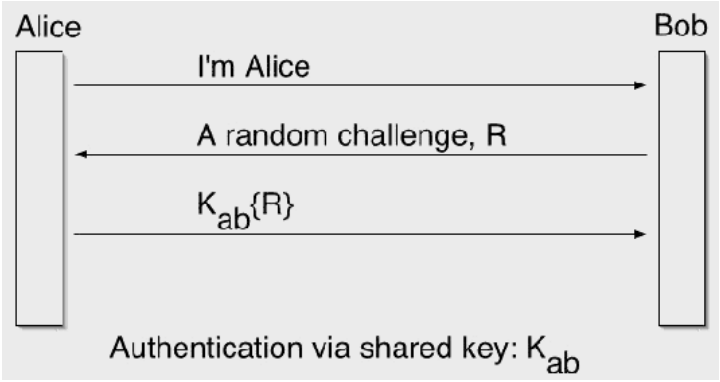
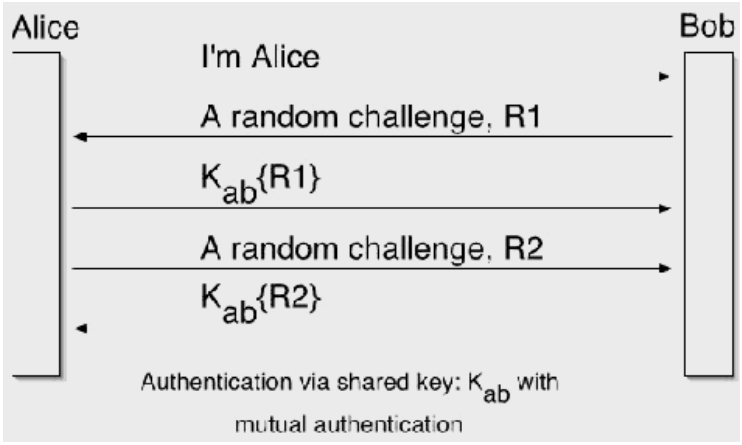
Protocole d'authentification simple

La figure suivante illustre un protocole d'authentification simple :



Le protocole suivant illustre une variante d'un protocole simple d'authentification mutuelle :



Attention : Problèmes avec ce schéma d'authentification

Ne s'adapte pas au facteur d'échelle: Avec n clients et m services, il va falloir distribuer a priori n*m clés symétriques

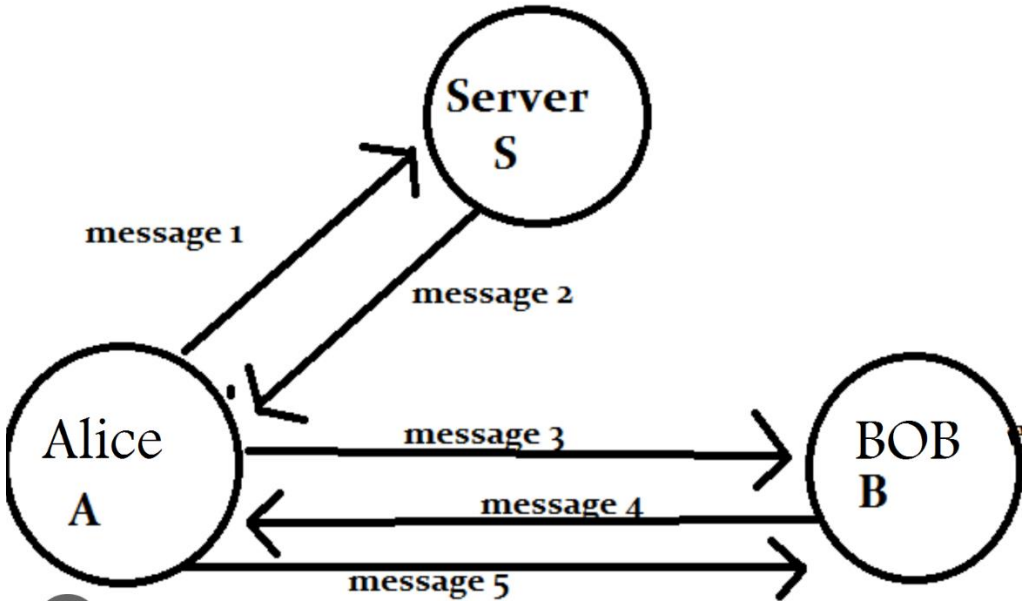
Amélioration possible: Partager une clé entre les clients et une tierce partie de confiance SERVEUR DE CONFIANCE

