Key distribution

The major thing that we see in all the major crypto algorithms is the key distribution, we said that in crypto algorithms key distribution will be a major deal. Because n square relation is required for the keys ,symmetric is required for the keys and we have to find somewhere, to distribute this keys securely and generally, algo's are quiet good in encryption link DES, AES or those kind of things, but what we do fail is in key distribution of the algorithms, if the key distribution is not secure enough ,whatever, the algorithm you are using doesn't make any sense, the major thing for us is to have to have a secure key distribution along with the secure algorithms, we will see later on, in the public key crypto systems in the public key crypto system ,we are going to solve this problem ,we will be able to distribute securely the symmetric encryption keys but, imagine that we don't have the public key crypto algo's , if that is the case how am I going to distribute the keys, that's my question ,we can say this, lets say ,A and B are trying to communicate and they are having a secret key that have but what they can do is A can select key and physically deliver to B. widely used in military services, and those security officers has the keys and everything they travel and go to other side, and in that type other side get the key and that person get the key and that person, get the security clearance and everything and in other words put trust on the person basically, and also the stuff they are going to carry protection and it is still used for big things and third party can select and deliver key to A and B ,remember the first week remember the network security model and the judge ,and it was difficulty to find some body to trust ,so this is going t o be little bit tough for us, but we will see a example and we will see a third party ,but this third party, not a person and it's a server and its going to distribute some keys and I will show the example for that , or if A and B are communicated previously ,if they have a previous communication they can use a previous key to communicate with a new key and it is generating the key randomly and it is encrypting it to the other side. So as you see over here, its pretty tough and it is having a secret key and I am using that secret key, to distribute my new secret key and what is the draw back for this one??..if the previous key is hacked or broken ,this is going to be broken immediately, when you use a key that is longer ,there is a great change that someone already discovered your key and never use old key to discover something , old keys are old keys, you cannot be sure of the security of the old keys just don't use it .if A and B has secure communication between C, C can relay key between A and B, this third party I am going to show you an example, by using only symmetric algorithm to distribute keys to the users and those users are going to use those keys for some symmetric algorithms ,before going into that we are going to name that keys as session keys, these secret keys that we are going to use in crypto algorithms are called session keys ,we are going to use session these keys to encrypt data and we are going need it to provide cryptographic protection, that's why these session keys will be used to encrypt the data, what about security of the session keys, it says cryptographic protection, to protect the session keys ,we are going to have master keys to encrypt the session and session keys to encrypt the data, this is the general idea behind the network security and if you look at the session keys and protection of the session keys we are going to see the protection of the keys you may ask what Is the protection of master keys, this is a never kind of never ending thing ,we are going to encrypt the master keys then encrypt those keys and so on , these are the requirements for the cryptographic protection ,and what they are doing Is they are writing and keeping in temper proof, and temporary resistance devices, and then they are putting in saves and they are hardware protection its hardware protection of the master keys, it doesn't require

any cryptographic protection, are you clear?..the life time of the session key message keys are short , master keys are quite large, session keys are arrange these keys in every session ,from 6 months to one year of your application ,but average is they do change the session and for every new session you must have a new session key and you can see the lifetime of the session is less than the master keys. These are the key distribution of the algorithms.

