Kerberos

Cross-platform authentication and single sign-on

Version 1.0 du 23 janvier 2004

Exposés de nouvelles technologies des réseaux

Septembre 2003 – Février 2004

Ecole d'ingénieur en Informatique et Réseaux de l'université de Marne-la-Vallée

Clément DEBON

Julien VICTOR

TABLE DES MATIÈRES

I. INTRODUCTION

aKuÁ åØÜ¤Üm6uNt¡¤ƒŠ>°zbÖ.gVÜ¼Ò
zêúgQ{ sè¼[œM¾´Žž7%Î¨,):€ÑwÅòO,
D*<¸ ½Òë•^ß

Le début de ce document est crypté. En effet, pourquoi déciderions-nous de laisser ces informations à la vue de tous ? Comment savoir dans quel but vous voulez y avoir accès ? Et puis, tout d'abord... qui êtes-vous ?

Voici autant de questions que l'on peut se poser dès le début d'une conversation entre deux personnes. Autant de questions qui peuvent survenir autant dans la vie de tous les jours que dans le monde virtuel.

Si, lors d'une conversation banale, la présence d'oreilles indiscrêtes (où qu'elles se trouvent) est tolérable, il n'en va pas de même dans le domaine de l'informatique. Des données circulent en permanence, représentant autant de données confidentielles que de données personnelles.

Lorsque l'on consulte des données confidentielles, comme l'état de notre compte bancaire via Internet, il est évident que l'on ne veut communiquer qu'avec notre banque. Et pour être certain qu'il s'agisse bien de notre banque, il va nous falloir les outils adéquats pour la reconnaître.

II. POURQUOI UTILISER UN SERVICE D'AUTHENTIFICATION ?

Tout d'abord, il faut se souvenir que dans des temps très reculés à l'échelle de l'informatique, dans les années 70, les terminaux étaient reliés au serveur par des liens spécialisés. Pour s'infiltrer, un cracker devait donc obligatoirement se brancher physiquement sur ces liens.

Lorsque les réseaux ont commencé à utiliser un modèle *client-serveur* et que les terminaux ont été remplacés par les PC, les administrateurs ne pouvaient plus avoir confiance dans les utilisateurs finaux. En effet, ceux-ci peuvent désormais modifier un logiciel ou écouter le réseau. Il a donc fallu mettre en place un système permettant de rétablir cette confiance sur le réseau.

Aujourd'hui, alors que nous consultons tous les jours nos e-mails, ou que nous échangeons des données que nous souhaiterions confidentielles, les mots de passe et les données circulent la plupart du temps « en clair » entre notre poste et le serveur ou le destinataire. Cela signifie que quiconque surveillant nos données pourra lire nos conversations, nos mots de passe et donc nos données.

La solution proposée est la mise en place d'un système d'authentification, permettant d'assurer que deux interlocuteurs se connaissent et savent qui est l'autre. Comme les communications peuvent, en principe, être vues par n'importe qui, il a été proposé de les sécuriser, afin que seules les personnes concernées puissent consulter ces informations confidentielles.

Kerberos est l'un des protocoles d'authentification disponibles. Il a été créé par le MIT pour solutionner ces différents problèmes de sécurité des réseaux.

III. LE PROTOCOLE NEEDHAM – SCHROEDER, BASE DE KERBEROS

1 Présentation du protocole Needham - Schroeder

Deux chercheurs du Xerox Palo Alto Center, Roger Needham et Michael Schroeder, ont défini, vers la fin des années 70, une plateforme sécurisée permettant d'authentifier les utilisateurs. Ils ont mis en place deux protocoles, dont l'un utilisant des clés privées de cryptage, et qui est à la base de Kerberos.

Le système défini est supposé vivre dans un environnement hostile, où n'importe qui est capable d'écouter les paquets sur le réseau et de les modifier. Cependant, ils ont pris comme hypothèse que l'utilisateur choisira un mot de passe respectant certains critères (en particulier, en évitant un mot de passe trop court ou un mot trop commun). Ainsi, ce protocole ne protège en rien contre une attaque de type « *force brute* » *(brut force)* ou utilisant un dictionnaire de mots.

Le protocole définit trois participants dans le réseau :

- un client ;
- un serveur proposant le service que l'utilisateur veut utiliser ;
- un serveur d'authentification.

Le client représente la machine effectuant la requête d'authentification : en général, il s'agit d'un PC. Le serveur est un serveur applicatif, comme par exemple un serveur de messagerie ou bien un serveur de fichiers. Enfin, le serveur d'authentification est un serveur dédié, détenant l'ensemble des clés de cryptage des clients et des serveurs du réseau.

Il ne s'agit pas ici d'authentifier l'utilisateur avec le serveur d'authentification en utilisant un simple mot de passe. Le protocole Needham- Schroeder fournit un mécanisme permettant de distribuer une clé de cryptage au client et au service, celle- ci disposant d'une validité limitée dans le temps. L'authentification des deux parties se fait alors au travers de cet échange.

2 Déroulement du protocole

Premièrement, le client contacte le serveur d'authentification. Il envoie un message contenant son identité ainsi que celle du serveur d'application qu'il tente de contacter. Il y joint également une valeur aléatoire.

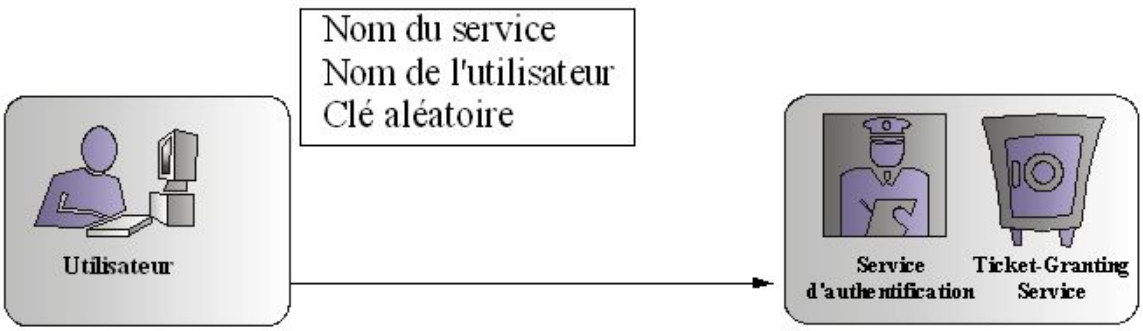


Figure 3.1 : Premier échange

Le serveur d'authentification reçoit ces informations, et recherche les clés privées de cryptage correspondant au service et à l'utilisateur. Il adjoint une troisième clé à utilisation unique, générée aléatoirement, appelée « clé de session » et utilisée pour sécuriser les communications entre le serveur d'application et le client.