Définition du protocole Needham-Schroeder:

Le protocole Needham-Schroeder est un protocole cryptographique utilisé pour l'établissement de clés de session sécurisées dans les systèmes de communication. Il a été développé par Roger Needham et Michael Schroeder. Le protocole existe en deux versions : une version symétrique et une version à clé publique.

A-Version symétrique du protocole Needham-Schroeder:

La version symétrique du protocole est utilisée dans les systèmes où les deux parties partagent une clé secrète avec un serveur de confiance (trusted third party (TTP)). Voici les étapes:



M1:A->S:(A,B,R*a*)
M2:S->A:{R*a*,B,K*ab*,{K*ab*,A}K*b.s*}K*a.s*
M3:A->B:{K*ab*,A}K*b.s*
M4:B->A:{R*b*}K*ab*
M5:A->B:{R*b*+*1*}K*ab*

1-Alice envoie une requête au serveur, demandant à communiquer avec Bob. Cette requête comprend un nonce **Ra** (un nombre aléatoire utilisé une seule fois) généré par Alice.

2-Le serveur génère une nouvelle clé de session pour Alice et Bob. Il envoie ensuite à Alice le nonce qu'elle a généré, la clé de session et le nom de Bob, le tout chiffré avec la clé partagée entre Alice et le serveur.

3-Alice déchiffre le message du serveur, récupère la clé de session et vérifie le nonce pour s'assurer que la réponse est bien pour sa demande. Elle envoie ensuite à Bob la clé de session et son nom, le tout chiffré avec la clé partagée entre Bob et le serveur.

4-Bob déchiffre le message d'Alice, récupère la clé de session et envoie à Alice un nonce **Rb** généré par lui, chiffré avec la nouvelle clé de session.

5-Alice reçoit le nonce de Bob, l'incrémente de un et le renvoie à Bob, chiffré avec la clé de session.

6-Bob vérifie que le nonce est bien celui qu'il a envoyé, incrémenté de un. Si c'est le cas, l'authentification est réussie et la session sécurisée peut commencer.

B-Version à clé publique du protocole Needham-Schroeder:

M1:A->S:(A,B)
M2:S->A:{B,PK$_b$}SK$_s$
M3:A->B:{Ra,A}PK$_b$
M4:B->S:(B,A)
M5:S->B:{A,PK$_a$}SK$_s$
M6:B->A:{Ra,Rb}PK$_a$
M7:A->B:{Rb}PK$_b$

1-Alice demande la clé publique de Bob au serveur, cette requête comprend (l'identité de A et B).

2-Le serveur répond à Alice avec la clé publique de Bob en même temps que son identité, le tout étant signé par la clé privée du serveur.

3-Alice choisit un nonce **Ra**(un nombre aléatoire utilisé une seule fois), le chiffre avec la clé publique de Bob avant de lui envoyer.

4-Bob demande la clé publique d'Alice au serveur, cette requête comprend (l'identité de A et B).

5-Le serveur répond à Bob avec la clé publique d'Alice en même temps que son identité, le tout étant signé par la clé privée du serveur.

6-Bob choisit un nonce **Rb**(un nombre aléatoire utilisé une seule fois), le chiffre avec la clé publique de Alice en même temps que le nonce que celle-ci a généré afin de prouver qu'il a pu déchiffrer le message envoyé à l'étape 3.

