Présentation

Kerberos est un protocole d’authentification AAA issu du projet « Athena » du MIT (Massachusetts Institute of Technology). Pour plus d’information sur le principe AAA, je vous invite à lire l’article « [AAA Authentication Authorization Accounting](http://www.devensys.com/blog/aaa-authentication-authorization-accounting) ». Son nom « Kerberos » vient de la mythologie grecque et correspond au nom du chien à trois têtes gardien des portes d’Hadès. Il porte relativement bien son nom puisqu’il est chargé d’authentifier, d’autoriser et de surveiller les utilisateurs voulant accéder aux ressources et services de votre réseau. Il agit en chien de garde contre les intrus sur vos services réseau. Le protocole Kerberos a été normalisé dans sa version 5 par l’IETF dans les RFC 1510 (en septembre 1993) et RFC 1964 (juin 1996).

C’est un standard qui résout de nombreux problèmes de sécurité, d’administration, et de productivité dans l’authentification des clients et des services au sein d’un réseau. En effet, kerberos introduit le principe de Single Sign-On (SSO). Ainsi avec une authentification unique, l’utilisateur aura accès à tous les services du réseau. Pour plus d’information sur le SSO je vous invite à lire l’article « [SSO Késako ?](http://www.devensys.com/blog/ssokesako) ». Pour couronner le tout, il a l’avantage d’être compatible aussi bien avec les systèmes libres que les systèmes propriétaires.

Tout bon administrateur réseau doit maintenant se demander si c’est sécurisé. Car confier une authentification SSO à un protocole faillible c’est signer l’arrêt de mort de son infrastructure. Au lieu de me croire sur parole, je vous invite à comprendre son fonctionnement et à juger par vous-même.

Kerberos se base sur une tierce partie de confiance pour gérer l’authentification, le **KDC** (Key Distribution Center). Tous les utilisateurs et services du réseau font confiance à cette tierce partie. Pour réaliser la sécurité de l’authentification, Kerberos utilise un mécanisme de chiffrement basé sur des algorithmes à clef symétrique. Chaque sujet et service du réseau a une clé secrète partagée avec le **KDC**.

Kerberos utilise un système de **ticket** pour réaliser l’authentification et introduit le principe de SSO (Single Sign On). L’utilisateur s’authentifie sur le **KDC** puis utilise un **ticket** pour s’authentifier sur chaque service demandé. L’utilisateur ne transmet jamais son mot de passe au service.

Le protocole Kerberos sépare le rôle du **KDC** en deux services :

* ***AS (Authentication Service)***: Il s’agit du service sur lequel l’utilisateur s’authentifie. Il délivre un TGT (Ticket Granting Ticket) en cas d’authentification réussie. Ce ticket est en fait une demande d’accès au TGS.
* ***TGS (Ticket Granting Service)***: Ce servir fourni les tickets d’accès aux différents services du réseau. On les appelle TS (Ticket Service).

Le principe de fonctionnement général est simple. L’utilisateur s’authentifie auprès du service d’authentification (le vigile). Celui-ci lui donne un ticket d’accès prouvant qu’il s’est authentifié (un badge). L’utilisateur se rend auprès du service de gestion des tickets. Il demande un accès au service de fichiers par exemple. Grâce au badge donné par le vigile le service de ticket sait que l’utilisateur s’est bien authentifié. Il va donc lui remettre un ticket d’accès au service de fichier. L’utilisateur peut maintenant aller voir le service de fichier avec son ticket. Le service lui donnera accès aux ressources correspondantes à ses autorisations.

Fonctionnement détaillé de Kerberos

Maintenant, détaillons ce processus afin de bien comprendre les mécanismes de sécurité de Kerberos.

*Note : Le chiffrement sera noté comme suit Kclé {partie chiffrée par la clé}*

Notation :

* ***AS***: service d’authentification.
* ***TGS***: Service délivrant des tickets d’accès aux services.
* ***TGT***: Ticket d’accès au TGS.
* ***TS***: Ticket d’accès au service demandé.
* ***Kutilisateur***: Clé secrète de l’utilisateur, connu de l’utilisateur et du TGS.
* ***Kservice***: Clé secrète du service demandé, connu du service et du TGS.
* ***Ktgs***: Clé secrète du TGS connu du TGS et de l’AS.
* ***KsessionTGS***: Clé de session entre l’utilisateur et le TGS.
* ***Ksession***: clé secrète de session entre l’utilisateur et le service demandé.
* ***TicketService***: Ticket d’accès au service demandé.

Fonctionnement :



[1]       L’utilisateur s’authentifie en entrant un couple login/password (généralement à l’ouverture de session sur la machine). Le client kerberos de la machine dérive à l’aide d’un algorithme de hash le mot de passe de l’utilisateur en clé secrète **Kutilisateur**. Le client kerberos va maintenant contacter l’**AS** pour authentifier l’utilisateur. Il envoie pour cela à l’AS l’identité de l’utilisateur (en clair) ainsi que **Kutilisateur** {message authentifiant définie dans le protocole kerberos, timestamp}. L’utilisateur n’aura pas à renouveler son authentification durant plusieurs heures (variable selon les réglages).

