Cryptography

Homework #6

Sam Quinn
CS427
03/07/2016

1)

```
? p = 6323947392563;
? A = 6233663610066;
? B = 4871694980854;
? g = 2;
? a = znlog(Mod(A,p), Mod(g,p))
%5 = 5111660011293
? b = znlog(Mod(B,p), Mod(g,p))
%6 = 6266770399842
? Mod(A,p)^b == Mod(B,p)^a
%7 = 1
```

The shared key between the two would be **3433487769333.**

2)

a)

Elgamal is susceptible to CCA attack, where we can make a ciphertext that will produce any value we want within the cyclic group that the original message **m** was created in. Elgamal is multiplicatively homomorphic, meaning you can perform multiplication operations on the ciphertext themselves that when decrypted will result in the same multiplication on the plaintext. So to get only the original **m** out in plaintext we can create a ciphertext **c'** with a random with message **m'.** We can multiply the ciphertexts **c** and **c'** together to get **c*c'.** We can send **c*c'** to the decryption oracle that will output the message **m*m',** which all we would have to do is pull out the **m'** from the resulting plaintext. After we extract the message **m'** the original message **m** will be left over.

b)

Given 2 unknown ciphertexts we can do the same attack as mentioned above with the multiplicatively homomorphic aspects of Elgamal encryption. The only change would be that we do not need to remove any of the messages from the returned plaintext.

3)

a)

There are always going to be more samples than there are numbers in the domain **N**. Because the function $q = \sqrt{2N}$ will always be larger than the domain **N.** With the idea of birthday bounds there is a %60 chance that the random number **ri** will be the same as **rj** since both values are translated onto the domain **N.** If we now take a random **x** value that we add to another **rj** then mod **N** we are given the same probability since that value will be within the same domain **N** based on the mod of **N.**

b)

Given the discrete logarithm problem $g^r \equiv_p g^x * g^s$ we can make the following reductions to show that you can solve for **x** with basic logarithm rules.

$g^r \equiv_p g^x * g^s$

$g^r \equiv_p g^{xs}$

$log_g(g^r) \equiv_p log_g(g^{xs})$

$r \equiv_p xs$

$\frac{r}{s} \equiv_p x$

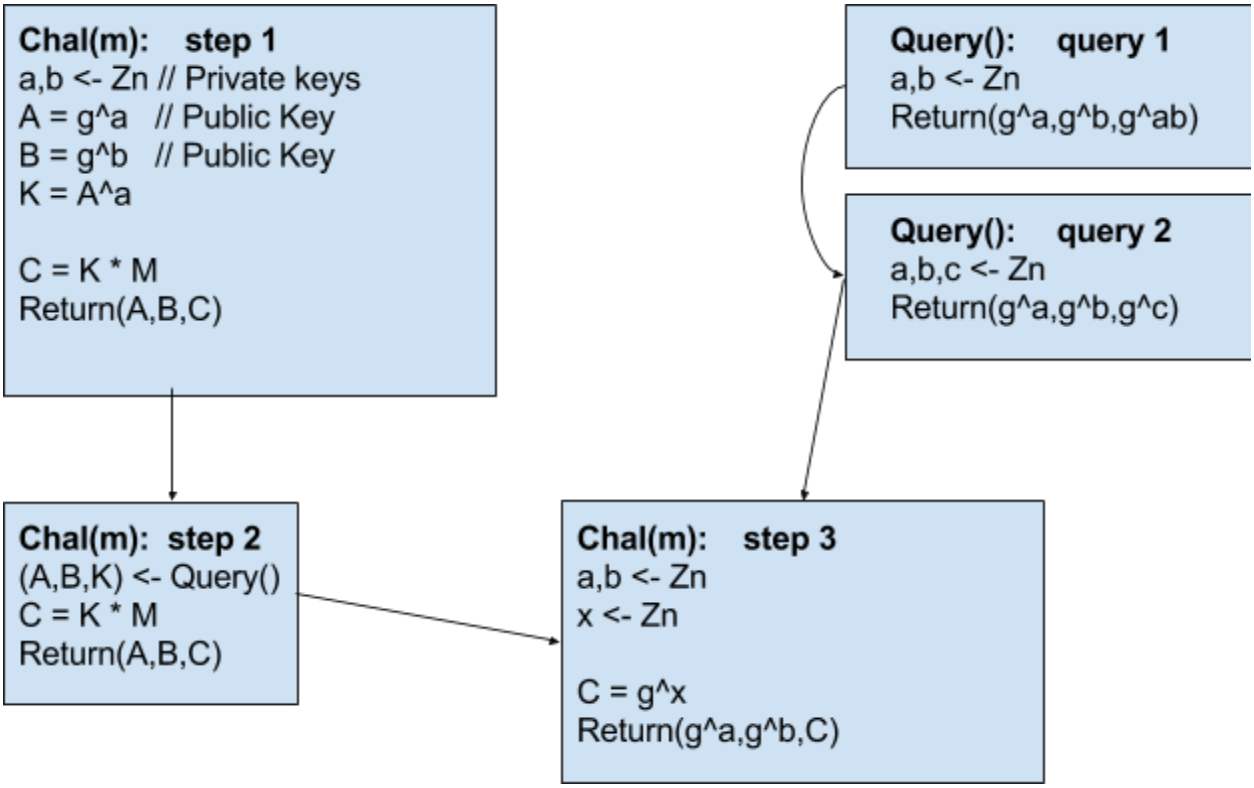
c)

Based on the size of Z_p^* we would be able to create an algorithm with the computational time of $O(\sqrt{p})$. This is due to the fact that **r** and **s** will be within the set of Z_p. To find both **r** and **s** we can use the same birthday bounds argument expressed in part a) therefore we can assume that we can find these values in $O(\sqrt{p})$ time, where p is the size of the domain. If we are to solve the discrete algorithm for a value **x** such that $g^r \equiv_p g^{xs}$, which would take a constant amount of time and does not add to the total $O(\sqrt{p})$. Therefore if we combine the too observations we are able to theoretically break a Diffie Hellman

Extra Credit

If we can prove that any public key encryption scheme is CPA secure for one ciphertext transfer then it will also be secure for many ciphertext transmissions.

For the key exchange we can use Diffie Hellman which is CPA-Secure when DDH assumption holds in group <g>.



In step 1, we generate the public and private keys the same as Diffie Hellman which is known to be CPA secure. We then encrypt the message with the **K** that both Alice and Bob would own after the key exchange.

In step 2, we can extract the key exchange from the Chal function to make modifications. Query 1 stage we have just standard Diffie Hellman key agreement where the generator is raised to both of the private keys to generate a shared key. Query 2 stage is where since **a** and **b** are generated uniformly random the multiplication of the two is also uniformly random which we can just pick randomly and name it **c**.

In step 3, we add back the query 2 stage to the Chal step 2 to get the final form. This is a secure encryption scheme based on the primitives of one time pad, but would not be possible without the public key exchange between the two parties.