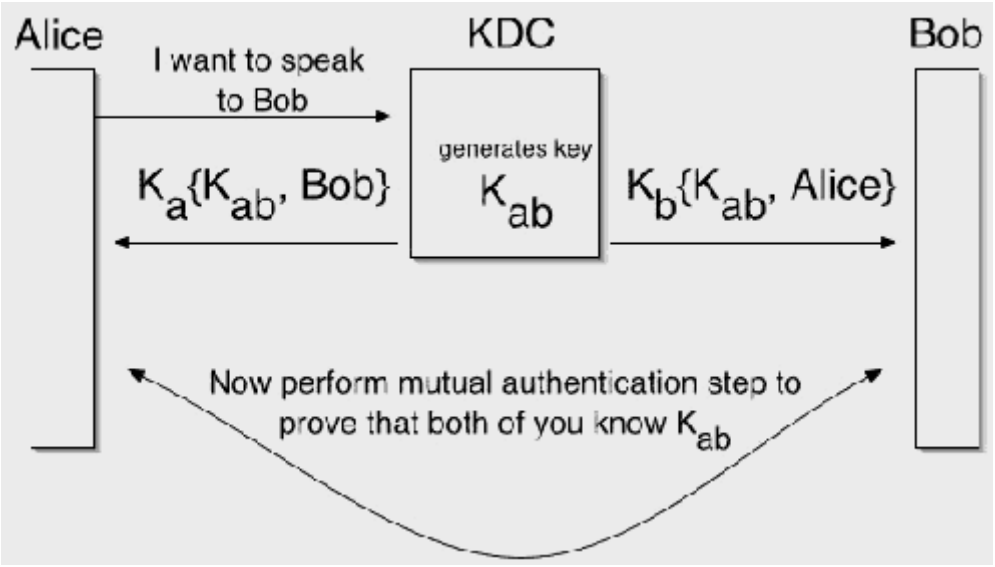
7-Alice envoie le nonce de Bob à Bob pour prouver qu'elle a pu déchiffrer le message envoyé à l'étape précédente.

KERBEROS SIMPLE

Authentification basée sur une tierce partie de confiance

Partager une clé entre chaque service et client avec une tierce partie de confiance

La tierce partie de confiance joue le rôle d'intermédiaire dans le processus d'authentification ; le KDC: Key Distribution Center. Chaque client et serveur partage une clé secrète avec le KDC. Le KDC génère une clé de session et la distribue aux parties communicantes confidentiellement. Les parties communicantes prouvent qu'elles connaissent la clé de session. Ce qui donne le schamé d'authentification suivant :

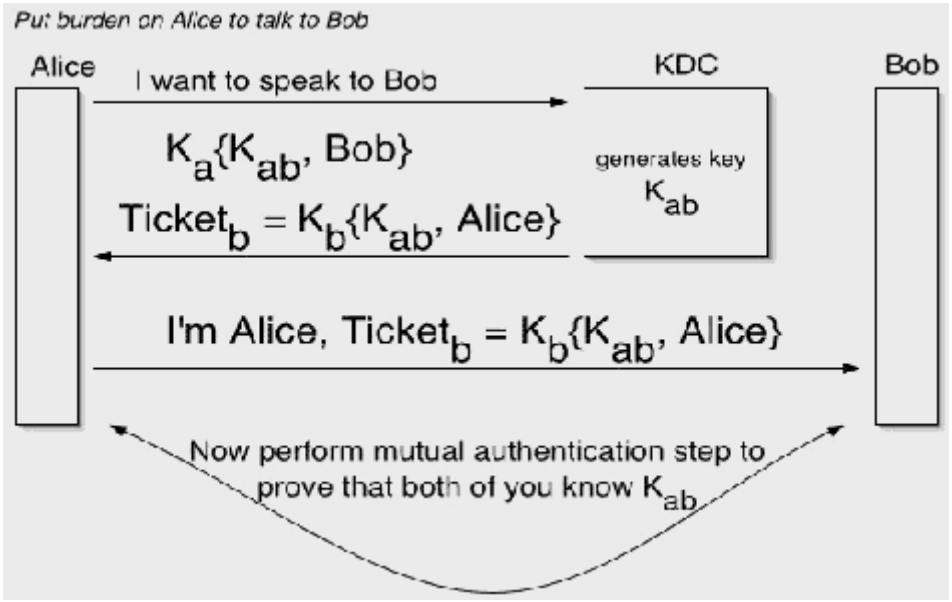


1-Alice envoie une requête au serveur KDC, demandant à communiquer avec Bob

2-Le serveur KDC produit une nouvelle clé de session pour Alice et Bob. Il transmet à Alice cette clé et l'identification de Bob, l'ensemble étant crypté avec la clé commune entre Alice et le serveur KDB. Simultanément, il envoie à Bob la clé et l'identité d'Alice, tout cela étant crypté avec la clé partagée entre Bob et le serveur KDB.

