CS 6324: Information Security **Spring 2021**

Homework #2

Due date & time: 11:59pm CST on March 12, 2021. Submit to eLearning by the due time.

Additional Instructions: (1) The submitted homework must be typed. Using Latex is recommended, but not required.

Problem 1 (10 pts) Cryptanalysis against Encryption Modes

- (4 pts) Suppose that everyone in the world is using the DES algorithm in the ECB encryption mode, and you can use a chosen plaintext attack against everyone. Show how to perform a dictionary attack such that, after an expensive initialization step, everyone's key can be recovered in very little time. Write pseudo code both for the initialization step and for the function to recover everyones key.
  Answer.
  Init step: Choose a plaintext P, and compute the ciphertext $C_i = DES_{K_i}(P)$ for all possible keys K_i.
  Key recovery: Get the corresponding cipher text C of P, and search the ciphertext dictionary for C to get the key.

- (2pts) How effective would the above attack be if AES, instead of DES, is used?
  Answer. The initial step will be much more expensive for 2^{128} possible keys.

- (2pts) How effective would the above attack be under known-plaintext (instead of chosen plaintext) attacks?
  Answer. We will need to create a ciphertext dictionary for each given plaintext.

- (2pts) How effective would the Chosen Plaintext attack described above be if everyone uses the CBC encryption mode with the IV randomly chosen?
  Answer. We will need to create a ciphertext dictionary for our chosen plaintext P combined with each IV.

Problem 2 (6 pts) Levels of security

- (2 pts) Assume that you are doing an exhaustive key search attack and can check 2^{32} (or about 4G) keys per second, about how many years would it take to for you to search through all keys while attacking ciphers of the following key lengths: 56, 64, 80?
  Answer.
  1. $2^{56}/2^{32}/(60*60*24*365) = 0.53$ years
  2. $2^{64}/2^{32}/(60*60*24*365) = 136$ years
  3. $2^{80}/2^{32}/(60*60*24*365) = 8.93*10^6$ years

- (2 pts) Still assuming that you can check 2^{32} keys per second, further assume that the age of universe is 13.7 billion years, how many ages of the universe does it take to search through a key space of 128 bits.
  Answer.
  $2^{128}/2^{32}/(60*60*24*365)/13.7^9 = 1.83*10^11$

- (2 pts) Does this mean that a cipher of 128 bit can never be broken by brute-force? Justify your answer.
  Answer. Any reasonable justification is acceptable.

Problem 3 (9 pts) We use birthday collision to illustrate breaking the three desirable properties for cryptographic hash functions: Preimage resistant, Second preimage resistant, and Collision-resistant. Assume that there are 365 days in a year, and the birthday of each person is distributed uniformly. We consider three groups: Group A has 25 persons, group B has 35 persons, and group C has 180 persons. We assume that the distribution of birthdays for group members are random. Answer the following questions for each of the three groups: A, B, and C. Write out the equation, then give the numerical results (which can be computed by using calculators or writing a small program. You don't need to provide your code).

- (3 pts) What is the probability that a group contains someone with date of birth being Jan 1? (This corresponds to preimage attack.)
  Answer.
  A: $1-(1-\frac{1}{365})^{25} = 0.0663$
  B: $1-(1-\frac{1}{365})^{35} = 0.0916$
  C: $1-(1-\frac{1}{365})^{180} = 0.3897$

- (3 pts) What is the probability that a group contains someone that has the same date of birth as you? You are not part of the group. (This corresponds to second preimage attack.)
  Answer. Same as first question

- (3 pts) What is the probability that one can find two members in a group such that they have the same date of birth? (This corresponds to collision attack.)
  Answer.
  A: $1-\frac{\frac{365!}{(365-25)!}}{365^{25}} = 0.57$

A: $1 - \frac{\frac{365!}{(365-35)!}}{365^{35}} = 0.81$

A: $1 - \frac{\frac{365!}{(365-180)!}}{365^{180}} = 1$

Problem 4 (10 pts) Message Authentication Code (MAC)

a (3 pts) When we say that an MAC needs to resist Existential Forgery under Chosen Plaintext Attack, what do we mean? State the precise definition.