Dans ce protocole, le serveur d'authentification ne communique jamais directement avec le serveur d'application. Il retourne au client un message contenant la clé de session ainsi que l'identité vérifiée des deux parties.

Comment ce message peut-il être protégé d'une personne écoutant sur le réseau ? Et avant cela, comment le serveur d'authentification peut-il être certain de l'identité du client lorsqu'il le contacte ?

La réponse est apportée par plusieurs couches de cryptage. Tout d'abord, le message est construit de manière à ce que seul le serveur d'authentification puisse le lire. Il contient le nom du client ainsi qu'une clé de session. Afin d'éviter la possibilité d'être capté sur le réseau, il est crypté à l'aide de la clé du serveur d'application. Or, seuls le serveur de service et le serveur d'authentification connaissent cette clé. Dans la terminologie de Kerberos, on parlera de « ticket » pour un tel message.

Le message de retour destiné au client contient le message d'origine, le nom du serveur d'application, une copie de la clé de session ainsi que la clé aléatoire du client. Ce message est crypté à l'aide de la clé privée du client. Le serveur d'authentification ne peut pas déterminer si le client est bien qui il prétend être. Il retournera ce message à quiconque le lui demandera, pour autant que l'utilisateur fasse bien partie de sa base d'informations. Cependant, comme le message est crypté à l'aide de la clé du client, seul celui- ci pourra décrypter ce message.

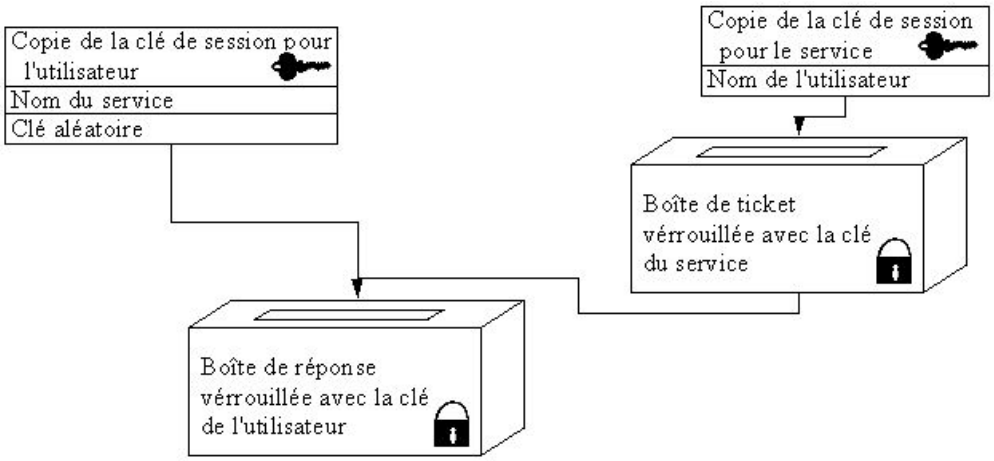


Figure 3.2 : Construction de la réponse pour le client

Le client reçoit donc le message, et doit alors entrer la clé (le mot de passe) permettant de le décrypter. S'il ne donne pas le bon, l'authentification aura échoué.

En cas de succès, le client aura alors à charge de faire suivre la clé de session au serveur d'application. Il pourra alors lui faire parvenir le « ticket », contenant une copie de la clé de session et le nom du client, le tout crypté à l'aide de la clé du serveur d'application. Seul le serveur de services est alors capable de décrypter le message, et donc de récupérer la clé de session.

Figure 3.3 : Envoi du ticket de service

A partir de là, la communication entre les deux parties ne peut être interpretée que par elles. Le client sait qu'il s'adresse au bon serveur, puisque lui seul était en mesure de décrypter le message contenant la clé de session. De son côté, le serveur d'application sait que le client est bien qui il prétend être, puisqu'il a été capable de décrypter le message contenant la clé de session et l'identité du client qui lui a été ensuite transféré.

Cependant, il reste une attaque à laquelle faire face. Le réseau étant considéré comme non sécurisé, une personne mal intentionnée serait en mesure de récupérer tous les messages circulant, et en particulier le message d'authentification envoyé par le client au serveur d'application. Elle pourrait alors utiliser ce message plus tard et le renvoyer au serveur d'application. Dans ce cas, si le serveur n'utilisait pas la clé de session pour sécuriser la communication avec le client, l'attaquant pourrait se faire passer pour sa victime.

La solution apportée par le protocole Needham – Schroeder oblige le client à prouver qu'il détient bien la clé de session : pour ce faire, le serveur d'application génère un autre nombre aléatoire, qu'il crypte à l'aide de la clé de session et fait parvenir au client. Le client décrypte alors ce nombre et doit, par exemple, lui ajouter une unité, puis le renvoyer crypté avec cette même clé de session. De cette manière, seul le « vrai » client, détenant la clé de session, est en mesure de répondre au serveur de service avec le bon nombre.

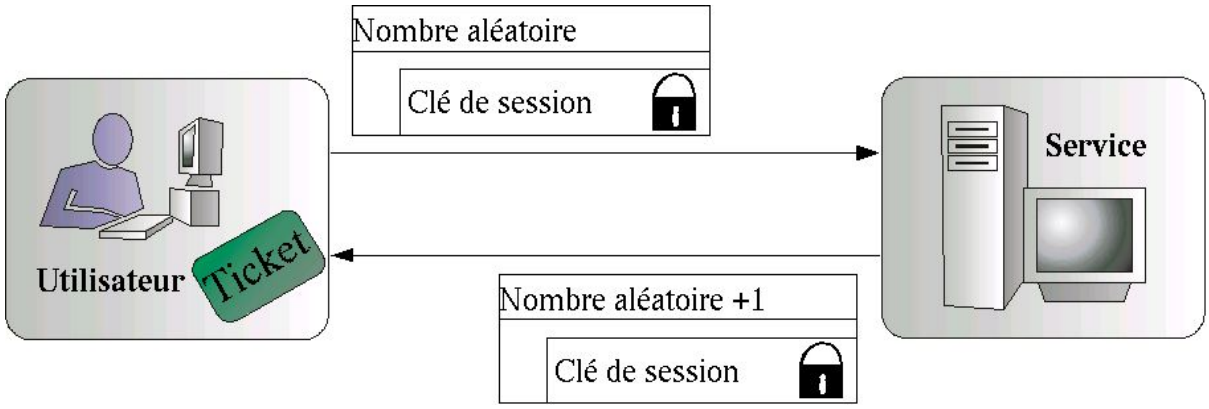


Figure 3.4 : Vérification mutuelle de l'identité

Le protocole Kerberos utilise une approche similaire pour éviter ce type d'attaque, mais il est basé sur un système d'horloges synchronisées.

