Authentification des utilisateurs - de LDAP à Kerberos

Pascal Aubry et François Dagorn (IFSIC / université de Rennes 1)

Introduction

Cet article explique comment il est possible de migrer l'authentification des utilisateurs de l'université de Rennes 1 depuis la solution actuelle basée sur l'annuaire LDAP vers une solution basée sur Kerberos, avec les objectifs suivants :

- Supprimer les failles de sécurité actuelles observées avec l'authentification LDAP.
- Proposer un mécanisme d'authentification autre que LDAP pour Windows 7.
- Disposer d'une authentification SSO depuis la session utilisateur sur les postes clients jusqu'aux applications web.

Nous commençons par décrire la situation actuelle, puis évoquons les solutions possibles. Après un très bref aperçu de Kerberos, nous montrons les tests qui ont été menés à bien puis proposons une stratégie de déploiement pour laquelle l'IFSIC serait pilote pour l'université.

1. La situation actuelle

1.1 Architecture

L'architecture informatique de l'université de Rennes 1 est aujourd'hui constituée de :

- **postes clients** (Windows XP, Linux et Mac OS X). A ces postes viendront s'ajouter à la rentrée 2010/2011 des postes Windows 7.
- **serveurs de fichiers**
  - Network Appliance (commun à toute l'Université) partagé en NFS pour les clients Unix et en CIFS pour les clients Windows
  - Serveurs de fichiers Unix partagés en NFS pour les clients Unix et/ou CIFS (Samba) pour les clients Windows
  - Serveurs de fichiers Windows partagés en CIFS pour les clients Windows
- **services applicatifs web**
- **serveurs d'impression** CUPS

1.2 Authentification

Le système d'authentification des usagers est architecturé autour de l'annuaire LDAP de l'établissement qui est dérivé journellement du système d'information. L'authentification LDAP est dans tous les cas réalisée par un *fast bind* sur l'annuaire du couple *uid*/*password* dont on

veut valider l'authenticité, les mécanismes suivants sont mis en oeuvre :

- Les clients Unix se basent sur le module PAM pam_ldap.
- Les clients Windows XP utilisent le module PGina, qui stocke les informations de connexion des utilisateurs (*uid*/*password*) et les rejoue quand nécessaire auprès des serveurs pour effectuer les montages (*net use*).
- Les montages CIFS du Network Appliance transmettent les couples (*uid*/*password*) à l'annuaire LDAP pour validation.
- Les montages NFS (v3) sont effectués sans authentification, par confiance envers les clients autorisés.
- Les services applicatifs web sont CASifiés, l'authentification au niveau du serveur CAS est déléguée à l'annuaire LDAP.
- Les clients Windows impriment via des montages SMB sur les serveurs d'impression, l'authentification est faite par Samba via pam_ldap.
- Les clients Unix impriment en IPP, également via pam_ldap.

Notons qu'il s'agit d'un usage détourné de l'annuaire LDAP, conçu initialement comme un service de pages blanches pour la recherche d'informations (utilisateurs, machines) et est aujourd'hui principalement utilisé pour l'authentification. Cet usage est néanmoins très largement adopté dans nos universités et répond fonctionnellement bien au besoin car il permet de contrôler l'accès à tous les types de services nécessaires sur un réseau informatique (ouverture de sessions, impressions, partage de fichiers, ...). Il est de plus disponible dans tous les environnements utilisés (Linux, Windows, MacOS, Solaris, ...).

2. Pourquoi évoluer ?

L'utilisation de LDAP décrite plus haut est aujourd'hui relativement générale dans les universités, plusieurs problèmes liés à la sécurité des réseaux se posent pourtant.

2.1 Pour plus de sécurité

Les mots de passe des usagers doivent impérativement être acheminés en clair depuis les postes de travail qui hébergent des services jusqu'aux serveurs LDAP chargés des opérations de contrôles. L'utilisation du protocole sécurisé LDAPS (LDAP sur TLS) permet de contourner ce problème puisqu'il impose une session TLS avant tout dialogue LDAP. L'université de Rennes 1 n'utilise pas LDAPS et tous les postes de travail du réseau académique exposent les mots de passe des usagers aux yeux d'utilisateurs indélicats (l'empoisonnement ARP est facile à mettre en oeuvre). L'utilisation de services de fichiers mutualisés contrôlés par LDAP accentuent ce problème car les mots de passe doivent circuler en clair jusqu'aux services avant d'être acheminés ensuite (éventuellement en LDAPS) jusqu'aux serveurs LDAP. Dans ce cadre, les serveurs d'impressions (Samba, pam_ldap) ainsi que le serveur de fichiers communautaire de l'université de Rennes 1 sont aujourd'hui des maillons faibles de la sécurité du réseau de l'établissement.

2.2 A cause de l'apparition de Windows 7

En 2009, le nouveau système d'exploitaion Windows 7 est apparu, il ne s'intègre pas facilement dans un environnement contrôlé par LDAP.

Windows XP utilisait PGina pour intercepter le mot de passe de l'usager au moment de la phase de connexion, il le ressortait ensuite quand celà s'avérait nécessaire (utisation de partages, ...). Ce schéma ne peut plus fonctionner sous Windows 7, mais on ne peut toutefois pas renoncer à Windows 7 pour cette raison.

L'IFSIC a développé un outil similaire à PGina (Regina), il est en place et fonctionne. La méthode utilisée par Regina et PGina pour intercepter et utiliser le mot de passe en clair n'est néanmoins pas compatible avec les impératifs de sécurité d'un réseau informatique. Plus généralement, c'est l'utilisation de LDAP pour authentifier des usagers qui pose de gros problèmes de sécurité.

2.3 Pour un SSO de bout en bout

La généralisation des ENT a permis la mise en oeuvre de systèmes d'authentification unique pour les environnements web, mais on reste pour l'instant d'en l'attente d'un système d'authentification allant de l'ouverture de session jusqu'aux applicatifs Web.

3. Les solutions

On distingue trois solutions possibles raisonnablement envisageables à la rentrée 2010/201.

3.1 Ne rien faire

Bien que non sécurisé, le réseau est aujourd'hui fonctionnel, il peut rester en l'état.

Si cette solution est la plus envisageable en terme de coût, elle oblige néanmoins à utiliser une « verrue » supplémentaire pour Windows 7 (Regina).

3.2 Utiliser Active Directory quand c'est possible

Les deux failles principales sont le serveur de fichiers et les services d'impressions. S'il est relativement simple de contrôler les accès CIFS au serveur de fichier communautaire (par l'intermédiaire d'un serveur Active Directory utilisant une authentification NTLM nettement plus sûre que LDAP), cette solution ne règle pas complètement le problème car :

- les postes Linux utilisent actuellement des montages NFS dans lesquels le serveur de fichiers fait confiance au client (sécurité zéro). Quelle solution adopter alors ?

- les services d'impressions continuent à utiliser Samba + pam_ldap, faut-il les faire basculer aussi vers Active Directory ? Comment se passeraient alors les impressions depuis Linux ?

3.3 Changer de mécanisme d'authentification

Dans ce cadre, seule la solution Kerberos est envisageable.

Deux implémentations peuvent être envisagées :

1. La mise en place d'un « vrai » service Kerberos (MIT ou Heimal).
2. L'utilisation des fonctionnalités Kerberos de Active Directory.

La deuxième solution, décrite par Emmanuel Blindauer en 2005 (Kerberos : Linux, Windows et le SSO), est possible. Néanmoins,

- Elle oblige à s'appuyer sur une solution non libre pour l'authentification, coeur de la sécurité du système.
- Elle comporte certains problèmes décrits par le même auteur quatre ans plus tard (Référentiel d'authentification interopérable et ouvert: Kerberos), dans lequel il confessait qu'il eut été plus judicieux de nommer l'article de 2005 « Active Directory = Windows + Linux + SSO + Problèmes ».

Notre choix se porte donc clairement sur la première solution.

4. Kerberos

Kerberos fonctionne en environnement hétérogène, assurant la sécurité des échanges sur un réseau non sûr et permettant la mise en place d'un véritable service d'authentification unique.

Kerberos utilise un système de chiffrement symétrique pour assurer un dialogue sécurisé entre deux protagonistes. Les dialogues s'opèrent en utilisant une clef secrète et partagée. Les algorithmes de chiffrement sont publics (AES, DES, 3DES, ...), toute la sécurité du système repose sur la confidentialité de la clef de chiffrement. Pour faciliter la gestion d'un tel système, Kerberos repose sur l'utilisation d'un tiers de confiance qui distribue les clefs aux utilisateurs et services abonnés (les *principals*). Un serveur Kerberos est appelé KDC (*Key Distribution Center*).

Kerberos est un service sûr qui assure la confidentialité, l'intégrité des données ainsi que la non-répudiation (les participants sont identifiés, y compris le serveur contrairement à NTLM). Le service d'authentification assure l'identification unique du client et lui procure un ticket de session qu'il pourra utiliser pour demander des tickets d'utilisation des services *kerbérisés*. Un ticket de session chiffré avec la clef d'un service *kerbérisé* constitue un ticket de service. On distingue deux fonctionnalités dans un service kerberos :

- le service d'authentification
- le service de délivrement de tickets de services.

Kerberos a été mis au point au MIT dans les années 1990, il est maintenant très largement déployé et est disponible dans tous les environnements aujourd'hui utilisés (Linux, Windows, MacOS, ...). Des universités françaises ont déjà migré leur systèmes d'authentification vers Kerberos, parmi celles-ci on peut citer les universités de Strasbourg et de Bordeaux 1.

5. Les tests effectués

Comme indiqué en 3.3, la possibilité d'utiliser le serveur Kerberos enlisé dans un service Active Directory de MicroSoft a été volontairement écartée et les tests ont été effectués avec un serveur Kerberos hébergé sur un serveur Linux.
Les éléments ci-dessous ont été validés.

5.1 Le service Kerberos

Un serveur kerberos (MIT 1.6.1) maître est fonctionnel sur **kerb1.univ-rennes1.fr**. Il est redondé par un second serveur esclave (**kerb2.univ-rennes1.fr**), dont la synchronisation avec le serveur maître est assurée par une *crontab* via le protocole *kprop*.

Une interface web (PHP, CASifiée) permet la gestion des principals clients (serveurs et stations de travail).

Voir : Installation des serveurs Kerberos

5.2 Le service CAS

Un serveur CAS (3.3.5) est fonctionnel sur **cas-kerb.univ-rennes1.fr**.

Ce serveur permet le SSO de bout en bout (depuis l'authentification sur les postes clients jusqu'à celle sur les applications web).

Il permet également l'alimentation du royaume Kerberos UNIV-RENNES1.FR par interception des authentifications LDAP.

Voir : Installation du serveur CAS

5.3 Les clients

L'authentification Kerberos s'intègre parfaitement (de manière native) dans les clients Linux, l'accès à tous les services a été validé : CAS,

NFS v3 et v4 (sur serveurs linux et NetApp), Samba, CUPS.

L'authentification Kerberos seule est possible pour les clients Windows (XP et 7), mais l'accès au serveur NetApp nécessite l'intégration d'un sous-domaine Active Directory.

L'authentification des services Samba (sur serveur Unix), CUPS a été validée.

Voir :

- Intégration d'un client Linux
- Intégration d'un client Windows XP
- Configuration de Firefox pour le SSO
- Configuration de Internet Explorer pour le SSO

5.4 Les services de fichiers

Le montage des volumes NetApp a été validé à la fois en NFS depuis les clients Unix et en CIFS depuis les clients Windows, en s'appuyant sur un Active Directory pour lequel une relation d'approbation mutuelle avec le royaume Kerberos a été mis en place.

Les montages NFS v3 et v4 ainsi que Samba ont également été validés, ce qui permet aux entités l'utilisation de services de fichiers autonomes.

Voir :

- Mise en place d'un serveur Samba
- Mise en place d'un serveur NFS (v4-Kerberos)

5.5 Le service CUPS

Les clients Windows et Unix peuvent imprimer sur un serveur CUPS Kerbérisé de manière transparente.

Voir : Mise en place d'un serveur CUPS

5.6 Le service 802.1X

Un serveur FreeRadius a été configuré pour utiliser une base d'authentification Kerberos.
Le dispositif fonctionne mais ne peut pas être intégré dans le cadre de l'authentification unique.

Voir : 802-1x, Radius et Kerberos

6. Une stratégie de déploiement

Les tâches à réaliser pour mettre l'authentification Kerberos en production sont listées ci-dessous.

Passage à Kerberos de l'IFSIC d'abord, du reste de l'université ensuite
La seule condition pour que le passage de l'authentification Kerberos soit possible d'abord sur l'IFSIC avant de passer à toute l'université est que le NetApp puisse partager les mêmes volumes à la fois avec authentification Kerberos en NFS v4 (pour les clients Unix de l'IFSIC) et sans authentification en NFS v3 (pour les clients Unix du reste de l'université, qui pourraient ainsi migrer à Kerberos ultérieurement).

Dans le cas où l'export mixte v4/Kerberos et v3/*Trusted* des mêmes volumes ne serait pas possible, il serait possible de rester en mode v3/*Trusted* en attendant que tous les clients NFS potentiels de l'université soient *Kerbérisés*. Les seuls prérequis pour le passage (dans une première étape) de l'IFSIC à Kerberos sont donc les tâches à réaliser au niveau du CRI et détaillées ci-dessous.

Cette stratégie (conservation de NFS v3/*Trusted* tant que tous les clients NFS n'ont pas migré à Kerberos) semble la plus raisonnable.

Le passage de toute l'université à l'authentification Kerberos nécessiterait au niveau de chaque cellule les mêmes tâches que celles à effectuer à l'IFSIC.

6.1 Tâches au niveau du CRI

Serveurs Kerberos

Recréer une infrastructure de production pour les serveurs Kerberos kerb1.univ-rennes1.fr et kerb2.univ-rennnes1.fr.

Voir : Installation des serveurs Kerberos

Serveur CAS

Modifier le serveur CAS pour qu'il permette :

- l'authentification des tickets Kerberos des clients
- l'alimentation du royaume Kerberos avec les utilisateurs qui s'authentifient sur l'annuaire LDAP

Voir : Installation du serveur CAS

Application Sésame

Modifier l'application Sésame de l'université pour qu'elle permette :

- l'ajout des utilisateurs dans le royaume Kerberos lors de l'activation des comptes
- la modification du mot de passe des utilisateurs dans le royaume Kerberos (en plus de celle dans l'annuaire LDAP)
- la suppression des utilisateurs dans le royaume Kerberos lors de la suppression des comptes

Voir : Modification de l'application Sésame

NetApp

Ajouter les exports NFS v4 avec authentification Kerberos (et vérifier qu'ils sont compatibles avec les mêmes exports v3 sans authentification, sinon rester en v3/Trusted).

Ajouter l'authentification Kerberos aux exports CIFS en s'appuyant sur l'Active Directory.

Voir : Mise en place d'un serveur NFS (v4-Kerberos)

6.2 Tâches au niveau de l'IFSIC

Clients

Modifier tous les clients pour l'authentification Kerberos.

Voir :

- Intégration d'un client Linux
- Intégration d'un client Windows XP
- Configuration de Firefox pour le SSO
- Configuration de Internet Explorer pour le SSO

Serveur de fichiers

Modifier l'authentification des serveurs Samba et NFS (passage à Kerberos).

Voir :

- Mise en place d'un serveur Samba
- Mise en place d'un serveur NFS (v4-Kerberos)

Serveur d'impression

Remonter un serveur d'impression Kerberisé pour faciliter la transition.

Voir : Mise en place d'un serveur CUPS

7. Conclusion

Tout est techniquement prêt pour migrer d'une authentification basée sur LDAP à une authentification basée sur Kerberos.

Nous sommes prêts dès à présent à contribuer à cette migration en assistant les équipes du CRI pour la mise en production (Kerberos, CAS et Sésame).

Références

- ARCHANN, une architecture d'annuaire et d'authentification interopérable pour un SSO unifié en environnement hétérogène, Pascal Levy (Université de Paris 1), JRES 2009
- Kerberos et la sécurité, Emmanuel Brouillon (CEA), SSTIC 2004
- Kerberos : Linux, Windows et le SSO, Emmanuel Blindauer (IUT Robert Schuman Strasbourg), JRES 2005
- Configuring a Kerberos 5 Server, Redhat 9 manual
- Replacing NIS with Kerberos and LDAP HOWTO

- Kerberos/LDAP/NFSv4 HOWTO
- Authenticate Windows to Unix Kerberos
- Making WindowsXP authenticate login to a UNIX MIT KDC
- Single sign-on "How To" Guide
- Référentiel d'authentification interopérable et ouvert: Kerberos, Emmanuel Blindauer (IUT R.Schuman Université de Strasbourg), JRES 2009
- http://pig.made-it.com/kerberos.html
- http://nfsv4.bullopensource.org/doc/kerberosnfs/krbnfs_howto_v3.pdf

Installation des serveurs Kerberos

Cette page montre comment installer deux serveurs Kerberos redondants (**kerb1** le maître et **kerb2** l'esclave).

Installation du serveur maître

Installation du système

Nom du serveur	**kerb1.univ-rennes1.fr**
Système	RedHat Entreprise 5
Ouverture de ports	**ssh** (22 tcp) **kinit** (88 tcp/udp) **kerberos password** (749 tcp) **kerberos auth** (750 tcp)

Configurer la synchronisation de l'horloge sur le serveur **ntp.univ-rennes1.fr** (cf **/etc/ntp.conf**) et s'assurer que le démon **ntpd** est en marche :

```
[root@kerb1 ~]# chkconfig ntpd on
[root@kerb1 ~]# service ntpd start
ntpd: Synchronizing with time server:                      [  OK  ]
Syncing hardware clock to system time                      [  OK  ]
Starting ntpd:                                             [  OK  ]
[root@kerb1 ~]#
```

Installation du KDC (Key Distribution Center)

Editer le fichier **/etc/krb5.conf** :

```
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 default_realm = UNIV-RENNES1.FR
 dns_lookup_realm = false
 dns_lookup_kdc = false
 ticket_lifetime = 24h
 forwardable = yes

[realms]
 UNIV-RENNES1.FR = {
  kdc = kerb1.univ-rennes1.fr:88
  admin_server = kerb1.univ-rennes1.fr:749
  default_domain = univ-rennes1.fr
 }

[domain_realm]
 .univ-rennes1.fr = UNIV-RENNES1.FR
 univ-rennes1.fr = UNIV-RENNES1.FR

[appdefaults]
 pam = {
   debug = false
   ticket_lifetime = 36000
   renew_lifetime = 36000
   forwardable = true
   krb4_convert = false
 }
```

Installer le package **krb5-server** (**yum install krb5-server**).