3- Maintenant, exécutez l'étape d'authentification mutuelle pour prouver que tous les deux connaissent la clé Kab.

B-avec ticket



1-Alice envoie une requête au serveur KDC, demandant à communiquer avec Bob

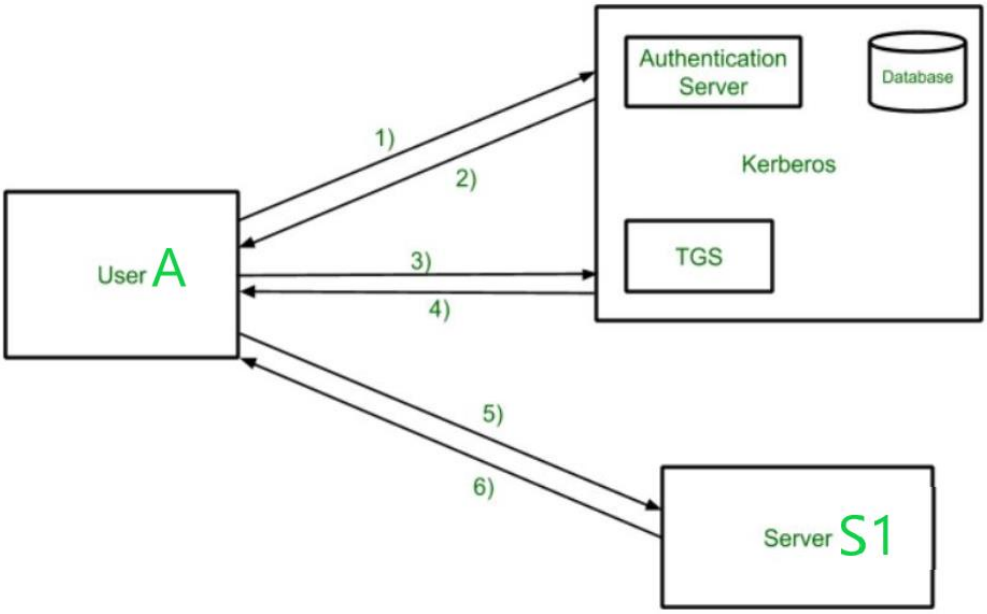
2-Le serveur KDC produit une nouvelle clé de session pour Alice et Bob. Il transmet à Alice cette clé et l'identification de Bob, l'ensemble étant crypté avec la clé commune entre Alice et le serveur KDB , il envoie aussi un Ticketb comprend (la clé et l'identité d'Alice) crypté avec la clé partagée entre Bob et le serveur KDB.

3- en troisième étape Alice envoie leur identité et le Ticketb à Bob

4-- Maintenant, exécutez l'étape d'authentification mutuelle pour prouver que tous les deux connaissent la clé Kab

Kerberos

Est un protocole d'authentification réseau qui fonctionne sur la base de "billets" pour fournir une authentification forte en environnement non sûr. Il repose sur la cryptographie à clé secrète et a été développé par le Massachusetts Institute of Technology (MIT).



Ticket Granting Ticket (TGT)

Un ticket pour accéder au TGS afin d'obtenir des tickets de service

Ticket Granting Service (TGS)

Permet à des utilisateurs d'obtenir des tickets pour des services.

Étapes du protocole Kerberos

1-Authentification initiale

M1:A->AS:(A,TGS,T*a*)

L'utilisateur A envoie une requête non cryptée au service d'authentification (AS) pour obtenir un "Ticket" pour le service de distribution de tickets (TGS). Cette requête contient l'identifiant de l'utilisateur A et le nom du service TGS et timestamp Ta.

