Kerberos et la Sécurité

Emmanuel Bouillon

Commissariat à l'Énergie Atomique
Direction des Applications Militaires
BP 12, 91680 Bruyères-le-Chatel, France
emmanuel.bouillon@cea.fr

Résumé Afin d'évaluer la pertinence de Kerberos dans un contexte donné, il convient de comprendre non seulement l'intérêt de ce protocole mais aussi ses limites, y compris en termes de sécurité. Cet article propose de rappeler brièvement le fonctionnement du protocole Kerberos, puis de décrire certains problèmes de sécurité qui peuvent se poser lors de son déploiement.

Les points abordés sont regroupés en deux catégories : d'une part les problèmes inhérents au protocole lui-même, d'autre part les problèmes liés aux difficultés pratiques de déploiement. Les aspects de la première catégorie rappelleront les limitations connues de Kerberos et seront présentés à la lumière des implémentations actuelles. Les aspects de la deuxième catégorie aborderont les difficultés liées à la mise en œuvre de ce protocole, ces difficultés pouvant aboutir à des compromis impactant la sécurité.

Les éléments présentés dans cet article visent à aider à évaluer : l'apport de Kerberos dans un environnement donné, la cohérence de cet apport au vu des mesures de sécurité existantes dans cet environnement et la pertinence de cette solution par rapport à d'autres mesures de sécurité.

1 Introduction

1.1 Qu'est-ce que Kerberos ?

Selon la mythologie, Kerberos est le chien à trois têtes gardant le pont menant aux portes de l'enfer. Plus "récemment", Kerberos fut le nom choisi pour le mécanisme d'authentification d'un projet mené au MIT appelé Athena. Aujourd'hui, Kerberos est un protocole d'authentification réseau. La version actuelle du protocole, la version 5, est un standard défini dans la RFC 1510 [1]. Les versions 1 à 3 ne sont restées que des versions de développement. La version 4 est décrite dans [2] et [3].

1.2 Intérêts du protocole

Peu de mécanismes résolvent le problème de l'authentification réseau unifiée en milieu hétérogène, qui plus est, mixte Unix/MS Windows. L'utilisation pour

l'authentification, des NIS (Network Information Services), de SSH pour la distribution de fichiers passwd et shadow, d'UIS (Unix Integration Services) service Microsoft qui n'est plus supporté, des SFU (Service For Unix) sous Windows, de LDAP, sont des techniques qui souffrent toutes de défauts de mise en œuvre et de sécurité. Kerberos présente l'avantage d'être un standard disponible sous les systèmes "libres" (Linux, *BSD) et sous les systèmes propriétaires, Solaris (SunMicrosystem), Irix (SGI), Windows 2000 et XP. La portabilité des implémentations "OpenSource" permet de compléter cette liste. Bien sûr, Kerberos résout plusieurs problèmes de sécurité classiques dans l'authentification des clients et des services au sein d'un réseau. Enfin, Kerberos est aujourd'hui le principal mécanisme de sécurité sous-jacent supporté dans les implémentations de la GSSAPI [4,5]. Cette API étant la base des RPCSEC_GSS, mécanisme désigné pour sécuriser la prochaine version de NFS (version 4, [6]).

1.3 Objectif de cet article

Pour mieux évaluer la pertinence de ce protocole dans un contexte donné, il convient d'en comprendre non seulement les avantages mais aussi les limites, y compris en termes de sécurité. Cet article propose de rappeler le fonctionnement du protocole Kerberos, puis de décrire certains problèmes de sécurité qui peuvent se poser lors du déploiement de Kerberos. Les points abordés sont regroupés en deux catégories :

- Les problèmes inhérents au protocole lui-même ;
- Les problèmes liés aux difficultés pratiques de déploiement.

Dans la suite, une application sera dite "kerbérisée" pour signifier qu'elle supporte le protocole Kerberos.