Answer

1. Challenger chooses a key K

2. Adversary chooses a number of messages M_1, M_2, ... , M_n, and get $t_i = MAC_k M_i$

3. Adversay should not be able to generate (M, t), where $M \notin \{M_1, M_2, ..., M_n\}$ and $t = MAC_k(M)$

b (3 pts) Consider the following MAC construction that uses AES. Given a message M, it is padded to so that its length in bits is a multiple of 128, and divided into blocks of 128 bits each: $M_1, M_2, \ldots, M_\ell$. The MAC value under a key k is defined as $MAC_k(M) = AES_k(M_1) \oplus AES_k(M_2) \oplus \cdots \oplus AES_k(M_\ell)$. Show that this MAC construction is insecure by giving a concrete attack.

Answer Adversary queries the MAC value of $M = A||B$, where A and B are two blockes of 128 bits. He gets $t = MAC_k(M)$.

Then, adversary can choose $M' = B||A$ and make sure $t = MAC_K(M')$

c (4 pts) Consider another MAC construction that uses AES. We limit ourselves to consider messages whose lengths are multiple of 96 bits, and are at most $2^{32} \times 96$ bits long. Given a message M, it is divided into blocks of 96 bits each: $M_1, M_2, \ldots, M_\ell$. Each M_i is then padded to 128 bits by including the binary encoding of i in the beginning of each block M_i. We use M'_i to denote the padded blocks. The MAC value under a key k is defined as $MAC_k(M) = AES_k(M'_1) \oplus AES_k(M'_2) \oplus \cdots \oplus AES_k(M'_\ell)$. Show that this MAC construction is insecure by giving a concrete attack.

Answer

1. Adversary chooses a number of messages $M_1 = A||B$, $M_2 = A$, $M_3 = C$ and gets the corresponding MAC t_1, t_2, t_3.

2. Adversary picks $M' = C||B$ and $t' = t_1 \oplus t_2 \oplus t_3$

Problem 5 (7 pts) CBC-MAC In class, we discussed HMAC. Another commonly used MAC scheme is Cipher Block Chaining Message Authentication Code, abbreviated CBC-MAC. This constructs a message authentication code scheme from a block cipher. To compute the CBC-MAC of a message given a key K, one uses the key K to encrypt the message in CBC mode, using 0 as the IV, and the final ciphertext block is the MAC value.

a (2 pts) Let E_K be the encryption function with block size b and key K, $B_0 = 00\cdots 0$ be the IV used in CBC-MAC, and $\oplus$ to denote bit-by-bite XOR. Write the formula for the CBC-MAC of a two-block message $M = M_1M_2$, i.e.

$$\text{CBC-MAC}_K(M_1M_2) = .$$

Answer
$\text{CBC-MAC}_K(M_1M_2) = E_k(M_2 \oplus E_k(M_1 \oplus B_0))$

b (5 pts) CBC-MAC is only secure for fixed-length messages. Show that when variable-length messages are allowed, CBC-MAC is insecure against a chosen-plaintext attack. That is, show that given M and $\text{CBC-MAC}_K[M]$, how to come up M' and $\text{CBC-MAC}_K[M']$ for some M'.
Answer
1. Adversary chooses $M_1 = A_1$ and get $t_1 = E_k(A_1 \oplus B_0)$, choose $M_2 = A_1||B_0$ and get $t_2 = E_k(B_0 \oplus E_k(A_1 \oplus B_0)) = E_k(B_0 \oplus t_1)$
2. Adversary picks $M' = t_1$ and then $t' = t_2$

Problem 6 (12 pts) RSA encryption. Assume that the message space consists of all 24-bit non-negative integer numbers. We use $N = 60529591$ for RSA encryption.

a (3 pts) Write a program that computes p and q such that $pq = N = 60529591$. Provide the values of p, q, and the value of $\Phi(N)$.
Answer p=6073, q=9967

b (3 pt) We choose $e = 31$. Provide the value of d such that $(ed \bmod \Phi(N)) = 1$.
(The algorithm for doing this is known as the extended Euclidean algorithm. If you do not know the algorithm, and do not want to learn the algorithm yourself, you can use an inefficient method such as enumerating over all positive integer numbers less than $\phi(N)$ and checking which one satisfies the requirement. For large N, this inefficient method takes too long time; but for the number here, it should be fast enough.)
Answer d = 29280751

c (0 pt) If you are unfamiliar with RSA, write a program that loops through all value x's such that $1 \le x \le N-1$, to verify that $(x^{ed} \bmod N) = x$. This is just to help convince yourself that RSA decryption works.
Hint: When computing $x^y \bmod N$, directly computing x^y may overflow the representation of integers you have. You can use the Square and Multiply method, which exploits the fact that $x^{ab} \bmod N = (x^a \bmod N)^b \bmod N$. You can also exploit the fact that $x^{y+1} \bmod N = (x^y \bmod N)x \bmod N$.

