Problem 1.

Solution: In this problem, we are given two message groups, one can denote as $M_1 = \{0,1\}^{1000}$ and the other is $M_2 = \{0,1\}^{2000}$, we are also given a hash output sets $Y = \{0,1\}^{128}$, this problem can be seen as to solve preimage problem for both hash fuctions, for a given $y = d$, find out two m, $m_1 \in M_1$, $m_2 \in M_2$, Such that $h_1(m_1) = h_2(m_2) = y$.

We know that the possiblity of solving this problem is equal to

$$\varepsilon = P(\text{successfuly finding } X) = 1 - (1 - \frac{1}{M})^{Q} \simeq \frac{Q}{M}$$

for $\varepsilon = \frac{1}{2}$, we have $Q = \frac{M}{2}$ times. since $M = |Y|$, we know that, the equal number of messages has to be tested for both h_1 and h_2 in order to find a message hashes to d, that number is $Q = 2^{127}$.

Problem 2.

prove it is easy to solve second preimage for any $x \in \mathbb{Z}_2^m$ without having to solve a quardratic equation.

Solution: in the problem, we are given that a hash fuction $h: \mathbb{Z}_2^n \to \mathbb{Z}_2^m$, where $n = m$.

$h(x) = x^2 + ax + b \pmod{2^m}$, it is not Second Preimage resisent, because, for every $x' \in \mathbb{Z}_2^n$ of the form: $x' = -x - a$ is a valid solution to the Second preimage problem.

Because we could put x' into our given hash fuction, we get:

$$h(x') = (-x-a)^2 + a(-x-a) + b$$
$$= x^2 + 2ax + a^2 - ax - a^2 + b$$
$$= x^2 + ax + b.$$

So we have no need to solve a quardratic equation to solve Second preime for any $x \in \mathbb{Z}_2^m$.

Problem 3.

(a). this question is about collision resistence. if the attackers would like to find two distinct messages that share the same hash output, $h(m) = h(m')$, beause SHA-1 output is of 160 bit long. this is kind of the same as the birthday problem, find a collision is equivalent to find two persons share same birthday. so $Q \approx 1.177\sqrt{M}$, in the problem, M is number of possible hash output, which is 2^{160}, which makes $Q = 1.177 * \sqrt{2^{160}} = 1.177 * 2^{80}$, so it is the queries that need to made to find the two messages.

(b). if we want to find some messages m for some observed hash output y, it is equivalent to find the preimage of the y, which means in this case, the attack will need 2^{512} queries, since in SHA-1 we split message into 512-bit blocks, which is kind of impossible for the compute ability of computers nowadays. it means SHA-1 has a strong preimage resistance.

(c). I don't think it is a good idea. although the appending operation seems to make the algorithm more complicated, the attacker could find out an appropriate message for this $h(m \| h(m))$.

Problem 4.

first, we could assume that there does not exsit $\bar{x} = \bar{x}' \| \bar{x}'' \in X$, that $h(x) = h(\bar{x})$ for some $x \in X$. which means, for every $x \in X$, if $x \neq \bar{x}$, then $h(x) \neq h(\bar{x})$, $f(x' \otimes x'') \neq f(\bar{x}' \otimes \bar{x}'')$

now we know that $f(x' \otimes x'') \neq f(\bar{x}' \otimes \bar{x}'')$

$$\Rightarrow \quad x' \otimes x'' \neq \bar{x}' \otimes \bar{x}'', \text{ for every } x, \bar{x} \in X.$$

if $m = 8$ and $x = 00111100$ and $\bar{x} = 11000011$.

$$x' = 0011 \quad x'' = 1100 \qquad \bar{x}' = 1100 \quad \bar{x}'' = 0011$$

$$\Rightarrow \quad x' \otimes x'' = 0011 \otimes 1100 = 1111$$

$$\bar{x}' \otimes \bar{x}'' = 1100 \otimes 0011 = 1111$$

so the result is contridict with what we assume above. which shows that the given hash fuction is not second preimage resistant.

Scanned with CamScanner

problem 5.

(a) Now we have $\begin{cases} \gamma = \alpha^k \pmod p \\ s = am + k\gamma \pmod{p-1} \end{cases}$, asked to prove $\alpha^s = (\alpha^a)^m \gamma^\gamma \pmod p$

$$\alpha^s = \alpha^{am+k\gamma} \pmod p \Leftarrow s = am + k\gamma$$
$$= \alpha^{am} * \alpha^{k\gamma} \pmod p \leftarrow \text{seperate}$$
$$= (\alpha^a)^m \cdot \gamma^\gamma \pmod p \leftarrow \gamma = \alpha^k$$
$$= \alpha^s$$

(b)

Now we have $\begin{cases} \gamma = \alpha^k \pmod p & ① \\ s = a^{-1}(m - k\gamma) \pmod{p-1} & ② \end{cases}$ asked to verify $\alpha^m = (\alpha^a)^s \gamma^\gamma \pmod p$

$$\alpha^m = (\alpha^a)^s \gamma^\gamma \pmod p$$
$$= (\alpha^a)^{a^{-1}(m-k\gamma)} \cdot \alpha^{k\gamma} \pmod p \quad \leftarrow ① + ②$$
$$= \alpha^{m-k\gamma} \cdot \alpha^{k\gamma} \pmod p \quad \leftarrow \text{combine.}$$
$$= \alpha^m$$

roblem 6.