3 Qu'est-ce que Kerberos ?

Kerberos a été conçu dans le but de proposer un protocole d'authentification multi-plateforme, disposant d'un système de demande d'identification unique, et permettant de contacter ensuite autant de services que souhaité. C'est pour ces raisons qu'il s'appuie sur le protocole de Needham – Schroeder.

Il s'agit d'un protocole sécurisé, dans le sens où il ne transmet jamais de mot de passe en clair sur le réseau.. Il transmet des messages cryptés à durée de vie limitée.

Le terme « *single sign-on* », utilisé en sous-titre de ce document, décrit le fait que l'utilisateur final n'a besoin de s'authentifier qu'une fois pour utiliser toutes les ressources du réseau supportant Kerberos au cours de sa journée de travail (en réalité, au cours du temps de session spécifié par l'administrateur : environ vingt heures, en général).

Le système Kerberos repose sur un « *tiers de confiance* » (Trusted third-party), dans le sens où il s'appuie sur un serveur d'authentification centralisé dans lequel tous les systèmes du réseau ont confiance. Toutes les requêtes d'authentification sont ainsi routées au travers de ce serveur Kerberos centralisé.

Le système d'authentification mutuelle utilisé permet non seulement de prouver que l'utilisateur derrière son clavier est bien qui il prétend être, mais aussi que le service qu'il tente d'utiliser correspond également. De cette manière, la communication instaurée assure la confidentialité des données sensibles.

Les trois concepts définis ci-dessus permettent de décrire les bases du service d'authentification réseau Kerberos.

4 Présentation de l'authentification avec Kerberos

Kerberos s'appuie donc sur ces bases lorsqu'un utilisateur du réseau souhaite utiliser un service. Les trois entités présentées ici, le client, le service et le serveur d'authentification, sont définies en tant que «*principals* » dans Kerberos.

Le service a besoin de savoir qui est l'émetteur de la requête. C'est pour cela que l'utilisateur lui présente un ticket, qui lui a été remis par le centre de distribution des clés Kerberos, ou KDC (*Kerberos Key Distribution Center*). Le ticket doit donc contenir des informations permettant d'identifier clairement l'utilisateur. Il doit montrer que l'émetteur détient une information que lui seul peut connaître, comme par exemple un mot de passe.

Kerberos nécessite que l'utilisateur et le service bénéficient de clés enregistrées auprès du KDC, et requiert la synchronisation des horloges des clients et des serveurs du réseau. La clé de l'utilisateur est dérivée d'un mot de passe que lui-seul connaît, tandis que celle du service est générée aléatoirement (personne ne peut taper de mot de passe dans ce cas).

A partir de ce point, le déroulement est très similaire au protocole décrit précédemment.

Le client émet une requête auprès du KDC, précisant son nom ainsi que celui du service souhaité. Lorsque le KDC reçoit ce message, il crée deux copies d'une clé générée, la clé de session, qui sera utilisée au cours de la communication entre le client et le service.