2-Réponse du Service d'Authentification

M2:AS->A:{K*a.tgs* ,TGS,T*as* ,LifetimeTicket*tgs* , Ticket*tgs*}Ka
Et Ticket*tgs*={K*a.tgs* ,A , ADDRS*a* , TGS,T*as* ,LifetimeTicket*tgs*}K*as.tgs*

L'AS vérifie si l'utilisateur est bien dans la base de données. Si c'est le cas, il crée le message :

* message contenant la clé de session client-TGS **K*a.tgs*** , l'identifiant du **TGS,** timestamp **Tas**, la validité du Ticket **LifetimeTicket*tgs* ,** un **Tickettgs** , Ce message est chiffré avec la clé Ka.as.

* Un Tickettgs (Ticket Granting Ticket) contenant l'identifiant du client **A**, Adresse de A **ADDRS*a***,la clé de session client-TGS **K*a.tgs*** et la validité du Ticket **LifetimeTicket*tgs*** ., l'identifiant du **TGS** , timestamp **Tas**, la validité du Ticket **LifetimeTicket*tgs*** ,Ce message est chiffré avec la clé Kas.tgs.

Ces deux messages sont envoyés à l'utilisateur A.

3-Demande de service

M3:A->TGS:(S1,Ticket*tgs*,Authenticator*a.tgs*)
Et Authenticator*a.tgs*={A , ADDRS*a* ,T'*a*}K*a.tgs*

Lorsque l'utilisateur veut accéder à un service, il envoie une demande au TGS avec le TGT obtenu précédemment et un Authentificateur (qui contient l'identifiant et adresse du client et un timestamp **T'a**) chiffré avec la clé de session client-TGS K*a.tgs*.

4-Réponse du Service de Distribution de Tickets

M4:TGS->A:{K*a.s1*,S1,T*tgs*,Tickets*1*}K*a.tgs*
Et Tickets*1* ={K*a.s1*,A, ADDRS*a*,S1,T*tgs*,LifetimeTickets*1*}K*tgs.s1*

Le TGS déchiffre le TGT avec sa clé secrète pour obtenir la clé de session client-TGS, qui est utilisée pour déchiffrer l'Authentificateur. Si les informations correspondent, le TGS crée un Ticket pour le service demandé S1(contenant l'identifiant du client, la clé de session client-serveur S1 et la validité du Ticket) chiffré avec la clé Ktgs.s1, et un message contenant la clé de session client-serveur Ka.s1 chiffré avec la clé de session client-TGS K*a.tgs*. Ces deux messages sont envoyés à l'utilisateur A.

5-Demande de service au serveur

M5:A->S1:(Tickets*1 ,* Authenticator*a.s1*)
Et Authenticator*a.s1* ={A, ADDRS*a*,S1,T"*a*}K*a.s1*

L'utilisateur déchiffre le deuxième message avec la clé de session client-TGS pour obtenir la clé de session client-serveur. Il envoie alors une requête au serveur avec le Ticket de service et un nouvel Authentificateur (qui contient l'identifiant du client et un timestamp) chiffré avec la clé de session client-serveur Ka.s1.

6-Validation du service

M6:S1->A:{T"*a+1*}K*a,s1*

Le serveur déchiffre le billet avec sa clé secrète pour obtenir la clé de session client-serveur, qui est utilisée pour déchiffrer l'Authentificateur. Si les informations correspondent et que le timestamp est valide, le serveur peut considérer que l'utilisateur est authentifié. Le serveur peut alors répondre à l'utilisateur en renvoyant le timestamp de l'Authentificateur incrémenté de 1, chiffré avec la clé de session client-serveur. Ceci prouve que le serveur a bien reçu l'Authentificateur et a déchiffré correctement le billet.

Accès au service

Une fois que l'utilisateur a reçu la confirmation du serveur, il peut accéder au service demandé.

En résumé, le protocole Kerberos utilise une approche basée sur des "billets" pour assurer une authentification sécurisée dans un environnement réseau. L'aspect clé de ce protocole est qu'il évite de transmettre des mots de passe en clair sur le réseau, ce qui augmente considérablement la sécurité de l'authentification.