HIT — Cryptography — Solutions 4

1603202-1150810613-Qiuhao Li

December 31, 2018

Problem 1. Let F be a pseudorandom function. Show that the following MAC for messages of length $2n$ is insecure: The shared key is a random $k \in \{0,1\}^n$. To authenticate a message $m_1 \| m_2$ with $|m_1| = |m_2| = n$, compute the tag $\langle F_k(m_1), F_k(F_k(m_2))\rangle$.

Solution 1.

For the adversary, he can first query $\mathsf{Mac}_k(\cdot)$ with $m_1||m_2$ as input, thus getting value of $F_k(m_1)$ and $F_k(F_k(m_2))$, and then queries $\mathsf{Mac}_k(\cdot)$ with $m_2||m_1$ as input, thus getting value of $F_k(m_2)$ and $F_k(F_k(m_1))$. Finally he can output a message $m_1||m_1$ with $\langle F_k(m_1), F_k(F_k(m_1))\rangle$ as tag, which lets the $\mathsf{Vrfy}_k(m,t) = 1$.

Problem 2. Let (Gen, H) be a collision-resistant hash function. Is $(\mathsf{Gen}, \hat{H})$ defined by $(\hat{H}^s(x) \stackrel{\text{def}}{=} H^s(H^s(x))$ necessarily collision resistant? Prove your answer.

Solution 2.

Yes, it is. I will prove it by contradiction.

Proof.

Truth: Since (Gen, H) is a collision-resistant hash function, we can conclude that if an polynomial-time adversary finds $H^s(u) = H^s(v)$, then u must equals v.

Let's say, there exists an polynomial-time adversary who finds x, x' let $\hat{H}^s(x) = \hat{H}^s(x')$, which indicates that $H^s(H^s(x)) = H^s(H^s(x'))$. As said above, if this happened, then $H^s(x)$ must equals $H^s(x')$, which contradicts the fact we know.

□

Problem 3. For each of following modifications to the Merkle-Damgård transform, determine whether the result is collision resistant or not. If yes, provide a proof; if not, demonstrate an attack. Hint: you may use two facts on hash function: (1) $h(x) = x$ is collision resistant. Although x is leaked, there is no collision. (2) A crhf h can be constructed from another crhf g by letting $h(x) = x\|0$ for $x = 0$ and letting $h(x) = g(x)\|1$ for $x \neq 0$.

1. Modify the construction so that the input length is not included at all (i.e, output z_B and not $z_{B+1} = h^s(z_B\|L)$).

2. Modify the construction so that instead of outputting $z = h^s(z_B \| L)$, the algorithm outputs $z_B \| L$

3. Instead of using an IV, just start the computation from x_1. That is, define $z_1 := x_1$ and then compute $z_i := h^s(z_{i-1} \| x_i)$ for $i = 2, \ldots, B+1$ and output z_{B+1} as before.

4. Instead of using a fixed IV, set $z_0 := L$ and then compute $z_i := h^s(z_{i-1} \| x_i)$ for $i = 1, \ldots, B$ and output z_B.

Solution 3.

1. No, it isn't.

   The adversary can outputs x, x' which meet $|x| \neq |x'|$ and x is a prefix of x'. Since **padding with 0s** will lead to the same input with these different messages, the collision will happen.

2. Yes, it is.

   Proof.

   $case_1$: $L \neq L'$: We can extract a collision in h_s from the final iterations of h_s in $H_s(x)$ and $H_s(x')$, which means $z_B \| L \neq z_{B'} \| L'$.

   $case_2$: $L = L'$: We proceed backward inductively and consider whether $x_{i^*} \neq x_{i^*}$. Since $x \neq x'$, so we can eventually find a collision in h_s, which means $z_{i^*-1} \| x_{i^*} \neq z'_{i^*-1} \| x'_{i^*}$.

   □

3. Yes, it is.

   Proof.

   The proof is pretty much the same as the one above and the only two differences is at the begin and the end of induction, where we consider $h(z_B || L)$ instead of $z_B || L$ and $x_1 || x_2, x'_1 || x'_2$ instead of $0^n || x_1, 0^n || x'_1$. □

4. No, it isn't.

   First of all, fix an arbitrary $x_1 \in (0,1)^n$. Let's say, $g : (0,1)^{2n} \Rightarrow (0,1)^{n-1}$ is a collision-resistant hash function. It is easy to verify that h_s defined as

   $$h_s(x) = \begin{cases} 0^{n-1}1 & if : x = 0^{n-2}10 || x_1 \\ 1 || g_s(x) & else \end{cases}$$

   is also collision resistant. Now we can notice that for any $x_2 \in (0,1)^n$, $H_s(x_1 || x_2) = H_s(x_2)$.

Problem 4. We have learned that CCA-secure encryption schemes can be constructed by Enc-then-MAC in the class. Is there any other way to achieve CCA-secure scheme but without MAC? For example, (1) do you think the following scheme is CCA-secure? And why?

- message $m \in \{0,1\}^{n/2}$ and key $k \in \{0,1\}^n$. In encryption, choose a random string $r \leftarrow \{0,1\}^{n/2}$ and ciphertext $c := F_k(r\|m)$, where F is a strong PRP.

Furthermore, no matter what is your answer to the above question, (2) do you think CCA-security implies secure Authenticated Encryption (A.E.)? And why?

Solution 4.

- Yes, it is, proven by reduction.

  Proof.

  (1) If true random f is used.

  $$\Pr[D^{f(\cdot),f^{-1}(\cdot)}(1^n) = 1] = \Pr\left[\mathsf{PrivK}^{\mathsf{cca}}_{\mathcal{A},\tilde{\Pi}}(n) = 1\right].$$

  (2) If F_k is used.

  $$\Pr[D^{F_k(\cdot),F_k^{-1}(\cdot)}(1^n) = 1] = \Pr\left[\mathsf{PrivK}^{\mathsf{cca}}_{\mathcal{A},\Pi}(n) = 1\right].$$

  Since F is a Strong PRP, so $\forall$ PPT distinguishers D

  $$\left|\Pr[D^{F_k(\cdot),F_k^{-1}(\cdot)}(1^n) = 1] - \Pr[D^{f(\cdot),f^{-1}(\cdot)}(1^n) = 1]\right| \leq \mathsf{negl}(n),$$

  $$\left|\Pr[D^{F_k(\cdot),F_k^{-1}(\cdot)}(1^n) = 1]\right| \leq \frac{1}{2} + \mathsf{negl}(n).$$

  where f is chosen $u.a.r$ from the set of permutations on n-bit strings.

  Thus we can conclude

  $$\Pr\left[\mathsf{PrivK}^{\mathsf{cca}}_{\mathcal{A},\Pi}(n) = 1\right] \leq \frac{1}{2} + \mathsf{negl}(n).$$

  □

- No, it doesn't. For example, the CCA-secure encryption scheme just mentioned above can't achieve authentication communication, because the F is a strong PRP (i.e., bijection), which means all ciphertext given to $Dec(c)$ will be decrypted successfully.

Problem 5. Show a message transmission scheme that achieves authentication communication (with integrity and authenticity) but is not a secure A.E (without confidentiality).

Solution 5.

Suppose $\langle S, V \rangle$ is a secure MAC, we can define a message transmission scheme $\langle S', V' \rangle$

$$S'_k(m) = (S_k(m), m)$$

$$V'_k(m, (t_1, t_2)) = V_k(m, t_1) \wedge (t_2 = m)$$

Obviously, this scheme achieves authentication communication (with integrity and authenticity) but not CCA-secure (actually it leaks the entire message), thus not asecure A.E.

Happy  Hacking!