2 Le protocole Kerberos

2.1 Caractéristiques du protocole

Le protocole Kerberos se base sur les deux articles [7] et [8]. Il permet l'authentification des utilisateurs et des services sur un réseau supposé non sûr. Par exemple, les données sur ce réseau peuvent être lues ou modifiées, les adresses des machines peuvent être faussées. Ces informations ne peuvent donc servir à l'authentification. Kerberos utilise une tierce partie de confiance. Toutes les entités du réseau, utilisateurs et services, appelés principaux, font confiance à cette tierce partie (le serveur Kerberos ou KDC pour Key Distribution Center). Enfin, Kerberos utilise des mécanismes de chiffrement basés sur des algorithmes à clef symétrique (ou secrète) : tous les principaux partagent cette clef secrète avec le serveur Kerberos.

2.2 Fonctionnement détaillé : de l'authentification par secret partagé à Kerberos v5

Dans toute la suite, Alice (A) sera l'entité qui initie la communication (analogie avec le client, l'utilisateur), Bob (B) l'entité qui répond (analogie avec le

service, serveur applicatif). K_{ab} sera la clef partagée entre Alice et Bob (on a $K_{ab} = K_{ba}$).

Authentification par secret partagé Kerberos utilise des algorithmes de chiffrement à clef secrète pour l'authentification. La manière la plus simple de faire de l'authentification à l'aide de tels algorithmes est la suivante :

1. Alice appelle Bob
2. Bob répond à Alice avec un nombre généré aléatoirement, N_b
3. Alice retourne $K_{ab}\{N_b\}$

A l'issue de cet échange, Bob en déchiffrant la réponse de Alice, vérifie que Alice connaît bien la clef secrète K_{ab}. Pour une authentification mutuelle, Alice émet à son tour un nombre aléatoire (N_a) que Bob retourne chiffré par K_{ab}.

Une autre méthode pour l'authentification mutuelle à l'aide d'un algorithme à clef secrète est de faire envoyer par Bob, non pas un nombre aléatoire, mais ce nombre chiffré par K_{ab}. Alice déchiffre ce message, effectue une opération triviale sur le nombre obtenu et le renvoie chiffré. On obtient le schéma suivant :

1. Alice appelle Bob
2. Bob envoie $K_{ab}\{N_b\}$
3. Alice déchiffre le message, obtient N_b et envoie $K_{ab}\{N_b - 1\}$
4. Alice envoie $\mathrm{K}_{ab}\{N_a\}$
5. Bob déchiffre le message, obtient N_a et envoie $K_{ab}\{N_a - 1\}$

Ce qui, en regroupant les messages indépendants, donne (schéma 1) :

1. Alice envoie $K_{ab}\{N_a\}$
2. Bob déchiffre le message, obtient N_a, envoie $K_{ab}\{N_a - 1\}$ et $K_{ab}\{N_b\}$
3. Alice déchiffre le message, obtient N_b et envoie $K_{ab}\{N_b - 1\}$

C'est ce schéma qui apparaît dans le protocole de Needham et Schroeder [7]. Avec une amélioration toutefois :

Utilisation d'une tierce partie de confiance L'un des inconvénients majeurs du schéma 1 est son manque d'extensibilité. La généralisation à m utilisateurs et n services, implique la distribution préalable de $m \times n$ clefs partagées. Une solution à cette problématique est l'introduction d'une tierce partie de confiance (une autre solution est l'utilisation d'un algorithme de chiffrement asymétrique pour négocier une clef de session symétrique). Cette tierce partie (TP) de confiance connaît les secrets partagés de chaque entité à authentifier (K_a pour Alice, K_b pour Bob). L'authentification mutuelle peut alors se faire comme suit :

1. Alice demande à TP d'accéder à Bob
2. TP génère une clef de session entre Alice et Bob, K_{ab}
3. TP envoie $K_a\{K_{ab}, Bob\}$ à Alice et $K_b\{K_{ab}, Alice\}$ à Bob

A ce niveau, on peut appliquer le schéma 1 pour l'authentification mutuelle par algorithme à clef secrète. En effet, Alice connaissant K_a, peut en déduire K_{ab}. De même, Bob connaissant K_b peut en déduire K_{ab}.