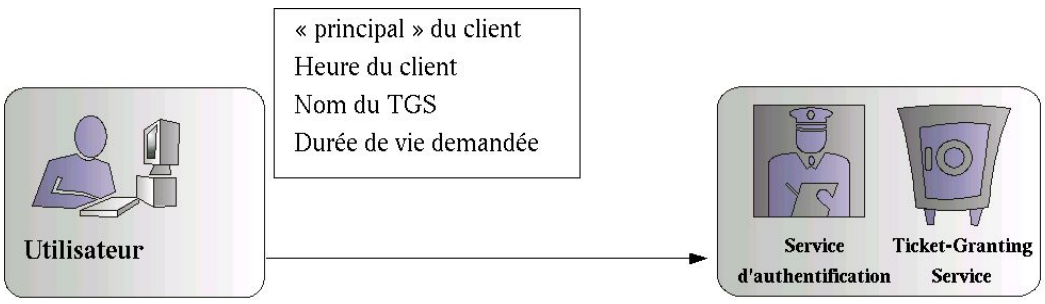


Figure 4.1 : Premier échange avec le service d'authentification

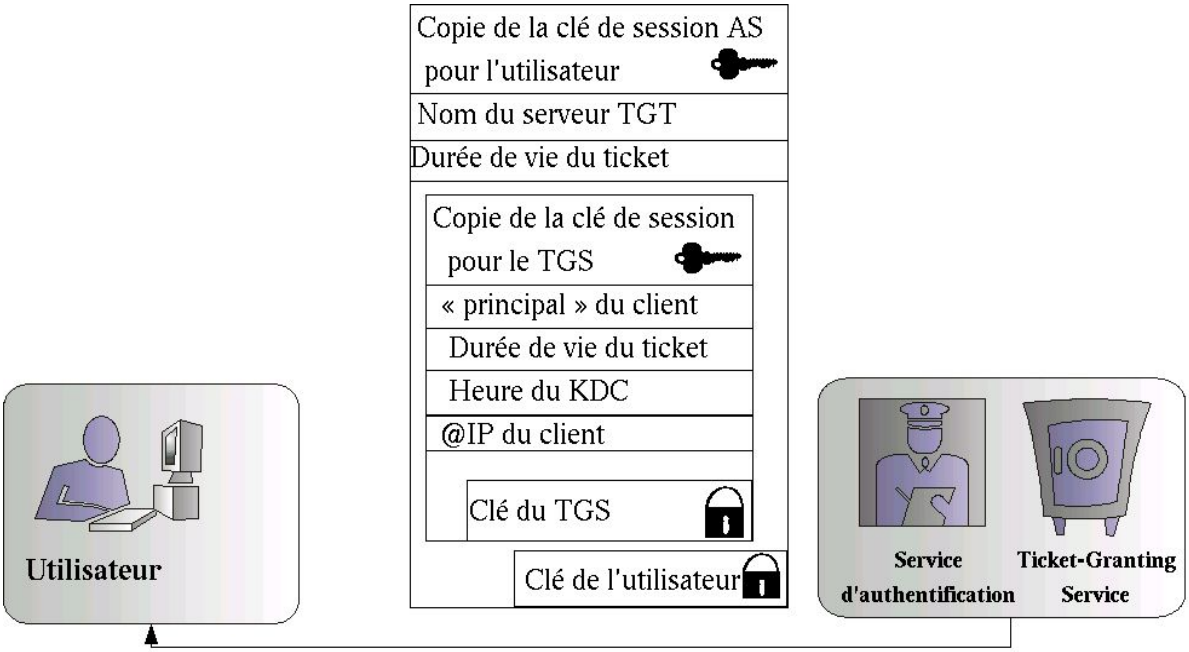


Figure 4.2 : réponse du service d'authentification

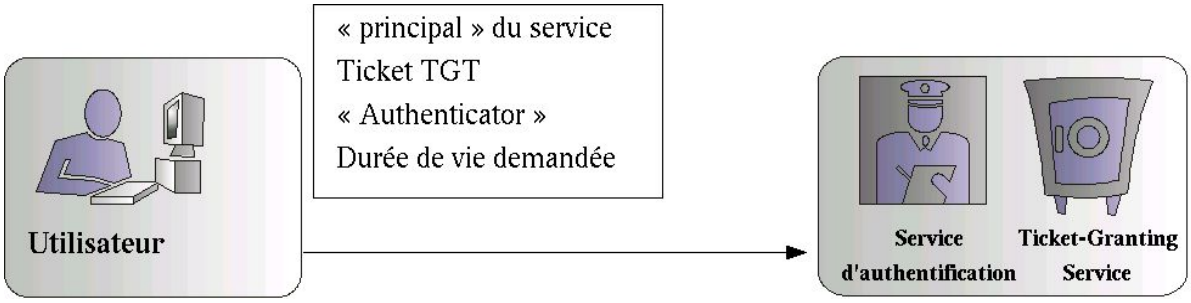


Figure 4.3 : requete au TGS

Le KDC envoie un message comportant deux « boîtes », l'une contenant une copie de la clé de session ainsi que le nom du service, cryptée à l'aide de sa clé privée ; la seconde contenant la deuxième copie de la clé de session, ainsi que le nom de l'utilisateur, et cryptée à l'aide de la clé du serveur d'application. Cette deuxième clé est appelée « *ticket* ».

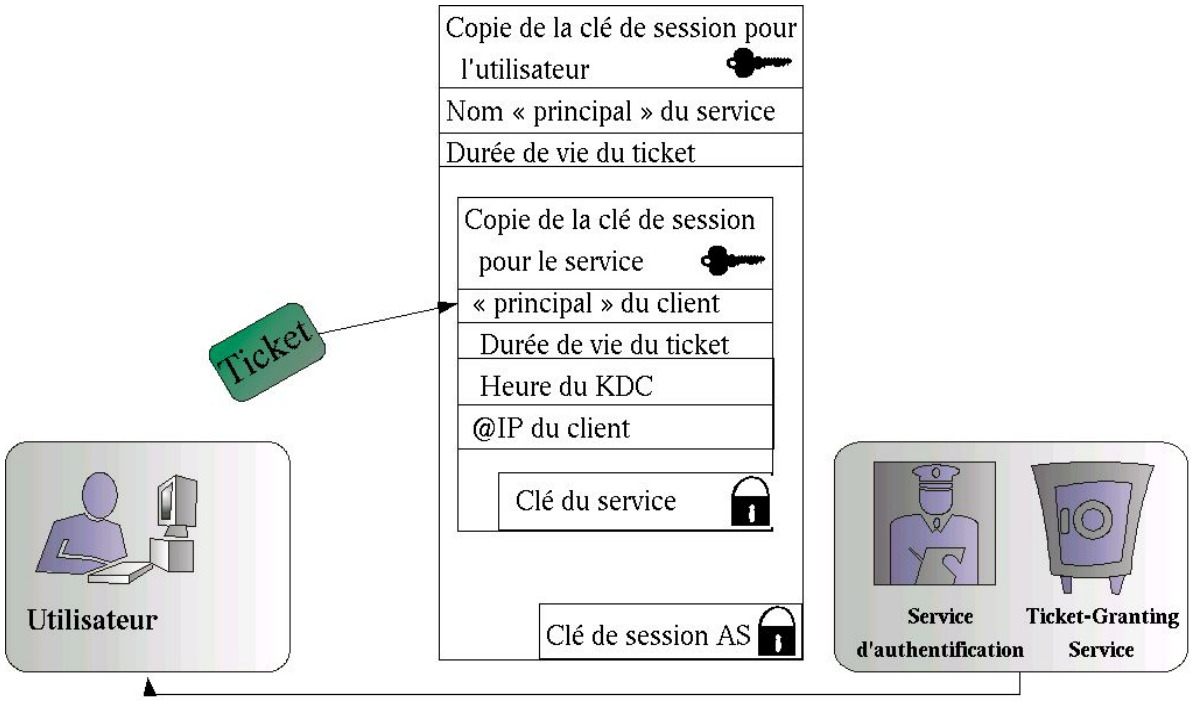


Figure 4.4 : réponse du TGS

L'utilisateur ouvre la première boîte avec sa clé et récupère le nom du service. Il ne peut pas ouvrir la seconde, puisqu'elle a été« fermée » avec la clé du serveur d'application. Il va alors composer un nouveau message contenant non seulement cette deuxième boîte, mais aussi une boîte contenant son nom et l'heure d'envoi et cryptée à l'aide de la clé de session. Cette troisième boîte est appelée « authenticator » dans le jargon Kerberos.

Le serveur d'application reçoit ce message, ouvre la deuxième boîte à l'aide de sa clé privée, obtient ainsi la clé de session qu'elle contient et peut enfin ouvrir la troisième boîte qui lui était destinée. Le service connaît ainsi le nom de son interlocuteur, et l'heure de son poste de travail. Cette information est très importante, puisqu'elle permet de s'assurer que personne n'a copié le ticket. Comme les horloges ne sont pas toujours en parfaite synchronie, une marge de cinq minutes est autorisée par défaut. De plus, le service maintient une liste des « *authenticator* » envoyés, afin de s'assurer qu'ils ne soient pas ré- émis.

