Public key application

We have three categories of application, encryption and decryption, is going to provide confidentiality and these 2 will provide authentication and the other will provide key exchange, these three things basically ,confidentiality ,authentication and key distribution ,some of the algorithm is going to used for all the three, but some of the algorithm will be specific to 1 ,either it could be confidentiality or key exchange or something like this , one is RSA they vary and encryption and decryption yes, digital signatures it is used and key exchange it is used as you see with RSA algorithm we can do three things. Elliptic curve it also accepts all the algorithm but generally, we are not going to use this one for encryption and decryption because we have to convert some things as points and so on, we will see the details later, but generally we are going to use it for digital and key exchange, Eljemal is used for all the algorithm which is not here, differ hellman is also very important , almost all the security application is going to used in RSA, and the differ hellman and it is used only in key exchange, it is perfect for key distribution and it comes with a new idea, In the distribution of key what we doing Is were transmitting the key to the places and this differ hellman says instead of transmitting like that why don't we calculate the keys on the both sides, I am going to do some calculate some things and it will be based on some calculations and no one will be able to calculate, if you don't know that parameters, so instead of calculating and distributing the keys we are going to calculate to both source and destination ,in a way that its not going to be easy for others to make the calculation and it is very good algorithm ,and the other one is DSS which uses only digital signatures and no for other two ,because it is a signature algorithm and the algorithm is called DSA (digital signature algorithm), and elliptic curve have differ hellman application ,it is called ECC differ hellman and also it has got , ECC elliptic curve cryptography DSA and I should also say there is also ECC Eljemal and also we don't have ECC RSA because RSA is a single algorithm ,we have elliptic curve version of differ hellman and DSA, and also it is widely used which we are going to see later, so these are the crypto algorithms and we see there is no design and we don't have 100's of algorithm

RSA

Designed by Rivest , Shamir and adleman professors of MIT,R stands for rivest ,Shamir stand for Shamir and a stands for adleman ,Best known and widely used in public key scheme, based on exponentiation in a finite field over integers modulo a prime we know that we have to work in field anyways will use modulo prime, and uses large prime integers which is 2048 bits and not 1024 ,security due to high cost of factoring large numbers, so lets have a look, first of all each user generate a private/ public key pair, first of all we are going to generate 2 large primes ,prime is also difficult , so P and q and multiply them and that is my modulus and I am going to calculate the quotient function and this the function what will be my modulus p-1 times q-1 ,this is the calculation of quotient and iam going to select a number in random and that is going to be quotient and GCE between E and quotient will be 1, you said this is for even number right and if it is a odd number then different condition, E is going to be your public key and I said that E and private key has some relation ships and that is called e.d = 1 in mod quotient, in other words D is multiplicative inverse of a in mod quotient, d is the private key, e is the public key ,so e.d = I in mod quotient so that is the idea basically and you know the relationship now and you are going to publish public key along with modulus ,e without modulus doesn't mean anything because we are going

to do encryption in mod n, if you don't know n value, just knowing e you cant encrypt anything, so you are going to keep p, q, d secret, we are actually deleting p and q and I am not going to use p and q, once I going to calculate d then iam going to delete p and q so d is known and n is known and also you know the relation ship between e and d and the question is this, knowing the algorithm and knowing modulus and knowing public and it is infeasible to find corresponding private key, is it true, ?..you know n , you e, how can you calculate d, where is the security ?.. to find the d ,you must calculate and d is inverse of d in quotient how are you are going to calculate quotient. An you have to know p and q to find p and q what you have to do is you have to factorize n and that's why I told you that you have to factorize n. if you cannot factorize n fast enough then you cannot find p and q , you cannot calculate modulus ,and you cannot calculate inverse of p and q, you see knowing the relation e.d in mod quotient doesn't reveal anything ,and it is a very clever design, look at this ,it seems that its not easy to that kind of stuff, we didn't understand properly like e and d ,cant we?..but as you see we cannot , calculation depends on quotient or knowledge of quotient ,prime factors of n which is n ,which is modulus to find n, if you cannot calculate n then you will not get the private key, {question from a student}. The value of d is going to be 0, when you find n number multiplied by e then the number will be less than n. because this number is less than n , {question from a student}. So the value of d is 1 to whatever the result of this multiplication, quotient n is true ,but n is also true because this number Is less than n, so if it less than n then n is less than d, then n is easily simplified, so this only the key setup this is not the encryption and decryption is easy, we said in the description that those things must be easy, and look at the encryption and decryption, and it is something like this, M^emod N, so this is you ciphertext and that's why to encrypt any message you have to know e and n , you have to know mod n, so that's why we have to post modulus n along with public key information , you see M^e mod n, but the problem is this M is a large number and e is a large number, if e is 200 decimal digit number , you have to multiply n many times and you have to reduce the result its going to take enormous amount of resources and time, although we say that this algorithm is easy, its not going to be that easy, but if you select ,e small ,that's going to be easy , you can select 200 decimal digit numbers as we discussed ,but this time if e is small ,then d will be bigger ,in the implementation what we are doing is ,we are trying to select same size for e and d, so that there security will be better, but because of performance problem we are going to select d a bit large number and e as small number, but do the decryption on the other side ,so for the encryption it should be fast, in other words encryption will be faster than decryption like C^dmod n, same kind of calculation but with private key, so this way as you see we are encrypting and decrypting the keys and todays words algorithm depends on this algorithm, its little bit compliment for keys as you have to find large prime numbers, and so on, but encryption ,and decryption is very straight forward process, all you have to do is power the message with private key in the same modulus ,and the sender will know public key and modulus and without modulus this will not calculate ciphertext and so you guys tell me this is block or stream cipher ??.. the value of n must be between 0 and n, and lets say that my modulus is 35 and my message value could be starting from 1 to 34, if my message has more than 34 as a number , then I am going to split it just like this, lets say my number is 3547, this is my message and my mod is 35 and I am going to do RSA encryption I cannot just go and calculate this ,because, my message is bigger than m, then I am going to get 2 digits from here which is 35 then I am going to get only one number that 3. So this Is going to be 3 power 3 ,so 547 is bigger than 35 and 54 is bigger than 35, and I am going to get 5, 47 is bigger than 35 and then I am going to get 4 and 7, so I am going to split the message into