Editer le fichier **/var/kerberos/krb5kdc/kdc.conf** :

```
[kdcdefaults]
 v4_mode = nopreauth
 kdc_ports = 88,750
 kdc_tcp_ports = 88

[realms]
 UNIV-RENNES1.FR = {
  #master_key_type = des3-hmac-sha1
  acl_file = /var/kerberos/krb5kdc/kadm5.acl
  dict_file = /usr/share/dict/words
  admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
  supported_enctypes = des3-hmac-sha1:normal arcfour-hmac:normal des-hmac-sha1:normal
des-cbc-md5:normal des-cbc-crc:normal des-cbc-crc:v4 des-cbc-crc:afs3
 }
```

Editer le fichier **/var/kerberos/krb5kdc/kadm5.acl** :

```
*/admin@UNIV-RENNES1.FR *
```

Créer la base Kerberos :

```
[root@kerb1 ~]# kdb5_util create -s
Loading random data
Initializing database '/var/kerberos/krb5kdc/principal' for realm 'UNIV-RENNES1.FR',
master key name 'K/M@UNIV-RENNES1.FR'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:
Re-enter KDC database master key to verify:
[root@kerb1 ~]#
```

Ajouter le premier utilisateur (**root/admin**) :

```
[root@kerb1 ~]# kadmin.local -q "addprinc root/admin"
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
WARNING: no policy specified for root/admin@UNIV-RENNES1.FR; defaulting to no policy
Enter password for principal "root/admin@UNIV-RENNES1.FR":
Re-enter password for principal "root/admin@UNIV-RENNES1.FR":
Principal "root/admin@UNIV-RENNES1.FR" created.
[root@kerb1 ~]#
```

Démarrer les services :

```
[root@kerb1 ~]# chkconfig kadmin on
[root@kerb1 ~]# service kadmin start
Starting Kerberos 5 Admin Server:                          [  OK  ]
[root@kerb1 ~]# chkconfig krb5kdc on
[root@kerb1 ~]# service krb5kdc start
Starting Kerberos 5 KDC:                                   [  OK  ]
[root@kerb1 ~]#
```

Vérification en affichant la liste des *principals* :

```
[root@kerb1 ~]# kadmin -p root/admin
Authenticating as principal root/admin with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  listprincs
K/M@UNIV-RENNES1.FR
kadmin/admin@UNIV-RENNES1.FR
kadmin/changepw@UNIV-RENNES1.FR
kadmin/history@UNIV-RENNES1.FR
kadmin/localhost.localdomain@UNIV-RENNES1.FR
krbtgt/UNIV-RENNES1.FR@UNIV-RENNES1.FR
root/admin@UNIV-RENNES1.FR
kadmin:  exit
[root@kerb1 ~]#
```

Installation du serveur esclave

Installation du système

Nom du serveur	**kerb2.univ-rennes1.fr**
Système	RedHat Entreprise 5
Ouverture de ports	**ssh** (22 tcp) **kinit** (88 tcp/udp) **kerberos auth** (750 tcp)

Installation du KDC (Key Distribution Center)

Installer le package krb5-server, puis répéter toutes les opérations faites sur le serveur **kerb1**, seule l'ouverture du port 749 n'est pas nécessaire.

Pour aller plus vite, copier les fichiers **/etc/krb5.conf**, **/var/kerberos/krb5kdc/kdc.conf** et **/var/kerberos/krb5kdc/kadm5.acl** depuis le serveur kerb1 :

```
[root@kerb2 ~]# scp root@kerb1:/etc/krb5.conf /etc
root@kerb1's password:
krb5.conf                                     100%  638     0.6KB/s   00:00
[root@kerb2 ~]# scp root@kerb1:/var/kerberos/krb5kdc/kdc.conf /var/kerberos/krb5kdc/
root@kerb1's password:
kdc.conf                                      100%  414     0.4KB/s   00:00
[root@kerb2 ~]# scp rootifsic@kerb1:/var/kerberos/krb5kdc/kadm5.acl /var/kerberos/krb5kdc/
root@kerb1's password:
kadm5.acl                                     100%   26     0.0KB/s   00:00
[root@kerb2 ~]#
```

Et modifier la partie **realms** du fichier **/etc/krb5.conf** (remplacer **kerb1** par **kerb2**):

```
[realms]
 UNIV-RENNES1.FR = {
  kdc = kerb2.univ-rennes1.fr:88
  admin_server = kerb2.univ-rennes1.fr:749
  default_domain = univ-rennes1.fr
 }
```

Créer la base Kerberos, ajouter le premier utilisateur (**root/admin**), démarrer les services et vérifier le fonctionnement en affichant les *principals*.

Mise en place de la réplication

Sur le serveur maître, créer les clés des serveurs **kerb1** et **kerb2** et les exporter dans la *keytab* par défaut du serveur (**/etc/krb5.keytab**) :

```
[root@kerb1 ~]# kadmin -p root/admin
Authenticating as principal root/admin with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey host/kerb1.univ-rennes1.fr
WARNING: no policy specified for host/kerb1.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to no
policy
Principal "host/kerb1.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  addprinc -randkey host/kerb2.univ-rennes1.fr
WARNING: no policy specified for host/kerb2.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to no
policy
Principal "host/kerb2.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  ktadd host/kerb1.univ-rennes1.fr
Entry for principal host/kerb1.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc mode
with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/kerb1.univ-rennes1.fr with kvno 3, encryption type ArcFour with HMAC/md5
added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/kerb1.univ-rennes1.fr with kvno 3, encryption type DES with HMAC/sha1
added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/kerb1.univ-rennes1.fr with kvno 3, encryption type DES cbc mode with
RSA-MD5 added to keytab WRFILE:/etc/krb5.keytab.
kadmin:  ktadd host/kerb2.univ-rennes1.fr
Entry for principal host/kerb2.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc mode
with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/kerb2.univ-rennes1.fr with kvno 3, encryption type ArcFour with HMAC/md5
added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/kerb2.univ-rennes1.fr with kvno 3, encryption type DES with HMAC/sha1
added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/kerb2.univ-rennes1.fr with kvno 3, encryption type DES cbc mode with
RSA-MD5 added to keytab WRFILE:/etc/krb5.keytab.
kadmin:  exit
[root@kerb1 ~]#
```

Sur le serveur esclave, copier le fichier /etc/krb5.keytab :

```
[root@kerb2 ~]# scp root@kerb1.univ-rennes1.fr:/etc/krb5.keytab /etc
krb5.keytab                                   100%  634     0.6KB/s   00:00
[root@kerb2 ~]#
```

Sur le serveur esclave, éditer le fichier **`/var/kerberos/krb5kdc/kpropd.acl`** de la manière suivante :

```
host/kerb1.univ-rennes1.fr@UNIV-RENNES1.FR
host/kerb2.univ-rennes1.fr@UNIV-RENNES1.FR
```

Et démarrer le service **kpropd** :

```
[root@kerb2 ~]# chkconfig kprop on
[root@kerb2 ~]# service kprop start
Starting Kerberos 5 Propagation Server:                    [  OK  ]
[root@kerb2 ~]#
```

Sur le serveur maître, créer le script **/usr/local/bin/krb5prop.sh** :

```
[root@kerb1 ~]# cat > /usr/local/bin/krb5prop.sh
#!/bin/sh
/usr/kerberos/sbin/kdb5_util dump /var/kerberos/krb5kdc/slave_datatrans
/usr/kerberos/sbin/kprop -f /var/kerberos/krb5kdc/slave_datatrans kerb2.univ-rennes1.fr > /dev/
null
[root@kerb1 ~]# chmod 700 /usr/local/bin/krb5prop.sh
[root@kerb1 ~]#
```

Exécuter le script « à la main » :

```
[root@kerb1 ~]# /usr/kerberos/sbin/kdb5_util dump /var/kerberos/krb5kdc/slave_datatrans
[root@kerb1 ~]# /usr/kerberos/sbin/kprop -f /var/kerberos/krb5kdc/slave_datatrans
kerb2.univ-rennes1.fr
Database propagation to kerb2.univ-rennes1.fr: SUCCEEDED
[root@kerb1 ~]#
```

Pour vérifier la bonne propagation des *principals*, ajouter un *principal* fictif sur le serveur maître et propager vers le serveur esclave :

```
[root@kerb1 ~]# kadmin -p root/admin
Authenticating as principal root/admin with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc dummy
WARNING: no policy specified for dummy@UNIV-RENNES1.FR; defaulting to no policy
Enter password for principal "dummy@UNIV-RENNES1.FR":
Re-enter password for principal "dummy@UNIV-RENNES1.FR":
Principal "dummy@UNIV-RENNES1.FR" created.
kadmin:  exit
[root@kerb1 ~]# /usr/local/bin/krb5prop.sh
[root@kerb1 ~]#
```

Sur le serveur esclave, vérifier la présence du nouveau principal :

```
[root@kerb2 ~]# kadmin.local -q "listprincs"
Authenticating as principal rootifsic/admin@UNIV-RENNES1.FR with password.
K/M@UNIV-RENNES1.FR
dummy@UNIV-RENNES1.FR
host/kerb1.univ-rennes1.fr@UNIV-RENNES1.FR
host/kerb2.univ-rennes1.fr@UNIV-RENNES1.FR
kadmin/admin@UNIV-RENNES1.FR
kadmin/changepw@UNIV-RENNES1.FR
kadmin/history@UNIV-RENNES1.FR
kadmin/localhost.localdomain@UNIV-RENNES1.FR
krbtgt/UNIV-RENNES1.FR@UNIV-RENNES1.FR
root/admin@UNIV-RENNES1.FR
[root@kerb2 ~]#
```

Ne pas oublier de supprimer le principal fictif ensuite (**kadmin.local -q "delprinc dummy"** sur **kerb1**).

Modifier le fichier **/etc/crontab** pour faire en sorte que la synchronisation entre les deux KDCs soient effectuée de manière automatique toutes les 5 minutes (par exemple) :

```
*/5 * * * * /usr/local/bin/krb5prop.sh
```

Les deux serveurs **kerb1** et **kerb2** sont maintenant installés.

Gestion des principals

La gestion des *principals* peut se faire à distance à l'aide **kadmin** depuis une machine d'administration (de confiance).

Pour cela, depuis la machine d'administration, on génère un *principal* **manager/admin** et on exporte sa clé dans une *keytab* locale :

```
[root@admin ~]# kadmin -p root/admin
Authenticating as principal root/admin with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey manager/admin
WARNING: no policy specified for manager/admin@UNIV-RENNES1.FR; defaulting to no policy
Principal "manager/admin@UNIV-RENNES1.FR" created.
kadmin:  ktadd -k /etc/manager.keytab manager/admin
Entry for principal manager/admin with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1
added to keytab WRFILE:/etc/manager.keytab.
Entry for principal manager/admin with kvno 3, encryption type ArcFour with HMAC/md5 added to
keytab WRFILE:/etc/manager.keytab.
Entry for principal manager/admin with kvno 3, encryption type DES with HMAC/sha1 added to keytab
WRFILE:/etc/manager.keytab.
Entry for principal manager/admin with kvno 3, encryption type DES cbc mode with RSA-MD5 added to
keytab WRFILE:/etc/manager.keytab.
kadmin:  exit
[root@admin ~]#
```

On utilise ensuite la commande **kadmin -p manager/admin -k -t /etc/manager.keytab -q "commande_kadmin"** pour exécuter la commande **commande_kadmin**. Par exemple :

```
[root@admin ~]# kadmin -p manager/admin -k -t /etc/manager.keytab -q "listprincs"
Authenticating as principal manager/admin with keytab /etc/manager.keytab.
HTTP/cas-kerb.univ-rennes1.fr@UNIV-RENNES1.FR
K/M@UNIV-RENNES1.FR
cas/admin@UNIV-RENNES1.FR
host/cas-kerb.univ-rennes1.fr@UNIV-RENNES1.FR
host/clinux.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR
host/cwinxp.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR
host/kerb1.univ-rennes1.fr@UNIV-RENNES1.FR
host/kerb2.univ-rennes1.fr@UNIV-RENNES1.FR
kadmin/admin@UNIV-RENNES1.FR
kadmin/changepw@UNIV-RENNES1.FR
kadmin/history@UNIV-RENNES1.FR
kadmin/localhost.localdomain@UNIV-RENNES1.FR
krbtgt/UNIV-RENNES1.FR@UNIV-RENNES1.FR
manager/admin@UNIV-RENNES1.FR
paubry@UNIV-RENNES1.FR
root/admin@UNIV-RENNES1.FR
[root@admin ~]#
```

On pourra écrire un script **/usr/local/bin/kexec** pour exécuter plus facilement les commandes sous **kadmin** :

```
[root@admin ~]# cd /usr/local/bin
[root@admin bin]# cat > kexec
#!/bin/bash
kadmin -p manager/admin -k -t /etc/manager.keytab -q "$*"
[root@admin bin]# chown root.root kexec
[root@admin bin]# chmod 700 kexec
[root@admin bin]#
```

La récupération d'un principal dupont dans la base Kerberos pourra ainsi se faire par :

```
[root@admin bin]# kexec getprinc dupont
Authenticating as principal manager/admin with keytab /etc/manager.keytab.
Principal: dupont@UNIV-RENNES1.FR
Expiration date: [never]
Last password change: Wed Mar 10 12:23:31 CET 2010
Password expiration date: [none]
Maximum ticket life: 1 day 00:00:00
Maximum renewable life: 0 days 00:00:00
Last modified: Wed Mar 10 12:23:31 CET 2010 (cas/admin@UNIV-RENNES1.FR)
Last successful authentication: [never]
Last failed authentication: [never]
Failed password attempts: 0
Number of keys: 6
Key: vno 1, Triple DES cbc mode with HMAC/sha1, no salt
Key: vno 1, ArcFour with HMAC/md5, no salt
Key: vno 1, DES with HMAC/sha1, no salt
Key: vno 1, DES cbc mode with RSA-MD5, no salt
Key: vno 1, DES cbc mode with CRC-32, Version 4
Key: vno 1, DES cbc mode with CRC-32, AFS version 3
Attributes:
Policy: [none]
[root@admin bin]#
```

Installation du serveur CAS

Adaptation d'un serveur CAS existant
Cette page décrit l'installation de A à Z d'un nouveau serveur CAS qui permet :

- l'authentification transparente des utilisateurs par la transmission de tickets Kerberos
- l'alimentation automatique d'un royaume Kerberos à partir des utilisateurs LDAP lors de leur connexion

Il est facile pour un administrateur CAs d'extraire de cette documentation les éléments nécessaires à l'adaptation d'un serveur CAS existant pour lui donner les fonctionnalités voulues.

Nom du serveur	**cas-kerb.univ-rennes1.fr**
Système	RedHat Entreprise 5

Ouverture de ports	**ssh** (22 tcp) **https** (443)

Installation système

Installer les packages Tomcat (**yum install tomcat5**), Apache (**yum install httpd**), puis Maven :

```
[root@cas-kerb ~]# cd /usr/local
[root@cas-kerb local]# wget ftp:
//ftp.inria.fr/pub/Apache/maven/binaries/apache-maven-2.2.1-bin.tar.gz
--2010-03-09 10:35:41--  ftp:
//ftp.inria.fr/pub/Apache/maven/binaries/apache-maven-2.2.1-bin.tar.gz
=> `apache-maven-2.2.1-bin.tar.gz'
Resolving ftp.inria.fr... 192.93.2.32
Connecting to ftp.inria.fr|192.93.2.32|:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD /pub/Apache/maven/binaries ... done.
==> SIZE apache-maven-2.2.1-bin.tar.gz ... 2840961
==> PASV ... done.    ==> RETR apache-maven-2.2.1-bin.tar.gz ... done.
Length: 2840961 (2.7M)
100%[==============================>] 2,840,961   2.01M/s   in 1.3s
2010-03-09 10:35:43 (2.01 MB/s) - `apache-maven-2.2.1-bin.tar.gz' saved [2840961]
[root@cas-kerb local]# tar xf apache-maven-2.2.1-bin.tar.gz
[root@cas-kerb local]# ln -s apache-maven-2.2.1 maven2
[root@cas-kerb local]# cd /etc/profile.d
[root@cas-kerb profile.d]# cat > maven2.sh
export PATH=$PATH:/usr/local/maven2/bin
[root@cas-kerb profile.d]# chmod 644 maven2.sh
[root@cas-kerb profile.d]# . maven2.sh
[root@cas-kerb profile.d]#
```

Installation de CAS

Télécharger la dernière version de CAS depuis http://www.jasig.org/cas/download et décompresser :

```
[root@cas-kerb ~]# cd /usr/local
[root@cas-kerb local]# wget http://www.ja-sig.org/downloads/cas/cas-server-3.3.5-release.tar.gz
--2010-03-09 09:25:51--  http://www.ja-sig.org/downloads/cas/cas-server-3.3.5-release.tar.gz
Resolving www.ja-sig.org... 128.112.131.108
Connecting to www.ja-sig.org|128.112.131.108|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 14467126 (14M) [application/x-gzip]
Saving to: `cas-server-3.3.5-release.tar.gz'
100%[=======================>] 14,467,126   427K/s   in 15s
2010-03-09 09:26:07 (919 KB/s) - `cas-server-3.3.5-release.tar.gz' saved [14467126/14467126]
[root@cas-kerb local]# tar xf cas-server-3.3.5-release.tar.gz
[root@cas-kerb local]# cd cas-server-3.3.5
[root@cas-kerb cas-server-3.3.5]#
```

Pour diminuer les temps de compilation, on ajoute la propriété **skipTests** au *plugin* **maven-sunfire** dans le fichier **pom.xml** à la racine du projet :

```
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-surefire-plugin</artifactId>
  <configuration>
    <skipTests>true</skipTests>
    <includes>
      <include>**/*Tests.java</include>
    </includes>
    <excludes>
      <exclude>**/Abstract*.java</exclude>
    </excludes>
  </configuration>
</plugin>
```

Créer un répertoire **cas-server-rennes1** dans lequel seront stockées toutes les personnalisations et y ajouter le fichier **pom.xml** suivant :

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <parent>
    <groupId>org.jasig.cas</groupId>
    <artifactId>cas-server</artifactId>
    <version>3.3.5</version>
  </parent>
  <modelVersion>4.0.0</modelVersion>
    <groupId>fr.univrennes1.cas</groupId>
    <artifactId>cas-server-rennes1</artifactId>
    <version>3.3.5</version>
    <packaging>war</packaging>
    <name>University of Rennes 1 CAS webapp</name>
    <organization>
        <name>University of Rennes 1</name>
        <url>http://www.univ-rennes1.fr.fr</url>
    </organization>
    <description>The University of Rennes 1 customizations to the JA-SIG CAS server.</description>
    <dependencies>
        <dependency>
            <groupId>org.jasig.cas</groupId>
            <artifactId>cas-server-webapp</artifactId>
            <version>${project.version}</version>
            <type>war</type>
        </dependency>
        <dependency>
            <groupId>org.jasig.cas</groupId>
            <artifactId>cas-server-core</artifactId>
            <version>${project.version}</version>
        </dependency>
      </dependencies>
    <build>
        <finalName>cas</finalName>
        <plugins>
            <plugin>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>RELEASE</version>
                <configuration>
                    <source>1.5</source>
                    <target>1.5</target>
                </configuration>
            </plugin>
        </plugins>
    </build>
    <repositories>
        <repository>
            <id>jasig-repository</id>
            <name>JA-SIG Maven2 Repository</name>
            <url>http://developer.ja-sig.org/maven2</url>
        </repository>
    </repositories>
    <reporting>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-project-info-reports-plugin</artifactId>
            </plugin>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-javadoc-plugin</artifactId>
            </plugin>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-changelog-plugin</artifactId>
            </plugin>
        </plugins>
    </reporting>
</project>
```

Plutôt que modifier les fichiers de la distribution CAS, il est préférable de maintenir toutes modifications effectuées dans ce répertoire, ce qui facilite les mises à jour. Tous les fichiers trouvés dans ce répertoire écraseront les fichiers de la distribution, on adoptera donc exactement la

même hiérarchie de fichiers que celle de la distribution.

Installation basique

Copier le fichier **src/main/webapp/WEB-INF/classes/log4j.properties** de **cas-server-webapp** dans **src/main/webapp/WEB-INF/classes** et indiquer le chemin des logs :