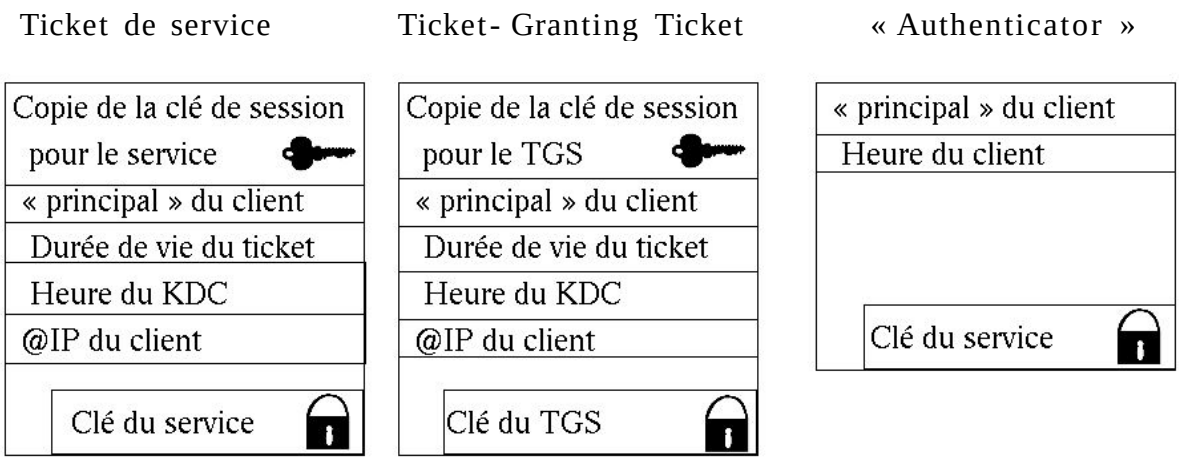


Figure 4.5 : les différents tickets circulant sur le réseau

En général, l'« *authenticator* » contient beaucoup plus d'informations que celles que nous exposons ici, comme par exemple un «*checksum* » pour valider l'intégrité des données, ou une clé de cryptage pour assurer la confidentialité des communications futures entre le client et le service.

Parfois, l'utilisateur souhaite que le service s'authentifie à son tour. Pour cela, le service récupère l'heure de la troisième boîte («authenticator »), le met dans une quatrième boîte avec son nom, et la crypte à l'aide de la clé de session. Quoi qu'il en soit, cette boîte doit contenir l'heure contenue dans la troisième, pour assurer au client la provenance du message, car lui-seul est en mesure de valider cette information.

5 TGS / TGT

Les échanges décrits jusqu'ici sont nécessaires à chaque fois qu'un utilisateur tente d'atteindre un service, et à chaque fois qu'il doit entrer un mot de passe (qui permet de décrypter la première boîte). Une solution consisterait à stocker la clé dérivée du mot de passe, mais cela engendrerait un problème de sécurité : une copie de ces clés permettrait de se faire passer pour l'utilisateur.

Kerberos offre une solution à ce problème, en répartissant les fonctionnalités du KDC : d'un côté le serveur d'authentification («*Authentication Server* ») ; de l'autre, le serveur d'obtention de ticket («*Ticket-Granting Server* »). Le serveur d'authentification est chargé de produire les tickets pour le TGS.

Le client fourni un mot de passe, en échange duquel le TGS lui donne un ticket (« *Ticket- Granting Ticket* ») et la clé de session associée (cf Figure 4.2). Celui- ci est valable en général pendant huit heure. Il s'agit de la seule communication entre le client et le serveur d'authentification. Comme le TGT est le premier ticket obtenu, il est aussi appelé « ticket initial ». Ensuite, quel que soit le service dont voudra bénéficier le client, il enverra le TGT au TGS afin d'obtenir un ticket ordinaire (cf Figures 4.3 et 4.4). Il n'aura donc plus besoin d'entrer son mot de passe, puisque le KDC et lui- même partagent la clé de session. L'utilisateur peut donc simplement adresser un message au TGS, contenant le TGT ainsi qu'un nouvel « authenticator », ce qui l'identifiera immédiatement.

6 Mon royaume pour Kerberos

Jusqu'à présent nous n'avons considéré qu'un seul serveur d'authentification et un seul TGS, installés ou non sur la même machine. Cependant, dans une grande entreprise, il n'est pas rare de décomposer le réseau en « royaumes » ou « realm ». En effet, avec une augmentation des requêtes d'authentification sur le réseau, le KDC deviendrait un goulot d'étranglement pour le processus, ce qui n'est pas permis pour un système distribué comme Kerberos.

Les royaumes correspondent généralement aux zones DNS. Par conséquent, le nom du royaume est, dans ce cas, le nom de domaine DNS en majuscule.

Pour permettre a un utilisateur d'un royaume d'atteindre un service situé dans un autre, le royaume de l'utilisateur doit s'authentifier auprès du TGS distant (« *Remote TGS* »). Les deux royaumes échangent alors une paire de clés, qu'ils utiliseront uniquement lors de la phase d'authentification entre royaumes. En fait, cette procédure est transparente pour l'utilisateur. Le programme Kerberos se charge de contacter le serveur d'authentification pour accéder au TGS, puis il contacte le TGS pour atteindre le TGS distant, et enfin le service.

Kerberos V5 propose même une structure hiérarchique des royaumes, car il n'est pas toujours utile de tous les traverser pour atteindre le service. En fait, l'ensemble des royaumes à traverser est enregistré dans les tickets.

Le cas d'un client voulant atteindre un service au travers d'un routeur mais dont l'adresse est translatée (NAT) pose une difficulté, car l'adresse IP de la machine émettant la requête apparaît dans les tickets. La solution consiste soit à paramétrer chaque client du réseau NAT afin qu'il utilise l'adresse interne du routeur pour les demandes de tickets, soit à utiliser l'option « -A » de l'application kinit, pour préciser une demande de tickets sans adresse IP. Il est aussi possible de modifier le fichier /etc/krb5.conf en y ajoutant, dans la section « libdefaults », la ligne :

```
noaddresses = true
```

Toutes ces procédures vous semblent sans doute bien compliquées, mais aux yeux de l'utilisateur, tout ceci est transparent : les programmes Kerberos se chargent de contacter le bon royaume, et l'utilisateur n'a plus qu'à simplement utiliser le service qu'il désire.

7 Audit

Il est bien entendu important de s'assurer qu'aucune personne extérieure ne soit capable d'attaquer une machine du réseau, mais il est tout aussi important de réaliser des audits en interne pour s'assurer de l'abscence de toute activité illicite. Pour cela, Kerberos propose de nombreuses options de journalisation de l'activité. Bien que Windows 2000 ne propose aucun journal par défaut, ils sont paramétrables depuis la console de gestion de la politique de sécurité.

8 Les ports utilisés par Kerberos

Le tableau suivant récapitule les ports utilisés par le serveur Kerberos, ainsi que la description du service correspondant.

Version	***Port***	***tcp***	***udp***	***Description***
Kerberos 5				
	88	oui	oui	Service de tickets Kerberos 5
	749	oui	non	Service Kerberos 5 pour le changement du mot de passe de l'utilisateur
	4444	non	oui	Service de conversion de tickets de Kerberos 5 à 4
	749	oui	non	Service d'administration de Kerberos 5
	464	non	oui	Service de changement de mot de passe pour une machine d'administration KDC
Kerberos 4				
	750	oui	oui	Service de tickets Kerberos 4
	751	oui	oui	Service d'administration Kerberos 4
	761	oui	non	Service de changement de mot de passe Kerberos 4

IV. L'ÉVOLUTION DE KERBEROS, V1 À V5

1 Athena

Le projet Athena, soutenu par un consortium de vendeurs d'ordinateurs, est lancé en mai 1983, pour une durée de 5 ans. Son but était de développer des stratégies, ainsi que des logiciels, dans le cadre d'un système réseau client-serveur. Il devait, à l'origine, être uniquement utilisé par le MIT. Vont ainsi être conçus des systèmes encore utilisés de nos jours, comme par exemple NFS (Network File System) et le serveur X (base de l'interface graphique des systèmes Unix).

