M1 - Outils de l'Internet (Année 2010/2011)

Dog Day Afternoon

Nous allons ici nous intéresser au protocole d'authentification *Kerberos*. Ce protocole a été développé au *Massachussetts Institute of Technology* (MIT) et permet à des utilisateurs de s'authentifier sur un réseau sans avoir à partager de mots de passe avec chaque autre membre du réseau et sans utiliser de système de clés privées et clés publiques.

1. Sachant que *Kerberos* a été développé pour protéger les communications d'un projet appelé *Athena*, justifiez le nom du protocole.

Le protocole fonctionne en plusieurs temps. Très brièvement, un utilisateur demande au serveur d'authentification une session en s'identifiant à l'aide d'un secret partagé avec le serveur au préalable. Puis lorsqu'il a une session d'authentification, il demande des tickets à un serveur de tickets. Enfin, il donne ces tickets aux différents serveurs de services qu'il veut utiliser sur le réseau.
À chaque utilisateur correspond une clé secrète. Celle-ci est obtenue en demandant un mot de passe à l'utilisateur puis en effectuant un hachage [1] de celui-ci. La clé est transmise aux serveurs d'authentification et de tickets une seule fois au moment où l'utilisateur est enregistré sur le réseau (en général les deux serveurs sont sur la même machine).

2. Pourquoi effectue-t-on un hachage du mot de passe ? (il y a plusieurs raisons)

L'identification de l'utilisateur auprès du *serveur d'authentification* se fait comme suit :

a. Le client envoie un message en clair au serveur demandant une session annonçant son identité.
b. Le serveur vérifie que l'utilisateur est bien dans sa base de données et si c'est le cas il lui envoie deux messages :
   - **Message A** : Un code de session encodé à l'aide de la clé secrète du client,
   - **Message B** : [nom du client, adresse réseau du client, durée de validité de la session, code de session] encodé à l'aide d'un clé secrète du serveur de tickets ;
c. Le client décrypte le message A et obtient ainsi un code de session qui lui permettra par la suite de communiquer avec le serveur de tickets. Le message B n'est pas compréhensible par l'utilisateur.

3. Comment le client a-t-il la confirmation qu'il s'adresse bien au serveur d'authentification et non pas à un usurpateur ?

Par la suite, lorsque le client veut accéder à un service, il doit demander un ticket à un *serveur de tickets* :

a. Il envoie alors deux messages :
   - **Message C** : [message B, nom du service demandé],
   - **Message D** : Le nom du client encodé à l'aide du code de session ;
b. Lorsqu'il reçoit les messages C et D, le serveur extrait alors le message B, et le décode à l'aide de sa clé. Il obtient ainsi le code de session et peut décrypter le message D.
c. Le serveur répond alors au client en lui envoyant les deux messages suivants :

1. Ici, une fonction de hachage est une fonction qui à une chaîne de caractères s associe une autre chaîne de caractères $h(s)$ telle qu'il est algorithmiquement très difficile d'obtenir des informations sur s connaissant $h(s)$.

- **Message E** : [nom du client, adresse réseau du client, durée de validité de la session, code de ticket] encodé à l'aide de la clé du service demandé,
- **Message F** : Le code de ticket (le même que dans le message précédent) encodé à l'aide du code de session.

4. Pourquoi le serveur de tickets sait-il que le client est bien celui qu'il prétend être ?

5. Comment le client sait-il qu'il est bien en train de communiquer avec le serveur de tickets ?

Enfin, le client peut s'authentifier auprès du serveur du service qu'il désire joindre :

a. Le client envoie les messages suivants au serveur du service :
   - Le message E précédent,
   - **Message G** : [nom du client, heure] encodé à l'aide du code de ticket des messages E et F ;

b. Le serveur du service décode alors le message E à l'aide de sa clé secrète et obtient ainsi le code de ticket ;

c. À l'aide de la clé, il décode le message G et envoie au client la réponse suivante :
   - **Message H** : L'heure trouvée dans le message G incrémentée de 1 encodée à l'aide du code de ticket ;

d. Le client décode le message reçu et vérifie que l'heure y est correctement incrémentée ;

e. Le client demande le service au serveur du service ;

f. Le service est fourni par le serveur.

6. Comment l'utilisateur sait-il que la machine à laquelle il s'adresse dans la dernière étape est bien le serveur du service qu'il voulait joindre ?

7. Comment le service contacté sait que l'utilisateur est bien celui qu'il prétend être ? (ce qui est, rappelons-le, ce que l'on essaie de faire depuis le début)

8. Représentez la totalité des communications sur un schéma simplifié (cette question sert principalement à vous forcer à représenter la totalité du processus afin de mieux le voir dans son ensemble).

9. En donnant quelques caractéristiques des modèles *client-serveur* et *peer-to-peer* dites si *Kerberos* est plutôt l'un ou l'autre. En tant que tel, expliquez comment une défaillance matérielle localisée peut bloquer la totalité du réseau.

10. Si le client et le serveur de service veulent crypter leurs échanges après l'authentification, comment peuvent-ils faire ?

11. Pourquoi utilise-t-on un serveur de tickets ? (pourquoi le serveur d'authentification ne donne-t-il pas directement le ticket lorsque le client s'authentifie ?)

12. Pourquoi donne-t-on une durée de validité limitée aux sessions d'authentification ? (la durée par défaut d'une session est de quelques heures)

13. Si un utilisateur extérieur parvient à obtenir le code de la session, qu'il est capable d'intercepter toutes les communications et d'envoyer des messages à tout le monde, que peut-il faire ?

14. Quels avantages et inconvénients voyez-vous à *Kerberos*, comment se compare-t-il aux autres solutions d'authentification ?