*Remarque : Le mot de passe n’est pas transmis à travers le réseau. Il sert uniquement à générer Kutilisateur. De même, Kutilisateur servira à chiffrer une partie du message et ne sera jamais transmise à travers le réseau. Il n’y a donc pas de possibilité d’interception. Seule une compromission de la machine cliente ou un brute force peut permettre à une personne malveillante de récupérer ces informations.*

**

[2]       L’**AS** lit l’identité de l’utilisateur (envoyé en clair). L’AS connait le mot de passe de l’utilisateur, car c’est lui qui gère la base de données utilisateur. Il peut donc obtenir **Kutilisateur**. Grâce à **Kutilisateur** il va déchiffrer le message authentifiant envoyé par l’utilisateur. Si une fois celui-ci déchiffré il correspond, c’est qu’il a bien été chiffré avec **Kutilisateur**. L’**AS** vérifie ensuite le timestamp pour éviter tout rejeu de l’authentification.

L’**AS** retourne maintenant au client **Kutilisateur** {**TGT, KsessionTGS**}. C’est-à-dire un Ticket **TGT** et une clé de session entre l’utilisateur et le TGS, le tout chiffré par la clé secrète de l’utilisateur pour que personne n’ait accès à ces informations.

*Remarque :* ***Kutilisateur*** *n’est connu que du serveur et du client et n’a jamais été communiqué à travers le réseau. Cette communication ne peut donc être lue par aucune tierce partie malveillante.*



[3]       L’utilisateur déchiffre la réponse de l’**AS**. Il récupère le **TGT** et la clé de session pour communiquer avec le **TGS**. Le **TGT** est un ticket contenant trois éléments chiffrés avec **Ktgs** : une date d’expiration, le nom de l’utilisateur et **KsessionTGS**. Donc **TGT** = **Ktgs** {date d’expiration, nom d’utilisateur, **KsessionTGS**}. L’utilisateur envoie au **TGS** : le **TGT**, un **KsessionTGS** {timestamp} (permet d’assurer l’unicité de la requête, antirejeu) ainsi que le service auquel il souhaite accéder. Cet envoi correspond à une demande d’accès à un service donné.

*Remarque : le* ***TGT*** *n’est pas lisible par l’utilisateur puisqu’il est chiffré avec la clé secrète* ***Ktgs*** *connue uniquement de l’****AS*** *et du* ***TGS****.*



[4]       Grâce au **TGT**, le **TGS** connait l’identité du sujet et la clé de session à utiliser pour chiffrer la communication avec l’utilisateur. Il vérifie grâce à **KsessionTGS** la valeur du timestamp (permet d’éviter le rejeu et de signer en quelque sorte la requête de l’utilisateur). En fonction des droits du sujet, il va accepter ou refuser la demande d’accès au service. Dans le cas où le sujet possède les droits d’accès au service, le **TGS** va lui envoyer **KsessionTGS** {timestamp, **Ksession**, service à joindre, **TicketService**}. Le **TicketService** est un ticket à transmettre au service pour assurer la connexion avec celui-ci. **TicketService** = **Kservice** {**Ksession**, identité sujet, date d’expiration}.

*Remarque :* ***KsessionTGS*** *a été transmise au* ***TGS*** *via le* ***TGT****, donc de manière sécurisée. Ainsi, seuls le* ***TGS****, l’utilisateur et l’****AS*** *(fournis* ***Ksession****) connaissent la clé. Cette communication ne pourra toujours pas être lue par une personne externe au procédé.*



[5]       Le sujet déchiffre la réponse du **TGS**. Il transmet ensuite **TicketService** au service pour initialiser la connexion. **TicketService** = **Kservice** {**Ksession**, identité sujet, date d’expiration}.

*Remarque :* ***TicketService*** *est chiffré avec* ***Kservice*** *une clé secrète connue uniquement du service et du* ***TGS****. L’utilisateur ne peut donc pas lire ou modifier son contenue.*



[6]       Le service déchiffre **TicketService** et récupère **Ksession**. Il renvoie **Ksession** {timestamp}.



[7]      Le sujet déchiffre le timestamp, vérifie que le temps écoulé entre l’envoi et la réception de la réponse n’est pas trop long. Ksession {timestamp}.

*Remarque : L’utilisation de timestamp permet de s’assurer que les parties connaissent la clé et d’éviter qu’une personne mal intentionnée rejoue une séquence d’accès intercepté.*



Les inconvénients de Kerberos

* Tous les services du réseau doivent être « Kerberisé », c’est-à-dire compatible avec le protocole Kerberos. Les services doivent être capables de comprendre le système de ticket, sinon aucune authentification ne sera possible.
* Il faut que toutes les machines du réseau soient synchronisées, sinon l’utilisation des timestamp et la durée de vie des tickets risquent d’être faussées et plus aucune authentification ne sera possible.
* Kerberos introduit un SPOF (Single Point Of Failure) dans le réseau. Si le serveur Kerberos tombe, il n’y aura plus aucun accès aux différents services du réseau. La machine serveur de Kerberos doit être parfaitement sure.
* Si l’AS de kerberos est compromis, un attaquant pourra accéder à tous les services avec un unique login.
* Kerberos chiffre uniquement la phase d’authentification, il ne chiffre pas les données qui seront transmises lors de la session.