L'idée d'un système d'authentification a germé au sein du MIT lorsqu'il est devenu évident que ses étudiants allaient être autorisés à accéder à des serveurs de fichiers sur un réseau assez grand.

Ce projet a été raccroché au projet Athena.

L'objectif de ce projet était de développer un protocole réseau d'authentification : celui-ci devait centraliser la confiance dans quelques machines d'un réseau, qui allaient être étroitement surveillées et contrôlées. Les communications d'authentification entre ces serveurs de confiance et les autres ordinateurs du réseaux devaient être cryptées afin de ne pas pouvoir être interceptées (sécurisation).

Depuis sa conception au sein du projet Athena, le protocole Kerberos a subit d'importantes modifications, lui permettant d'acquérir une meilleure facilité d'utilisation, une grande modularité et plus de sécurité.

2 Des versions internes

Les versions 1, 2 et 3 du protocole sont internes au MIT et auront servi à expérimenter de nouveaux concepts, pour finalement concevoir un système stable. Elles sont donc plutôt considérées comme des versions de test du protocole.

3 Versions 4 et 5

3.1 Des origines à la version 4

Mise à disposition le 24 janvier 1989, la première version publique du protocole a été rapidement approuvée et mise en oeuvre par plusieurs concepteurs de systèmes d'exploitation.

En réponse à la législation américaine leur interdisant l'exportation de leur produit, en particulier à cause de l'utilisation de DES (algorithme de cryptographie), une version spéciale dédiée à l'exportation a été conçue. Surnommée Bone, elle ne contient donc plus aucun code de cryptage.

Dans l'université australienne Bond, Errol Young développe sa propre implémentation de l'algorithme DES et l'intègre à la version Bones. Cela va donner eBones qui, développée en dehors des Etats-Unis, ne tombe plus sous sa législation et peut être librement exportée.

Plusieurs implémentations du protocole Kerberos 4 ont été développées. Même si certaines sont encore assez activement utilisées, la version 4 de l'implémentation du MIT est dorénavant en phase de maintenance, et peut être considérée comme terminée. Il est aujourd'hui recommandé d'utiliser la version 5, dernière en date.

3.2 Quoi de neuf dans la version 5?

La version 5 du protocole Kerberos a été développée afin d'apporter de nouvelles fonctionnalités et de nouvelles améliorations du point de vue de la sécurité.

La nouveauté la plus importante pour la pérénité du protocole est sa documentation dans la RFC1510.

Ont été améliorés :

- la délégation des tickets
- des types de cryptage extensibles
- des authentifications inter-royaumes améliorées
- ...

3.3 Les différences entre la version 4 et la version 5

Généralités

Kerberos V5 est ainsi supérieur à la V4 dans bien des domaines. Cependant, ces améliorations ont un coût : de plus faibles performances.

Cependant, cette nouvelle version permet entre autres d'utiliser des algorithmes de cryptage plus puissants et fiables, car DES a depuis été reconnu vulnérable à certaines attaques. La plupart de ces vulnérabilités sont pourtant dues au MIT, et à sa propre implémentation des algorithmes, et peuvent ne pas être présentes dans d'autres implémentations de Kerberos. Malheureusement, tout comme sa version précédente, l'implémentation du protocole par le MIT est toujours soumise aux lois américaines (malgré le nouveau statut des logiciels libres dans ce cadre) et ne peut être légalement exportée en dehors des territoires US.

La version 4 de Kerberos est, pour le moment encore, assez utilisée. Cette version n'est pas abordée dans ce document, au profit de la version 5 (qui reste compatible avec la version 4).

Une nouvelle norme de codage

Le plus difficile, dans une communication informatique (comme humaine), est de se faire comprendre par son interlocuteur. Tout va bien lorsque, d'un côté comme de l'autre, les normes de codage (ou de langage) est identique. Cela devient plus difficile avec la diversité des ordinateurs, et des systèmes que l'on peut rencontrer.

En informatique, ce problème se pose essentiellement dans la représentation des nombres : *little-endian* (bit de poids faible à la fin), et *big-endian* (bit de poids fort à la fin). Le protocole TCP/IP utilise la représentation *big-endian*, mais les défenseurs du *little-endian* restent nombreux.

La version 4 utilise un bit supplémentaire (appelé bit B) définissant la représentation utilisée dans le message :

- à 0 dans le cas de *big-endian*
- à 1 pour *little-endian*

Le protocole V5 utilise quant à lui une représentation totalement différente. Il utilise un standard de formatage des données, nommé ASN.1 (pour Abstract Syntax Notation One), standardisé par l'ISO (International Standards Organization).

Cela permet à Kerberos d'être encore plus flexible, en autorisant des champs optionnels et/ou de taille variable.

Cependant, ce choix entraine le doublement de la taille des messages envoyés.

Une durée de vie des tickets modifiée

Le système de gestion de la durée de vie des tickets avec Kerberos V4 ne permettait pas une grande marge de manoeuvre.

En effet, la durée était codée sur 1 octets, et pouvait donc avoir en tout 255 valeurs différentes. Chaque valeur étant un multiple de 5 minutes, un ticket ne pouvait rester valide que 21 heures au maximum.

Si cette valeur maximum restait correcte et appropriée pour la plupart des cas, il n'en reste pas moins que de nombreux utilisateurs et administrateurs devaientt redemander un nouveau ticket toutes les 21 heures.

Dans la version V5, cette durée de vie est représentée sur 17 octets, avec 1 seconde par unité. Les temps représentés sont donc virtuellement illimités, puisqu'il est maintenant possible de faire expirer un ticket Kerberos jusqu'au dernier jour de l'année 9999.

De plus, la division en seconde du temps permet une plus grande maîtrise et une plus grande précision dans les processus d'authentification.

Une délégation améliorée

La version 5 améliore grandement le processus de délégation des tickets. Ainsi, contrairement à la version 4, il devient possible de transférer un ticket à un autre ordinateur.

La version 4 ne permet qu'une délégation partielle, autorisant un utilisateur à se connecter à un autre royaume que le sien.