On remarque cependant que pour la distribution de K_{ab}, Alice, le client envoie un message alors que la tierce partie commune à tous les utilisateurs en envoie deux. On préfère en général faire porter la charge sur le client et non sur la tierce partie qui doit gérer tous les clients. On modifie donc légèrement le schéma précédent qui devient :

1. Alice demande à TP d'accéder à Bob
2. TP génère une clef de session entre Alice et Bob, K_{ab}
3. TP envoie $K_a\{K_{ab}, Bob\}$ et $K_b\{K_{ab}, Alice\}$ à Alice
4. Alice appelle Bob et lui envoie $K_b\{K_{ab}, Alice\}$

Alice, qui ne connaît pas K_b, ne peut rien faire de $K_b\{K_{ab}, Alice\}$ si ce n'est l'envoyer à Bob. Cette information est le ticket pour dialoguer avec Bob.

On abouti alors au schéma de Needham et Schroeder, le nombre aléatoire N_1, envoyé avec la première requête n'est là que pour rendre unique cette requête et sa réponse pour éviter des attaques de type "rejeu".

1. Alice demande à TP d'accéder à Bob et joint N_1
2. TP génère une clef de session entre Alice et Bob, K_{ab}
3. TP envoie $K_a\{N_1, K_{ab}, Bob, Ticket_b\}$ à Alice, où $Ticket_b = K_b\{K_{ab}, Alice\}$
4. Alice déchiffre la réponse, en tire $Ticket_b$ et K_{ab}
5. Alice appelle Bob avec $Ticket_b$ et $K_{ab}\{N_a\}$ (schéma 1)
6. Bob déchiffre le Ticket, en tire K_{ab}, répond avec $K_{ab}\{N_a - 1, N_b\}$
7. Alice répond avec $K_{ab}\{N_b - 1\}$

Kerberos Le protocole Kerberos améliore le schéma de Needham et Schroeder d'une part en remplaçant les nombres aléatoires par des dates (timestamps), d'autre part en séparant le rôle de la tierce partie de confiance en deux services :

- Le service d'authentification (AS pour Authentication Service)
- Le générateur de ticket de service (TGS pour Ticket Granting Service)

L'utilisation des timestamps permet d'introduire la péremption de ticket et de réduire la fenêtre de temps pendant laquelle le rejeu de ticket est possible. Ce type d'attaque est surtout empêché par l'utilisation d'un cache stockant les authentificateurs reçus et dont la date de validité n'a pas expiré. On note que cette propriété introduit une dépendance entre Kerberos et le service de temps.

La séparation du rôle du KDC en deux services induit deux types de ticket différents :

- Le TGT (Ticket Granting Ticket) est envoyé par l'AS. Une demande de TGT est en fait une demande d'accès au service TGS. Le TGT est donc le ticket du service TGS. La réponse de l'AS contient aussi la clef de session entre le client (A) et le service TGS ($K_{a,tgs}$) chiffrée par la clef secrète du client (K_a).

– Le TS (Ticket Service) pour un service donné (S) est envoyé par le TGS sur présentation d'un TGT et d'un timestamps chiffré avec la clef de session $K_{a,tgs}$. Le TGS sait donc que le client a pu déchiffrer la réponse de l'AS et donc connaît la clef secrète K_a. Le TGS génère une clef de session entre A et S, $K_{a,s}$. La réponse du TGS contient donc notamment, cette clef de session chiffrée par $K_{a,tgs}$ (A pourra ainsi la déduire) et le TS proprement dit, constitué, entre autre, de $K_{a,s}$ chiffré par K_s, la clef secrète du service S (S, sur présentation de ce ticket pourra en déduire $K_{a,s}$).

La même clef $K_{a,tgs}$ permet d'obtenir les clefs de session des services accédés pendant toute la validité du TGT. Le principal avantage de cette séparation des rôles du KDC est de permettre le SSO (Single Sign On). En effet, seul l'exploitation du TGT nécessite l'utilisation de la clef secrète du client. Ainsi, tant que son TGT est valide, le client pourra acquérir des TS sans réutiliser sa clef, qui est directement déduite du mot de passe dans le cas d'un utilisateur. Cette propriété, couplée avec la possibilité de rendre le TGT "forwardable", c'est-à-dire réutilisable une fois connecté au service, permet à Kerberos de fournir un SSO.

La pré-authentification Au cours du dialogue précédent, on constate qu'il n'est pas nécessaire de connaître K_a pour obtenir de la tierce partie de confiance un TGT et un message chiffré par K_a. Il suffit de le demander à l'AS pour l'obtenir.

