Fundamentals of Cryptography

Homework 6

Dr. Mohammad Dakhilalian

Fall 2024

Theory Part

Thoroughly review **Chapters 10 & 11** of the book *Understanding Cryptography* to confidently address the questions.

Question 1

In an RSA digital signature scheme, Bob signs messages x_i and sends them together with the signatures s_i and her public key to Alice. Bob's public key is the pair (n, e); her private key is d.
Oscar can perform man-in-the-middle attacks, i.e., he can replace Bob's public key with his own on the channel. His goal is to alter messages and provide these with a digital signature which will check out correctly on Alice's side. Show everything that Oscar must do for a successful attack.

Question 2

Considering the Elgamal signature scheme, you are given Bob's private key $k_{\text{pr}} = (d) = (67)$ and the corresponding public key $k_{\text{pub}} = (p, \alpha, \beta) = (97, 23, 15)$.

1. Calculate the Elgamal signature (r, s) and the corresponding verification for a message from Bob to Alice with the following messages x and ephemeral keys k_{E}:
   (a) $x = 17$, $k_E = 31$
   (b) $x = 17$, $k_E = 49$
   (c) $x = 85$, $k_E = 77$
2. You receive two alleged messages x_1, x_2 with their corresponding signatures (r_i, s_i) from Bob. Verify whether the messages $(x_1, r_1, s_1) = (22, 37, 33)$ and $(x_2, r_2, s_2) = (82, 13, 65)$ both originate from Bob.

Question 3

The parameters of DSA are given by $p = 59$, $q = 29$, $\alpha = 3$, and Bob's private key is $d = 23$. Show the process of signing (Bob) and verification (Alice) for the following hash values $h(x)$ and ephemeral keys k_E:

1. $h(x) = 17$, $k_E = 25$
2. $h(x) = 2$, $k_E = 13$
3. $h(x) = 21$, $k_E = 8$

Question 4

Answer the following questions:

1. Find the probability of at least two students having the same birthday as a function of K and N, where N is the number of days in the year and K is the number of students.
2. We consider three different hash functions which produce outputs of lengths 64, 128, and 160 bits. After how many random inputs do we have a probability of $\epsilon = 0.5$ for a collision?

Question 5

Draw a block diagram for the following hash functions built from a block cipher $e()$:

1. $e(H_{i-1}, x_i \oplus H_{i-1}) \oplus x_i \oplus H_{i-1}$
2. $e(H_{i-1}, x_i) \oplus x_i \oplus H_{i-1}$
3. $e(x_i \oplus H_{i-1}, H_{i-1}) \oplus x_i$

Question 6

Assume the block cipher **PRESENT** (block length 64 bits, 128-bit key) is used in a Hirose hash function construction. The algorithm is used to store the hashes of passwords in a computer system. For each user i with password PW_i, the system stores:

$$h(PW_i) = y_i$$

where the passwords (or passphrases) have an arbitrary length. Within the computer system, only the values y_i are actually used for identifying users and giving them access.

Unfortunately, the password file that contains all hash values falls into your hands and you are widely known as a very dangerous hacker. This in itself should not pose a serious problem as it should be impossible to recover the passwords from the hashes due to the one-wayness of the hash function. However, you discovered a small but momentous implementation flaw in the software: The constant c in the hash scheme is assigned the value $c = 0$. Assume you also know the initial values ($H_{0,L}$ and $H_{0,R}$).

1. What is the size of each entry y_i?
2. Assume you want to log in as user U (you might be the CEO of the organization). Provide a detailed description that shows that finding a value PW_{hack} for which:

   $$PW_{\text{hack}} = y_U$$

   takes only about 2^{64} steps.
3. Which of the three general attacks against hash functions do you perform?
4. Why is the attack not possible if $c \neq 0$?

Programming Part

Question 7

Implement the Digital Signature Algorithm (DSA) as described below. You are allowed to use the built-in SHA algorithm and the Miller-Rabin algorithm for prime number generation as provided in the following steps:

Inputs

- Message x: The message to be signed.

Outputs

- Public Key k_{pub}: A tuple (p, q, α, β), where $\beta = \alpha^d \mod p$.
- Private Key k_{pr}: The private key d.
- Signature (r, s): The signature of the message, where r and s are computed as described in the DSA Signature Generation section.
- Verification Result: Whether the signature is valid or invalid, is determined by the verification steps.

Digital Signature Algorithm (DSA):

Key Generation for DSA

1. Generate a prime p with $2^{1023} < p < 2^{1024}$.
2. Find a prime divisor q of $p-1$ with $2^{159} < q < 2^{160}$.
3. Find an element α with $\text{ord}(\alpha) = q$, i.e., α generates the subgroup with q elements.
4. Choose a random integer d with $0 < d < q$.
5. Compute $\beta \equiv \alpha^d \mod p$.

The keys are now:

$$k_{\text{pub}} = (p, q, \alpha, \beta)$$
$$k_{\text{pr}} = (d)$$

DSA Signature Generation

1. Choose an integer as a random ephemeral key k_E with $0 < k_E < q$.
2. Compute $r \equiv (\alpha^{k_E} \mod p) \mod q$.
3. Compute $s \equiv (\text{SHA}(x) + d \cdot r) k_E^{-1} \mod q$.

DSA Signature Verification

1. Compute auxiliary value $w \equiv s^{-1} \mod q$.
2. Compute auxiliary value $u_1 \equiv w \cdot \text{SHA}(x) \mod q$.
3. Compute auxiliary value $u_2 \equiv w \cdot r \mod q$.
4. Compute $v \equiv (\alpha^{u_1} \cdot \beta^{u_2} \mod p) \mod q$.
5. The verification $\text{ver}_{k_{\text{pub}}}(x, (r, s))$ follows from:

$$v \equiv r \mod q \implies \text{valid signature}$$
$$v \not\equiv r \mod q \implies \text{invalid signature}$$

Prime Generation for DSA

You can use the Miller-Rabin algorithm for generating prime numbers p and q as described below:

Output: two primes (p, q), where $2^{1023} < p < 2^{1024}$ and $2^{159} < q < 2^{160}$, such that $p-1$ is a multiple of q.

Algorithm:

1. Find prime q with $2^{159} < q < 2^{160}$ using the Miller-Rabin algorithm.
2. FOR $i = 1$ TO 4096
   (a) Generate random integer M with $2^{1023} < M < 2^{1024}$.
   (b) $M_r \equiv M \mod 2q$.
   (c) $p - 1 \equiv M - M_r$ (note that $p-1$ is a multiple of $2q$).
   (d) IF p is prime (use Miller-Rabin primality test)
      i. RETURN (p, q)
   (e) $i = i + 1$
3. GOTO Step 1