Un hachage des mots de passe prenant en compte la pratique

Les clés Kerberos sont des chaînes d'octets (une clé DES, par exemple, a une longueur de 8 octets), comportant des nombres entre 0 et 255. Comme il est difficile pour un utilisateur de se souvenir de ces nombres, Kerberos intègre un mécanisme de hachage permettant de transcrire une chaîne de caractères en une chaîne d'octets. Ce mécanisme ne va pas permettre, par contre, de retrouver la chaîne de caractères originelle.

Un inconvénient majeur n'avait pas été pensé lors de la version 4 : un utilisateur possédant des comptes sur plusieurs royaumes va vouloir garder le même mot de passe sur tous les royaumes. Cependant, si la clé est récupérée en sniffant le réseau, cela signifie alors que la personne mal-intentionnée pourra aussi accéder à tous les royaumes sur lequel l'utilisateur possède un compte.

Pour limiter les dégâts, la version 5 du protocole va aussi intégrer le nom du royaume dans le processus de hachage, ce qui permettra de générer des clés différentes, pour un même mot de passe, mais sur des royaumes différents.

De nouveaux algorithmes de cryptographie

Comme expliqué plus haut, l'algorithme DES n'est plus, de nos jours, considéré comme fiable. Pouvant être décrypté en quelques heures en utilisant un cluster de machines, il est même considéré comme inutilisable d'un point de vue sécurité par de nombreuses personnes.

Le protocole V4 ne permet pas, à moins d'énormes modifications dans sa structure, d'utiliser un système de cryptage alternatif.

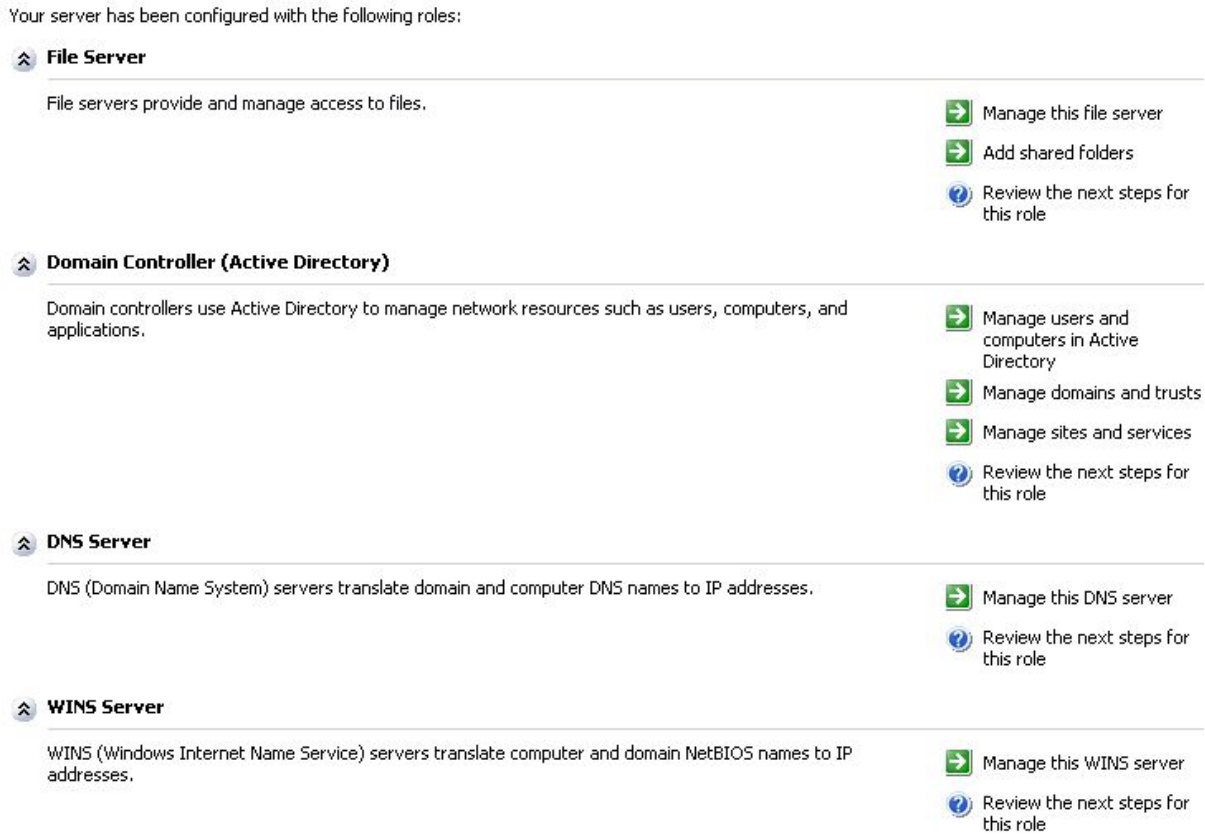
Le protocole V5 permet donc de gérer les algorithmes de manière modulaire, et intègre par défaut Triple-DES (clés de 168 bits, contre 56 bits avec DES), qui est bien plus sécurisant et solide. Cependant, le système de cryptage DES est gardé pour des raisons de compatibilité avec le protocole V4 de Kerberos.

V. MISE EN OEUVRE SUR DIFFÉRENTES PLATEFORMES

1 Le serveur

Dans le cadre de la rédaction de ce document, nous avons procédé à l'installation d'un serveur Windows 2003 beta.

Nous avons configuré notre serveur en tant que contrôleur principal de domaine, serveur DNS et serveur de fichiers. De cette manière, la base Active Directory a été installée, autorisant l'utilisation de Kerberos 5 dans la gestion des mots de passe des utilisateurs. En effet, Kerberos est directement intégré à LDAP.



La console d'administration des machines et des utilisateurs d'Active Directory permet ensuite de paramétrer l'utilisation de Kerberos dans la gestion des mots de passe. Nous avons ainsi pu paramétrer les options suivantes

- durcissement de la politique de mots de passe
- durée de vie
- ...etc

Kerberos utilise DNS pour la localisation des « enregistrements de service DNS » (SRV, RFC 2052) et peut, à l'aide des enregistrements TXT, localiser le royaume approprié correspondant à un nom d'hôte ou un nom de domaine.

Par contre, lorsqu'il s'agit de contacter un royaume non-Windows, il est nécessaire d'utiliser un programme particulier, `ksetup`, pour entrer les informations manuellement.

Sous Unix, la translation entre nom de machine et nom de royaume est réalisée dans le fichier `/etc/krb5.conf`, de la même manière que `/etc/hosts` est utilisé pour résoudre les noms de machines en adresses IP.

2 Le client ou le serveur d'application

Sous Windows 2000, XP ou 2003, le simple fait d'ajouter la machine au domaine permet de bénéficier de Kerberos lors de l'authentification de l'utilisateur.