La pré-authentification résout ce problème. Elle exige que la requête à l'AS, émise par le client, soit accompagnée d'un timestamp chiffré par K_a. L'AS peut ainsi valider que la requête émane bien d'une entité connaissant sa clef secrète.

2.3 Les relations de confiance inter-royaume

Kerberos prévoit la possibilité qu'un principal puisse s'authentifier auprès de principaux n'appartenant pas à son royaume : Il s'agit de l'authentification inter-royaume. Cela implique d'établir une relation de confiance entre les deux royaumes. Cette relation peut être unilatérale ou bilatérale.

Deux méthodes sont possibles pour établir une telle relation.

Méthode directe Les deux KDC partagent une clef secrète utilisée pour prouver l'identité d'un principal lorsqu'il change de royaume. L'inconvénient de cette méthode est son manque d'extensibilité. La généralisation à n royaumes impose l'échange de $n \times (n-1)/2$ clefs ou $n \times (n-1)$ clefs dans le cas de relations bilatérales. Ceci peut être résolu par l'utilisation de la méthode transitive.

Méthode transitive Dans ce cas, on définit le chemin des royaumes partageant une clef secrète. Ce chemin peut être soit explicite (mécanisme des CApath, Certificate Authority Path) soit hiérarchique via le DNS.

3 Kerberos et la sécurité

Cette partie rappelle certains des problèmes de sécurité qui peuvent être rencontrés au cours du déploiement de Kerberos sur un réseau.

3.1 Problèmes inhérents au protocole

Les problèmes décrits ci-après sont connus de longue date. Pour certains, Dug Song [9] notamment a démontré la faisabilité de leur exploitation. L'objectif de ce paragraphe est d'en rappeler le principe et de montrer que sans précaution, ces attaques sont toujours valables dans les implémentations actuelles du protocole.

Attaque par dictionnaire et pré-authentification Dans la version 4 du protocole, le mécanisme de pré-authentification n'était pas prévu. N'importe qui pouvait obtenir un message chiffré avec la clef secrète d'un utilisateur donné. Toute la sécurité reposait sur l'incapacité de déduire la clef secrète de ce message. Bien sûr, compte tenu de la propension des utilisateurs à choisir des mots de passe faibles, une attaque par dictionnaire avait clairement de fortes chances d'être fructueuse.

Sans pré-authentification, écrire un outil qui ferait l'équivalent d'un `ypcat passwd` avec les NIS ne pose pas de difficultés.

Dès lors, adapter un outil de craquage de mots de passe pour lui permettre d'attaquer ces messages chiffrés est assez simple. Dug Song propose un patch pour John the Ripper [10] et la version 4 de Kerberos. Avec l'avènement de la version 5 de Kerberos et l'apport de la pré-authentification, la conscience de cette faiblesse du protocole s'est estompée. Les documentations insistent peu sur son utilité. Il convient alors de faire les remarques suivantes sur ce sujet :

- La pré-authentification n'est pas activée par défaut sur nombre d'implémentations du protocole. Elle est notamment impossible si l'on souhaite une compatibilité avec Kerberos version 4.
- L'activation de la pré-authentification diminue le risque mais ne l'élimine pas puisque dans le cas d'une authentification licite, le message chiffré peut être lu sur le réseau. Modifier un sniffer à cette fin est simple.
- Adapter un outil de craquage de mot de passe à la version 5 du protocole est simple.

En conséquence, il est important de rappeler que la pré-authentification est une option indispensable à la sécurité de Kerberos mais n'élimine pas complètement le risque d'attaque par dictionnaire tant que la clef secrète sera dérivée d'un mot de passe statique choisi par l'utilisateur.

La mise en place de mesures assurant la robustesse des mots de passe choisis par les utilisateurs est un moyen efficace de lutter contre ce problème. Le support par Kerberos d'algorithmes de chiffrement asymétrique (PKINIT [11]) couplé avec l'utilisation d'un deuxième facteur pour l'authentification, une carte à puce par exemple, supprimerait le risque d'une telle attaque. Espérons que les efforts actuels pour l'intégration de PKINIT et de la pré-authentfication "matérielle" aboutiront rapidement.

