UPM **Computer Security** Date: October 18, 2023

Cryptography Module - Homework

Deadline: October 27, 2023

(1) Randomized encryption is not enough (6 Points)

In class we discussed that a deterministic encryption scheme cannot be secure in the IND-CPA sense. Also, we saw that for this reason the ECB mode of operation is not IND-CPA secure. In this problem you will prove that simply randomizing an encryption method is not sufficient to make the encryption IND-CPA secure.

Let $E_K : \{0,1\}^n \to \{0,1\}^n$ be a *blockcipher* which supports inputs of n bits, and consider the following randomized version of ECB:

- $\underline{\mathsf{RandECB\text{-}Encrypt}(K, M)}$: the algorithm takes as input the key K and a message M consisting of L blocks of n bits each, $M = M_1||M_2||M_3||\cdots||M_L$, and performs the following steps:
  - Sample a random n-bit string $IV \leftarrow \{0,1\}^n$;
  - For $i = 1$ to L:
  - Compute $C_i \leftarrow E_K(IV \oplus M_i)$;
  - Output $IV||C_1||\cdots||C_L$.

(a) (1 point) Show a decryption algorithm for this encryption, and prove its correctness.

(b) (5 points) Show an IND-CPA attack against the scheme. Namely, describe an algorithm which: chooses two specific messages of two blocks each, $M = M_1||M_2$ and $M' = M'_1||M'_2$; receives a ciphertext $C^* = (IV^*||C_1^*||C_2^*)$ that is computed as either $\mathsf{RandECB\text{-}Encrypt}(K, M)$ or $\mathsf{RandECB\text{-}Encrypt}(K, M')$; and efficiently determines which is the case. Concretely, how can you choose M and M'? What test can you do on C^*?

(2) Encryption is not a secure MAC method (4 points)

In class we discussed that using a secure encryption as a method to generate MACs may not guarantee integrity. In this problem you are asked to prove that this is the case for the OFB mode of operation.

Let $E_K : \{0,1\}^n \to \{0,1\}^n$ be a *blockcipher* which supports inputs of n bits, and consider the OFB encryption method that we studied in class. For a message M of length $2n$, OFB with E_K works as follows.

- $\underline{OFBEncrypt(K, M)}$: on input the key K and a message M consisting of two blocks of n bits each, $M = M_1||M_2$, choose $IV \leftarrow \{0,1\}^n$ uniformly at random, compute

$$C_1 \leftarrow E_K(IV) \oplus M_1, \quad C_2 \leftarrow E_K(E_K(IV)) \oplus M_2$$

and return (IV, C_1, C_2).

- $\underline{OFBDecrypt(K,(IV,C_1,C_2))}$: on input the key K and a ciphertext (IV,C_1,C_2), compute

$$M_1 \leftarrow E_K(IV) \oplus C_1, \quad M_2 \leftarrow E_K(E_K(IV)) \oplus C_2$$

and return (M_1, M_2).

Next, consider the following MAC candidate scheme

- $\underline{MAC(K,M)}$: output $T \leftarrow OFBEncrypt(K,M)$;
- $\underline{Verify(K,M,T)}$: output 'accept' iff $M = OFBDecrypt(K,T)$.

and consider an attacker who sees a MAC T on a message M.

Describe an attack algorithm that, without the knowledge of the secret key K, can create a valid MAC on any other message M^* (different from M). Justify the correctness of the attack. Namely, describe an algorithm that on input a message M and a valid MAC $T = (IV, C_1, C_2)$ for it, outputs a valid MAC $T^* = (IV^*, C_1^*, C_2^*)$ for a given message M^*.

$Attack(M, T, M^*)$:

(3) Security of Hash-and-Sign Digital Signatures (6 points)

Let $\Sigma = (Sign, Ver)$ be a digital signature that supports messages of fixed length ℓ, i.e., any $M \in \{0,1\}^\ell$. As we have seen in the class, we can use Σ to construct a scheme for arbitrarily long messages – i.e., any $M \in \{0,1\}^*$ – by combining it with a collision-resistant hash function $H : \{0,1\}^* \to \{0,1\}^\ell$.

