is *KR-secure* if $\mathsf{Adv}^{\mathrm{kr}}_{\Pi}(A)$ is negligible for all polynomial-time (randomized) adversaries A. Show that if Π is IND-CPA secure and $\ell = \omega(\log k)$ (i.e., $2^{-\ell}$ is negligible), then Π is also KR-secure.

Hint: Give an oracle O such that $\Pr\left[A^{\mathsf{O}^{\mathsf{LR}_0[\Pi]}} \Rightarrow 1\right] \geq \mathsf{Adv}^{\mathrm{kr}}_{\Pi}(A)$ and $\Pr\left[A^{\mathsf{O}^{\mathsf{LR}_1[\Pi]}} \Rightarrow 1\right] \leq \frac{1}{2^\ell}$ for any adversary A.

Task 2 – When IVs Collide (8 points)

When using block cipher mode of operations, in particular, CTR and CBC modes, it is important that the initialization vectors (IV) be chosen uniformly and randomly upon every invocation of the encryption algorithm so that the probability that the same IV

is chosen in two invocations of the encryption algorithm is very small. So what really happens when the IVs do collide?

Concretely, let $\mathsf{E} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be the block cipher used in CTR or CBC mode. Consider encrypting two plaintexts

$$M_0 = M_0[1]M_0[2]\cdots M_0[q] \quad \text{and} \quad M_1 = M_1[1]M_1[2]\cdots M_1[q],$$

each consisting of exactly q blocks of length n.

a) [4 points] Suppose we encrypted M_0, M_1 using CTR mode and obtained ciphertexts

$$C_0 = C_0[0]C_0[1]C_0[2]\cdots C_0[q] \quad \text{and} \quad C_1 = C_1[0]C_1[1]C_1[2]\cdots C_1[q].$$

If it happens that $C_0[0] = C_1[0]$, what can you learn about M_0, M_1 from C_0, C_1?

b) [4 points] Instead of CTR mode, CBC mode is used to encrypt the plaintexts. Again, if it happens that $C_0[0] = C_1[0]$, what can you learn about M_0, M_1 from C_0, C_1?

Task 3 – IND-CPA Security (10 points)

Let us consider the following randomized encryption scheme, based on the AES block cipher (which we denote as AES). Upon encrypting with a 16-byte key K, it first pads the plaintext M into 16-byte blocks $M[1], \ldots, M[\ell]$ using PKCS#7 padding, then generates a random 16-byte mask R, and lastly produces the ciphertext

$$C \leftarrow R \| \mathsf{AES}(K, M[1] + 1 + R) \| \cdots \| \mathsf{AES}(K, M[\ell] + \ell + R).$$

Here, addition is to be interpreted as in CTR encryption defined in class.

a) [5 points] Argue, informally, that the scheme is indeed IND-CPA secure, assuming AES is a good pseudorandom function, when encrypting plaintexts of length less than 16 bytes. You do not need to give a formal proof, but you need to rely on an intuitive understanding of the PRF security notion to argue that IND-CPA security would hold.

b) [5 points] Show that the scheme is, in general, not IND-CPA secure when encrypting plaintexts longer than 16 bytes.

Remember to use the definition of IND-CPA security for variable-length plaintexts! Here, we assume that plaintexts are sequences of bytes, i.e., only plaintexts for which bit length is a multiple of 8 are considered.

Task 4 – Padding-Oracle Attack (22 points)

In class, we have seen an example of a padding-oracle attack which recovers one plaintext byte from a ciphertext encrypted with CBC encryption using PKCS#7 padding. The attack invokes the padding oracle to make so-called padding validity checks, each telling us only whether the content recovered from a ciphertext is correctly padded or not. We want now to extend this attack to recover the full plaintext, and then implement it. Throughout this task, numbers in monospaced font (e.g., `01`) are hexadecimal.

a) **[6 points]** Give a strategy to recover the last byte of the last plaintext block, **which may or may not be validly padded**, with help of the padding oracle.

Hint: The strategy given in class only works if the plaintext is padded correctly. If this is not the case, show that there may be two byte-values $X_1, X_2 \in \{0,1\}^8$ such that XORing $X_1 \oplus \texttt{01}$ and $X_2 \oplus \texttt{01}$ to the last byte of the second-last block leads to correct decryption. Explain then how we can determine which one between X_1 and X_2 is the correct value for the last byte.

b) **[8 points]** Explain how to recover the **entire** plaintext M given its CBC encryption C. How many padding validity checks does your attack need?

c) **[8 points]** We now want to implement the padding-oracle attack from **c)** against CBC. To this end, we provide `hw3-oracle.py` on Ed, which contains a function `PadOracle` taking as argument a byte-string representing a ciphertext (whose length must be a multiple of 16 bytes) and checking whether it encrypts a correctly padded plaintext, for a hardcoded fixed key. In particular, it returns either `True` or `False` to indicate whether the padding is valid or not.

Extend `hw3-oracle.py` into a Python program that decrypts any given ciphertext (in a file whose name is passed as an argument) encrypted under the hardcoded key by **only using calls** to `PadOracle`. (Do not make use of the hardcoded key!) We also provide some ciphertext examples for you to test your code on.