ECE 471/571: Sample Exam Problem Solutions

Instructor: Ming Li

Electrical and Computer Engineering, University of Arizona

Problem 1: (*True or False*)
These questions require YES or NO answers with explanations.
(1). In theory, if the key is truly random, never reused, and kept secret DES and AES are both provably secure against known plaintext attacks.

Answer: False

(2). A Feistel cipher structure lets you use the same hardware or software for decryption as for encryption.

Answer: True

(3). Recall the modes of encryption. For the ECB and counter modes, if we change one bit in one ciphertext block, then only one block in the decrypted plaintext will be garbled.

Answer: True

(4). The textbook RSA cryptosystem is secure against chosen ciphertext attacks.

Answer: False

(5). It is impossible for both public key encryption and digital signature schemes to achieve perfect secrecy.

Answer: True

Problem 2: Symmetric Key Encryption
a) What is a one-time pad?

One-time pad is a perfectly secure symmetric key encryption algorithm.

Plaintext space, key space and ciphertext space all consist of length n bit strings;
The key vector k is freshly and randomly generated.
Encryption rule: $x \oplus k = y$
Decryption rule: $y \oplus k = x$

b) Any good random number generator can be used as a secret-key encryption algorithm. Explain how?

Answer: You can use a pesudorandom number generator $F_K(\cdot) : \{0,1\}^m \times \{0,1\}^k \to \{0,1\}^*$ to generate a key stream, and then XOR it with the plaintext to encrypt. For example, when $F_k(\cdot)$ is implemented by repeatedly applying a keyed hash function $H(K||\cdot)$:
Choose a random IV each time; Let K be the shared key.

$K_1 = H(K||IV)$;
$K_2 = H(K||K_1)$;
......
$K_n = H(K||K_{n-1})$;

$C_1 = P_1 \oplus K_1$;
$C_2 = P_2 \oplus K_2$;
......
$C_n = P_n \oplus K_n$;

Problem 3: Suppose Alice and Bob know each other's public key. Alice sends a message to Bob. How can she encrypt the message so that, when Bob receives it, he is sure about all of the following:

– Nobody else can view the content (confidentiality),
– The message is from Alice and no one has modified it (authentication, integrity).
– Nobody else (Eve) could trick Bob into thinking that Eve also generated the same message.

Answer:
Alice can send Bob: $E_{pk_B}(x||Sig_{sk_A}(x))$, or $E_{pk_B}(x||Sig_{sk_A}(x||ID_B))$.

Problem 4: (Graduate students only) What is a practical method for finding a triple of keys that maps a given plaintext to a given ciphertext using EDE (where E stands for DES encryption, and D is for DES decryption)?

Answer: It is like the meet-in-the-middle attack against encrypting twice with two keys (refer to Kaufman's book Section 4.4.1.2, page 112, posted material on Piazza).

Here we have three different keys. With one plaintext and ciphertext pair (x_1, y_1), we can build two tables, the first one contains $D_{K_2}(E_{K_1}(x_1)$ for all the key combinations of K_1, K_2 (there are 2^{112} of these), and the second one contains $D_{K_3}(y_1)$ (there are 2^{56} of these). Then we can find the tuple of keys (K_1, K_2) and K_3 that gives the same value in both tables, and those are candidate keys. Using more plaintext and ciphertext pairs, we can continue this process to filter out the false keys, and finally only one key tuple remains which is the correct (K_1, K_2, K_3). This process has complexity $O(2^{112})$.

Problem 4: (Undergraduate students only) Suppose that the F function of DES mapped every input R to (a) a 32-bit string of ones and, (b) to a bitwise complement of R denoted as R', regardless of what the key k is. Find the ciphertext y for a given plaintext x.

Answer:
(a): Let's denote $L^0R^0 = IP(x)$. f denote a 32-bit string of all ones. Then:
$L^1 = R^0$, $R^1 = L^0 \oplus f$.
$L^2 = R^1 = L^0 \oplus f$, $R^2 = L^1 \oplus f = R^0 \oplus f$.
$L^3 = R^2 = R^0 \oplus f$, $R^3 = L^2 \oplus f = L^0$.
$L^4 = R^3 = L^0$, $R^4 = L^3 \oplus f = R^0$.

Thus we can see that the pair L^0, R^0 appears periodically for every four rounds. So in the end, we have 16 rounds, and the output of the 16-th round is the same as the 0-th round ($L^{16}R^{16} = L^0R^0$). Applying the inverse permutation, we obtain: $y = IP^{-1}(L^{16}R^{16}) = IP^{-1}(IP(x)) = x$.