```
log4j.appender.logfile.File=/var/log/tomcat5/cas.log
```

Générer le WAR, le copier dans Tomcat et redémarrer :

```
[root@cas cas-server-rennes1]#  mvn package install
[root@cas cas-server-rennes1]#  cp target/cas.war /var/lib/tomcat5/webapps/ROOT.war
[root@cas cas-server-rennes1]#  /etc/init.d/tomcat5 restart
```

Debug
Ajouter dans le fichier **src/main/webapp/WEB-INF/classes/log4j.properties** la ligne suivante :

```
log4j.logger.org.jasig.cas=DEBUG
```

Les logs se trouvent dans le répertoire **/var/log/tomcat5**.

La mise au point la plus difficile est celle de Kerberos (les logs en debug de **Krb5LoginModule** se trouvent dans **catalina.out**).

Test

Tester http://cas-kerb.univ-rennes1.fr:8080 (user = test, password = test).

Script de déploiement de CAS

Pour faciliter le déploiement du serveur CAS et le redémarrage de Tomcat, on pourra ajouter le script **/usr/local/cas-server-3.3.5/deploy-restart.sh** suivant :

```
[root@cas-kerb ~]# cd /usr/local/cas-server-3.3.5
[root@cas-kerb cas-server-3.3.5]# cat > deploy-restart.sh
#!/bin/bash
service tomcat5 stop
pushd /var/lib/tomcat5/webapps
rm -rf ROOT ROOT.war
popd
rm -rf /usr/share/tomcat5/work/_
pushd /var/log/tomcat5
rm -rf cas.log catalina.out
touch cas.log catalina.out
chown tomcat.tomcat cas.log catalina.out
popd
pushd /usr/local/cas-server-3.3.5
pushd cas-server-rennes1
mvn package install
cp target/cas.war /var/lib/tomcat5/webapps/ROOT.war
popd
popd
service tomcat5 start
[root@cas-kerb cas-server-3.3.5]# chmod 744 deploy-restart.sh
[root@cas-kerb cas-server-3.3.5]#
```

Ajout d'un frontal Apache

On va dans cette partie configurer un frontal Apache sur le port 80, qui va accéder au Tomcat du serveur CAS en AJP sur le port 8009.

⚠ Il n'est pas obligatoire de mettre un frontal Apache devant Tomcat, mais cela délègue le chiffrement à Apache au lieu de Tomcat et simplifie l'administration système (cette architecture est employée de manière générale sur les plateformes d'exploitation).

Configuration de Apache

Insérer dans **/etc/httpd/conf.d/proxy_ajp.conf** les lignes suivantes :

```
ProxyPass / ajp://cas-kerb.univ-rennes1.fr:8009/ min=0 max=100 smax=50 ttl=10
```

Configuration de Tomcat

S'assurer que le connecteur AJP sur le port 8009 n'est pas commenté et a bien le paramètres **tomcatAuthentication** positionné à **false** :

```
<Connector port="8009"
    debug="0"
    enableLookups="false"
    redirectPort="8443"
    protocol="AJP/1.3"
    tomcatAuthentication="false" />
```

Test

Après redémarrage de Apache et Tomcat, le serveur CAS doit désormais répondre sur l'URL https://cas-kerb.univ-rennes1.fr (sur le port 80 par défaut en HTTP).

Passage en HTTPS

Installer si nécessaire le package **mod_ssl** (**yum install mod_ssl**).

Installerle certificat x509 et la cléprivée du serveur dans les répertoires appropriés (/etc/pki/tls/

Configuration de Apache

Installer le certificat du serveur en éditant **/etc/httpd/conf.d/ssl.conf** et modifier les lignes suivantes dans le *virtual host* **_default_:443** :

```
#SSLCertificateFile /etc/pki/tls/certs/localhost.crt
SSLCertificateFile /etc/pki/tls/certs/cas-kerb.univ-rennes1.fr.pem
#SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
SSLCertificateKeyFile /etc/pki/tls/private/cas-kerb.univ-rennes1.fr.key
```

Test

Après redémarrage de Apache, le serveur CAS doit désormais répondre sur l'URL https://cas.ifsic.univ-rennes1.fr (sur le port 443 par défaut en HTTPS).

Suppression de l'accès en HTTP
Ne pas oublier de supprimer la ligne suivante de **/etc/httpd/conf/httpd.conf** :

```
Listen 80
```

Ajout de l'authentification LDAP

Configuration de CAS pour LDAP

Ajouter la dépendance vers le module **cas-server-support-ldap** dans le fichier **pom.xml** :

```
<dependency>
  <groupId>org.jasig.cas</groupId>
  <artifactId>cas-server-support-ldap</artifactId>
  <version>${project.version}</version>
</dependency>
```

Copier le fichier **src/main/webapp/WEB-INF/deployerConfigContext.xml** de **cas-server-webapp** dans **src/main/webapp/WEB-INF**, ajouter le bean suivant pour déclarer le contexte LDAP :

```
<bean id="contextSource" class="org.springframework.ldap.core.support.LdapContextSource">
  <property name="pooled" value="true"/>
  <property name="urls">
    <list>
      <value>ldap://ldapglobal.univ-rennes1.fr/</value>
    </list>
  </property>
  <property name="userDn" value=""/>
  <property name="password" value=""/>
  <property name="baseEnvironmentProperties">
    <map>
      <entry>
       <key>
         <value>java.naming.security.authentication</value>
       </key>
       <value>simple</value>
      </entry>
     </map>
  </property>
</bean>
```

puis changer le handler **SimpleTestUsernamePasswordAuthenticationHandler** par celui-ci :

```
<bean
   class="org.jasig.cas.adaptors.ldap.FastBindLdapAuthenticationHandler" >
     <property name="filter" value="uid=%u,ou=people,dc=univ-rennes1,dc=fr" />
     <property name="contextSource" ref="contextSource" />
</bean>
```

Test

Redéployer le serveur CAS, et tester l'authentification d'un utilisateur LDAP.

Ajout de l'authentification Kerberos

La documentation de référence est http://www.ja-sig.org/wiki/display/CASUM/SPNEGO .

Configuration de Kerberos

Editer le fichier **/etc/krb5.conf** pour intégrer le serveur au domaine Kerberos :

```
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 default_realm = UNIV-RENNES1.FR
 dns_lookup_realm = false
 dns_lookup_kdc = false
 ticket_lifetime = 24h
 forwardable = yes

[realms]
 UNIV-RENNES1.FR = {
  kdc = kerb1.univ-rennes1.fr:88
  admin_server = kerb1.univ-rennes1.fr:749
  default_domain = univ-rennes1.fr
 }

[domain_realm]
 .univ-rennes1.fr = UNIV-RENNES1.FR
 univ-rennes1.fr = UNIV-RENNES1.FR

[appdefaults]
 pam = {
   debug = false
   ticket_lifetime = 36000
   renew_lifetime = 36000
   forwardable = true
   krb4_convert = false
 }
```

Ajouter le *principal* **HTTP/cas-kerb.univ-rennes1.fr** (casse importante) au royaume et l'exporter dans le fichier **/etc/http.keytab** :

```
[root@cas-kerb ~]# kadmin
Authenticating as principal rootifsic/admin@UNIV-RENNES1.FR with password.
kadmin: Client not found in Kerberos database while initializing kadmin interface
[rootifsic@cas-kerb ~]# kadmin -p root/admin
Authenticating as principal root/admin with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey HTTP/cas-kerb.univ-rennes1.fr
WARNING: no policy specified for HTTP/cas-kerb.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to no
policy
Principal "HTTP/cas-kerb.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  ktadd -k /etc/http.keytab HTTP/cas-kerb.univ-rennes1.fr
Entry for principal HTTP/cas-kerb.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc mode
with HMAC/sha1 added to keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas-kerb.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas-kerb.univ-rennes1.fr with kvno 3, encryption type DES with HMAC/sha1
added to keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas-kerb.univ-rennes1.fr with kvno 3, encryption type DES cbc mode with
RSA-MD5 added to keytab WRFILE:/etc/http.keytab.
kadmin:  exit
[root@cas-kerb ~]#
```

Le fichier **/etc/http.keytab** sera utilisé par la librairie JCIFS, il doit être lisible par l'utilisateur **tomcat** :

```
[root@cas-kerb ~]# cd /etc
[root@cas-kerb etc]# chown root:tomcat http.keytab
[root@cas-kerb etc]# chmod 640 http.keytab
[root@cas-kerb etc]#
```

Debug
Ajouter dans le fichier **/etc/tomcat5/tomcat5.conf** les lignes suivantes :

```
JAVA_OPTS="$JAVA_OPTS -Dcom.ibm.security.jgss.debug=all"
JAVA_OPTS="$JAVA_OPTS -Dsun.security.jgss.debug=true"
JAVA_OPTS="$JAVA_OPTS -Dsun.security.krb5.debug=true"
JAVA_OPTS="$JAVA_OPTS -Djava.security.debug=logincontext,policy,scl,gssloginconfig
```

Les logs se trouvent dans **/var/log/tomcat5/catalina.out**.

Configuration de CAS

Ajouter le support du *handler* spnego

Editer le fichier **pom.xml** et ajouter la dépendance suivante (par exemple juste après la dépendance vers le module **cas-server-support-ldap**) :

```
<dependency>
     <groupId>org.jasig.cas</groupId>
     <artifactId>cas-server-support-spnego</artifactId>
     <version>${project.version}</version>
</dependency>
```

Modifier le login webflow

Editer le fichier **src/main/webapp/WEB-INF/login-webflow.xml** et ajouter l'état suivant juste avant l'état *viewLoginForm* :

```
<action-state id="startAuthenticate">
  <action bean="negociateSpnego" />
  <transition on="success" to="spnego" />
</action-state>

<action-state id="spnego">
  <action bean="spnego" />
  <transition on="success" to="sendTicketGrantingTicket" />
  <transition on="error" to="viewLoginForm" />
</action-state>
```

Dans ce même fichier, remplacer les références à *viewLoginForm* par *startAuthenticate* pour les deux *decision-state gatewayRequestCheck* et *renewRequestCheck* :

```
<decision-state id="gatewayRequestCheck">
  <if
    test="${externalContext.requestParameterMap['gateway'] != '' &amp;&amp;
externalContext.requestParameterMap['gateway'] != null &amp;&amp; flowScope.service != null}"
    then="redirect"
    else="startAuthenticate" />
</decision-state>
```

```
<decision-state id="renewRequestCheck">
  <if
    test="${externalContext.requestParameterMap['renew'] != '' &amp;&amp;
externalContext.requestParameterMap['renew'] != null}"
    then="startAuthenticate"
    else="generateServiceTicket" />
</decision-state>
```

Déclarer le bean implémentant le nouvel état du webflow en ajoutant les lignes suivantes dans le fichier **src/main/webapp/WEB-INF/cas-servlet.xml** (par exemple juste avant le bean **authenticationViaFormAction**) :

```
<bean
    id="negociateSpnego"
    class="org.jasig.cas.support.spnego.web.flow.SpnegoNegociateCredentialsAction" />

<bean
    id="spnego"
    class="org.jasig.cas.support.spnego.web.flow.SpnegoCredentialsAction">
 <property name="centralAuthenticationService" ref="centralAuthenticationService"/>
</bean>
```

Modifier le schéma d'authentification

Pour modifier le schéma d'authentification, éditer le fichier **src/main/webapp/WEB-INF/deployerConfigContext.xml** et modifier le bean **authenticationManager** en ajoutant :

- **PrincipalBearingCredentialsToPrincipalResolver** après les *resolvers* existants de **credentialsToPrincipalResolvers**
- **PrincipalBearingCredentialsAuthenticationHandler** avant les *handlers* existants de **authenticationHandlers**

```
<bean id="authenticationManager" class="org.jasig.cas.authentication.AuthenticationManagerImpl">
  <property name="credentialsToPrincipalResolvers">
    <list>
      <!--  ... the others credentialsToPrincipalResolvers ... -->
      <bean class=
"org.jasig.cas.support.spnego.authentication.principal.SpnegoCredentialsToPrincipalResolver" />
    </list>
  </property>
  <property name="authenticationHandlers">
    <list>
      <bean class=
"org.jasig.cas.support.spnego.authentication.handler.support.JCIFSSpnegoAuthenticationHandler">
 <property name="authentication">
   <bean class="jcifs.spnego.Authentication" />
 </property>
 <property name="principalWithDomainName" value="false" />
 <property name="NTLMallowed" value="false"/>
      </bean>
      <!--  ... the others authenticationHandlers... -->
    </list>
  </property>
</bean>
```

Le bean **authenticationManager** doit ainsi ressembler à :

```
<bean   id="authenticationManager"
        class="org.jasig.cas.authentication.AuthenticationManagerImpl">
    <property name="credentialsToPrincipalResolvers">
      <list>
        <bean class=
"org.jasig.cas.authentication.principal.UsernamePasswordCredentialsToPrincipalResolver" />
        <bean class=
"org.jasig.cas.authentication.principal.HttpBasedServiceCredentialsToPrincipalResolver" />
        <bean class=
"org.jasig.cas.support.spnego.authentication.principal.SpnegoCredentialsToPrincipalResolver" />
      </list>
    </property>
    <property name="authenticationHandlers">
      <list>
        <bean class=
"org.jasig.cas.support.spnego.authentication.handler.support.JCIFSSpnegoAuthenticationHandler">
   <property name="authentication">
     <bean class="jcifs.spnego.Authentication" />
   </property>
   <property name="principalWithDomainName" value="true" />
   <property name="NTLMallowed" value="false"/>
      </bean>
      <bean class=
"org.jasig.cas.authentication.handler.support.HttpBasedServiceCredentialsAuthenticationHandler"
             p:httpClient-ref="httpClient" />
 <bean class="org.jasig.cas.adaptors.ldap.FastBindLdapAuthenticationHandler" >
          <property name="filter" value="uid=%u,ou=people,dc=univ-rennes1,dc=fr" />
          <property name="contextSource" ref="contextSource" />
        </bean>
      </list>
    </property>
</bean>
```

Ajouter enfin le bean **jcifsConfig**, qui done les options de configuration de JCIFS :

```
<bean name="jcifsConfig" class=
"org.jasig.cas.support.spnego.authentication.handler.support.JCIFSConfig">
  <property
      name="jcifsServicePrincipal"
      value="HTTP/cas-kerb.univ-rennes1.fr" />
  <property
      name="kerberosDebug"
      value="true" />
  <property
      name="kerberosRealm"
      value="UNIV-RENNES1.FR" />
  <property
      name="kerberosKdc"
      value="kerb1.univ-rennes1.fr" />
  <property
      name="loginConf"
      value="/etc/jcifs/login.conf" />
</bean>
```

Configuration de JCIFS

La configuration de JCIFS se fait également dans le fichier login.conf pointé par le bean **jcifsConfig**. Créer ce fichier avec le contenu suivant :

```
com.sun.security.jgss.krb5.accept {
  com.sun.security.auth.module.Krb5LoginModule
  required
  debug=true
  storeKey=true
  useKeyTab=true
  keyTab="/etc/http.keytab"
  principal="HTTP/cas-kerb.univ-rennes1.fr";
};
```

Configuration de Tomcat

Il faut passer à la JVM qui exécute Tomcat l'option **-Djavax.security.auth.useSubjectCredsOnly=false**, par exemple en éditant le fichier **/etc/tomcat5/tomcat5.conf** et en ajoutant la ligne suivante :

```
JAVA_OPTS="$JAVA_OPTS -Djavax.security.auth.useSubjectCredsOnly=false"
```

Test

Un navigateur bien configuré et possédant des credentials Kerberos valides doit maintenant se connecter au serveur CAS sans aucune interaction....

Ajout de l'alimentation Kerberos

L'alimentation Kerberos désigne le processus qui permet d'alimenter un royaume Kerberos avec comptes utilisateurs lors de la connexion au serveur CAS. Elle peut être utilisée pour amorcer le remplissage du royaume à partir d'un annuaire LDAP, ce que nous montrons dans cet exemple.

Configuration de Kerberos

En premier lieu, déclarer le *principal* qui servira à la mise à jour du royaume Kerberos (ici **cas/admin**) et l'exporter dans le fichier /etc/cas-admin.keytab :

```
[root@cas-kerb ~]# kadmin -p root/admin
Authenticating as principal root/admin with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey cas/admin
WARNING: no policy specified for cas/admin@UNIV-RENNES1.FR; defaulting to no policy
Principal "cas/admin@UNIV-RENNES1.FR" created.
kadmin:  ktadd -k /etc/cas-admin.keytab cas/admin
Entry for principal cas/admin with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1
added to keytab WRFILE:/etc/cas-admin.keytab.
Entry for principal cas/admin with kvno 3, encryption type ArcFour with HMAC/md5 added to keytab
WRFILE:/etc/cas-admin.keytab.
Entry for principal cas/admin with kvno 3, encryption type DES with HMAC/sha1 added to keytab
WRFILE:/etc/cas-admin.keytab.
Entry for principal cas/admin with kvno 3, encryption type DES cbc mode with RSA-MD5 added to
keytab WRFILE:/etc/cas-admin.keytab.
kadmin:  exit
[root@cas-kerb ~]#
```

Le fichier **cas-admin.keytab** doit être lisible par l'utilisateur **tomcat** :

```
[root@cas-kerb ~]# cd /etc
[root@cas-kerb etc]# chown root:tomcat cas-admin.keytab
[root@cas-kerb etc]# chmod 640 cas-admin.keytab
[root@cas-kerb etc]#
```

Il est également nécessaire de modifier les permissions du fichier de *log* de **kadmin** sans quoi l'appel de **kadmin** par l'utilisateur **tomcat** provoquerait une erreur.

