CS411 HW 3

Samuel Lee

24774

11/12/2019

1. a. The answer is x^6+x^5+1. First of all, we have to set the parameters of the binary finite field GF(2^m)/g(x), where the m is the extension degree of the binary field. In order to do to multiplication, we also need to know the coefficients of the irreducible polynomial(which is given in the question). Once we have the information, we can just multiply the 2 equations using bitwise shift operation and xor operation. We can find the multiplicative inverse of A(x), T(x)= A−1(x). The multiplicative inverse of x7+ x6+ x4+x3+x2+1 in GF(28) is x7+ x6+x5+x4+x3.
2. There are two possible cases for this question. The first is: we can reduce the given digest, and we iterate through the second value of the RainbowTable. If we find the corresponding password, then we hash it t-2 times, and find the password. However, there is a possibility that we may not find the password after the first reduction. Therefore, we keep hashing and reducing it 'n' number of times until we find a corresponding password match. Then, like case 1, we iterate through it t-n times, hashing and reducing it until we are left with the final password.
3. For part a, if you have either Cp or Cq, you can find the gcd(Cp, n), which will give you the p value. This is because Cp*Cq = k^2e((p*q)^e), where p*q is n, and n is a multiple of Cp and Cq. There, once we have Cp or Cq, we can find the necessary values to decrypt the messages. For part b, in order to find n, we need to know the p and q values. We can determine the p value by finding the gcd(n,cq), given Cp or Cq. Once we have the p value, q value can be calculated by doing some simple division. After we have the values, we can simply plug in the necessary variables like d(which is the inverse of e and Phi of (N)). Then we can decrypt the ciphertext by implementing and using a simple power modulus function by giving the ciphertext, d and n

value as the parameters. The decrypted ciphertext is : Insanity is doing the same thing, over and over again, but expecting different results.

4. For part a, suppose we choose a number k (relatively prime to N), where x = k^e mod N. If we multiply this with m^e mod N, you will get

   x*C = k^e * m^e (mod N). Once we have this value, we feed it into the oracle, which will give us: (x*c)^d = ((k*m)^c)^d (mod (N)). The c and d values will cancel out because they are inverse of each other. We can then find the modular inverse of k, to find m.

   For part b, we did the implementation of part a. We chose a k (=2) which is relatively prime to N. Then, we made created a c1= k^e mod N. Then we multiplied c1*c. We fed the numbers from the c1*c to the oracle, which gave us a response. With this given numbers, we found the inverse of K in relation to N(modinv(k,N)) and solved for the plaintext with message*inverseOfK mod N, and converted the byte form to text form.

   The answer is: You discovered my verry secret message:) Bravo.