Introduction to cryptology
TD#4

2022-W10,...

Exercise 1: MAC with a small state

A designer wants to design a MAC using a block cipher $E : \{0,1\}^{128} \times \{0,1\}^{32} \to \{0,1\}^{32}$. He wants to use a variant of CBC-MAC, but with larger tags than what a direct application using E would allow. Specifically, he wishes for 128-bit tags. The result is the following. On input $(k, k_0, k_1, k_2, k_3, m)$, compute:

$$x := \mathsf{CBC\text{-}Encrypt}[E](k, 0, m) \quad y_0 := E(k_0, x) \quad y_1 := E(k_1, x) \quad y_2 := E(k_2, x) \quad y_3 := E(k_3, x),$$

and output $y := y_0||y_1||y_2||y_3$.

Q. 1: How many possible values can be taken by x (for any k, m).

Q. 2: How many possible values can be taken by y, for a fixed MAC key (k, k_0, k_1, k_2, k_3)?

Q. 3: Give a strategy that allows to gather all possible tags for a fixed MAC key, with time, memory and query cost $\approx 2^{32}$ (assuming for simplicity that if the input message is 32-bit long, no padding is performed in the CBC encryption).

Q. 4 Assuming that the precomputation of the previous question has been performed, what is the forgery probability for a random message? Is this MAC a good MAC?

Q. 5 Is the modified scheme that on input $(k, k_0, k_1, k_2, k_3, m)$ computes:

$$x := \mathsf{CBC\text{-}Encrypt}[E](k, 0, m) \quad y_0 := E(k_0, x) \quad y_1 := E(k_1, y_0) \quad y_2 := E(k_2, y_1) \quad y_3 := E(k_3, y_2),$$

and outputs $y := y_0||y_1||y_2||y_3$ protected against the above attack?

Exercise 2: MAC definitions; RC4-MAC *(Exam '21)*

We first consider a deterministic MAC $M : \{0,1\}^\kappa \times \mathcal{X} \to \{0,1\}^n$.

Q.1: Suppose that you know a universal forgery A for M that wins the universal forgery game with probability p^U and that runs in time t^U and makes q^U queries to its oracle.

1. Specify an existential forgery A' for M that uses A as a black box.
2. Analyse the cost t^E and q^E of A' and its success probability p^E.

Q.2: Suppose that you know an existential forgery A for M that wins the existential forgery game with probability p^E and that runs in time t^E and makes q^E queries to its oracle.

1. Specify a PRF distinguisher for M that runs in time $t^F \approx t^E$ and makes $q^F \approx q^E$ queries to its oracle.

2. Give a lower bound for $\mathrm{Adv}_M^{\mathrm{PRF}}(q^F, t^F)$ by analysing the advantage of your distinguisher.

3. Is the following (informally stated) scenario possible: "M is vulnerable to an existential forgery attack, but it is hard to distinguish from a random function"?

4. Show that the following (informally stated scenario) is possible: "There is no efficient existential forgery attack on M, but it is easy to distinguish it from a random function". Only a sketch of proof is required here.

Q.3: Recall that an assumption A_1 is said to be *stronger* than an assumption A_2 if breaking A_2 implies breaking A_1 with a similar cost, but breaking A_1 does not necessarily imply breaking A_2 with a similar cost. Consider the three following (informally stated) assumptions: A_1: M is hard to distinguish from a random function; A_2: there is no efficient universal forgery on M; A_3: there is no existential forgery on M.

1. Order the assumptions A_1, A_2, A_3 from weakest to strongest. *Be careful to justify your answer.*

2. Suppose that you need a MAC algorithm, and are magically given access to one that satisfies an assumption that you are free to choose; which of A_1, A_2 or A_3 would you pick (and why)?

RC4 is a stream cipher that can be used to (poorly) encrypt binary strings of arbitrary length in the following way:

1. Two communicating parties share a secret key k.

2. For each new plaintext p to be encrypted, one picks a unique initialisation vector v.

3. One runs a setup algorithm on the pair (k, v) that returns an initial state s (that depends on both k and v).

4. One runs the RC4 keystream generator on s, producing a keystream z of the same length as p.

5. The encryption of p is returned as $c := p \oplus z$, along with the initialisation vector v.

A designer suggests to use RC4 as the basis of a MAC algorithm. For simplicity, we assume that the input is at least 128-bit long, or that it has otherwise been padded up to that length (or longer) using an appropriate injective padding scheme. To authenticate a message one runs RC4 encryption on the input and returns the last 128 bits of the ciphertext as a tag. In more details:

1. Two communicating parties share a secret key k.

2. One runs a setup algorithm on the pair $(k, 0)$ that returns an initial state s.

3. For each new input x to be authenticated, one runs the RC4 keystream generator on s, producing a keystream z of the same length as x.

4. One encrypts x as $c := x \oplus z$; the last 128 bits of c are returned as the authentication tag of x.

Q.4:

1. Give (and analyse) a very efficient attack on RC4-MAC with respect to one of the three security notions studied in this exercise.