```
[root@cas-kerb ~]# cd /var/log
[root@cas-kerb log]# touch kadmind.log
[root@cas-kerb log]# chown root:tomcat kadmind.log
[root@cas-kerb log]# chmod 664 kadmind.log
[root@cas-kerb log]#
```

Intégration du module cas-server-integration-kerberosfeed

L'alimentation du royaume Kerberos se fait en utilisant le module **cas-server-integration-kerberosfeed** (téléchargeable ici), qui doit être installé au même niveau que les autres modules du projet puis compilé (lancer la commande **mvn package install** depuis le répertoire du module).

Il faut ensuite ajouter la dépendance de **cas-server-rennes1** vers **cas-server-integration-kerberosfeed** en ajoutant dans le fichier **pom.xml** du module **cas-server-rennes1** les lignes suivantes :

```
<dependency>
  <groupId>org.jasig.cas</groupId>
  <artifactId>cas-server-integration-kerberosfeed</artifactId>
  <version>${project.version}</version>
</dependency>
```

Configuration de CAS

On utilise la classe **KerberosFeedAuthenticationHandlerWrapper** pour enrober l'appel des *authenticationHandlers* pour lesquels on souhaite mettre en place une alimentation du royaume Kerberos. Les lignes suivantes de **src/main/webapp/WEB-INF/deployerConfigContext.xml**

```
<bean class="org.jasig.cas.adaptors.ldap.FastBindLdapAuthenticationHandler" >
  <property name="filter" value="uid=%u,ou=people,dc=univ-rennes1,dc=fr" />
  <property name="contextSource" ref="contextSource" />
</bean>
```

Seront ainsi remplacées par :

```
<bean class="org.esupportail.cas.adaptors.kerberosfeed.KerberosFeedAuthenticationHandlerWrapper" >
  <property name="authenticationHandler">
    <bean class="org.jasig.cas.adaptors.ldap.FastBindLdapAuthenticationHandler" >
      <property name="filter" value="uid=%u,ou=people,dc=univ-rennes1,dc=fr" />
      <property name="contextSource" ref="contextSource" />
    </bean>
  </property>
  <property name="config" ref="kerberosFeedConfig" />
  <property name="registry" ref="kerberosFeedRegistry" />
</bean>
```

Lorsqu'un *authenticationHandler* est enrobé de la sorte et que l'authentification du bean enrobé (ici **FastBindLdapAuthenticationHandler**) s'effectue avec succès, alors elle est suivie de l'ajout dans le royaume de l'utilisateur à l'aide d'une commande **kadmin**, en utilisant la configuration donnée par le bean **kerberosFeedConfig** :

```
<bean
    id="kerberosFeedConfig"
    class="org.esupportail.cas.adaptors.kerberosfeed.KerberosFeedConfig">
  <property name="kadminPath" value="/usr/kerberos/bin/kadmin" />
  <property name="realm" value="UNIV-RENNES1.FR" />
  <property name="principal" value="cas/admin" />
  <property name="useKeytab" value="true" />
  <property name="keytab" value="/etc/cas-admin.keytab" />
  <!-- property name="password" value="secret" /-->
  <property name="passwordAllowedChars"
    value=
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789&amp;~#{([-|`\\_^@)]=+}$%*!:/;.,?&gt;&
/>
</bean>
```

La propriété **password** n'est utilisée que lorsque **useKeytab** est positionnée à **false**, dans le cas ci-dessous on utilise la *keytab* générée précédemment.

L'ajout d'un utilisateur déjà présent dans le royaume Kerberos provoque une erreur de **kadmin**, qui n'est pas prise en compte (rien n'est fait dans ce cas, l'erreur n'est pas dommageable). Néanmoins, afin de ne pas rejouer l'ajout d'un utilisateur déjà présent dans le royaume, le bean **KerberosFeedAuthenticationHandlerWrapper** s'appuie sur un registre (propriété **registry**), dans lequel il mémorise les utilisateurs déjà ajoutés dans le royaume. Par défaut (lorsque la propriété **registry** n'est pas positionnée), la mémorisation est faite en mémoire (implémentation **InMemoryRegistryImpl**) et les informations sont perdues à chaque redémarrage du serveur CAS. On peut aussi utiliser un registre basé sur BerkeleyDb, dont les informations seront permanentes :

```
<bean
    id="kerberosFeedRegistry"
    class="org.esupportail.cas.adaptors.kerberosfeed.registry.BerkeleyDbRegistryImpl">
  <property name="dbPath" value="/tmp" />
</bean>
```

Les informations seront ici stockées dans le répertoire **/tmp**.

Test

Se connecter sur le serveur CAS avec un utilisateur de l'annuaire LDAP (par exemple **dupont**) et vérifier qu'il est bien ajouté dans la base Kerberos (**getprinc dupont** sous **kadmin** sur le KDC).

Intégration d'un client Windows XP

Configuration Kerberos

Sur le KDC, ajouter le principal correspondant au client :

```
[root@kerb1 ~]# kadmin -p root/admin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -pw password -e des-cbc-crc:normal host/cwinxp.ifsic.univ-rennes1.fr
WARNING: no policy specified for host/cwinxp.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to
no policy
Principal "host/cwinxp.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  exit
[root@kerb1 ~]#
```

Installer les outils d'administration supplémentaires (sur le CD d'installation, exécuter **SUPTOOLS.EXE** dans le répertoire **\SUPPORT\TOOLS**). Cela installe l'utilitaire **ksetup**.

Executer **cmd.exe** en tant qu'administrateur :

```
C:\>ksetup /addkdc UNIV-RENNES1.FR kerb1.univ-rennes1.fr
NOTE: /AddKdc requires a reboot to take effect on pre-SP1 Win2000 computers
C:\>ksetup /addkdc UNIV-RENNES1.FR kerb2.univ-rennes1.fr
NOTE: /AddKdc requires a reboot to take effect on pre-SP1 Win2000 computers
C:\>ksetup /addkpasswd UNIV-RENNES1.FR kerb1.univ-rennes1.fr
NOTE: /AddKpasswd requires a reboot to take effect on pre-SP1 Win2000 computers
C:\>ksetup /setcomputerpassword password
Setting computer password
NOTE: /SetComputerPassword requires a reboot to take effect.
C:\>ksetup /setrealm UNIV-RENNES1.FR
Setting Dns Domain
NOTE: /SetRealm requires a reboot to take effect
C:\>
```

Rebooter le client puis exécuter (toujours en tant qu'administrateur) :

```
C:\> ksetup /mapuser * *
```

Configuration réseau

Le client doit rejoindre le domaine AD (AD.UNIV-RENNES1.FR).

Intégration d'un client Linux

Authentification

Configurer l'authentification des utilisateurs avec **system-config-authentication** :

- User information : Enable LDAP support, LDAP search base DN : **ou=people,dc=univ-rennes1,dc=fr**, LDAP server : **ldap://ldapglobal.univ-rennes1.fr**
- Authentication : Enable Kerberos support, Realm : **UNIV-RENNES1.FR**, KDCs : **kerb1.univ-rennes1.fr:88**, Admin servers :

kerb1.univ-rennes1.fr:749

- sur les gentoo de l'IFSIC : il faut installer les paquets **mit-krb5** et **pam_krb5** et au final le fichier **/etc/pam.d/system-auth** doit avoir l'allure suivante :

```
auth       required     pam_env.so
auth       sufficient   pam_unix.so likeauth nullok
auth       sufficient   pam_krb5.so try_first_pass
auth       required     pam_deny.so

account     required      pam_unix.so broken_shadow
account     sufficient    pam_localuser.so
account     sufficient    pam_succeed_if.so uid < 500 quiet
account     [default=bad success=ok user_unknown=ignore] pam_krb5.so
account     required      pam_permit.so

password   required pam_cracklib.so difok=2 minlen=8 dcredit=2 ocredit=2 retry=3
password   sufficient pam_unix.so nullok md5 shadow use_authtok
password   required pam_deny.so

session     optional      pam_keyinit.so revoke
session     required      pam_limits.so
session     [success=1 default=ignore] pam_succeed_if.so service in crond quiet use_uid
session     required      pam_unix.so
session     optional      pam_krb5.so
```

Configuration Kerberos

Vérifier le fichier **/etc/krb5.conf** :

```
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 default_realm = UNIV-RENNES1.FR
 ticket_lifetime = 24h
 forwardable = yes

[realms]
 UNIV-RENNES1.FR = {
  kdc = kerb1.univ-rennes1.fr:88
  kdc = kerb2.univ-rennes1.fr:88
  admin_server = kerb1.univ-rennes1.fr:749
  default_domain = univ-rennes1.fr
 }

[domain_realm]
 .univ-rennes1.fr = UNIV-RENNES1.FR
 univ-rennes1.fr = UNIV-RENNES1.FR

[appdefaults]
 pam = {
  debug = false
  ticket_lifetime = 36000
  renew_lifetime = 36000
  forwardable = true
  krb4_convert = false
 }
```

Créer le principal du client sous **kadmin** (depuis le client) et générer stocker la clé localement (dans **/etc/krb5.keytab**) :

```
[root@clinux log]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey host/clinux.ifsic.univ-rennes1.fr
WARNING: no policy specified for host/clinux.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to
no policy
Principal "host/clinux.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  ktadd -k /etc/krb5.keytab host/clinux.ifsic.univ-rennes1.fr
Entry for principal host/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-256 CTS
mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-128 CTS
mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc
mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with
HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode
with RSA-MD5 added to keytab WRFILE:/etc/krb5.keytab.
kadmin:  exit
[root@clinux log]#
```

Le shell des utilisateurs est renvoyé par l'attribut **loginShell** de l'annuaire LDAP. Si le shell des utilisateurs n'est pas installé, il faut alors l'installer (par exemple **yum install csh**).

Vérification de l'authentification des utilisateurs :

```
[root@clinux ~]# su - paubry
su: warning: cannot change directory to /private/staff/y/ry/paubry: No such file or directory
id: cannot find name for group ID 20857
su: /bin/csh: No such file or directory
[paubry@clinux ~]$ exit
logout
[root@clinux log]#
```

Monter les homedirs des utilisateurs en ajoutant dans **/etc/fstab** les lignes suivantes :

```
sf1ifsic:/vol/vol1/private/student /private/student nfs
exec,nolock,dev,suid,rw,rsize=8192,wsize=8192 1 1
sf1ifsic:/vol/vol2/private/staff /private/staff nfs exec,nolock,dev,suid,rw,rsize=8192,wsize=8192
1 1
```

Créer puis monter les répertoires d'accueil :

```
[root@clinux ~]#
[root@clinux ~]# cd /
[root@clinux /]# mkdir -p /private/staff
[root@clinux /]# mkdir -p /private/student
[root@clinux /]# mount -a
[root@clinux /]# mount
[...]
sf1ifsic:/vol/vol1/private/student on /private/student type nfs
(rw,nolock,rsize=8192,wsize=8192,addr=148.60.4.42)
sf1ifsic:/vol/vol2/private/staff on /private/staff type nfs
(rw,nolock,rsize=8192,wsize=8192,addr=148.60.4.42)
[root@clinux /]#
```

Ajouter si nécessaire le groupe des utilisateurs en local en ajoutant dans le fichier **/etc/group** :

```
staff:x:20857:
```

Enfin vérifier à nouveau le login des utilisateurs :

```
[root@clinux ~]# su - paubry
[paubry@clinux ~]$ exit
logout
[root@clinux ~]#
```

Configuration Firefox

Pour que l'authentification Kerberos soit propagée par Firefox, une petite configuration est nécessaire comme indiqué sur cette page : Configuration de Firefox pour le SSO.

Configuration de Firefox pour le SSO

pour transmettre l'authentification Kerberos de Firefox à mod_auth_kerb :

- entrer **about:config** dans la barre de navigation
- entrer **nego** dans le filtre
- positionner les variables **network.negociate-auth.delegation-uris** et **network.negociate-auth.trusted-uris** à **ifsic.univ-rennes1.fr**.

Client Linux

Les préférences de Firefox sont situées dans le répertoire d'accueil de l'utilisateur dans le fichier .mozilla/firefox/83g4wyjr.default/prefs.js :

```
user_pref("network.negotiate-auth.delegation-uris", "ifsic.univ-rennes1.fr");
user_pref("network.negotiate-auth.trusted-uris", "ifsic.univ-rennes1.fr");
```

Client Windows

A compléter.

Configuration de Internet Explorer pour le SSO

Indiquer le domaine univ-rennes1.fr comme étant dans l'intranet (Outils, Options Internet, Sécurité, Intranet local, Sites..., Avancé..., Ajoute ce site site web à la zone, *.univ-rennes1.fr).

Indiquer qu'il faut passer l'authentification Kerberos dans l'intranet local (Outils, Options Internet, Sécurité, Intranet local, Personnaliser le niveau, Authentification utilisateur, Connexion, Connexion automatique uniquement dans la zone intranet).

Modification des clés de registre
A compléter : trouver les clés de registre associées.

Mise en place d'un serveur Samba

Nous montrons dans cette partie comment configurer Samba pour authentifier les utilisateurs avec Kerberos.

Configuration de Samba

Editer **/etc/samba/smb.conf** comme suit :

```
[global]
   kerberos method = system keytab
   realm = UNIV-RENNES1.FR
   security = ADS
   log file = /var/log/samba/log.%m
   hosts allow = 148.60.10. 127.

[tmp]
   comment = Temporary file space
   path = /tmp
   read only = no
   public = yes
```

Configuration Kerberos

Il faut à la fois déclarer le client (host) et le service SMB (cifs) dans le royaume Kerberos :

```
[root@server ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey host/server.ifsic.univ-rennes1.fr
WARNING: no policy specified for host/server.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to
no policy
Principal "host/server.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  ktadd host/server.ifsic.univ-rennes1.fr
Entry for principal host/server.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc
mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/server.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/server.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with
HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/server.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode
with RSA-MD5 added to keytab WRFILE:/etc/krb5.keytab.
kadmin:  addprinc -randkey cifs/server.ifsic.univ-rennes1.fr
WARNING: no policy specified for cifs/server.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to
no policy
Principal "cifs/server.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  ktadd cifs/server.ifsic.univ-rennes1.fr
Entry for principal cifs/server.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc
mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal cifs/server.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal cifs/server.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with
HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal cifs/server.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode
with RSA-MD5 added to keytab WRFILE:/etc/krb5.keytab.
kadmin:  exit
[root@server ~]#
```

Tests

Clients Windows

Connecter un lecteur réseau sur **\\server.ifsic.univ-rennes1.fr\tmp**.

Clients linux

mount -t cifs étant réservé à l'utilisateur root, on valide le passage de l'authentification Kerberos avec **smbclient** en utilisant l'option **-k** :

```
[paubry@clinux ~]$ smbclient //server.ifsic.univ-rennes1.fr/tmp -k
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 3.4.2-47.fc12]
smb: > ls
  .                                   D        0  Fri Jan 22 15:25:32 2010
  ..                                 DR        0  Fri Jan 15 15:15:44 2010
  [...]
  36048 blocks of size 2097152. 32054 blocks available
smb: > exit
[paubry@clinux ~]$
```

Mise en place d'un serveur NFS (v4-Kerberos)

Nous montrons dans cette partie comment configurer NFS (v4) pour authentifier les utilisateurs avec Kerberos.

Les tests sont fait sur la machine cas.ifsic.univ-rennes1.fr, sur laquelle on installe le serveur NFS (v4 par défaut).

La dernière partie montre al configuration d'un filer NetApp.

Configuration du serveur

Editer le fichier de configuration qui donne le mapping des utilisateurs pour tous les services basés sur RPC, dont NFS (**/etc/idmapd.conf**) :

```
Domain = univ-rennes1.fr
Local-Realms = UNIV-RENNES1.FR
```

Ajouter un principal pour le service NFS (nfs/cas.ifsic.univ-rennes1.fr) et l'ajouter au fichier /etc/krb5.keytab.

Préciser dans le fichier **/etc/exports** les répertoires à exporter :

```
/tmp      gss/krb5(sync,rw,fsid=0,no_subtree_check,anonuid=65534,anongid=65534)
```

Editer le fichier **/etc/sysconfig/nfs** et indiquer que l'on veut utiliser un NFS sécurisé :

```
SECURE_NFS="yes"
```

(re)Démarrer les services NFS et rpcidmapd.

Configuration du client

Ajouter dans la keytab du client le principal de root pour pouvoir faire les montages NFS :

```
[root@clinux ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey root/clinux.ifsic.univ-rennes1.fr
WARNING: no policy specified for root/clinux.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to
no policy
Principal "root/clinux.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  ktadd -k /etc/keytab root/clinux.ifsic.univ-rennes1.fr
Entry for principal root/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-256 CTS
mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/keytab.
Entry for principal root/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-128 CTS
mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/keytab.
Entry for principal root/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc
mode with HMAC/sha1 added to keytab WRFILE:/etc/keytab.
Entry for principal root/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/keytab.
Entry for principal root/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with
HMAC/sha1 added to keytab WRFILE:/etc/keytab.
Entry for principal root/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode
with RSA-MD5 added to keytab WRFILE:/etc/keytab.
kadmin:  exit
[root@clinux ~]#
```

Activer le module **rpcsec_gss_krb5** dans le noyau si nécessaire :

```
[root@clinux ~]# lsmod| grep rpc
auth_rpcgss            31232  1 nfs
sunrpc                158428  9 nfs,lockd,nfs_acl,auth_rpcgss
[root@clinux ~]# modprobe rpcsec_gss_krb5
WARNING: All config files need .conf: /etc/modprobe.d/anaconda, it will be ignored in a future
release.
[root@clinux ~]# lsmod| grep rpc
rpcsec_gss_krb5         8824  0
auth_rpcgss            31232  2 rpcsec_gss_krb5,nfs
sunrpc                158428  10 rpcsec_gss_krb5,nfs,lockd,nfs_acl,auth_rpcgss
[root@clinux ~]#
```

*Ajouter ici comment insérer le module à chaque redémarrage.

Editer le fichier **/etc/sysconfig/nfs** et indiquer que l'on veut utiliser un NFS sécurisé :

```
SECURE_NFS="yes"
```

Démarrer le démon **rpcgssd** :

```
[root@clinux ~]# /etc/init.d/rpcgssd status
rpc.gssd is stopped
[root@clinux ~]# chkconfig rpcgssd on
[root@clinux ~]# /etc/init.d/rpcgssd start
Starting RPC gssd: WARNING: All config files need .conf: /etc/modprobe.d/anaconda, it will be
ignored in a future release.
                                                           [  OK  ]
[root@clinux ~]# /etc/init.d/rpcgssd status
rpc.gssd (pid 29697) is running...
[root@clinux ~]#
```

Effectuer les mêmes modification de **/etc/idmapd.conf** que sur le serveur et redémarrer le démon rpcidmapd :

```
[root@clinux ~]# /etc/init.d/rpcidmapd restart
Stopping RPC idmapd:                                       [ OK ]
Starting RPC idmapd:                                       [ OK ]
[root@clinux ~]
```

Monter à la main les répertoires :

```
[root@clinux ~]# mount -t nfs4 -osec=krb5 cas.ifsic.univ-rennes1.fr:/ /mnt
[root@clinux ~]#
```

Pour un montage automatique des répertoires, modifiers le fichier **/etc/fstab** :

```
cas.ifsic.univ-rennes1.fr:/ /mnt nfs4 sec=krb5
```

Installation Gentoo
Installer le package **nfs-utils** avec l'option kerberos:

```
USE="kerberos" emerge nfs-utils
```

Configuration NFS v4 avec un filer NetApp

Créer le principal du service nfs (**nfs/netapp.univ-rennes1.fr**) en utilisant l'option **-e des_cbc_crc:normal** (le seul chiffrement compris par NetApp), l'exporter dans **Unix_krb5.keytab** (toujours avec l'option **-e des_cbc_crc:normal**), puis copier ce fichier dans la hiérarchie **/etc** du filer (après un montage NFS v3 par exemple ou un FTP).

Executer **nfs setup** sur le filer, en spécifiant que l'on s'appuie sur un KDC Unix.

Lors de l'ajout d'un partage, spécifier **krb5** dans le paramètre **SECURITY** (égal à **sys** par défaut).

Enfin, la récupération des identités des utilisateurs doit être configuré de la manière suivante :