blocks and what is the blocks size, the concept of the block sizeis 10 or 100 bits it depends on value lets say my modulus is 336, if that is the case then I can get 354 as my message , lets say 76 , now it is 354 so I can get this one what is the block size ,I don't know it depends on this one ,is it binary ,what we are doing in binary Is ,If the modulus 2048 bit number then you are going to select less than 2048 bits basically, and the modulus and the keys are not goingto have1000 zeros or 2000 zeros, and the last digit is 1 and generally 1's are distributed 1 randomly and the value is going to be less than 1, if the message is text you can convert into numbers like ASCII number and that number is 1 big large number if you are using 200 decimal digit number that means that you can get 200 decimal digit number, from the message as 1 block of the message to play it safe you are not going to get 200 decimal digit number but you will get 150 decimal digit number because most probably that number, is going tobe less than modulus and that's going to 150 decimal digit number , but when you decipher it ,you are going to recalculate those numbers again back and if you use ASCII coding you are going to get the ASCII number of the message, generally you are going to use the ASCII representation of the message ,was it a stream or block cipher and the answer is block cipher.

RSA use

In RSA algorithm what we did so far is this, first of all we selected large two prime numbers, and calculate the modulus that is multiplication of these two numbers and calculate the quotient that Is p-1 times q-1 an we selected an integer e that's 1 and quotient then calculated d that is inverse of e and mod quotient of n then we publicly know e and n and keep it secret and not p and q secret and encryption process, the value of M will be less than modulus and in decryption is very straight forward it is c^dmod n.

Example.

Lets see a example for better understanding, you choose RSA, lets select two prime numbers 17 and 11 .we are going to calculate p times q that is 17 times 11 which is 187, so that is my modulus right now and then iam going to calculate quotient which is p-1 times q-1 which 16 times 10 = 160, as I said quotient is going to be even number ,so only way the quotient must be even is gcd must be 1 between quotient and e , e is relatively prime to quotient if you select any odd number ,you are going to satisfy that condition basically, that's a random selection no other extra requirement ,any number between 1 to quotient can be chosen , but you have to satisfy this gcd condition with those numbers, I found e and I selected e and that's my public key, I have to calculate the private key and what e times d = 1 mod 160 and what is the inverse of 7 in mod 160 , Iam going to find a number and if I multiply that number the result is going to be 160, that's the inverse 23, times 7 = 160 = 1 mod 160, so that means that 23 is my private key and that key I am going to keep secret ,but I am going to send 187and 7 information to everybody so that everybody will know that my public key is 7 and my modulus is 187, but no one knows my 23 that is my private and the prime numbers, 17 and 11 and I am going to remove it as I am not going to use it anymore, so I am going to keep it 23 and 7 is public and modulus is public and by knowing187 again it is difficult to find 23 is it?..i am wrong the number is very small it is very easy to find 17 and 11, it doesn't provide any security at all for all numbers this must be secure but in this case, it is easy to find 23, but for large number its going to be difficult my number is 88 and I can encrypt like

that 88^7 mod 187 as 88 is less than 187 the result is 11, and it is ciphertext and iam going to transfer it to destination and when the recipient receives 11, its going to use private key 23 , 7 and 23 are public and private key pair and by using 187 it is going to recover, plaintext back, so what we did is this ,lets say this is the sender and this is the recipient sender has 88 and sender knows public key and modulus that is 7 and 187, by using the public key and modulus it is going to encrypt 88 ,88 in mod 187 that's equal to 1 1and transmits it to destination and destination has the knowledge of 23 and its going to use that power of 11^{23} mod 187 and it will recover 88 back, so that is the plaintext, so you see how easy this algorithm is , you can use this algorithm quiet frequently, the major thing is this algorithm to distribute the symmetric encryption algorithm keys , like key of 88 basically, for small messages we can use RSA algorithm like 128 bits, 256 bits key size, it is pretty easy to encrypt those size packets and RSA is used to sign digital signatures and it is also used for, session key encryption and not for large encryption, you can say it is very difficult, imagine that you have one gigabyte of data, and then you are going to do this calculation where key length will be huge, modulus will be huge and its going to be, too much time, that's why we are not using RSA for bulk encryption.