d (3 pts) For each of the following ciphertexts $10000, 20000, 30000, 40000, 50000, 60000$, show the decrypted message, as a base-10 number. Here we allow these messages to be beyond a 24-bit integer.
Answer 34910012, 33207991, 54348463, 31535938, 5844860, 46207307

e (3 pts) Describe how an adversary can recover any message encrypted in this way without factoring N. That is, the message space remains the same. However, the value of N is so large that factoring it becomes infeasible.

Answer We can enumerate all the possible plaintext and calculate the corresponding ciphertext to build a dictionary

Problem 7 (12 pts) ElGamal encryption. Again assume that the message space consists of all 16-bit integers. Consider ElGamal encryption. Let us choose $p = 96737$. Then we have

$$Z_p^* = \{1, \dots, 96736\}$$

a (3 pts) Write a program to find g, the smallest generator of Z_p^*, i.e., g is the smallest number in Z_p^* such that the set $\{g^1 \bmod p,\ g^2 \bmod p, \cdots,\ g^{96736} \bmod p\} = Z_p^*$. Provide the value g.

Answer g = 3

(The program can simply tries each number starting with 2 and see whether it is a generator. A number a is a generator for Z_p^* if and only if the smallest positive integer j such that $g^j \bmod p = 1$ is $j = p - 1$.)

b (3 pts) Write a program that computes the discrete logarithm in Z_p^*, and use it to compute the value $x = \log_g 90307$ in Z_p^*. That is, find x such that $(g^x \bmod p) = 90307$. (You can use the inefficient brute-force method.)

Answer x = 90620

c (3 pts) Write a program that implements ElGamal decryption. Then decrypt the following ciphertexts: $(5000, 5001)$, $(10000, 20000)$, $(30000, 40000)$, $(50000, 60000)$, $(70000, 80000)$, $(90000, 100000)$, which were encrypted under the public key given above, i.e., $(p, g, 90307)$. Show each decrypted message as a base-10 number.

(In El Gamal decryption of (c_1, c_2), you can compute x and need to find M such that $(xM \bmod p) = c_2$. To do this, you need to first compute $z = (x^{-1} \bmod p)$, that is, compute z such that $(xz \bmod p) = 1$. Given z, you can compute $M = (zc_2 \bmod p)$. Refer to Part b of the RSA question for how to find z.)

Answer 85101, 4996, 31532, 28134, 51546, 46927

d (3 pts) In Problem 6e, you showed how to break RSA encryption without factoring N. Can you recover a the plaintext from an ElGamal ciphertext without solving the discrete log problem or the Computational Diffie-Hellman problem, which are infeasible when p is very large? If yes, describe how. If no, explain why.

Answer. ElGamal encryption cannot be broken by enumerating all plaintexts and computing the ciphertexts. This is because ElGamal encryption is randomized. For one plaintext, choosing different random numbers result in different ciphertexts.

Problem 8 (9 pts) Hash Collisions and Digital Signatures. In a brute-force attack to find collisions for a hash function with a n-bit output, one chooses a group of $m = 2^{n/2}$ different messages. The probability that among them there are two messages that have the same hash is quite high, because the m messages give $\frac{m(m-1)}{2}$ pairs, each of which having the probability $\frac{1}{2^n}$ to give a collision.

a (4 pts) Given two groups X and Y of messages, we aim to finding a collision between a message in X and a message in Y. Suppose that each of X and Y contains $m = 2^{n/2}$ messages, what is the probability that at least one collision between a message in X and one in Y exists? Write out the formula, and compute/estimate it for $n = 128$.

Answer:

$$1 - \left(1 - (1 - 2^{-n})^{m*m}\right) = \quad 1 - \left(\left(1 - \frac{1}{2^n}\right)^{2^n}\right) \approx 1 - \frac{1}{e} \approx 0.63.$$

b (5 pts) Suppose that Bob is preparing a contract that Alice purchases a product from Bob at a cost of 1 million dollars for Alice to digitally sign. Bob wants to generate also a contract that sets the price at 10 million dollars. Bob is willing to put in $O(2^{n/2})$ amount of computation for the chance to cheat 9 million.