```
ldap.base                    dc=univ-rennes1,dc=fr
ldap.base.passwd             ou=people,dc=univ-rennes1,dc=fr
ldap.enable                  on
ldap.minimum_bind_level      anonymous
ldap.port                    389
ldap.servers                 ldapglobal.univ-rennes1.fr
ldap.servers.preferred       ldapglobal.univ-rennes1.fr
ldap.usermap.attribute.unixaccount uid
ldap.usermap.attribute.windowsaccount uid
ldap.usermap.enable          on
```

Les principals **root/client.ifsic.univ-rennes1.fr** doivent également être créés (et exportés dans le **/etc/krb5.keytab** des clients) en utilisant l'option **-e des_cbc_crc:normal**.

Mise en place d'un serveur CUPS

Installation du serveur CUPS

Configuration Système

Ajouter dans le fichier **/etc/group** un groupe pour les administrateurs du serveur CUPS :

```
admin:x:19999:ayello,dagorn,diascorn,frlemass,paubry
```

Ces utilisateurs seront autorisés pour certaines opérations spéciales (par exemple l'ajout d'imprimantes, la destruction de jobs ne leur appartenant pas, ...).

Configuration de Kerberos

Deux *principals* sont nécessaires :

- **ipp/server.ifsic.univ-rennes1.fr** pour les impressions
- **HTTP/server.ifsic.univ-rennes1.fr** pour l'accès à l'interface web de CUPS

```
[root@server ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey ipp/server.ifsic.univ-rennes1.fr
WARNING: no policy specified for ipp/server.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to
no policy
Principal "ipp/server.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  ktadd ipp/server.ifsic.univ-rennes1.fr
Entry for principal ipp/server.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc
mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal ipp/server.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal ipp/server.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with
HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal ipp/server.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode
with RSA-MD5 added to keytab WRFILE:/etc/krb5.keytab.
kadmin:  addprinc -randkey HTTP/server.ifsic.univ-rennes1.fr
WARNING: no policy specified for HTTP/server.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to
no policy
Principal "HTTP/server.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  ktadd HTTP/server.ifsic.univ-rennes1.fr
Entry for principal HTTP/server.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc
mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal HTTP/server.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal HTTP/server.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with
HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal HTTP/server.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode
with RSA-MD5 added to keytab WRFILE:/etc/krb5.keytab.
kadmin:  exit
[root@server ~]#
```

Configuration de CUPS

La configuration de CUPS se fait dans le fichier **/etc/cups/cupds.conf** :

On précise le groupe des administrateurs du serveur :

```
#SystemGroup sys root
SystemGroup admin
```

On autorise l'administration distante :

```
#Listen localhost:631
Port 631
Listen /var/run/cups/cups.sock
```

On indique que l'authentification par défaut sera Kerberos :

```
DefaultAuthType Negotiate
```

On indique que la politique par défaut des imprimantes sera kerberos (cf plus bas) :

```
DefaultPolicy kerberos
```

On écrit enfin la politique kerberos en adaptant légèrement la politique prédéfinie authenticated :

```
<Policy kerberos>
  <Limit Create-Job Print-Job Print-URI>
    AuthType Default
    Require valid-user
    Order deny,allow
  </Limit>
  <Limit Send-Document Send-URI Hold-Job Release-Job Restart-Job Purge-Jobs Set-Job-Attributes
Create-Job-Subscription Renew-Subscription Cancel-Subscriptio\
n Get-Notifications Reprocess-Job Cancel-Current-Job Suspend-Current-Job Resume-Job CUPS-Move-Job
CUPS-Get-Document>
    AuthType Default
    Require user @OWNER @SYSTEM
    Order deny,allow
  </Limit>
  <Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class
CUPS-Set-Default>
    AuthType Default
    Require user @SYSTEM
    Order deny,allow
  </Limit>
  <Limit Pause-Printer Resume-Printer Enable-Printer Disable-Printer
Pause-Printer-After-Current-Job Hold-New-Jobs Release-Held-New-Jobs Deactivate-Printer \
Activate-Printer Restart-Printer Shutdown-Printer Startup-Printer Promote-Job Schedule-Job-After
CUPS-Accept-Jobs CUPS-Reject-Jobs>
    AuthType Default
    Require user @SYSTEM
    Order deny,allow
  </Limit>
  <Limit Cancel-Job CUPS-Authenticate-Job>
    AuthType Default
    Require user @OWNER @SYSTEM
    Order deny,allow
  </Limit>
  <Limit All>
    Order deny,allow
  </Limit>
</Policy>
```

NB : par rapport à la politique **authenticated**, on rajoute simplement la directive **Require valid-user** pour les opérations *Create-Job*, *Print-Job* et *Print-URI*, qui limitera l'impression aux utilisateurs authentifiés.

Debug

Pour obtenir plus d'informations, positionner la directive **LogLevel** à **debug** dans **/etc/cups/cupds.conf** :

```
LogLevel debug
```

Les logs se trouvent dans **/var/log/cups/error_log**.

Intégration des clients

Linux

Chaque client Linux a son propre serveur CUPS, qui ne fait que rediriger vers le serveur principal.

Pour cela, on indique simplement dans le fichier **/etc/cups/client.conf** vers quel serveur rediriger toutes les requêtes :

```
ServerName server.ifsic.univ-rennes1.fr
```

Windows

802-1x, Radius et Kerberos

802.1X est un protocole qui a pour but d'ouvrir l'accès au réseau en fonction d'une authentification des usagers ou des machines qui

essaient de s'y raccorder. Le processus d'authentification peut être varié et déporté vers un service d'authentification centralisé. De nombreux cas d'usage utilisent un serveur *freeRadius* pour l'authentification. Les clients **802.1X** (commutateurs, bornes Wi-Fi, ...) sont alors configurés pour interroger un serveur *freeRadius* qui gère différents scénarios d'authentification. Pour configurer un serveur *freeRadius* s'appuyant sur une base d'usagers Kerberos, on peut procéder de la manière suivante :

```
- compiler un freeRadius à partir de la distribution SRC
- un module  rlm_krb5 est alors produit
- dans le fichier radiusd.conf insérer ce qui suit dans la configuration des modules
 krb5 {
           keytab = /etc/krb5.keytab
           service_principal = radius/fqdn.du.serveur.radius      }
- toujours dans radiusd.conf dans la section authenticate ajouter
  Auth-Type Kerberos {
    krb5
  }
- la configuration du reste dépend du cas d'usage, ci-dessous un ajout effectué dans le fichier
users :

  tutu Auth-Type := kerberos
 Fall-Through = No
```

Il convient également de créer les principaux suivants :

- le host qui héberge le serveur radius (addprinc -randkey host/fqdn.du.serveur.radius)
- le service radius (addprinc -randkey radius/fqdn.du.serveur.radius)
- extraire le fichier keytab correspondant et l'installer comme indiqué dans le radiusd.conf

Si ce scénario permet d'intégrer une base kerberos dans le processus d'ouverture des accès au réseau, notons que la propagation des tickets ne s'effectue pas jusqu'aux machines connectées (à suivre ...).

Modification de l'application Sésame

L'application Sésame de l'université doit être modifiée pour permettre :

- l'ajout des utilisateurs dans le royaume Kerberos lors de l'activation des comptes
- la modification du mot de passe des utilisateurs dans le royaume Kerberos (en plus de celle dans l'annuaire LDAP)
- la suppression des utilisateurs dans le royaume Kerberos lors de la suppression des comptes

Cela nécessite le branchement dans l'application, au même niveau que les actions effectuées sur l'annuaire LDAP et l'Active Directory, de l'appel des procédures équivalentes pour la maintenance de la cohésion du Royaume Kerberos avec la base des utilisateurs du S.II.

La mise à jour du royaume se fait en utilisant la commande **kadmin**, qui permet d'interagir avec la base de données de kerberos. **kadmin** n'offrant pas d'API, l'appel se fait à l'aide d'un appel système, de la même manière que décrit dans Installation du serveur CAS.

Configuration de Kerberos

En premier lieu, déclarer le *principal* qui servira à la mise à jour du royaume Kerberos (ici **sesame/admin**) et l'exporter dans le fichier **/etc/sesame-admin.keytab** :

```
[root@sesame ~]# kadmin -p root/admin
Authenticating as principal root/admin with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey sesame/admin
WARNING: no policy specified for sesame/admin@UNIV-RENNES1.FR; defaulting to no policy
Principal "sesame/admin@UNIV-RENNES1.FR" created.
kadmin:  ktadd -k /etc/sesame-admin.keytab sesame/admin
Entry for principal sesame/admin with kvno 3, encryption type Triple DES cbc mode with HMAC/sha1
added to keytab WRFILE:/etc/cas-admin.keytab.
Entry for principal sesame/admin with kvno 3, encryption type ArcFour with HMAC/md5 added to
keytab WRFILE:/etc/cas-admin.keytab.
Entry for principal sesame/admin with kvno 3, encryption type DES with HMAC/sha1 added to keytab
WRFILE:/etc/cas-admin.keytab.
Entry for principal sesame/admin with kvno 3, encryption type DES cbc mode with RSA-MD5 added to
keytab WRFILE:/etc/cas-admin.keytab.
kadmin:  exit
[root@sesame ~]#
```

Le fichier **sesame-admin.keytab** doit être lisible par l'utilisateur **tomcat** :

```
[root@sesame ~]# cd /etc
[root@sesame etc]# chown root:tomcat sesame-admin.keytab
[root@sesame etc]# chmod 640 sesame-admin.keytab
[root@sesame etc]#
```

Il est également nécessaire de modifier les permissions du fichier de *log* de **kadmin** sans quoi l'appel de **kadmin** par l'utilisateur **tomcat** provoquerait une erreur.

```
[root@sesame ~]# cd /var/log
[root@sesame log]# touch kadmind.log
[root@sesame log]# chown root:tomcat kadmind.log
[root@sesame log]# chmod 664 kadmind.log
[root@sesame log]#
```

Exécution des requêtes kadmin

L'exécution d'une requête **kadmin** se fait de la manière suivante :

```
/usr/kerberos/sbin/kadmin -r UNIV-RENNES1.FR -p sesame/admin -k -t /etc/sesame-admin.keytab -q
<query>
```

Pour ajouter un utilisateur dans le royaume, on utilisera la requête (**password** est le mot de passe de l'utilisateur, **uid** son identifiant) :

```
add_principal -pw password uid
```

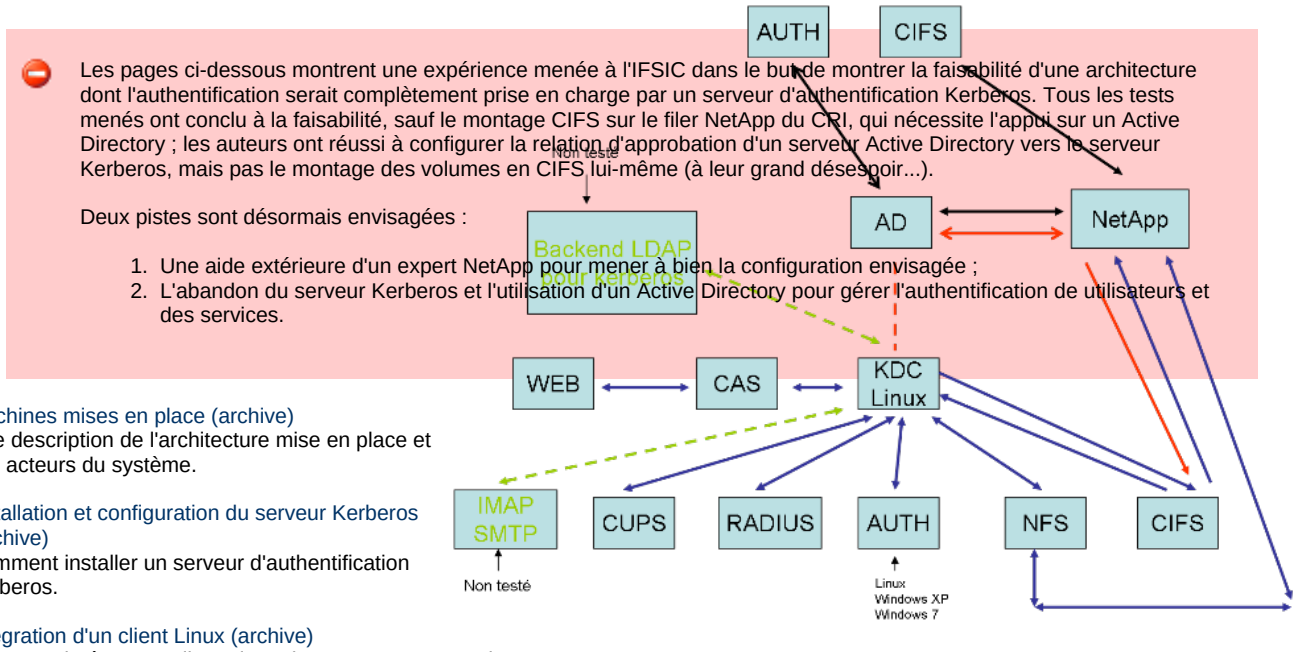
Pour modifier le mot de passe d'un utilisateur dans le royaume, on utilisera la requête :

```
change_password -pw password uid
```

Enfin pour supprimer un utilisateur du royaume, on utilisera la requête :

```
delete_principal -force uid
```

Mise en place d'une maquette à l'IFSIC (archive)



Les pages ci-dessous montrent une expérience menée à l'IFSIC dans le but de montrer la faisabilité d'une architecture dont l'authentification serait complètement prise en charge par un serveur d'authentification Kerberos. Tous les tests menés ont conclu à la faisabilité, sauf le montage CIFS sur le filer NetApp du CRI, qui nécessite l'appui sur un Active Directory ; les auteurs ont réussi à configurer la relation d'approbation d'un serveur Active Directory vers le serveur Kerberos, mais pas le montage des volumes en CIFS lui-même (à leur grand désespoir...).

Deux pistes sont désormais envisagées :

1. Une aide extérieure d'un expert NetApp pour mener à bien la configuration envisagée ;
2. L'abandon du serveur Kerberos et l'utilisation d'un Active Directory pour gérer l'authentification de utilisateurs et des services.

Machines mises en place (archive)
Une description de l'architecture mise en place et des acteurs du système.

Installation et configuration du serveur Kerberos (archive)
Comment installer un serveur d'authentification Kerberos.

Intégration d'un client Linux (archive)
Comment intégrer un client Linux dans un royaume Kerberos.

Passage de l'authentification Kerberos aux applications web (mod_auth_kerb)
Comment configurer Apache et mod_auth_kerb pour que l'authentification Kerberos passe du niveau système au niveau applicatif (web).

Installation et configuration du serveur CAS (archive)
Comment installer et configurer un serveur CAS pour qu'il authentifie les utilisateurs par Kerberos et LDAP.

Passage de l'authentification Kerberos à Samba (archive)
Comment mettre en place un serveur Samba et faire passer l'authentification Kerberos des clients lors du montage des volumes.

Passage de l'authentification Kerberos à NFS (v4)
Comment mettre en place un serveur NFS v4 et faire passer l'authentification Kerberos des clients lors du montage des volumes.

CASKERB:802.1X, radius et Kerberos802.1X, radius et Kerberos (archive)
Comment configurer un serveur freeRadius pour qu'il authentifie les usagers sur une base kerberos

Intégration d'un client Windows XP (archive)
Comment intégrer un client Windows XP dans un royaume Kerberos.

Intégration d'un client Windows 7 (archive)
Comment intégrer un client Windows 7 dans un royaume Kerberos.

Problèmes liés au clonage des stations de travail (archive)

Migration de l'authentification de LDAP à Kerberos (archive)
Comment envisager la migration de l'authentification des utilisateurs d'un annuaire LDAP à un serveur Kerberos.

Points restants à voir :

- CUPS et Kerberos
- NetApp et Kerberos
- AD et Kerberos

Passage de l'authentification Kerberos à NFS (v4) (archive)

Nous montrons dans cette partie comment configurer NFS (v4) pour authentifier les utilisateurs avec Kerberos.

Les tests sont fait sur la machine cas.ifsic.univ-rennes1.fr, sur laquelle on installe le serveur NFS (v4 par défaut).

La dernière partie montre al configuration d'un filer NetApp.

Configuration du serveur

Editer le fichier de configuration qui donne le mapping des utilisateurs pour tous les services basés sur RPC, dont NFS (**/etc/idmapd.conf**) :

```
Domain = univ-rennes1.fr
Local-Realms = UNIV-RENNES1.FR
```

Ajouter un principal pour le service NFS (nfs/cas.ifsic.univ-rennes1.fr) et l'ajouter au fichier /etc/krb5.keytab.

Préciser dans le fichier **/etc/exports** les répertoires à exporter :

```
/tmp      gss/krb5(sync,rw,fsid=0,no_subtree_check,anonuid=65534,anongid=65534)
```

Editer le fichier **/etc/sysconfig/nfs** et indiquer que l'on veut utiliser un NFS sécurisé :

```
SECURE_NFS="yes"
```

(re)Démarrer les services NFS et rpcidmapd.

Configuration du client

Ajouter dans la keytab du client le principal de root pour pouvoir faire les montages NFS :

```
[root@clinux ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey root/clinux.ifsic.univ-rennes1.fr
WARNING: no policy specified for root/clinux.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to
no policy
Principal "root/clinux.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  ktadd -k /etc/keytab root/clinux.ifsic.univ-rennes1.fr
Entry for principal root/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-256 CTS
mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/keytab.
Entry for principal root/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-128 CTS
mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/keytab.
Entry for principal root/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc
mode with HMAC/sha1 added to keytab WRFILE:/etc/keytab.
Entry for principal root/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/keytab.
Entry for principal root/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with
HMAC/sha1 added to keytab WRFILE:/etc/keytab.
Entry for principal root/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode
with RSA-MD5 added to keytab WRFILE:/etc/keytab.
kadmin:  exit
[root@clinux ~]#
```

Activer le module **rpcsec_gss_krb5** dans le noyau si nécessaire :

```
[root@clinux ~]# lsmod| grep rpc
auth_rpcgss            31232  1 nfs
sunrpc                158428  9 nfs,lockd,nfs_acl,auth_rpcgss
[root@clinux ~]# modprobe rpcsec_gss_krb5
WARNING: All config files need .conf: /etc/modprobe.d/anaconda, it will be ignored in a future
release.
[root@clinux ~]# lsmod| grep rpc
rpcsec_gss_krb5         8824  0
auth_rpcgss            31232  2 rpcsec_gss_krb5,nfs
sunrpc                158428  10 rpcsec_gss_krb5,nfs,lockd,nfs_acl,auth_rpcgss
[root@clinux ~]#
```

*Ajouter ici comment insérer le module à chaque redémarrage.

Editer le fichier **/etc/sysconfig/nfs** et indiquer que l'on veut utiliser un NFS sécurisé :

```
SECURE_NFS="yes"
```

Démarrer le démon **rpcgssd** :

```
[root@clinux ~]# /etc/init.d/rpcgssd status
rpc.gssd is stopped
[root@clinux ~]# chkconfig rpcgssd on
[root@clinux ~]# /etc/init.d/rpcgssd start
Starting RPC gssd: WARNING: All config files need .conf: /etc/modprobe.d/anaconda, it will be
ignored in a future release.
                                                           [  OK  ]
[root@clinux ~]# /etc/init.d/rpcgssd status
rpc.gssd (pid 29697) is running...
[root@clinux ~]#
```

Effectuer les mêmes modification de **/etc/idmapd.conf** que sur le serveur et redémarrer le démon rpcidmapd :

```
[root@clinux ~]# /etc/init.d/rpcidmapd restart
Stopping RPC idmapd:                                       [ OK ]
Starting RPC idmapd:                                       [ OK ]
[root@clinux ~]
```

Monter à la main les répertoires :

```
[root@clinux ~]# mount -t nfs4 -osec=krb5 cas.ifsic.univ-rennes1.fr:/ /mnt
[root@clinux ~]#
```

Pour un montage automatique des répertoires, modifiers le fichier **/etc/fstab** :

```
cas.ifsic.univ-rennes1.fr:/ /mnt nfs4 sec=krb5
```

Installation Gentoo
Installer le package **nfs-utils** avec l'option kerberos:

```
USE="kerberos" emerge nfs-utils
```

Configuration NFS v4 avec un filer NetApp

Créer le principal du service nfs (**nfs/netapp.univ-rennes1.fr**) en utilisant l'option **-e des_cbc_crc:normal** (le seul chiffrement compris par NetApp), l'exporter dans **Unix_krb5.keytab** (toujours avec l'option **-e des_cbc_crc:normal**), puis copier ce fichier dans la hiérarchie **/etc** du filer (après un montage NFS v3 par exemple ou un FTP).

Executer **nfs setup** sur le filer, en spécifiant que l'on s'appuie sur un KDC Unix.

Lors de l'ajout d'un partage, spécifier **krb5** dans le paramètre **SECURITY** (égal à **sys** par défaut).

Enfin, la récupération des identités des utilisateurs doit être configuré de la manière suivante :

```
ldap.base                    dc=univ-rennes1,dc=fr
ldap.base.passwd             ou=people,dc=univ-rennes1,dc=fr
ldap.enable                  on
ldap.minimum_bind_level      anonymous
ldap.port                    389
ldap.servers                 ldapglobal.univ-rennes1.fr
ldap.servers.preferred       ldapglobal.univ-rennes1.fr
ldap.usermap.attribute.unixaccount uid
ldap.usermap.attribute.windowsaccount uid
ldap.usermap.enable          on
```

Les principals **root/client.ifsic.univ-rennes1.fr** doivent également être créés (et exportés dans le **/etc/krb5.keytab** des clients) en utilisant l'option **-e des_cbc_crc:normal**.

Passage de l'authentification Kerberos à Samba (archive)

Nous montrons dans cette partie comment configurer Samba pour authentifier les utilisateurs avec Kerberos.

Les tests sont fait sur la machine cas.ifsic.univ-rennes1.fr, sur laquelle on installe Samba.

Configuration de Samba

Editer **/etc/samba/smb.conf** comme suit :

```
[global]
   use kerberos keytab = yes
   realm = UNIV-RENNES1.FR
   security = ADS
   log file = /var/log/samba/log.%m
   max log size = 50
   log level = 3
   hosts allow = 148.60.10. 127.