(a). Now we know $p = 31847$, $\alpha = 5$, $\beta = 26379$, and signature $(\gamma, \delta) = (20679, 11082)$ and also $x = 20543$, beacause we know the verification in ElGamal is like this $\beta^\gamma \gamma^\delta = \alpha^m \pmod p$.

we first compute $\beta^\gamma \bmod p$, and $\gamma^\delta \bmod p$, then their product mod p.

second compute $\alpha^m \pmod p$.

finally compare the two results, check if they are equal so to verify.

here is the result: $\alpha^m = 20688 \pmod p$, $\beta^\gamma * \gamma^\delta = 20688 \pmod p$.

(b). To find the private key a, we need to compute this $\beta = \alpha^a \pmod p$.

I search online, there is an algorithm Shanks, called baby step-giant step.

first assume $a^k \equiv b \pmod p$, $k = in - j$. where $n = \text{ceil}(\text{sqrt}(p))$.

we have $a^{i \cdot n - j} = b \pmod p$, $\Rightarrow a^{in} = a^j \cdot b \pmod p$.

we could first store all the values in the leftside, then loop through the right side try each j to see when right value will equals left.

from the problemtic file, we get the private key is 7973.

(c) This problem is kind of the same as the (b). we are given $\gamma = \alpha^k \pmod p$ if we would like to compute k, we use the same algorithm as (b), and we get $k = 19387$.

Problem 7.

This problem is kind of the same as Diffie-Hellman key Exchange problem.. the difference is that it has four parties rather than two.

In the original problem, Alice and Bob shared a secret key, K_{AB}.

Now Bob Ted Carol and Alice could do the same, that first chose a large prime p and a primitive root α.

Then the four of them all generate a random number, which is b, t, c, a.

Then they send message one to another like

$$\begin{tikzcd} \text{Bob} \arrow[r, "\alpha^b"] & \text{Ted} \arrow[r, "\alpha^t"] & \text{Carol} \arrow[r, "\alpha^c"] & \text{Alice} \arrow[lll, bend left, "\alpha^a"] \end{tikzcd}$$

$$\begin{tikzcd} \text{Bob} \arrow[r, "\alpha^{ba}"] & \text{Ted} \arrow[r, "\alpha^{tb}"] & \text{Carol} \arrow[r, "\alpha^{ct}"] & \text{Alice} \arrow[lll, bend left, "\alpha^{ac}"] \end{tikzcd}$$

$$\begin{tikzcd} \text{Bob} \arrow[r, "\alpha^{bac}"] & \text{Ted} \arrow[r, "\alpha^{tba}"] & \text{Carol} \arrow[r, "\alpha^{ctb}"] & \text{Alice} \arrow[lll, bend left, "\alpha^{act}"] \end{tikzcd}$$

after this kind of communication, Bob, Ted, Carol, Alice will get act, bac, tba and ctb respectively, they just need to raise the number to their own random number, they will get the $btca$. then they can communicate

Scanned with CamScanner

```
t_repo_yuanjieyue/assignment_3/src (master)
$ ls
problem6.c  README.md

Administrator@PC-20171119GJTZ MINGW64 /f/courses/cs5770_software_security/Studen
t_repo_yuanjieyue/assignment_3/src (master)
$ gcc problem6.c -o problem6

Administrator@PC-20171119GJTZ MINGW64 /f/courses/cs5770_software_security/Studen
t_repo_yuanjieyue/assignment_3/src (master)
$ ./problem6
---------------------
(a). prod is 20688 and exp3 is 20688
Verified
---------------------
(b). the private key a is: 7973
the ver num is: 26379
Verified
--------------------
(c). the k is: 19387
```

problem 8.

we know that SSN has 9 digits, and we can assume that we have Q students in the class, so the problem becomes find the possibility of two students share the same last 4 digits of their SSN, we could transform it to P[at least two out of Q students have a same last 4 digits SSN]

$= 1 - P$[nobody in the class of Q students has the same last 4 digit SSN]

it is kind of the same as birthday problem.

$$P = 1 - e^{\frac{-Q(Q-1)}{2M}}.$$