Usurpation du KDC Cette deuxième attaque a aussi été mise en évidence par Dug Song notamment. Le but de ce paragraphe est d'en expliquer le principe, de donner les moyens de s'en protéger et leurs limites, ceux-ci n'étant que rarement préconisés dans les documentations.

Fondamentalement, Kerberos permet d'authentifier un utilisateur auprès d'un service kerbérisé et réciproquement. Cependant, on peut dans le cadre du déploiement de Kerberos, utiliser ce protocole comme point d'entrée sur un réseau, c'est-à-dire au niveau de l'authentification système.

Le moyen le plus simple pour y parvenir est l'utilisation d'un module PAM, d'autres font exécuter la commande `kinit` à la connexion de l'utilisateur ou encore changent la commande de login. Dans le cas de l'utilisation d'un module PAM, le système se sert de la capacité à déchiffrer la réponse du KDC avec le mot de passe fourni par l'utilisateur pour l'authentifier. Il ne s'agit pas d'une réelle authentification Kerberos auprès d'un service. Le défaut de cette configuration est qu'il n'y pas authentification de la réponse du KDC. La requête et l'analyse de sa réponse sont faites par le client. Si la clef donnée par l'utilisateur permet de déchiffrer le TGT obtenu, l'utilisateur est autorisé à se connecter et possède un TGT. Ce n'est que s'il cherche à accéder à un service kerbérisé qu'il s'en servira pour faire une demande de TS.

Ce manque d'authentification de la réponse du KDC permet une attaque par usurpation du KDC si l'on a la possibilité d'écouter et d'injecter du trafic entre le système cible et le KDC. Si la seule présentation d'un TGT correspondant à la clef secrète fournie au login permet de se connecter à un système, il est possible de contourner cette authentification comme suit :

1. L'attaquant se connecte au nom de la victime en entrant un mot de passe quelconque
2. Le système émet une requête de TGT
3. L'attaquant répond en utilisant la clef dérivée du mot de passe qu'il a lui-même fournie au système
4. La réponse peut être déchiffrée par la clef : l'utilisateur est autorisé à se connecter

Bien sûr, le TGT ainsi obtenu ne permettra pas d'obtenir de TS valide. Mais une fois connecté, l'utilisateur peut accéder au disque et à tous les services non kerbérisés (Home en NFSv3 non kerbérisé par exemple).

Une contre-mesure pour cette attaque est de vérifier l'authenticité de la réponse du KDC en demandant un TS avec ce TGT pour le service `host` du système local et d'en vérifier la validité. Si la réponse est valide, le KDC partage non seulement la clef secrète de l'utilisateur mais aussi celle du service host du système. Il est à noter que le module PAM Kerberos le plus utilisé [12] prévoit cette fonctionnalité mais elle n'est pas documentée.

Kerberos et son environnement On décrit parfois Kerberos comme un protocole faisant peu d'hypothèses sur la sécurité du réseau où il est déployé. Il

convient cependant de comprendre les conséquences d'une compromission même partielle du réseau sur l'authentification.

La compromission, au niveau adminstrateur, d'une machine quelconque du réseau aboutit à la compromission des clefs des services de cette machine et des tickets des utilisateurs qui y sont connectés. L'attaquant peut alors usurper l'identité :

- des services de cette machine tant que leurs clefs ne sont pas changées
- des utilisateurs authentifiés sur cette machine pendant la durée de validité de leurs tickets

L'installation d'un cheval de Troie pourra aboutir à la compromission de mots de passe des utilisateurs, rendant l'usurpation de leur identité possible jusqu'au changement de leur mot de passe. En conséquence, le déploiement de Kerberos ne permet pas de s'abstenir d'autres mesures élémentaires de sécurité comme l'installation des mises à jour de sécurité. Enfin, dans un environnement kerbérisé, la compromission d'une seule machine spécifique, un KDC, conduit au contrôle total par l'attaquant de tous les authentifiants de entités Kerberos (utilisateurs et services) de ce royaume. On comprend la nécessité de sécuriser avec soin les machines assurant cette fonction et aussi de cloisonner les différents groupes d'utilisateurs dans des royaumes différents.