[tmp]
   comment = Temporary file space
   path = /tmp
   read only = no
   public = yes
```

Configuration Kerberos

Il faut à la fois déclarer le client (host, déjà fait précédemment) et le service SMB (cifs) :

```
[root@kerb ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin: addprinc -randkey cifs/cas.ifsic.univ-rennes1.fr
WARNING: no policy specified for cifs/cas.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to no
policy
Principal "cifs/cas.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin: ktadd -k /etc/krb5.keytab cifs/cas.ifsic.univ-rennes1.fr
Entry for principal cifs/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-256 CTS mode
with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal cifs/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-128 CTS mode
with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal cifs/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc
mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal cifs/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal cifs/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with HMAC/sha1
added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal cifs/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode with
RSA-MD5 added to keytab WRFILE:/etc/krb5.keytab.
kadmin: exit
You have new mail in /var/spool/mail/root
[root@kerb ~]#
```

Le fichier **/etc/krb5.conf** du serveur samba ne doit pas permettre l'usage du chiffrement 3DES. Le fichier **/etc/krb5.conf** du serveur kerberos doit être répliqué sur tous les principaux de services (HTTP, cifs, ...).

Tests

Clients Windows

Connecter un lecteur réseau sur **\\cas.ifsic.univ-rennes1.fr\tmp**.

Clients linux

mount -t cifs étant réservé à l'utilisateur root, on valide le passage de l'authentification Kerberos avec **smbclient** en utilisant l'option **-k** :

```
[paubry@clinux ~]$ smbclient //cas.ifsic.univ-rennes1.fr/tmp -k
Domain=[WORKGROUP] OS=[Unix] Server=[Samba 3.4.2-47.fc12]
smb: \> ls
  .                                   D        0  Fri Jan 22 15:25:32 2010
  ..                                 DR        0  Fri Jan 15 15:15:44 2010
  [...]
  36048 blocks of size 2097152. 32054 blocks available
smb: \> exit
[paubry@clinux ~]$
```

Passage de l'authentification Kerberos aux applications web (mod_auth_kerb) (archive)

Nous montrons dans cette partie comment configurer le couple Apache/mod_auth_kerb pour faire passer le SSO de l'authentification système jusqu'aux applications web.

Les tests sont fait sur la machine **cas.ifsic.univ-rennes1.fr**, sur laquelle on installe Apache et mod_auth_kerb. Cette partie ne sert pas pour les applications CASifiées, mais elle peut être envisagée pour des application dont on ne dispose pas des sources et qui seraient capable de s'appuyer sur une authentification externe de type **REMOTE_USER**.

Authentification

Il n'est pas nécessaire de configurer l'authentification des utilisateurs avec **system-config-authentication** sur ce serveur (les utilisateurs n'ont pas à se connecter sur le serveur CAS). Il faut néanmoins installer le fichier **/etc/krb5.conf** :

```
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 default_realm = UNIV-RENNES1.FR
 default_etypes = des3-hmac-sha1 des-cbc-crc
 default_tkt_enctypes = des3-hmac-sha1 des-cbc-crc
 default_tgs_enctypes = des3-hmac-sha1 des-cbc-crc
 permitted_enctypes = des3-hmac-sha1 des-cbc-crc rc4-hmac
 ticket_lifetime = 24h
 forwardable = yes

[realms]
 UNIV-RENNES1.FR = {
  kdc = kerb.ifsic.univ-rennes1.fr:88
  admin_server = kerb.ifsic.univ-rennes1.fr:749
  default_domain = univ-rennes1.fr
 }

[domain_realm]
 .univ-rennes1.fr = UNIV-RENNES1.FR
 univ-rennes1.fr = UNIV-RENNES1.FR

[appdefaults]
 pam = {
   debug = false
   ticket_lifetime = 36000
   renew_lifetime = 36000
   forwardable = true
   krb4_convert = false
 }
```

Installation basique Apache

Installer httpd (Apache) et mod_auth_kerb et démarrer Apache :

```
[root@cas ~]# chkconfig httpd on
[root@cas ~]# service httpd start
Starting httpd:                            [OK]
[root@cas ~]#
```

Création d'un script de test

Ecrire un simple script **test.php** dans le répertoire **/var/www/html/kerb** :

```
<?php
echo "<p>REMOTE_USER=[".$_SERVER['REMOTE_USER']."]</p>";
echo "<p>PHP_AUTH_USER=[".$_SERVER['PHP_AUTH_USER']."]</p>";
phpinfo();
?>
```

Debuggage

Modifier la directive **LogLevel** du fichier **/etc/httpd/conf/httpd.conf** :

```
LogLevel debug
```

Les logs sont dans le répertoire **/var/log/httpd**.

Test

Tester en accédant à http://cas.ifsic.univ-rennes1.fr/kerb/test.php .

Installation mod_auth_kerb

Configuration Kerberos

Déclarer le client Kerberos :

```
[root@cas ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey HTTP/cas.ifsic.univ-rennes1.fr
WARNING: no policy specified for HTTP/cas.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to no
policy
Principal "HTTP/cas.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  exit
[root@cas ~]#
```

Configuration mod_auth_kerb

Exporter la clé du client dans le fichier le fichier **/etc/httpd/conf/mod_auth_kerb.keytab** (ce fichier sera utilisé par mod_auth_kerb) :

```
[root@cas ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  ktadd -k /etc/httpd/conf/mod_auth_kerb.keytab HTTP/cas.ifsic.univ-rennes1.fr
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-256 CTS mode
with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/httpd/conf/mod_auth_kerb.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-128 CTS mode
with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/httpd/conf/mod_auth_kerb.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc
mode with HMAC/sha1 added to keytab WRFILE:/etc/httpd/conf/mod_auth_kerb.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/httpd/conf/mod_auth_kerb.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with HMAC/sha1
added to keytab WRFILE:/etc/httpd/conf/mod_auth_kerb.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode with
RSA-MD5 added to keytab WRFILE:/etc/httpd/conf/mod_auth_kerb.keytab.
kadmin:  exit
[root@cas ~]#
```

Modifier les permissions de la clé :

```
[root@cas ~]# chown apache /etc/httpd/conf/mod_auth_kerb.keytab
[root@cas ~]# chmod 640 /etc/httpd/conf/mod_auth_kerb.keytab
[root@cas ~]#
```

Protéger un répertoire par Kerberos en éditant **/etc/httpd/conf.d/auth_kerb.conf** :

```
<Location /kerb>
  #SSLRequireSSL
  AuthType KerberosV5
  AuthName "Kerberos Login"
  KrbMethodNegotiate On
  KrbMethodK5Passwd Off
  KrbAuthRealms UNIV-RENNES1.FR
  Krb5KeyTab /etc/httpd/conf/mod_auth_kerb.keytab
  require valid-user
</Location>
```

Test

Avant de tester, ne pas oublier d'ouvrir le port 80 entrant (**system-config-firewall**).

Tester en accédant http://cas.ifsic.univ-rennes1.fr/kerb/test.php. Le nom de l'utilisateur doit apparaître dans les variables **$_SERVER["REMOTE_USER"]** et **$_SERVER["PHP_AUTH_USER"]** (quelque chose comme **paubry@IFSIC.UNIV-RENNES1.FR**).

Note : il faut configurer les navigateurs clients pour que l'authentification kerberos soit transmise au serveur web.

- Configuration de Firefox (archive)
- Configuration de Internet Explorer (archive)

Il est possible de supprimer le domaine Kerberos de l'identifiant renvoyé par mod_auth_kerb en ajoutant l'option suivante à **mod_auth_kerb** :

```
KrbLocalUserMapping On
```

Configuration de Firefox (archive)

pour transmettre l'authentification Kerberos de Firefox à mod_auth_kerb :

- entrer **about:config** dans la barre de navigation
- entrer **nego** dans le filtre
- positionner les variables **network.negociate-auth.delegation-uris** et **network.negociate-auth.trusted-uris** à **ifsic.univ-rennes1.fr**.

Client Linux

Les préférences de Firefox sont situées dans le répertoire d'accueil de l'utilisateur dans le fichier .mozilla/firefox/83g4wyjr.default/prefs.js :

```
user_pref("network.negotiate-auth.delegation-uris", "ifsic.univ-rennes1.fr");
user_pref("network.negotiate-auth.trusted-uris", "ifsic.univ-rennes1.fr");
```

Client Windows

A compléter.

Configuration de Internet Explorer (archive)

Indiquer le domaine ifsic.univ-rennes1.fr comme étant dans l'intranet (Outils, Options Internet, Sécurité, Intranet local, Sites..., Avancé..., Ajoute ce site site web à la zone, *.fsic.univ-rennes1.fr).

Indiquer qu'il faut passer l'authentification Kerberos dans l'intranet local (Outils, Options Internet, Sécurité, Intranet local, Personnaliser le niveau, Authentification utilisateur, Connexion, Connexion automatique uniquement dans la zone intranet).

A compléter : trouver les clés de registre associées.

Installation et configuration du serveur CAS (archive)

Installer un JDK, Maven et Tomcat comme spécifié sur cette page : Installation Java, Maven et Tomcat

Installation d'un serveur CAS basique

Installation

Télécharger la dernière version de CAS depuis http://www.jasig.org/cas/download et décompresser :

```
[root@cas ~]# cd /usr/local
[root@cas local]# wget http://www.ja-sig.org/downloads/cas/cas-server-3.3.5-release.tar.gz
--2010-01-18 10:47:55--  http://www.ja-sig.org/downloads/cas/cas-server-3.3.5-release.tar.gz
Resolving www.ja-sig.org... 128.112.131.108
Connecting to www.ja-sig.org|128.112.131.108|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 14467126 (14M) [application/x-gzip]
Saving to: “cas-server-3.3.5-release.tar.gz”
100%[======================================================================>] 14,467,126   271K/s
in 84s
2010-01-18 10:49:19 (168 KB/s) - “cas-server-3.3.5-release.tar.gz” saved [14467126/14467126]
[root@cas local]# tar xf cas-server-3.3.5-release.tar.gz
[root@cas local]# cd cas-server-3.3.5
[root@cas cas-server-3.3.5]# cd cas-server-webapp
[root@cas cas-server-webapp]#
```

Modifier le fichier **src/main/webapp/WEB-INF/classes/log4j.properties** en indiquant le chemin des logs :

```
log4j.appender.logfile.File=/var/log/tomcat5/cas.log
```

Générer le WAR, le copier dans Tomcat et redémarrer :

```
[root@cas cas-server-webapp]#  mvn package install
[root@cas cas-server-webapp]#  cp target/cas.war /var/lib/tomcat5/webapps/ROOT.war
[root@cas cas-server-webapp]#  /etc/init.d/tomcat5 restart
```

Debuggage

Ajouter dans le fichier **src/main/webapp/WEB-INF/classes/log4j.properties** la ligne suivante :

```
log4j.logger.org.jasig.cas=DEBUG
```

Les logs se trouvent dans le répertoire **/var/log/tomcat5**.

La mise au point la plus difficile est celle de Kerberos. Les logs en debug de **Krb5LoginModule** se trouvent dans **catalina.out**.

Test

Désactiver si nécessaire le firewall pour le port 8080 (**system-config-firewall**) et tester http://cas.ifsic.univ-rennes1.fr:8080 (user = test, password = test).

Passage en HTTPS

Génération du keystore

Rapatrier *Jetty* (par exemple dans **/usr/local**) depuis http://static.roopindersingh.com/jetty-6.1.7.jar .

Copier les clés publique (**cas.ifsic.univ-rennes1.fr.pem**) et privée (**cas.ifsic.univ-rennes1.fr.key**) dans **/etc/pki/tls/private** pour générer le keystore dans **/etc/tomcat5** (en donnant comme mot de passe changeit) :

```
[root@cas private]# openssl pkcs12 -export \
>     -out cas.ifsic.univ-rennes1.fr.pkcs12 \
>     -in cas.ifsic.univ-rennes1.fr.pem \
>     -inkey cas.ifsic.univ-rennes1.fr.key
Enter Export Password:
Verifying - Enter Export Password:
[root@cas private]# java -cp /usr/local/jetty-6.1.7.jar org.mortbay.jetty.security.PKCS12Import \
>     cas.ifsic.univ-rennes1.fr.pkcs12 /etc/tomcat5/cas.ifsic.univ-rennes1.fr.keystore
Enter input keystore passphrase: changeit
Enter output keystore passphrase: changeit
Alias 0: 1
Adding key for alias 1
[root@cas private]#
```

Régler les permissions du keystore :

```
[root@cas private]# cd /etc/tomcat5/
[root@cas tomcat5]# chgrp tomcat cas.ifsic.univ-rennes1.fr.keystore
[root@cas tomcat5]# chmod 640 cas.ifsic.univ-rennes1.fr.keystore
[root@cas tomcat5]#
```

Configuration de Tomcat

Dans **/etc/tomcat5/server.xml**, commenter le connecteur HTTP sur le port 8080 et décommenter le connecteur HTTPS sur le port 8443 en ajoutant l'attribut :

```
keystoreFile="/etc/tomcat5/cas.ifsic.univ-rennes1.fr.keystore"
```

Test

Redémarrer Tomcat, désactiver si nécessaire le firewall pour le port 8443 et tester https://cas.ifsic.univ-rennes1.fr:8443 (user = test, password = test).

Ajout de l'authentification LDAP

Script de déploiement de CAS

Pour faciliter le déploiement du serveur CAS, on pourra ajouter le script **/usr/local/cas-server-3.3.5/deploy.sh** suivant :

```
#!/bin/bash
/etc/init.d/tomcat5 stop
pushd /usr/local/cas-server-3.3.5/cas-server-webapp
mvn package install && rm -f /var/lib/tomcat5/webapps/ROOT.war && cp target/cas.war /var
/lib/tomcat5/webapps/ROOT.war
popd
/etc/init.d/tomcat5 start
```

Configuration de CAS pour LDAP

Dans le fichier **src/main/webapp/WEB-INF/deployerConfigContext.xml**, ajouter le bean suivant pour déclarer le contexte LDAP :

```
<bean id="contextSource" class="org.springframework.ldap.core.support.LdapContextSource">
  <property name="pooled" value="true"/>
  <property name="urls">
    <list>
      <value>ldap://ldapglobal.univ-rennes1.fr/</value>
    </list>
  </property>
  <property name="userDn" value=""/>
  <property name="password" value=""/>
  <property name="baseEnvironmentProperties">
    <map>
      <entry>
       <key>
         <value>java.naming.security.authentication</value>
       </key>
       <value>simple</value>
      </entry>
     </map>
  </property>
</bean>
```

puis changer le handler **SimpleTestUsernamePasswordAuthenticationHandler** par celui-ci :

```
<bean
   class="org.jasig.cas.adaptors.ldap.FastBindLdapAuthenticationHandler" >
     <property name="filter" value="uid=%u,ou=people,dc=univ-rennes1,dc=fr" />
     <property name="contextSource" ref="contextSource" />
</bean>
```

Test

Redéployer le serveur CAS et tester l'authentification d'un utilisateur LDAP.

Ajout d'un frontal Apache

On va dans cette partie configurer un frontal Apache sur le port 443, qui va accéder au Tomcat du serveur CAS en AJP sur le port 8009.

> ⚠ Il n'est pas obligatoire de mettre un frontal Apache devant Tomcat, mais cela délègue le chiffrement à Apache au lieu de Tomcat et simplifie l'admisnitration système (cette architecture est employée de manière générale sur les plateformes d'exploitation).

Configuration de Apache

Installer le certificat du serveur en éditant **/etc/httpd/conf.d/ssl.conf** et modifier les lignes suivantes dans le *virtual host* **_default_:443** :

```
#SSLCertificateFile /etc/pki/tls/certs/localhost.crt
SSLCertificateFile /etc/pki/tls/private/cas.ifsic.univ-rennes1.fr.pem
#SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
SSLCertificateKeyFile /etc/pki/tls/private/cas.ifsic.univ-rennes1.fr.key
```

Ajouter la ligne suivante pour indiquer à Apache de passer les requêtes à Tomcat :

```
ProxyPass / ajp://cas.ifsic.univ-rennes1.fr:8009/ min=0 max=100 smax=50 ttl=10 route=ori-indexing
```

Configuration de Tomcat

S'assurer que le connecteur AJP sur le port 8009 est décommenté et a bien le paramètres **tomcatAuthentication** positionné à **false** :