(b): Note that, $\overline{R} = R \oplus f$.
$L^1 = R^0$, $R^1 = L^0 \oplus (R^0 \oplus f)$.
$L^2 = R^1 = L^0 \oplus (R^0 \oplus f)$, $R^2 = L^1 \oplus (R^1 \oplus f) = L^0$.
$L^3 = R^2 = L^0$, $R^3 = L^2 \oplus R^2 \oplus f = R^0$.
$L^4 = R^3 = R^0$, $R^4 = L^3 \oplus R^3 \oplus f = L^0 \oplus R^0 \oplus f$.

Again, the above result in round 1 appears in a periodic fashion with period of 3. In the end of 16-th round, it will be the same as the 4-th round. Therefore, $L^{16}R^{16} = L^1R^1 = R^0(L^0 \oplus \overline{R^0})$, and $y = IP^{-1}(L^{16}R^{16}) = IP^{-1}(R^0(L^0 \oplus \overline{R^0}))$

Problem 5:
Suppose you are designing a processor that would compute with encrypted data. For example, given two encrypted data values $E_K(x)$ and $E_K(y)$, the processor would compute $E_K(x) + E_K(y)$, where "+" is an encrypted addition operator that performs addition on encrypted numbers. And the decryption: $D_K(E_K(x)+E_K(y))$ will be the same as $x+y$. None of the encryption algorithms we have learnt so far has the property that $E_K(x) + E_K(y) = E_K(x + y)$, although the encrypted addition operator does not necessarily have to be arithmetic addition. For DES and AES, is there any relationship between $E_K(x), E_K(y), E_K(x + y)$? Is this property desirable? How about for RSA?

Answer: The point for this question is not to find such a system, rather to realize why such a system (linear relationship between plaintext and ciphertext) is not desirable. For confusion purposes, one does not want patterns to appear in the ciphertext. Furthermore, the ordinary addition property is seldom used in encryption systems because they are not based in arithmetic (e.g., substitutions). For those that are (e.g. some public key encryption algorithm such like RSA), the arithmetic involves exponentiation and logarithms in finite fields because it is far more difficult to compute inverses. Thus, the only things left are trivial encryption schemes, such as variations on bitwise complement, in which it may be possible to find one for which the additive property holds. But for secure encryption schemes, it is most unlikely.

(Note: secure encryption schemes having this property do exist, which are called homomorphic encryption. They are still secure, since they have a different notion of security than what we have learned in this class so far. However, those are probabilistic encryption schemes, i.e., even the same plaintext will be encrypted to different ciphertext each time, so there will not be patterns in the ciphertext anyway. That said, in general, homomorphic encryption schemes are not secure against chosen-ciphertext attacks.)

DES and AES are all deterministic encryption algorithms. Deterministic encryption algorithms should not have the linear property. For RSA, there is no linear property like this, but if we change the addition operator to modular multiplication, then yes, it has the multiplicative homomorphic property. Textbook RSA is not secure since it is a deterministic algorithm. We can make it more secure by padding, such like using OAEP (with which it will lose its homomorphic property).

<u>Problem 6</u>: (Undergraduate students only)
We consider the RSA encryption.
(1) 3 and 65537 are commonly used as the public key. Can they be used as the private key instead? Why or why not?

Answer: No. Private key should be a large number.

(2) To illustrate the RSA system, we use primes $p = 23$ and $q = 17$. As public encryption key we use $e = 3$. Show that the private key $d = 235$.

Answer: $(p - 1) \times (q - 1) = 352$;
$e \times d \;\; mod \;\; (p - 1) \times (q - 1) = 705 mod \;\; 352 = 1$.

(3) Suppose Bob has an RSA Cryptosystem with a large modulus n for which the factorization cannot be found, e.g., n is 1024 bits long and Alice sends a message to Bob by representing each alphabetic character as an integer between 0 and 25 (i.e., $A \to 0$; $B \to 1$; ... $Z \to 25$) and then encrypting each letter as a separate plaintext character. Describe how Oscar can easily decrypt a message which is encrypted in this way.

Answer: A message consists of, let's say, m pieces of ciphertext $y_0, y_1, \cdots, y_m$. However, the plaintext space is restricted to 26 possible values and the ciphertext space too. That means we only have to test 26 possible plain-text letters for each cipher-text letter.