More specifically, Bob can apply the hash function $2 \times 2^{n/2} = 2^{n/2+1}$ times, which is still $O(2^{n/2})$. In addition, Bob can run an algorithm that takes time $O(n2^{n/2})$. For all practical purposes, we can view $O(n2^{n/2})$ as just $O(2^{n/2})$, since n is small as a multiplication factor.

Describe a way for Bob to come up with two versions of the contracts with the two different prices, but have the same hash value. For simplicity, let us assume that the document is a long and wordy (as legal documents are) ASCII text file. You will need to generate many candidates for the contracts. Describe how you can systematically generate the ones you need.

Answer. Bob constructs two contracts X and Y. X sets price at \$1M and Y sets price at \$10M. In X, Bob chooses $n/2$ places where one can use an alternate ASCII string without affecting the meaning, for example, replacing one space with two spaces after a period sign, adding an extra space at the end of a line or not, or replacing one word with another of the same meaning. For each of the $n/2$ places, Bob thus has two choices. This gives rise of $2^{n/2}$ different variants of X. Similarly, Bob can generate $2^{n/2}$ different variants of Y. Bob can thus try to find a collision between a variant of X and a variant of Y. Bob ask Alice to sign that X variant, and can claim that it is a signature on a variant of Y.

Problem 9 (9 pts) The UNIX `crypt` function is a hash function that only looks at the first eight bytes of the input message. For example, `crypt(helloworld)` returns the same value as `crypt(hellowor)`.

Some web sites use the following authentication method to authenticate users: (1) the user types in a user-id and a password P into his web browser, (2) the site, upon verification of the password P, computes $T =$ `crypt`(user-id$||K$), where $||$ denotes string concatenation, and K is a ℓ-byte site secret key $\ell \leq 8$, (3) the site sends a cookie back to the user containing T, (4) the user can use T to authenticate himself to the site in future connections.

Show that by choosing clever user-id's (of varying length) an attacker can expose the site's secret key K in time approximately 128ℓ. More concretely, the user creates an account, logs in and receives the corresponding T; he then creates another account (with a different user-id, logs in and receives another T. By repeating this sufficient times, the user recovers K completely. We are assuming there are 128 possible values for each character in a string.

Hint: Try to recover one character of K for each account created. The attack is described in the paper "Dos and Don'ts of Client Authentication on the Web" in USENIX Security Symposium, 2001. Reading the paper is allowed.

Answer. First create a user id of 7 characters, e.g., "aaaaaaa", and obtain the cookie $T =$ `crypt` ("aaaaaaa"$||k_1$), where k_1 is the first byte of the secret. For each byte x, compute `crypt` ("aaaaaaa"$||x$), and check whether the value equals T, when it equals, $x = k_1$. After obtaining k_1, create a user id of 6 characters, e.g., "aaaaaa". Now $T =$ `crypt` ("aaaaaaa"$||k_1||k_2$). Since we already know k_1, we can try all bytes (including empty to check for the case that the secret has only 1 byte) to find k_2. Then we create a user id of 5 characters "aaaaa" and find k_3. This way we can find out the secret in $O(128\ell)$, assuming that the secret uses a character set of at most 127 letters.

Problem 10 (16 pts) Read the article "New Directions in Cryptography" by Diffie and Hellman (`dh.pdf` is attached in eLearning), and answer the following questions. [To get a good grade in this question make sure you answer is short and directly to the point. Bullet-points answers are recommended.]

a (4 pts) The paper gives rationales for building encryption schemes that are secure against known plaintext attacks and chosen plaintext attacks, by discussing how such schemes remove restrictions that are placed on the ways of using them. Discuss these rationale in your own words.

b (4 pts) List all the limitations and shortcomings discussed in the paper about symmetric encryption schemes.

c (4 pts) List all the limitations and shortcomings discussed in the paper about symmetric message authentication schemes.

d (4 pts) The paper establishes the relationships among (1) public-key encryption, (2) public key distribution, and (3) digital signature (referred to in the paper as one-way authentication). By relationships, we mean it is possible to use one scheme to implement another. List these relationships, and explain the constructions involved to use one scheme to implement another.