```
<Connector port="8009"
    debug="0"
    enableLookups="false"
    redirectPort="8443"
    protocol="AJP/1.3"
    tomcatAuthentication="false" />
```

Le connecteur HTTPS sur le port 8443 peut être commenté.

Test

Le serveur CAS doit désormais répondre sur l'URL https://cas.ifsic.univ-rennes1.fr (sur le port 443 par défaut en HTTPS).

Ajout de l'authentification Kerberos

La documentation de référence est http://www.ja-sig.org/wiki/display/CASUM/SPNEGO .

Configuration de Kerberos

Procéder comme vu précédemment pour générer le fichier /etc/http.keytab qui sera utilisé ultérieurement par la librairie JCIFS :

```
[root@cas ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  ktadd -k /etc/http.keytab HTTP/cas.ifsic.univ-rennes1.fr
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-256 CTS mode
with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-128 CTS mode
with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc
mode with HMAC/sha1 added to keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with HMAC/sha1
added to keytab WRFILE:/etc/http.keytab.
Entry for principal HTTP/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode with
RSA-MD5 added to keytab WRFILE:/etc/http.keytab.
kadmin:  exit
[root@cas ~]
```

Configuration de CAS

Ajouter le support du handler spnego

Editer le fichier **<cas-home>/cas-server-webapp/pom.xml** et ajouter la dépendance suivante (par exemple juste après la dépendance vers le module **cas-server-support-ldap**) :

```
<dependency>
     <groupId>${project.groupId}</groupId>
     <artifactId>cas-server-support-spnego</artifactId>
     <version>${project.version}</version>
</dependency>
```

Modifier le login webflow

Editer le fichier **<cas-home>/cas-server-webapp/src/main/webapp/WEB-INF/login-webflow.xml** et ajouter l'état suivant juste avant l'état *viewLoginForm* :

```
<action-state id="startAuthenticate">
  <action bean="negociateSpnego" />
  <transition on="success" to="spnego" />
</action-state>

<action-state id="spnego">
  <action bean="spnego" />
  <transition on="success" to="sendTicketGrantingTicket" />
  <transition on="error" to="viewLoginForm" />
</action-state>
```

Dans ce même fichier, remplacer les références à *viewLoginForm* par *remoteAuthenticate* pour les trois decision-state *gatewayRequestCheck* et *renewRequestCheck* :

```
<decision-state id="gatewayRequestCheck">
  <if
    test="${externalContext.requestParameterMap['gateway'] != '' &amp;&amp;
externalContext.requestParameterMap['gateway'] != null &amp;&amp; flowScope.service != null}"
    then="redirect"
    else="startAuthenticate" />
</decision-state>
```

```
<decision-state id="renewRequestCheck">
  <if
    test="${externalContext.requestParameterMap['renew'] != '' &amp;&amp;
externalContext.requestParameterMap['renew'] != null}"
    then="startAuthenticate"
    else="generateServiceTicket" />
</decision-state>
```

Déclarer le bean implémentant le nouvel état du webflow en ajoutant les lignes suivantes dans le fichier **<cas-home>/cas-server-webapp/src/main/webapp/WEB-INF/cas-servlet.xml** (par exemple juste avant le bean **authenticationViaFormAction**) :

```
<bean
    id="negociateSpnego"
    class="org.jasig.cas.support.spnego.web.flow.SpnegoNegociateCredentialsAction" />

<bean
    id="spnego"
    class="org.jasig.cas.support.spnego.web.flow.SpnegoCredentialsAction">
 <property name="centralAuthenticationService" ref="centralAuthenticationService"/>
</bean>
```

Modifier le schéma d'authentification

Pour modifier le schéma d'authentification, éditer le fichier **<cas-home>/cas-server-webapp/src/main/webapp/WEB-INF/deployerConfigContext.xml** et modifier le bean **authenticationManager** en ajoutant :

- **PrincipalBearingCredentialsToPrincipalResolver** après les *resolvers* existants de **credentialsToPrincipalResolvers**
- **PrincipalBearingCredentialsAuthenticationHandler** avant les *handlers* existants de **authenticationHandlers**

```
<bean id="authenticationManager" class="org.jasig.cas.authentication.AuthenticationManagerImpl">
  <property name="credentialsToPrincipalResolvers">
    <list>
      <!--  ... the others credentialsToPrincipalResolvers ... -->
      <bean class=
"org.jasig.cas.support.spnego.authentication.principal.SpnegoCredentialsToPrincipalResolver" />
    </list>
  </property>
  <property name="authenticationHandlers">
    <list>
      <bean class=
"org.jasig.cas.support.spnego.authentication.handler.support.JCIFSSpnegoAuthenticationHandler">
 <property name="authentication">
   <bean class="jcifs.spnego.Authentication" />
 </property>
 <property name="principalWithDomainName" value="true" />
 <property name="NTLMallowed" value="false"/>
      </bean>
      <!--  ... the others authenticationHandlers... -->
    </list>
  </property>
</bean>
```

Le bean **authenticationManager** doit ainsi ressembler à :

```
<bean   id="authenticationManager"
        class="org.jasig.cas.authentication.AuthenticationManagerImpl">
    <property name="credentialsToPrincipalResolvers">
      <list>
        <bean class=
"org.jasig.cas.authentication.principal.UsernamePasswordCredentialsToPrincipalResolver" />
        <bean class=
"org.jasig.cas.authentication.principal.HttpBasedServiceCredentialsToPrincipalResolver" />
        <bean class=
"org.jasig.cas.support.spnego.authentication.principal.SpnegoCredentialsToPrincipalResolver" />
      </list>
    </property>
    <property name="authenticationHandlers">
      <list>
        <bean class=
"org.jasig.cas.support.spnego.authentication.handler.support.JCIFSSpnegoAuthenticationHandler">
   <property name="authentication">
     <bean class="jcifs.spnego.Authentication" />
   </property>
   <property name="principalWithDomainName" value="true" />
   <property name="NTLMallowed" value="false"/>
      </bean>
      <bean class=
"org.jasig.cas.authentication.handler.support.HttpBasedServiceCredentialsAuthenticationHandler"
              p:httpClient-ref="httpClient" />
 <bean class="org.jasig.cas.adaptors.ldap.FastBindLdapAuthenticationHandler" >
          <property name="filter" value="uid=%u,ou=people,dc=univ-rennes1,dc=fr" />
          <property name="contextSource" ref="contextSource" />
        </bean>
      </list>
    </property>
</bean>
```

Ajouter enfin le bean **jcifsConfig**, qui done les options de configuration de JCIFS :

```
<bean name="jcifsConfig" class=
"org.jasig.cas.support.spnego.authentication.handler.support.JCIFSConfig">
  <property
      name="jcifsServicePrincipal"
      value="HTTP/cas.ifsic.univ-rennes1.fr" />
  <property
      name="kerberosDebug"
      value="true" />
  <property
      name="kerberosRealm"
      value="UNIV-RENNES1.FR" />
  <property
      name="kerberosKdc"
      value="kerb.ifsic.univ-rennes1.fr" />
  <property
      name="loginConf"
      value="/usr/local/cas-server-3.3.5/cas-server-webapp/src/main/webapp/WEB-INF/login.conf" />
```

Configuration de JCIFS

La configuration de JCIFS se fait également dans le fichier login.conf pointé par le bean **jcifsConfig**. Créer ce fichier avec le contenu suivant :

```
jcifs.spnego.initiate {
  com.sun.security.auth.module.Krb5LoginModule
  required
  useKeyTab=true
  keyTab="/etc/http.keytab"
};
jcifs.spnego.accept {
  com.sun.security.auth.module.Krb5LoginModule
  required
  useKeyTab=true
  keyTab="/etc/http.keytab"
};
```


On peut également ajouter l'option **debug=true** pour obtenir des informations dans **catalina.out**.

Configuration de Tomcat

Il faut passer à la JVM qui exécute Tomcat l'option **-Djavax.security.auth.useSubjectCredsOnly=false**, par exemple en éditant le fichier **/etc/tomcat5/tomcat5.conf** et en ajoutant la ligne suivante :

```
JAVA_OPTS="$JAVA_OPTS -Djavax.security.auth.useSubjectCredsOnly=false"
```

Test

Un navigateur bien configuré et possédant des credentials Kerberos valides doit maintenant se connecter au serveur CAS sans aucune interaction....

Installation Java, Maven et Tomcat

Télécharger le dernier JDK depuis http://java.sun.com/javase/downloads/index.jsp , puis exécuter :

```
[root@cas Download]# chmod +x jdk-6u18-linux-i586-rpm.bin
[root@cas Download]# chmod +x jdk-6u18-linux-i586-rpm.bin
[...]
Done.
[root@cas Download]#
```

Ajouter le fichier **/etc/profile.d/java.sh** contenant les lignes suivantes :

```
export JAVA_HOME=/usr/java/default
export PATH=$JAVA_HOME/bin:$PATH
```

Vérifier l'installation :

```
[root@cas ~]# java -version
java version "1.6.0_18"
Java(TM) SE Runtime Environment (build 1.6.0_18-b07)
Java HotSpot(TM) Client VM (build 16.0-b13, mixed mode, sharing)
[root@cas ~]#
```

Installer *Maven* et *Tomcat* :

```
[root@cas ~]# yum install maven2 tomcat5
```

Version de Maven
Si jamais la version de Maven est avant 2.0.9, faire une installation manuelle depuis http://maven.apache.org/download.html

```
[root@cas ~]# mvn --version
/usr/java/default
Maven version: 2.0.8
Java version: 1.6.0_18
OS name: "linux" version: "2.6.31.9-174.fc12.i686" arch: "i386" Family: "unix"
[root@cas ~]# yum remove maven2
[...]
Complete!
[root@cas ~]# cd /usr/local
[root@cas local]# wget [ftp://ftp.inria.fr/pub/Apache/maven/binaries/apache-maven-2.2.1-bin.
--2010-01-18 11:14:08-- [ftp://ftp.inria.fr/pub/Apache/maven/binaries/apache-maven-2.2.1-bir
=> “apache-maven-2.2.1-bin.tar.gz”
Resolving ftp.inria.fr... 192.93.2.32
Connecting to ftp.inria.fr\|192.93.2.32\|:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD (1) /pub/Apache/maven/binaries ... done.
==> SIZE apache-maven-2.2.1-bin.tar.gz ... 2840961
==> PASV ... done.    ==> RETR apache-maven-2.2.1-bin.tar.gz ... done.
Length: 2840961 (2.7M) (unauthoritative)
100%[=========================================================================================
987K/s   in 2.8s
2010-01-18 11:14:12 (987 KB/s) - “apache-maven-2.2.1-bin.tar.gz” saved [2840961]
[root@cas local]# tar xf apache-maven-2.2.1-bin.tar.gz
[root@cas local]# ln -s apache-maven-2.2.1 maven2
[root@cas local]#
```

Ajouter Maven au PATH dans **/etc/profile.d/java.sh** :

```
export JAVA_HOME=/usr/java/default
export MVN_HOME=/usr/local/maven2
export PATH=$JAVA_HOME/bin:$PATH:$MVN_HOME/bin
```

Puis vérifier que Maven 2.2 est bien installé :

```
[root@cas local]# mvn -version
Apache Maven 2.2.1 (r801777; 2009-08-06 21:16:01+0200)
Java version: 1.6.0_18
Java home: /usr/java/jdk1.6.0_18/jre
Default locale: en_US, platform encoding: UTF-8
OS name: "linux" version: "2.6.31.9-174.fc12.i686" arch: "i386" Family: "unix"
[root@cas local]#
```

Installation et configuration du serveur Kerberos (archive)

Installation système

Boot sur CD Fedora 10 puis upgrade vers Fedora 12.

- FQDN : kerb.ifsic.univ-rennes1.fr
- IP : 148.60.10.50

Packages additionnels installés :

- Servers -> Network servers -> kerb5-server

Configuration SELinux

Pour éviter de se prendre la tête lors de l'utilisation des utilitaires de configuration modifiant les fichiers système, on passe SELinux en mode permissif par la commande :

```
[root@kerb ~] setenforce 0
```

On peut aussi le faire pour les sessions suivantes (après reboot de la machine) en ajoutant la ligne :

```
SELINUX=disabled
```

dans le fichier **/etc/selinux/config** (ou bien simplement **SELINUX=permissive**).

Configuration Kerberos

Modification de quelques fichiers de configuration pour créer le royaume UNIV-RENNES1.FR.

/etc/krb5.conf

```
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 default_realm = UNIV-RENNES1.FR
 default_etypes = des3-hmac-sha1 des-cbc-crc
 default_tkt_enctypes = des3-hmac-sha1 des-cbc-crc
 default_tgs_enctypes = des3-hmac-sha1 des-cbc-crc
 permitted_enctypes = des3-hmac-sha1 des-cbc-crc rc4-hmac
 ticket_lifetime = 24h
 forwardable = yes

[realms]
 UNIV-RENNES1.FR = {
  kdc = kerb.ifsic.univ-rennes1.fr:88
  admin_server = kerb.ifsic.univ-rennes1.fr:749
  default_domain = univ-rennes1.fr
 }

[domain_realm]
 .univ-rennes1.fr = UNIV-RENNES1.FR
 univ-rennes1.fr = UNIV-RENNES1.FR

[appdefaults]
 pam = {
   debug = false
   ticket_lifetime = 36000
   renew_lifetime = 36000
   forwardable = true
   krb4_convert = false
 }
```

/var/kerberos/krb5kdc/kdc.conf

```
[kdcdefaults]
 v4_mode = nopreauth
 kdc_ports = 88,750
 kdc_tcp_ports = 88

[realms]
 UNIV-RENNES1.FR = {
  #master_key_type = aes256-cts
  acl_file = /var/kerberos/krb5kdc/kadm5.acl
  dict_file = /usr/share/dict/words
  admin_keytab = /var/kerberos/krb5kdc/kadm5.keytab
  supported_enctypes = aes256-cts:normal aes128-cts:normal des3-hmac-sha1:normal
arcfour-hmac:normal des-hmac-sha1:normal des-cbc-md5:normal des-cbc-crc:normal des-cbc-crc:v4
des-cbc-crc:afs3 rc4-hmac:normal
 }
```

/var/kerberos/krb5kdc/kadm5.acl

```
*/admin@UNIV-RENNES1.FR *
```

/etc/gssapi_mech.conf

En 64 bits seulement :

```
# library                               initialization function
# ================================      =========================
# The MIT K5 gssapi library, use special function for initialization.
libgssapi_krb5.so                       mechglue_internal_krb5_init
```

Création de la base Kerberos :

```
[root@kerb ~]# kdb5_util create -s
Loading random data
Initializing database '/var/kerberos/krb5kdc/principal' for realm 'UNIV-RENNES1.FR',
master key name 'K/M@UNIV-RENNES1.FR'
You will be prompted for the database Master Password.
It is important that you NOT FORGET this password.
Enter KDC database master key:
Re-enter KDC database master key to verify:
[root@kerb ~]#
```

Ajout du premier utilisateur (root) :

```
[root@kerb ~]# kadmin.local -q "addprinc root/admin"
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
WARNING: no policy specified for root/admin@UNIV-RENNES1.FR; defaulting to no policy
Enter password for principal "root/admin@UNIV-RENNES1.FR":
Re-enter password for principal "root/admin@UNIV-RENNES1.FR":
Principal "root/admin@UNIV-RENNES1.FR" created.
[root@kerb ~]#
```

Démarrage des services :

```
[root@kerb ~]# chkconfig kadmin on
[root@kerb ~]# service kadmin start
Starting Kerberos 5 Admin Server:                          [  OK  ]
[root@kerb ~]# chkconfig krb5kdc on
[root@kerb ~]# service krb5kdc start
Starting Kerberos 5 KDC:                                   [  OK  ]
[root@kerb ~]#
```

Vérification en affichant la liste des *principals* :

```
[root@kerb ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  listprincs
K/M@UNIV-RENNES1.FR
kadmin/admin@UNIV-RENNES1.FR
kadmin/changepw@UNIV-RENNES1.FR
kadmin/history@UNIV-RENNES1.FR
kadmin/kerb.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR
krbtgt/UNIV-RENNES1.FR@UNIV-RENNES1.FR
root/admin@UNIV-RENNES1.FR
kadmin:  exit
[root@kerb ~]#
```

Si à cette étape **kadmin** affiche le message d'erreur **Cannot contact any KDC for requested realm while initializing kadmin interface**, cela signifie que le serveur ne se trouve pas lui-même comme KDC et il faut vérifier sa configuration réseau.

Ajout d'un principal pour le KDC lui-même (indispensable pour la réplication) :

```
[root@kerb ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey host/kerb.ifsic.univ-rennes1.fr
WARNING: no policy specified for host/kerb.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to no
policy
Principal "host/kerb.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  exit
[root@kerb ~]#
```

Ajout d'un utilisateur (paubry) pour les tests :

```
[root@kerb ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc paubry
WARNING: no policy specified for paubry@UNIV-RENNES1.FR; defaulting to no policy
Enter password for principal "paubry@UNIV-RENNES1.FR":
Re-enter password for principal "paubry@UNIV-RENNES1.FR":
Principal "paubry@UNIV-RENNES1.FR" created.
kadmin:  exit
[root@kerb ~]#
```

Configuration Firewall

Exécuter system-config-firewall et ouvrir les ports entrants suivants :

- 88 (pour kinit)
- 749 (pour les changements de mot de passe)
- 750 (pour l'authentification)

Intégration d'un client Windows 7 (archive)

Déclaration du client Kerberos

Sur le KDC, ajouter le principal correspondant au client :

```
[root@kerb ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -pw password -e des-cbc-crc:normal host/cwin7.ifsic.univ-rennes1.fr
WARNING: no policy specified for host/cwin7.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to
no policy
Principal "host/cwin7.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  exit
[root@kerb ~]#
```

Configuration Kerberos

Executer **cmd.exe** en tant qu'administrateur :

```
c:\> ksetup /addkdc UNIV-RENNES1.FR kerb.ifsic.univ-rennes1.fr
c:\> ksetup /setmachpassword password
c:\> ksetup /setrealm UNIV-RENNES1.FR
```

Rebooter le client puis exécuter (toujours en tant qu'administrateur) :

```
c:\> ksetup /mapuser * guest
```

Nous n'avons pas fait :