Kerberos et l'autorisation L'objectif d'une solution de sécurité est de s'attaquer à la problématique des trois 'A' (Authentication, Autorization, Accounting). Kerberos cherche à résoudre le problème du premier 'A', l'authentification. Il fournit au service l'assurance que l'utilisateur qui se présente à lui est bien celui qu'il prétend être (à la sécurité de la clef secrète près). Réciproquement, l'utilisateur sait qu'il s'adresse bien au service escompté (à la sécurité du fichier `keytab` près). De plus, en centralisant les demandes (échouées ou réussies) de connexions des clients aux services, Kerberos permet de tracer précisément l'accès de qui à quoi. Il participe ainsi à la traçabilité, le troisième 'A'. Par contre, Kerberos ne participe pas au processus d'autorisation. Celui-ci revient au service, confiant dans l'identité de son client. Cependant, les services classiques des implémentations de Kerberos, ne fournissent pas de mécanisme d'autorisation générique. L'utilisation à des fins d'autorisation du fichier `.k5login` est limitée. L'intégration du support des PAM, par exemple, est difficile compte tenu principalement du fait que ce mécanisme ne garantit pas la distinction entre authentification et autorisation.

3.2 Problèmes liés aux difficultés pratiques de déploiement

Kerberos peut améliorer notablement la sécurité d'un réseau. Cependant son déploiement est difficile, à tel point qu'on le déconseille parfois [13] :

"In our opinion, most sites are better off without it [sic]."

Ces difficultés aboutissent souvent à un compromis entre la meilleure solution du point de vue de la sécurité et les contraintes d'administration opérationnelle. Ce paragraphe présente une liste non exhaustive de difficultés pratiques qui

peuvent être rencontrées lors du déploiement de Kerberos et dont les solutions ou contournements auront des conséquences sur la sécurité.

Installation via le réseau À son installation, la machine n'est pas kerbérisée. Ce processus automatique ne peut donc pas complètement être sécurisé par Kerberos. Un moyen de distribuer les fichiers `keytab` d'une machine est d'utiliser la commande `kadmin`, de donner le mot de passe d'un principal privilégié et d'extraire le fichier `keytab`. Toute solution entièrement automatique aboutit à un certain niveau d'exposition du fichier *keytab*. Il est évidemment des environnements où l'intervention d'un administrateur à chaque (ré-)installation n'est pas envisageable.

Accès à un service sans mot de passe Plusieurs tâches d'administration classiques nécessitent l'accès à un service sans fournir de manière interactive l'authentifiant de l'identité utilisée. De telles procédures sont bien évidemment difficilement compatibles avec une méthode d'authentification sûre. Citons quelques exemples :

- Exécution de procédures automatiques :
  L'obtention d'un TGT nécessite toujours l'utilisation de la clef secrète du principal concerné. Si une procédure automatique a besoin d'accéder à un service kerbérisé (un script en crontab par exemple), comment lui faire obtenir un TGT puis un TS pour ce service. Plusieurs solutions ont été proposées pour résoudre ce problème [14,15]. Elles correspondent chacune à un compromis entre la sécurité et la faisabilité.
- Dépannage des utilisateurs :
  Il est parfois nécessaire à un administrateur de prendre l'identité d'un utilisateur, par exemple pour déboguer un problème qu'il n'arrive pas à reproduire dans son environnement. Que faire si ce problème nécessite l'accès à un service kerbérisé. Comment l'administrateur peut-il obtenir un TGT au nom de l'utilisateur ?
  Une solution évidente pour les implémentations qui utilisent un cache de ticket local sur le disque dur est d'utiliser ce fichier sous root. Ceci nécessite toutefois que l'utilisateur soit simultanément connecté. Cette contrainte peut être jugée trop importante. Dans ces conditions, comment un utilisateur peut obtenir le TGT d'un autre utilisateur ?
  Fondamentalement, un administrateur du KDC est omnipotent sur tous les principaux. Il peut donc obtenir un TGT pour un utilisateur. Cependant, certaines implémentations rendent ceci compliqué si l'on ne souhaite pas réinitialiser le mot de passe de l'utilisateur. C'est le cas pour l'implémentation du MIT, pas pour Heimdal. De plus, la population des administrateurs des KDC n'est pas nécessairement la même que celle qui dépanne les utilisateurs.
- Exécution de commandes en mode batch :
  Cette problématique regroupe les deux difficultés précédentes puisqu'il s'agit pour un processus système d'exécuter des commandes de manière

automatique, au nom d'un utilisateur quelconque. Situation que l'on peut rencontrer au niveau de l'ordonnancement d'un serveur de calcul parallèle par exemple. [16] notamment propose une solution. Les relations inter-royaume peuvent aussi être utilisées efficacement. Cela revient toutefois à dénaturer la sécurité de Kerberos en redonnant aux administrateurs une partie de leur pouvoir.

