1. Given the public key (N, e) = (3127,17)
    a. To find all M such that M = C, we need to find M^e such that M^e mod N = C and since C = M which implies to us that M^e mod N = M. This can be the same thing as M^e mod N = M. This means we need to find M such that M^{17} mod 3127 = M. We also know that M is in the range 2 till 3126 (as we are finding values for M not being 0 and 1).

       To accomplish this, we perform repeated squaring for all these values to see when M^{17} mod 3127 = M mod 3127.

       Running the program, we get the following values

       [235, 295, 530, 531, 825, 1061, 2066, 2302, 2596, 2597, 2832, 2892, 3126]
    b. Our smallest value is 235 so in this case we aim to find 235^{17} mod 3127

       First we convert 17 into binary which is 1001

       From here we build the following list from (1,10,100,1001) which is the same as (1,2,4,17)

       235 = 235 (mod 3127)

       $235^2 = 235^2 = 55225 = 2066$ (mod 3127)

       $235^4 = (235^2)^2 = 2066^2 = 1$ (mod 3127)

       $235^{17} = (235^{4*4+1}) = (235^4)^4 * (235) = 1*235 = 235$ (mod 3127)

    c. In practice, it shouldn't be a huge security concern as modulus N is much larger than 3127, especially if we use two very large prime numbers for p and q (Recall p * q =N). This means that as p and q are large, trying to factor N becomes very hard which means that an attacker will struggle to find the private key d.

2. X: Specified Constant which is public, M is a Single Block Message which is the size of key in block cipher. The hash is Y = E(X,M)
    a. One-Way Property:
    The one way property is that given Y, is it is hard to find X such that Hash(X) = Y. In this case, given Y is hard to find the message M such that Y = E(X,M). Here we assume the block cipher is secure. As we use the message M in place of the key and X in place of the plaintext, the attackers aim is to get M from X and Y alone since they don’t have access to E. Since they don’t have access to the block cipher, it makes it impossible to retrieve M from this, thus satisfying the condition
    Weak Collision Resistance:
    Weak Collision Resistance is when given a message M and a hash function, it is unfeasible to find another M’ such that M != M’ but Hash(M)=Hash(M’). So in this case we need to find M!=M’ such that E(X,M) = E(X,M’). As we assume the cipher is secure, based off the properties of a secure block cipher (pseudorandom nature) we know that the hash is weak collision resistant.
    b. Since we have two keys K_0 and K_1 and that $E(P,K_0) = E(P,K_1)$, this leads to a collision in the hash function. Due to the fact that $E(P,K_0) = E(P,K_1)$, and assuming K_0 and K_1 are not the same, this means the weak collision resistance property no longer holds thus leading to an insecure hashing method.
    c. To extend the definition of the hash to include a second block, we can change the method to make it $Y = E(X, M_1 \oplus M_2)$ or even more complex by doing $Y= E(E(X, M_1 \oplus M_2) , M_2)$

3. Given an 8-character password which each character can be one of 64 different values which are then salted and stored in a file with 256 (same as 2^8) password

hashes. This means that we have 64^8 possible passwords which is the same as 2^{48} possible passwords. Mallet can test 2^{10} possible passwords per second and has access to a dictionary with 2^{30} possible passwords with the probability that the password is in the dictionary being 1/3

a. To crack the password of one user, we test for two different possible paths in that the password is either in the dictionary or it isn't.

   If the password is in the dictionary: $2^{30}/2^{10} = 2^{20}$

   If the password isn't in the dictionary $(2^{48}/2)/\ 2^{10} = 2^{47}/2^{10} = 2^{37}$

   Expected time would be: $1/3 * 2^{20} + 2/3 * 2^{37}$

b. Pr(Not in Dictionary) = 2/3. This is for one given password. Therefore to find the probability that at least one is in it would be $1 - (2/3)^{256}$ as $(2/3)^{256}$ is the probability that none of the passwords are in the dictionary and by subtracting one, we get the probability that is it possibly in the file. As that value is very small, the possibility that it is in the dictionary is approximately 1, meaning it is always likely to be in the dictionary

c. Average work ≈ size of dictionary / Pr(Password is in the dictionary)
   $= 2^{30} / 1/3$
   $= 3* 2^{30}$