Key distribution scenario

This is the key distribution technique which is called (…….) which is a key distribution protocol. This is the basic protocol that is used in Kerberos, which is totally a symmetric encryption control and authentication protocol which is Kerberos and in that Kerberos they are using this key distribution protocol and it is based on (…….) protocol, let see how it is used. Kerberos is protocol and it does only use symmetric, encryption algorithm and not public key algorithm and first we are going have this KDC, which is called key distribution center, and these two guys initiator A and responder B is trying to establish a secret key ,that secret key will be used later on to encrypt any data but before going to that session key , imagine that there is a software, and I am installing that software in the KDC center ,KDC server, and in the client side I am going to have client part of that software and while I am installing that software, what I am going to do is I am going to write some secret keys, shared between A and KDC and B and KDC and it is given and it is given during the installation process, of the lets say Kerberos of this protocol basically ,in other words KDC and this will share a secret key and I am going to call that master key and that is called K_a and also KDC will share a secret key with B ,and is called K_b ,I am going to call master key B ,which is also called long term key. And it is installed during the installation of the whole protocol ,ok, so initiator A initiates the request saying that I, want to communicate with a responder and I need a key and its sends the request with a number and this number is called N_1 and it is called nonce. This is random number, it sends the random number with a request ,why he or she is doing that??. Any idea??..to avoid any replays ,and this nonce guarantee the freshness of the request. So that they can understand that this is a new request or a old request and also we are going to use this nonce in the respond packages so that I can think that this is a response for that request ,so on receiving this request N_1 ,key distribution center will check this nonce first of all and it will see that it is a new noun and it will say that it is a new request and it will accept the new request and from the new request it will send the address of responder B and it will create a session key and that is going to be shared between A and B and I am going t o call that KS and it will put that KS in the packet along with the nonce so that we can understand this reply for this request and same case, is going to be put in same packet with the idea of A, this is network ideas of packet ,encrypted with key, that is shared between KDC and B, and the whole message is encrypted with the key that is shared between KDC and A, are we clear ?..when KDC receives this request it is going to generate a session key that is used between A and B. and it is going to put it in this packet and also it is going to put it in this packet K_s and also between the key that is shared between KDC and B .and the whole packet is encrypted between the key that is shared between KDC and A. so imagine that you are A and you receive this message, what are you going to do?..you receive message 2., first of all while using that K, I have that K information and remember that definition ,K_a is the secret key between KDC and A, we call it master secrets and it is installed and it is given ,and I am going to use same key to decrypt the message ,when I decrypt the message what I am going to do is K_s and requested N_1 and I will understand that this K_s is for this one ,that is for this, guy it recovered that K_s , so its going to use that K_s and what about this part E_{Kb} (K_s ,ID_a) this is encryption of key B ,that is shared secret between KDC and B , then A doesn't have that information, it cannot recover inside of this message. It can recover the encrypted message ,but it cannot see the content of the message. So it recovered the encrypted part and that is called ticket in Kerberos. So it recovered the

ticket in the third step it is going to forward that ticket to destination B because there is only one ticket In that scenario, that is B can decrypt this message ,no one else ,as no one else has the knowledge of K_b except B and KDC, on receiving this packet ,responder will use K_b and its going to recover K_s and from this packet it will understand K_s is for A ,K_s that is going to use for A and B ,so then if you understand this is key for A , so any message comes encrypted will be decrypted using this key {question from a student}..but this time initiator doesn't know that whether this guy received K_s correctly or not, so to assure both of them has the same K_s, responder B will generate a random number, and 2 another nonce will be recovered with the encrypted K_s ,and its going to be sent to A and if A can recover N_2 that means both entities have the K_s and what we will do is we are going to decipher this N_2 and A is going to be F_{N2} that means it is a function of N_2 and what I am going to do is I am going to send back to B ,when B receives this one and recovers this N_2 it will know that A has the same session key, now both entities A and B are ready, to exchange any encrypted data, but in that encryption the key that we are going to use is session key that we distributed in this protocol, ok, you see it canbe done only using the symmetric encryption algorithm so we are only using symmetric algorithms to distribute session key ,right now both entities have the session keys they can encrypt the data and they can transfer it in both ways, A to B and B to A. are we clear?..but if you look at this protocol you may say that why we are sending this to A ,as A is forwarding to this guy , why don't we send this packet directly to B, it can be done, there are some variations to this protocol that is available where KDC can transfer ticket directly to responder, then back t initiator we can do that kind of stuffs, in step 3 ,I am having some problem ,what is that problem in step 3??..any problem in step 3?..you may include ID_b, lets say there is some ID_b so that it can understand it for B, but the actually authentication comes from the Key,so if they say its for key you can understand that its for the key, so you don't have to put extra effort for B, but if you want then you can put it ,instead of packet structure there could be some attacks over here, what are those attacks?..imagine that I captured this one and then I recorded and I exchanged ,I am going to replay it, and there is no random number and this guy everytime will accept this number ,it going to recover K_s send N_2 to K, all the time this time A will say whats going on ,I dint send anything, it canbe done anything and you can make responder so busy to recover this K_s all the time, in other words the responder B in packet doesn't check the origin of the message, this message can come from anywhere and we don't know whether B is receiving this message or not ,but what we know is this , there is only one key in this scenario that can recover the ticket, from this encrypted message whoever has the secret key K_a, who has the secret key K_a, ?..A..if I am receiving ticket like this then that must be A, I am receiving from, but I am not considering the replay attack,but if some one is doing like storing some where and sending it back to B, there is a possibility ,so instead of making like this we are going to put extra effort to calculate and make it one time ticket and its not valid for all the times ,so you can put some random number or expected number from A to B so on and so forth, that is the drawback for this kind of key scenario, this is the (……) protocol used in some version, but Kerberos has slight changes ,and it is based on this protocol basically ok?.