Sous Linux, que l'on utilise le client MIT ou Heimdal, il convient de paramétrer le fichier de configuration `/etc/krb5.conf`. Ce fichier contient le nom et l'adresse des KDC avec lesquels le client est autorisé à communiquer. En fait, il correspond au fichier de configuration du KDC, qu'il est donc possible de déployer facilement. Cependant, il est possible d'y apporter des modifications, suivant le poste concerné, mais il est nécessaire de toujours entrer les bonnes valeurs pour trois des attributs qu'il contient :

- *libdefault/default_realm* : cette option, utilisée par toute application Kerberos, précise le royaume à contacter ;
- Ensuite, chacun des royaumes que le client sera susceptible de contacter doivent être décrits, soit dans la partie « realms » de ce fichier, soit précisés dans le DNS. Concernant le fichier krb5.conf, l'attribut le plus important ici est « kdc » ;
- Enfin, la section « domain_realm » fait le lien entre royaume et DNS, en faisant correspondre un domaine DNS à un nom de royaume.

Sous Linux, si l'on veut utiliser une application avec Kerberos, il convient d'utiliser une version « kerbérisée » de cette application. Ces versions sont généralement disponibles sur Internet.

VI. L'AVENIR DE KERBEROS

1 Une évolution permanente

Kerberos a toujours été en constante évolution, entre autres pour y intégrer les nouvelles technologies et pour palier aux nouvelles menaces.

Plusieurs extensions de Kerberos V existent donc, et proposent des aperçus des nouveautés qui seront implémentées dans les versions à venir du protocole. Elles sont disponibles en tant que brouillon Internet (Internet Draft) au sein de l'IETF (the Internet Engineering Task Force).

2 De nouvelles bases

Il existe plusieurs façons pour authentifier une personne (ou plus généralement, une entité) ; on peut se baser sur :

- ce qu'il sait
- ce qu'il a
- ce qu'il est

« *Ce qu'il sait* » est tout simplement un mot de passe qui va permettre, par comparaison ou par décryptage, par exemple, de vérifier que la personne est bien celle qu'elle dit être. Seule cette technique est actuellement utilisée par le protocole Kerberos.

« *Ce qu'il a* » est plus subtil car il fait appel à un objet que la personne cherchant à s'authentifier a en sa possession. Le plus souvent, il s'agit d'un appareil générant 'aléatoirement' une suite de nombres/caractères. Cette suite ne sera valable uniquement que pendant un temps donné, et servira à l'authentification. Une version future de Kerberos devrait permettre l'utilisation de supports de type SmartCard, qui permettraient d'utiliser des clés beaucoup plus longues (mais très difficiles à utiliser dans le premier cas), pour augmenter encore la sécurité.

« *Ce qu'il est* » est justement une méthode encore peu utilisée pour le moment, et fait appel à des mécanismes de biométrie (empreinte digitale, rétine...). Cette technique n'est pas encore utilisée, entre autres par le manque de logiciels l'utilisant réellement, ainsi que du peu de matériels encore

onéreux.

L'intérêt de ce système est qu'il supprime la duplication, le vol, l'oubli et la perte. Cependant, à long terme, cette technique risque de ne pas être exempte de piratage.

Une autre amélioration dans ce domaine serait la généralisation, non pas de l'utilisation d'une de ces techniques d'authentification, mais de plusieurs d'entre- elles combinées.

3 Une refonte totale du système d'authentification ?

Les débuts de Kerberos se déroulaient en même temps que les débuts (ou presque) de la cryptographie asymétrique (dite à clés publiques). A cette époque, ces algorythmes n'étaient pas très utilisés, entre autres à cause de la puissance de calcul nécessaire, mais aussi à cause des brevets déposés.

La loi de Moore et le temps ont maintenant permis à cette technologie d'être plus généralement utilisée en matière de sécurité (SSL, et maintenant TSL, par exemple), et son implémentation dans Kerberos est plus que jamais à l'ordre du jour.

En matière de sécurité, les clés asymétriques présentent des atouts non négligeables face aux clés symétriques (ou cryptographie à clés privées). Elles vont permettre, en plus de la cryptographie des messages, de signer des messages (et donc authentifier son emetteur).

Cependant, la médaille possède quelques revers : le temps de traitement (utilisation de nombres à plusieurs centaines de chiffres) reste, même de nos jours, encore assez problématique pour les grands ensembles de données. De plus, il est toujours difficile de s'assurer que la personne qui fournit sa clé publique est réellement celle qu'elle prétend être.

Le premier problème se résoud en utilisant (comme de nombreux protocoles à l'heure actuelle) un mélange de clés publiques (phase d'initialisation) et de clés privées (phase de transmission de données).

Le second peut être corrigé de deux manières différentes. Globalement, l'utilisation d'un PKI (Public Key Infrastructure), sorte de base de données regroupant les clés publiques accompagnées d'informations sur son propriétaire, permet de s'assurer de l'identitité du propriétaire d'une clé publique donnée, car les organismes les valident eux- mêmes. Moins

centralisé, des outils comme le package PGP vont faire intervenir une tierce personne, connaissant personnellement le propriétaire, qui va alors signer la clé publique, et donc permettre une « identification formelle ». Ces types de mesures sont autant de protections contre des attaques *man-in-the-middle*.

L'utilisation des clés asymétriques avec Kerberos est donc désormais d'actualité, mais nécessiterait la révision complète des mécanismes d'authentification dans le protocole Kerberos.

VII. CONCLUSION

Le protocole Kerberos permet l'authentification des utilisateurs et des services du réseau. Il assure ainsi un certain degré de sécurité.

Du point de vue de l'utilisateur, son utilisation est transparente. Cependant, il perd son anonymat sur le réseau.

Cependant, une réelle politique de sécurité passe aussi par la mise en place de moyens tels que :

- paramétrage correct des autorisations
- détection des intrusions
- utilisation de logiciels/systèmes régulierement mis à jour
- ...

Kerberos, en tant que système d'autentification, n'est donc qu'un des maillons indissociables de toute la chaîne de sécurité.

Son rôle est primordial, car il va permettre la certification de la validité des interlocuteurs sur le réseau, mais sera totalement inutile si des portes sont laissées grandes ouvertes aux crackers, leur permettant de prendre possession des comptes avec privilèges.

VIII. BIBLIOGRAPHIE

Kerberos, The Definitive Guide – Jason Garman – Edition O'Reilly

ISBN 0-596-00403-6

Kerberos, A Network Authentication System – Brian Tung – Edition Addison-Wesley

ISBN 0-201-37924-4

redhat magazine #1 (comptes réseau avec LDAP et Kerberos)