Relation de confiance unilatérale et ticket "forwardable" Etablir une relation de confiance inter-royaume est le moyen souvent préconisé pour permettre à un utilisateur authentifié dans un royaume (ROYAUME-1) d'accéder à un service d'un autre royaume (ROYAUME-2). Ceci implique que les responsables de ROYAUME-2 font confiance dans l'authentification et la sécurité mise en place dans ROYAUME-1. Si cette confiance n'est pas réciproque, il convient de ne mettre en place qu'une relation de confiance inter-royaume unilatérale : Un utilisateur avec un TGT de ROYAUME-2 ne pourra pas obtenir un TS pour un service du ROYAUME-1.

Comme rappelé précédemment, l'option "forwardable" d'un ticket permet d'obtenir un SSO. Par exemple, à chaque rebond de machine en machine, c'est-à-dire à chaque obtention de TS, le TGT est dupliqué sur la machine cible. Ainsi, le mot de passe de l'utilisateur n'a pas à être fourni de nouveau, l'authentification Kerberos est toujours possible. C'est pourquoi cette option est si prisée des utilisateurs quand il s'agit du service `host/machine`.

Cependant, si cette option est activée dans ROYAUME-1 et acceptée dans ROYAUME-2, le TGT d'un utilisateur authentifié dans ROYAUME-1 et accédant à un service de ROYAUME-2 sera dupliqué sur la machine cible. Dès lors, un administrateur de ROYAUME-2 peut accéder au TGT de cet utilisateur. Ainsi, la relation de confiance unilatérale entre ROYAUME-2 et ROYAUME-1 nécessite toutefois, dans ces conditions (ticket "forwardable" et donc intégration dans le SSO de ROYAUME-1), que les responsables de ROYAUME-1 fassent confiance aux administrateurs de ROYAUME-2. Ce qui fait nettement évoluer la signification de la relation de confiance "unilatérale".

Protection des administrateurs Les points précédents confirment une fois de plus que les administrateurs occupent une place cruciale pour la sécurité d'un réseau, quand bien même kerbérisé. Les efforts consentis pour le déploiement de Kerberos doivent donc être en rapport avec ceux dédiés à la protection des administrateurs et de leurs stations de travail. Les premiers seraient inutiles si les homes des administrateurs sont exportés par NFS à tous et leurs displays accessibles, par exemple.

Authentification applicative Le support de l'authentification système via Kerberos est maintenant largement répandu. Son déploiement peut donc être envisagé même en environnement hétérogène. Cependant, le système d'exploitation n'est souvent pas l'unique partie du système d'information où une authentification est requise. L'accès au mail, via les protocoles POP ou IMAP, doit être

pris en considération. L'accès à une base de données ou à une appplication Web par exemple, peut exiger une authentification supplémentaire ou simplement la même mais renouvelée pour accorder son accès.

Dans le cas d'une authentification renouvelée, c'est-à-dire où l'application utilise le même protocole que l'authentification système, il est évident que le déploiement de Kerberos aura des conséquences sur le fonctionnement de l'application. Même si des standards, le plus souvent basés sur la GSSAPI, voient progressivement le jour [17,18], leur intégration n'est jamais transparente. Cet impact est donc à prendre en compte dès l'origine du projet de déploiement. Dans le cas d'une authentification applicative reposant sur un protocole différent, le problème repose de nouveau sur la cohérence des mécanismes de sécurité mis en œuvre. Par exemple, si les deux méthodes d'authentification reposent sur un mot de passe au choix de l'utilisateur, le mécanisme d'authentification le plus faible affaiblit le plus fort compte tenu de la probabilité d'un choix identique pour ces mots de passe.

