CS 166 ASSIGNMENT 2 SOLUTION

Overview

Marks on question:

- No mark: 1-part short answer question (the key point(s) for explanation is underlined)
- !: 1 part auto-graded question (answer in **bold**)
- +: n-parts question with autograding part and explanation part. The answer for autograding part is in **bold**, the key point(s) for explanation is underlined.

Basic rules of grading: Full points if your explanation/equation was correct, even if the answer is wrong (because points already deducted in the fill-in-blank part). And you get 0.5 (on whole question, not each sub-question) as long as you wrote something.

Week 3 - Public Key Cryptos - Part 1 & 2 (Q1 - Q7)

!Q1: Usages of public key cryptos (L5 P5 & P6)

I. When used for encryption, should use receiver's public key to encrypt and receiver use his/her private key to decrypt.
So, Alice encrypts message with **Bob**'s **public key** and Bob decrypts it with **Bob**'s **private key**.

II. When used for signature, should use signee's private key to encrypt and others use signee's public key to decrypt to verify.
So, Alice encrypts message with **Alice**'s **private key** and Bob decrypts it with **Alice**'s **public key**.

!Q2: Public key crypto for digital signature (L5 P6)

To verify a signature, Bob needs to get the signed message = $\mathbf{[M]_{Alice}}$, and use **Alice's public key** to decrypt to verify that $\{[M]_{Alice}\}_{Alice}$ = M. That is, Bob also needs to know the original message **M** to verify.

+Q3: Knapsack crypto example with SIK = (3,5,12,23) and m = 6, n = 47 (L6 P6 & P7)

I. Public key: GK, n

II. Private key: SIK, m (and m^{-1} mod n).

III. To get GK from SIK, the equation to use is $GK_i \equiv (SIK_i * m) \bmod n$.
So, (3 * 6) mod 47 = **18**; (5 * 6) mod 47 = **30**; (12 * 6) mod 47 = **25**; (23 * 6) mod 47 = **44**.
(Show one example is enough)

IV. GK = (18, 30, 25, 44) so encrypting 1110 = (1*18 + 1*30 + 1*25 + 0*44) mod n = 73 mod 47 = **26**

V. SIK = (3, 5, 12, 23), m^{-1} mod n = 6^{-1} mod 47 = 8. So, S = (26 * 8) mod 47 = 20.
Solve SIK, we get 20 = 1*3 + 1*5 + 1*12 + 0*23, that is, plaintext = 1110. Verified.

Yan Chen
Computer Science Department

!Q4: Knapsack from GK to SIK (L5 P12 - P14 & L6 P6)

I. $SIK_i \equiv (GK_i * p) \bmod n$, where $p = m^{-1} \bmod n$

II. Either show how to derive the above equation or prove the above equation is correct.

- Derive from $GK_i \equiv (SIK_i * m) \bmod n$:
  let $p = m^{-1} \bmod n$, that is, $p * m \equiv 1 \bmod n$
  So $GK_i * (p * m) \equiv (SIK_i * m) * 1 \equiv (SIK_i * m) \bmod n$
  Since m and n are relatively prime to each other, m can be cancelled:
  $GK_i * p \equiv SIK_i \bmod n$, i.e, $SIK_i \equiv GK_i * p \bmod n$
- Or prove that $SIK_i * m \equiv (GK_i * p * m) \equiv GK_i \bmod n$ where $p = m^{-1} \bmod n$
  $p = m^{-1} \bmod n$ means $p * m \equiv 1 \bmod n$
  So $(GK_i * p * m) \equiv (GK_i * 1) \equiv GK_i \bmod n$, Q.E.D.

+Q5: RSA example with N = 55 (L6 P9 - P11)

Q5.1 & 5.2: e must be relatively prime to $\varphi(N) = (p - 1) * (q - 1) = (5 - 1) * (11 - 1) = 40$, so 5 cannot be a possible e, but **9**, **11**, and **39** all can be possible e for N = 55.

Q5.3: The smallest e in Q5.1 is 9.

I. $d = e^{-1} \bmod \varphi(N)$, i.e., find a d such that $9*d = k\varphi(N) + 1 = 40k + 1$, where k can be any integer.
Since, $9 * 9 = 40 * 2 + 1$, d can be **9**.
(Oops I randomly chose the number, and it turns out d and e can be the same!)
Note that can be other d's as long as $d = e^{-1} \bmod \varphi(N)$.

II. To encrypt, should use the public key e, that is, $C = M^e \bmod N = 10^9 \bmod 55 =$ **10**. (Oops I just randomly used 10 as M and it turns out the resulting ciphertext is the same as plaintext!)

