A more secure authentication dialogue

Last time we were talking about secure authentication and we are trying to know how secure the authentication is ,we saw that we are making the first message secure and we are going to transfer to the authentication protocol and then we understand we have to have something should be added to the logon session and that granting to TGT from the AS and after authentication is successful and we have to go to tgs to get SGT ,so you are going to get the ticket once and you can use it to get the lot of different SGT for differentiation service and eventually your are going to reach the server you want to access by using SGT and you receive that one from TGS, there is one thing here that we dint solve ,there were some problem in the previous and some problem now and what is the problem repay attacks for this tickets and if someone use the ticket because tickets can user the ticket and impersonating the IV and other was a big problem so we dint solve that problem , we discussed that we are not going to send this ticket like this and we are going to send like this ,that kind of authentication ,lets start and have a look at the actual Kerberos version of the protocol.

Kerberos V4 exchange messages

Lets have a look at first message and take a look at the client messages ,client sends ID_c and ID_{tgs} client 1 to access and timestamp and it sends It as a new request {question from a student}. ID_{tgs} is the name of the server and nothing else, there is nothing secret about it, and if it is a web server and I call it Mozilla then I want to access Mozilla to that network , just telling that this guy wants to get ticket from this guy who is TGS , this guy will be eventually will get SGT from TGS and then he will get the ID and that ID is ID_{tgs} , it could be multiple tgs , depends on the domains , if the domain is multiple you are going to have multiple tgs and multiple AS and based on the network structure you have, so authentication server will receive this message and you will do what?.. it has to do something before creating the ticket, it is going to go and find the password of this id and from that password it is going to generate ,remember the key that we used for the client, K_c , so it is going to generate K_c , and it is going to generate $Ticket_{TGS}$ which is TGT. Look at the format of the ticket, it will have life time and we will have limited life time and timestamp is the creation date of the ticket ,then we have AD_c and ID_c and there is one more thing that Is KCTGS and this K_{ctgs} is the session key and it is going to use between client and local TGS, selected by AS and this whole thing is going to be encrypted with the key K_{tgs} and this whole key will be sent to AS. During the installation of these Kerberos we are installing these TGS and this K_{tgs} is going to be installed manually when you install Kerberos and that is the key shared when install AS and TGS. And that ticket will be included in this message as I said this is TGT and then life time too is the same life time and the same timestamp and the same TGS and even the session key by the AS all of them will be encrypted with key generated by the user password , and this thing is going to be transmitted to c ,so client receives this one and what is the job of client ?..why it has to enter the password?. To generate K_c else it cant generate the password. , in other words when this message received client will have ask password from the user and user will enter the password and when user enters the password, Kerberos client will generate the key , K_c as AS created ,remember they use des algorithm and it is able to generate the key if client has the correct password. Then it will recover the session key and it will understand that this key is used between key and this TGS and ticket life time is this and it is generated on this time and it cannot anything b about ticket because ticket id encrypted with the key that is shared between ticket

and authentication server ,it cannot see the content of the message . in step 2 its going to ask the password, because client will receive the message in step 2 ,when client receives this message Kerberos client will ask password form the client otherwise this client cannot decipher this cannot recover the message , because this client is generated from this password. AS generated the key with algorithm that I showed you guys and that algorithm is last time and that algorithm is the function of DES and what they are doing is they are folding the password, generating 1 and we call it K_{pv} or K_{pw} that K_{pw} was the input to the des algorithm and that input was encrypted with the key of the des algorithm , and the result was going to the cbc and result was going to the second block with the des and with the same key K_{pw} and we are going to select the same block , same ciphertext block is going to be the K_c information so server will calculate that one and when client sends this message, client will calculate that one to recover the content of this message to recover the ticket and not the content of the ticket. There is still no authentication, remember authentication will come when this ticket will send to the TGS. Because who ever has correct password can recover K_c and recover that password . if you dont have the correct password you cannot recover the password and if you recover this ticket TGS will know that you are the one who has the password for that key .key is not encrypted , this Ekc is the encryption of this whole message with eky and the password key is K_c , K_{ctgs} is the key that is going to transferred between client and TGS how we are going to do that we are going to see, what we are doing is we are distributing the session key in this pattern. There is key distribution right there at the point ,now we will go to step 3, remember in the previous slide we send the ticket like this, we said that sending the ticket like this will be a problem which is a repay attack problem . and we are going to solve that problem by sending the ticket with the authenticator , ticket along with an authenticator , look at the authenticator right now, there is a time stamp now on the authenticator. Address of c with the id of c encrypted with the K_{ctgs}, and don't get confuse with the TGS, this is going to be K_{ctgs} change it and also ticket and id_v along with the ticket is transmitted to the client so lets have a look now when TGS receives the ticket ,what is this, TGT is encrypted and that key is known by the TGS and by using that key TGS will decipher this ticket and it will get the session key , and by using that session it is going to decipher the authenticator and it will check the ID from that authenticator content and if those are the same in the ticket and that means this ticket is coming from client no one can create authenticator without the knowledge of K_{ctgs} ,so whoever creates those later must know the password and from that password it will generate K_c and from that it will be able to recover the K_{ctgs} so then that must be the authentication part, now TGS is saying that this client is saying that this ticket is authenticated by authentication and it is valid from that client and based on the requirement it says it wants ID_v the remote server that client wants to access and based on that one tgs is going to generate SGT that is Ticket $_v$, and it will put timestamp of the message to show it's the new message and another key that is session key and this key is going to be used between client and destination server, so its sending that key to client ,but look at the ticket content ,K_{cv} , so what TGS is doing is it sending these to the client and the destination to client and destination server. {question from a student}. $Ticket_{tgs}$ is being decrypted here, client cannot decrypt it because it doesn't have K_v information , it is the functionality of that function for that ticket is concerned , but during that it is going to prepare the authenticator , because if he doesn't know the source of the ticket there could be replay attack and authenticator prepare the key session key shared between client and TGS and we are going to apply same logic for client and server and that's why TGS is preparing $ticket_{tgs}$ with the lifetime so SGT will have lifetime ,timestamp showing that ticket is fresh and AD_C and

ID_c showing that this client is accessing this guy and session key and kcv , and all of them is encrypted with the K_c tgs and that K_v is installed during the installation between TGS and the server, and when client receives this packet by using the K_{ctgs} received from the ASit will be able to decipher this packet and tis going to recover K_{cv} and $ticket_v$ and by using this K_{cv} we are going to prepare the authenticator and it is encrypted with the K_{cv} and it will forward this ticket to destination server.