

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 functions, 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 possibility of solving this problem is equal to

$$\epsilon = P(\text{successfully finding } x) = 1 - \left(1 - \frac{1}{M}\right)^Q \approx \frac{Q}{M}$$

for $\epsilon = \frac{1}{2}$, 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 quadratic equation.

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

$h(x) = x^2 + ax + b \pmod{2^m}$, it is not second preimage resistant, 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 function, we get:

$$\begin{aligned} h(x') &= (-x-a)^2 + a(-x-a) + b \\ &= x^2 + 2ax + a^2 - ax - a^2 + b \\ &= x^2 + ax + b. \end{aligned}$$

so we have no need to solve a quadratic equation to solve second preimage for any $x \in \mathbb{Z}_2^m$.

Problem 3.

- (a). this question is about collision resistance. if the attackers would like to find two distinct messages that share the same hash output $h(m) = h(m')$, because 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 exist $\bar{x} = \bar{x}' || \bar{x}'' \in X$, that $h(x) = f(\bar{x})$ 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 $f(x' \otimes x'') \neq f(\bar{x}' \otimes \bar{x}'')$

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

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

$$\begin{array}{cc} \swarrow & \searrow \\ x' = 0011 & x'' = 1100 \end{array} \quad \begin{array}{cc} \swarrow & \searrow \\ \bar{x}' = 1100 & \bar{x}'' = 0011 \end{array}$$

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

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

so the result is contradict with what we assume above, which shows that the given hash function is not second preimage resistant.

Problem 5.

(a) Now we have $\begin{cases} r = 2^k \pmod{p} \\ s = am + kr \pmod{p-1} \end{cases}$, asked to prove $a^s = (2^a)^m r^r \pmod{p}$

$$a^s = 2^{am+kr} \pmod{p} \leftarrow s = am + kr$$

$$= 2^{am} * 2^{kr} \pmod{p} \leftarrow \text{separate}$$

$$= (2^a)^m * r^r \pmod{p} \leftarrow r = 2^k$$

$$= 2^s$$

(b)

Now we have $\begin{cases} r = 2^k \pmod{p} \\ s = a^{-1}(m - kr) \pmod{p-1} \end{cases}$ ^① asked to verify $2^m = (2^a)^s r^r \pmod{p}$, _②

$$2^m = (2^a)^s r^r \pmod{p}$$

$$= (2^a)^{a^{-1}(m-kr)} * 2^{kr} \pmod{p} \leftarrow ① + ②$$

$$= 2^{m-kr} * 2^{kr} \pmod{p} \leftarrow \text{combine.}$$

$$= 2^m$$

Problem 6.

(a). Now we know $p = 31847$, $a = 5$, $\beta = 26379$, and signature $(r, s) = (20679, 11082)$ and also $x = 20543$, because we know the verification in ElGamal is like this $\beta^r r^s = 2^m \pmod{p}$.

we first compute $\beta^r \pmod{p}$, and $r^s \pmod{p}$, then their product mod p - second compute $2^m \pmod{p}$.

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

here is the result: $2^m = 20688 \pmod{p}$, $\beta^r * r^s = 20688 \pmod{p}$.

(b). To find the private key a , we need to compute this $\beta = 2^a \pmod{p}$.

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

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

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

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

from the problematic file, we get the private key 257973.

- (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.

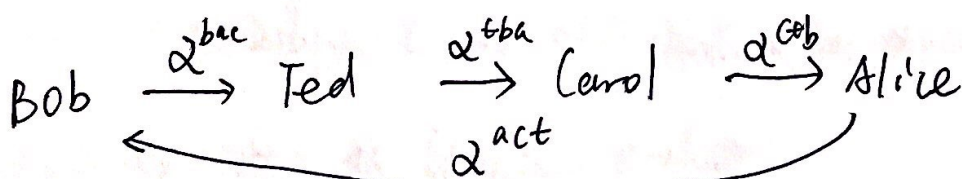
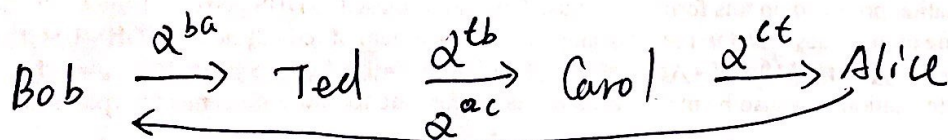
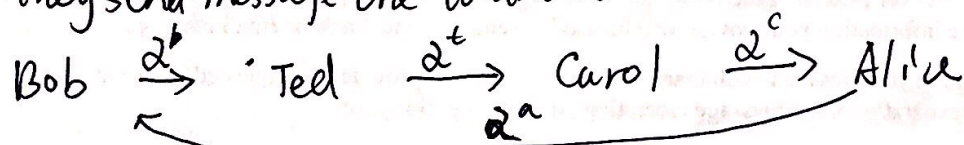
The difference is that it has four parties rather than two.

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 choose 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



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


```

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.

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