III. To sign, should use the private key d, that is, $C = M^d \bmod N = 20^9 \bmod 55 =$ **5**.

Q6: Cube root attack on RSA when e = 3 (L6 P12)

I. (Mention one of two possibilities is enough)
Possibility 1: when $M^e = M^3 < N$, then $C = M^3 \bmod N = M^3$. That is, attacker can easily get $M = \sqrt[3]{C}$.
Possibility 2: send same message M to 3 users using e = 3, so $C_1 = M^3 \bmod N_1$, $C_2 = M^3 \bmod N_2$, and $C_3 = M_3 \bmod N_3$. By Chinese remainder theorem, can get $C = M^3 \bmod N_1N_2N_3$. Rest is the same as possibility 1.

II. Given (N, e) = (33, 3) and d = 7.
If M = 3, since $M^3 = 3^3 = 27 < N = 33$, $C = M^3$, so cube root attack is possible;
If M = 4, since $M^3 = 4^3 = 64 > N = 33$, $C \neq M^3$, so cube root attack is not possible.

+Q7: Double encryption using same N for RSA (L5 P12, P23 & P24)

Since $C_0 = M^{e_0} \bmod N$ and $C_1 = C_0^{e_1} \bmod N$, we get $C_1 = ((M^{e_0} \bmod N)^{e_1}) \bmod N$.

Then based on ((a mod n)(b mod n)) mod n = ab mod n, we can get $((a \bmod n)^x) \bmod n = a^x \bmod n$,

That is, $C_1 = M^{e_1e_2} \bmod N$. This is same as using another $e = e_0e_1$. This only increase the size of e. But the size of e will not affect the security of the RSA. The security of RSA is highly related to the size of N.

So, **false**, this double encryption will not increase the security.

Week 4 - Public Key Cryptos Part 3 & Hash Functions (Q8 - Q15)

!Q8: Common misinterpretation of public key cryptos (L7 P10 & P11)

False. Signature CANNOT identify the sender. The signed message $[M]_{Alice}$ is public to everyone, and everyone can encrypt M by Bob's public key, so, everyone can send $\{[M]_{Alice}\}_{Bob}$, not just Alice herself.

+Q9: MIM attack on Diffie-Hellman (L7 P5 & P6)

False, MIM will not be successful.
Although Trudy can get $g^{at} \bmod p = (g^a \bmod p)^t \bmod p$ and $g^{bt} \bmod p = (g^b \bmod p)^t \bmod p$,
she cannot compute $g^{abt} \bmod p$ using $g^{at} \bmod p$ and $g^{bt} \bmod p$ without knowing a or b.

Q10: Data integrity and non-repudiation (L7 P9)

I. Non-repudiation means signee cannot deny the signature later

II. No, MAC doesn't provide non-repudiation since the key is shared between the two parties that are communicating.

III. Yes, digital signature provides non-repudiation, since the signee is the ONLY one who knows his/her private key.

Q11: Digital certificates (L7 P12)

I. A digital certificate contains the username, and his/her public key.

II. Can also include other info such as birthday, blood type, etc. (or any info that's reasonable)

III. Minimize the amount, so the certificate won't be too big. Also, no need to expose that much unnecessary personal information (or any reason that's reasonable).

!Q12: Collisions in hash functions (L8 P6 & P11)

False. Collision definitely will happen since the input space is larger than the output space.
For a secure hash function, collisions still happen, but it is hard to find any collisions by computation.

!Q13: Hash function to reduce spam emails (L8 P21)

False. The usage mention in class will only make it harder/more costly to send spam emails, NOT to detect and block them.

Q14: Hash used in online-bid system (L8 P20)

IV. Two properties involved
- One-way, that is, given h(bid), hard to find the bid.
- Collison, or more precisely, weak collision, since even knowing the bid and h(bid), bidders cannot easily find another bid' ≠ bid such that h(bid') = h(bid).
  That is, bidders cannot deny or change their bid afterwards.

V. The hash is secure since Trudy can only do a brute-force attack to try all the hashes to find collisions. But if Trudy can guess the range of the bids, then she can just hash those bids in the range (i.e., has fewer hashes to compute), which means, it is possible to compute all possibilities in a reasonable amount of time.

Q15: n-bit hash, m collisions, how many hashes (L8 P8 & P9)

I. n-bit hash means there are 2^n possible hashes.
Suppose we need to compute x hashes, then the number of comparisons (compare pairs) = ${}_xC_2 = x^2$.
Solving $x^2 = 2^n$, we get x = $\sqrt{2^n} = 2^{n/2}$ hashes to find a collision.

II. For every 2^n comparisons, we have 1 collision. So, we need $m*2^n$ comparisons to have m collisions.
Still, suppose we need to compute x hashes, the number of comparisons = ${}_xC_2 = x^2$.
Solving $x^2 = m * 2^n$, we get x = $\sqrt{m * 2^n} = \sqrt{m} * 2^{n/2}$ hashes to find m collisions.