Compatibilité des implémentations Le protocole Kerberos dans sa version 5 est un standard bien défini [1]. Lors de son déploiement se pose la question du choix d'une ou plusieurs implémentations de ce protocole. Certaines sont "OpenSource" comme l'implémentation du MIT ou Heimdal, d'autres propriétaires comme SEAM, fournie par Sun Microsystem ou Active Directory. Deux catégories de problèmes peuvent apparaître :

- Les incompatibilités avec le protocole lui-même : l'implémentation est-elle respectueuse de la RFC de Kerberos, de la RFC de la GSSAPI ?
- Les incompatibilités entre implémentations si le choix a été fait d'en utiliser plusieurs : notamment, des problèmes peuvent apparaître au niveau de l'API d'administration, du changement de mot de passe, des relations d'approbation et des algorithmes supportés. Sur ce dernier point par exemple, pour établir une relation d'approbation entre un KDC MIT et un KDC Windows 2000 Server SP4, il est nécessaire de dégrager la liste des algorithmes supportés. Un tel contournement a évidemment un impact sur la sécurité. Ce problème est résolu avec un KDC sous Windows 2003.

4 Conclusion

Kerberos offre un moyen puissant et efficace de sécuriser un réseau. Son adoption par un grand nombre de systèmes d'exploitation est gage de pérennité. Néanmoins, il est important d'en connaître le fonctionnement, les limites ainsi que les difficultés de son déploiement et de son administration. Ces dernières pouvant éventuellement aboutir à des compromis impactant la sécurité. On pourra alors appréhender de manière cohérente la sécurité d'un réseau et éviter le danger d'en surévaluer le niveau.

Références

1. J. Kohl et B. Clifford Neuman, The Kerberos Network Authentication Service (Version 5), *Internet Request for Comments RFC 1510*, septembre 1993.
2. B. Clifford Neuman et T. Ts'o, Kerberos : An Authentication Service for Computer Networks, *IEEE Communications*, Vol. 32, number 9, pp. 33-38, septembre 1994.
3. J. Kohl, B. Clifford Neuman et T. T'so, The Evolution of the Kerberos Authentication System In Distributed Open Systems, *IEEE Computer Society Press*, pp. 78-94, 1994.
4. E. Bouillon et P. Deniel, Premiers pas avec la GSSAPI, *Linux Magazine*, volume 52, juillet 2003.
5. J. Linn, The Kerberos Version 5 GSS-API Mechanism, *Internet Request for Comments RFC 1964*, juin 2001.
6. S. Shepler, B. Callaghan, D. Robinson, R. Thurlow, C. Beame, M. Eisler et D. Noveck, Network File System (NFS) verion 4 Protocol, *Internet Request for Comments RFC 3050*, avril 2003.
7. R. Needham et M. Schroeder, Using Encryption for Authentication in Large Networks of Computers, *Communications of the ACM*, Vol. 21, number 12, pp. 993-999, décembre 1978.
8. D. Denning et G. Sacco, Time stamps in Key Distribution Protocols, *Communications of the ACM*, Vol. 24 numéro 8, pp. 533-536, août 1981.
9. `http://www.monkey.org/~dugsong/`
10. `http://www.openwall.com/john`
11. B. Tung, C. Neuman, M. Hur, A. Medvinski, J. Wray et J. Trostle, Public Key Cryptography for Initial Authentication in Kerberos, *IETF Internet-Draft.*
12. `http://sourceforge.net/projects/pam-krb5/`
13. E. Nemeth, G. Snyder, S. Seebass et T. Hein, *Unix system administration Handbook*, Prentice Hall, 3ème édition, 2001.
14. Kerberos FAQ, `http://www.cmf.nrl.navy.mil/CCS/people/kenh/kerberos-faq.html`
15. I. Hollander, P. Rajaram, et C. Tanno, Morgan Stanley & Co, Kerberos on Wall street, *Usenix*, 1996.
16. A. Rubin et P. Honeyman, Long Running Jobs in an Authenticated Environment, *Usenix*, 1993.
17. E. Baize et D. Pinkas, The Simple and Protected GSS-API Negotiation Mechanism, *Internet Request for Comments RFC 2478*, décembre, 1998.
18. R. J. Detry, S. D. Kleban et P. C. Moore, The Generalized Security Framework, `http://www.prod.sandia.gov/cgi-bin/techlib/access-control.pl/2001/018338.pdf`, juin 2001.