Precisely, the new digital signature scheme for arbitrary messages is $\Sigma^* = (Sign^*, Ver^*)$ where the signing and verification algorithms are defined as follows:

$$Sign^*(sk, M) := Sign(sk, H(M))$$

$$Ver^*(pk, M, \sigma) := Ver(pk, H(M), \sigma)$$

The security claim is that Σ^* is a secure digital signature scheme (for messages in $\{0,1\}^*$) if: Σ is a secure digital signature (for messages in $\{0,1\}^\ell$), and H is a collision-resistant hash function.

a) (4 points) Suppose that the hash function H used in the construction of Σ^* is *not* collision resistant. How can you break the security (unforgeability) of Σ^*? Describe an attack against the security of Σ^*, and justify why the attack works. (**Hint:** The attacker can obtain one signature (produced using $Sign^*$) on a message M_1 of its choice, and then it must produce a forgery, that is a valid signature on a new message $M_2 \neq M_1$. Recall that this adversary must use the fact that the collision-resistance of H can be broken.)

b) (1 point) Suppose that in the digital signature scheme Σ^* above we use some function H that when applied to messages that have more than ℓ bits, say $(x_1, x_2, x_3, \ldots, x_m)$ with $m > \ell$, then

$$H(x_1, x_2, x_3, \ldots, x_m) = (x_1, x_2, \ldots, x_{\ell-1}, x_\ell \oplus x_{\ell+1} \oplus \ldots \oplus x_m)$$

To be more clear

– Define the first $\ell - 1$ bits of the output to be the first $\ell - 1$ bits of the input.
– Define the last bit of the output to be the XOR of the remaining bits of the input.

Regardless of how H is defined for messages of ℓ bits or less, H can not be collision resistant (and then the digital signature scheme Σ^* cannot be secure because of part a)). Why is it not collision resistant?

c) (1 point) Suppose instead we use some function H that when applied to messages that have ℓ bits or less, pads the message with zeros until we get ℓ bits and then outputs that string. That is, whenever $m \leq \ell$:

$$H(x_1, x_2, \ldots, x_m) = (x_1, x_2, \ldots, x_m, \underbrace{0, \ldots, 0}_{\ell - m \text{ zeros}})$$

Assume ℓ is at least 2. Regardless of how H is defined for more than ℓ bits, H can not be collision resistant (and then the digital signature scheme Σ^* cannot be secure). Why is it not collision resistant?

(4) Authenticated communication (4 points)

Suppose that Bob has a public key pk_{Bob} certified by a certification authority (where he knows the corresponding secret key sk_{Bob}) trusted by both Alice and Bob, but Alice does not have such a certified key pair. We assume that Bob's key can be used both for public key encryption and for digital signatures. Alice wants to communicate with Bob through an insecure channel where Eve can see and modify messages. In order to have faster communication, Alice wants to use a symmetric key encryption scheme to encrypt messages. Alice and Bob come up with the following protocol:

Alice: Chooses a key K and encrypts K under Bob's public key pk_{Bob}, i.e., she sends $c = Enc(K, pk_{Bob})$

Bob: Decrypts the key K using his secret key (i.e. $K = Dec(c, sk_{Bob})$) and sends the message $m =$ ``received'' signed with his secret key, i.e. he computes $\sigma = Sign(m, sk_{Bob})$ and sends (m, σ) to Alice.

Alice: Verifies the signed message by Bob by applying $Ver(m, \sigma, pk_{Bob})$. If this outputs 1, she will start to use K to send encrypted messages to Bob with the symmetric key encryption scheme.

(a) (2 points) Can Bob be sure that it is Alice who is sending the message c?

(b) (2 points) Suppose that Bob has executed the same protocol before with Amanda. When Alice and Bob execute this protocol, can Alice be sure that she has received the message m from Bob, even though the verification of its signature outputs 1?