```
c:\> ksetup /addkpasswd UNIV-RENNES1.FR kerb.ifsic.univ-rennes1.fr
```

qui semble permettre le changement des mots de passe depuis les clients puisque ce n'est pas désiré (procédure centralisée depuis une page web qui devrait à terme mettre à jour l'annuaire LDAP et la base Kerberos.

Modification le chiffrement

Chiffrements autorisés

Demarrer -> Panneau de configuration -> Système et sécurité -> Outils d'administration -> Stratégies de sécurité locale -> Stratégies locales -> Sécurité réseau : configurer les types de chiffrement autorisés pour Kerberos : DES_CBC_CRC et DES_CBC_MD5.

cf http://technet.microsoft.com/en-us/library/dd560670%28WS.10%29.aspx

NB : cela modifie la clé de registre suivante : HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System\Kerberos\Parameters\SupportedEncryptionTypes (et la positionne à 0x03).

Chiffrement par défaut

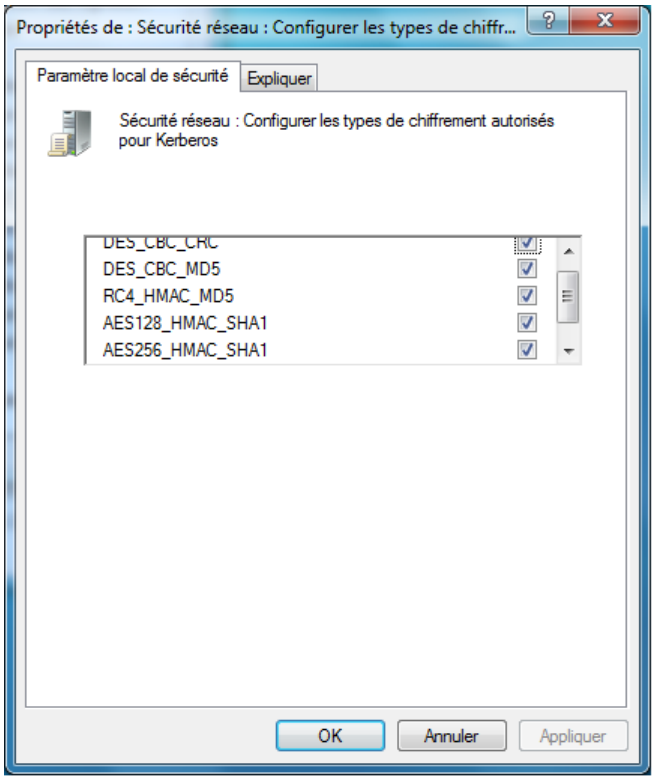
Mettre la valeur 0x17 (23) dans la clé de registre HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\Kerberos\Parameters.

Sur le serveur kerberos

Malgré les modifications indiquées ci-dessus, les paquets TGS-REP (réponse à une demande de TGS) ne sont pas compris par les postes Windows 7. Le TGS-REP est en effet chiffré en 3DES qui semble -t-il pose problème. Une modification du krb5.conf (sur le serveur kerberos) règle problème :

```
default_realm = UNIV-RENNES1.FR
default_etypes = des3-hmac-sha1 des-cbc-crc
default_tkt_enctypes = des3-hmac-sha1 des-cbc-crc
default_tgs_enctypes = des3-hmac-sha1 des-cbc-crc
#permitted_enctypes = des3-hmac-sha1 des-cbc-crc rc4-hmac
permitted_enctypes =  des-cbc-crc rc4-hmac
ticket_lifetime = 24h
forwardable = yes
```

Reste à vérifier si d'autres types d'encodage peuvent être introduits, l'image ci-dessous montre les types d'encodage supportés par Windows



:

Intégration d'un client Linux (archive)

Authentification

Configurer l'authentification des utilisateurs avec **system-config-authentication** :

- User information : Enable LDAP support, LDAP search base DN : ou=people,dc=univ-rennes1,dc=fr, LDAP server : ldap:ldapglobal.univ-rennes1.fr
- Authentication : Enable Kerberos support, Realm : UNIV-RENNES1.FR, KDCs : kerb.ifsic.univ-rennes1.fr:88, Admin servers : kerb.ifsic.univ-rennes1.fr:749
- sur les gentoo de l'IFSIC : il faut installer les paquets **mit-krb5** et **pam_krb5** et au final le fichier /etc/pam.d/system-auth doit avoir l'allure suivante :

```
auth       required     pam_env.so
auth       sufficient   pam_unix.so likeauth nullok
auth       sufficient   pam_krb5.so try_first_pass
auth       required     pam_deny.so

account     required      pam_unix.so broken_shadow
account     sufficient    pam_localuser.so
account     sufficient    pam_succeed_if.so uid < 500 quiet
account     [default=bad success=ok user_unknown=ignore] pam_krb5.so
account     required      pam_permit.so

password   required pam_cracklib.so difok=2 minlen=8 dcredit=2 ocredit=2 retry=3
password   sufficient pam_unix.so nullok md5 shadow use_authtok
password   required pam_deny.so

session     optional      pam_keyinit.so revoke
session     required      pam_limits.so
session     [success=1 default=ignore] pam_succeed_if.so service in crond quiet use_uid
session     required      pam_unix.so
session     optional      pam_krb5.so
```

Configuration Kerberos

Vérifier le fichier **/etc/krb5.conf** :

```
[logging]
 default = FILE:/var/log/krb5libs.log
 kdc = FILE:/var/log/krb5kdc.log
 admin_server = FILE:/var/log/kadmind.log

[libdefaults]
 default_realm = UNIV-RENNES1.FR
 default_etypes = des3-hmac-sha1 des-cbc-crc
 default_tkt_enctypes = des3-hmac-sha1 des-cbc-crc
 default_tgs_enctypes = des3-hmac-sha1 des-cbc-crc
 permitted_enctypes = des3-hmac-sha1 des-cbc-crc rc4-hmac
 ticket_lifetime = 24h
 forwardable = yes

[realms]
 UNIV-RENNES1.FR = {
  kdc = kerb.ifsic.univ-rennes1.fr:88
  admin_server = kerb.ifsic.univ-rennes1.fr:749
  default_domain = univ-rennes1.fr
 }

[domain_realm]
 .univ-rennes1.fr = UNIV-RENNES1.FR
 univ-rennes1.fr = UNIV-RENNES1.FR

[appdefaults]
 pam = {
  debug = false
  ticket_lifetime = 36000
  renew_lifetime = 36000
  forwardable = true
  krb4_convert = false
 }
```

Créer le principal du client sous **kadmin** (depuis le client) et générer stocker la clé localement (dans **/etc/krb5.keytab**) :

```
[root@clinux log]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey host/clinux.ifsic.univ-rennes1.fr
WARNING: no policy specified for host/clinux.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to
no policy
Principal "host/clinux.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  ktadd -k /etc/krb5.keytab host/clinux.ifsic.univ-rennes1.fr
Entry for principal host/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-256 CTS
mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-128 CTS
mode with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc
mode with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with
HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal host/clinux.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode
with RSA-MD5 added to keytab WRFILE:/etc/krb5.keytab.
kadmin:  exit
[root@clinux log]#
```

Le shell des utilisateurs est renvoyé par l'attribut **loginShell** de l'annuaire LDAP. Si le shell des utilisateurs n'est pas installé, il faut alors l'installer (par exemple **yum install csh**).

Vérification de l'authentification des utilisateurs :

```
[root@clinux ~]# su - paubry
su: warning: cannot change directory to /private/staff/y/ry/paubry: No such file or directory
id: cannot find name for group ID 20857
su: /bin/csh: No such file or directory
[paubry@clinux ~]$ exit
logout
[root@clinux log]#
```

Monter les homedirs des utilisateurs en ajoutant dans **/etc/fstab** les lignes suivantes :

```
sf1ifsic:/vol/vol1/private/student /private/student nfs
exec,nolock,dev,suid,rw,rsize=8192,wsize=8192 1 1
sf1ifsic:/vol/vol2/private/staff /private/staff nfs exec,nolock,dev,suid,rw,rsize=8192,wsize=8192
1 1
```

Créer puis monter les répertoires d'accueil :

```
[root@clinux ~]#
[root@clinux ~]# cd /
[root@clinux /]# mkdir -p /private/staff
[root@clinux /]# mkdir -p /private/student
[root@clinux /]# mount -a
[root@clinux /]# mount
[...]
sf1ifsic:/vol/vol1/private/student on /private/student type nfs
(rw,nolock,rsize=8192,wsize=8192,addr=148.60.4.42)
sf1ifsic:/vol/vol2/private/staff on /private/staff type nfs
(rw,nolock,rsize=8192,wsize=8192,addr=148.60.4.42)
[root@clinux /]#
```

Ajouter si nécessaire le groupe des utilisateurs en local en ajoutant dans le fichier **/etc/group** :

```
staff:x:20857:
```

Enfin vérifier à nouveau le login des utilisateurs :

```
[root@clinux ~]# su - paubry
[paubry@clinux ~]$ exit
logout
[root@clinux ~]#
```

Configuration Firefox

Pour que l'authentification Kerberos soit propagée par Firefox, une petite configuration est nécessaire comme indiqué sur cette page : Configuration de Firefox (archive)

Machines mises en place (archive)

Rôles des machines

Nom	IP	Rôle
kerb.ifsic.univ-rennes1.fr	148.60.10.50	Serveur Kerberos
cas.ifsic.univ-rennes1.fr	148.60.10.51	Serveur web (Apache + mod_auth_kerb) Serveur CAS
clinux.ifsic.univ-rennes1.fr	148.60.10.52	Client Linux
cwinxp.ifsic.univ-rennes1.fr	148.60.10.53	Client Windows XP
cwin7.ifsic.univ-rennes1.fr	148.60.10.54	Client Windows 7

Configuration commune des machines

Netmask : 255.255.255.0
Gateway : 148.60.10.254
DNS : 148.60.4.1
NTP : ntp1.univ-rennes1.fr

Intégration d'un client Windows XP (archive)

Déclaration du client Kerberos

Sur le KDC, ajouter le principal correspondant au client :

```
[root@kerb ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -pw password -e des-cbc-crc:normal host/cwinxp.ifsic.univ-rennes1.fr
WARNING: no policy specified for host/cwinxp.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to
no policy
Principal "host/cwinxp.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  exit
[root@kerb ~]#
```

Configuration Kerberos

Installer les outils d'administration supplémentaires (sur le CD d'installation, exécuter SUPTOOLS.EXE dans le répertoire \SUPPORT\TOOLS). Cela installe l'utilitaire KSETUP.

Executer **cmd.exe** en tant qu'administrateur :

```
C:\>ksetup /addkdc UNIV-RENNES1.FR kerb.ifsic.univ-rennes1.fr
NOTE: /AddKdc requires a reboot to take effect on pre-SP1 Win2000 computers
C:\>ksetup /addkpasswd UNIV-RENNES1.FR kerb.ifsic.univ-rennes1.fr
NOTE: /AddKpasswd requires a reboot to take effect on pre-SP1 Win2000 computers
C:\>ksetup /setcomputerpassword password
Setting computer password
NOTE: /SetComputerPassword requires a reboot to take effect.
C:\>ksetup /setrealm UNIV-RENNES1.FR
Setting Dns Domain
NOTE: /SetRealm requires a reboot to take effect
C:\>
```

Rebooter le client puis exécuter (toujours en tant qu'administrateur) :

```
C:\> ksetup /mapuser * guest
```

Configuration réseau

Modifier le nom de l'ordinateur (Poste de travail, clic droit, Propriétés, Nom de l'ordinateur, Modifier, Autres)

- Nom NetBios : CWINXP
- Nom complet de l'ordinateur : cwinxp.ifsic.univ-rennes1.fr
- Groupe de travail : UNIV-RENNES1.FR

Décocher la case Modifier le suffixe DNS principal lorsque les adhésions au domaine sont modifiées.

Une fois tous ces paramétrages effectués, il est désormais possible de se connecter sur le royaume Kerberos UNIV-RENNES1.FR.

Problèmes liés au clonage des stations de travail (archive)

La configuration d'un client dans un royaume kerberos nécessite quelques paramètres clonables (nom du royaume, serveurs du royaume,...). Ces paramètres sont positionnés dans le fichier /etc/krb5.conf (Linux) ou par des commandes qui modifient des clefs de registre (ksetup sous Windows). Dans le cadre d'un système de déploiement, l'image à déployer peut donc être configurée avec les valeurs adéquates (des constantes).

Le problème principal est de mettre en place la confiance entre le serveur kerberos et les clients (fichier keytab sous Linux, clefs de registre sous Windows, ...). Un fichier keytab ou son équivalence Windows contient la clef attribuée au principal par le serveur kerberos. Cette clef est évidemment propre à chaque client (principal), comment éviter une intervention sur chaque poste à l'issue d'un déploiement ? Il faut aussi noter que la sécurité d'un environnement kerbérisé repose sur la confidentialité du contenu des keytab et qu'on se place ici forcément dans un compromis entre sécurité et faisabilité.

- Linux : les keytab de toutes les machine figurent dans l'image à déployer, la première phase de boot va éliminer les keytab inutiles et positionner le bon en fonction du nom du poste de travail. Pour créer tous les keytab on peut procéder de la manière suivante :
  - sur le serveur kerberos on execute kadmin/addprinc (randkey) pour créer tous les principaux;
  - on génère ensuite un keytab par poste de travail. Les fichiers keytab portent un nom explicite et sont placés dans un répertoire qui va être exporté sur le poste étalon.
- Windows : concernant la création des principaux, on ne peut pas ici utiliser l'option -randkey car l'installation de la clef sur le client est effectuée par une commande. On va donc sur le serveur kerberos créer tous les principaux Windows avec le même mot de passe. Sur le poste étalon il faut exécuter "ksetup /setcomputerpassword password". Cette commande affecte le mot de passe qui va être utilisé dans les échanges avec le serveur kerberos dans le but d'obtenir le secret partagé.

Question: le scénario Windows est-il reproductible sous Linux ?

802.1X, radius et Kerberos (archive)

802.1X est un protocole qui a pour but d'ouvrir l'accès au réseau en fonction d'une authentification des usagers ou des machines qui essaient de s'y raccorder. Le processus d'authentification peut être varié et déporté vers un service d'authentification centralisé. De nombreux cas d'usage utilisent un serveur *freeRadius* pour l'authentification. Les clients **802.1X** (commutateurs, bornes Wi-Fi, ...) sont alors configurés pour interroger un serveur *freeRadius* qui gère différents scénarios d'authentification. Pour configurer un serveur *freeRadius* s'appuyant sur une base d'usagers Kerberos, on peut procéder de la manière suivante :

```
- compiler un freeRadius à partir de la distribution SRC
- un module  rlm_krb5 est alors produit
- dans le fichier radiusd.conf insérer ce qui suit dans la configuration des modules
 krb5 {
           keytab = /etc/krb5.keytab
           service_principal = radius/fqdn.du.serveur.radius      }
- toujours dans radiusd.conf dans la section authenticate ajouter
  Auth-Type Kerberos {
    krb5
  }
- la configuration du reste dépend du cas d'usage, ci-dessous un ajout effectué dans le fichier
users :

  tutu Auth-Type := kerberos
 Fall-Through = No
```

Il convient également de créer les principaux suivants :

- le host qui héberge le serveur radius (addprinc -randkey host/fqdn.du.serveur.radius)
- le service radius (addprinc -randkey radius/fqdn.du.serveur.radius)
- extraire le fichier keytab correspondant et l'installer comme indiqué dans le radiusd.conf

Si ce scénario permet d'intégrer une base kerberos dans le processus d'ouverture des accès au réseau, notons que la propagation des tickets ne s'effectue pas jusqu'aux machines connectées (à suivre ...).

Configuration de CUPS pour Kerberos (archive)

Dans l'architecture de test, le serveur CUPS est installé sur la machine cas.ifsic.univ-rennes1.fr.

Configuration du serveur

Générer le principal **ipp/cas.ifrsic.univ-rennes1.fr** et le stocker dans **/etc/krb5.keytab** :

```
[root@cas ~]# kadmin
Authenticating as principal root/admin@UNIV-RENNES1.FR with password.
Password for root/admin@UNIV-RENNES1.FR:
kadmin:  addprinc -randkey ipp/cas.ifsic.univ-rennes1.fr
WARNING: no policy specified for ipp/cas.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR; defaulting to no
policy
Principal "ipp/cas.ifsic.univ-rennes1.fr@UNIV-RENNES1.FR" created.
kadmin:  ktadd -k /etc/krb5.keytab ipp/cas.ifsic.univ-rennes1.fr
Entry for principal ipp/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-256 CTS mode
with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal ipp/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type AES-128 CTS mode
with 96-bit SHA-1 HMAC added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal ipp/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type Triple DES cbc mode
with HMAC/sha1 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal ipp/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type ArcFour with
HMAC/md5 added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal ipp/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type DES with HMAC/sha1
added to keytab WRFILE:/etc/krb5.keytab.
Entry for principal ipp/cas.ifsic.univ-rennes1.fr with kvno 3, encryption type DES cbc mode with
RSA-MD5 added to keytab WRFILE:/etc/krb5.keytab.
kadmin:  exit
[root@cas ~]#
```

Exécuter la commande suivante pour mettre en place l'authentification Kerberos :

```
[root@cas ~]# cupsctl DefaultAuthType=Negotiate
[root@cas ~]#
```

x

Migration de l'authentification de LDAP à Kerberos (archive)

Sous réserve de complétude des tests en cours, les pages de cet espace montrent qu'il est effectivement possible de complètement

déléguer l'authentification des utilisateurs à un serveur Kerberos.

Cette page discute de la migration d'une architecture où l'authentification des utilisateurs est confiée à un annuaire LDAP à une architecture où l'annuaire LDAP n'aurait en charge que son métier propre (le service d'annuaire) et où l'authentification des utilisateurs serait assurée par Kerberos.

Pourquoi ne pas basculer directement de LDAP à Kerberos ?

Parce que c'est impossible.

Cette phase de migration, dans laquelle l'authentification serait assurée à la fois par l'annuaire LDAP et le serveur Kerberos est obligatoirement à envisager pour les raisons suivantes :

- L'ajout d'un utilisateur (de son principal) dans le KDC nécessite la connaissance de son mot de passe ; puisque le mot des utilisateurs dans l'annuaire LDAP est chiffré de manière non réversible, il est impossible de créer automatiquement à un instant donné les comptes des utilisateurs dans le royaume Kerberos à partir des informations contenues dans l'annuaire LDAP.
- Certaines applications (*Legacy*) peuvent nécessiter une authentification LDAP et doivent continuer à fonctionner.

> **Application Legacy qui nécessitent une authentification LDAP**
> Au terme de la phase de migration, c'est-à-dire quand tous les utilisateurs existants avant la mise en place de Kerberos seront présents dans le royaume, il sera possible (sous réserve de tests) de modifier l'authentification pour qu'un bind user/password soit validé en s'appuyant sur le serveur Kerberos et non les mots de passe contenus dans l'annuaire LDAP.
>
> Il est peut-être même possible de faire en sorte de configurer l'annuaire LDAP de telle manière que les binds user/password soient validés en s'appuyant sur les mots de passe de l'annuaire LDAp présents, et sur le serveur Kerberos sinon, grâce à un démon **sals_authd** utilisant la GSSAPI (à confirmer).

Quand et comment migrer les utilisateurs ?

Il faut donc disposer d'un moyen de créer les utilisateurs dans le royaume Kerberos a la volée, à un moment où on dispose de leur mot de passe.
Les moments candidats sont les suivants :

- Lors de l'authentification sur un poste client. Cette solution, envisageable grâce au module PAM **pam_krb5_migrate**, est rejetée car elle nécessiterait un accès privilégié de création des comptes sur les postes clients, estimé comme un trop gros risque en matière de sécurité.
- Lors de l'authentification à un service, consulté par les utilisateurs de manière suffisamment régulière pour que la migration soit la plus rapide possible. Le seul service candidat suivant cette contrainte est le serveur de mail. Cette solution se heurte néanmoins au fait que dans de nombreux établissements, la consultation des mails se fait (au moins pour les étudiants) à travers un webmail CASifié, qui ne reçoit pas les mots de passe des utilisateurs (c'est dans ce cas le module **pam_cas** qui valide l'authentification des utilisateurs grâce à des ST ou PT émis par le serveur CAS).
- Lors de l'authentification sur le serveur CAS. C'est cette solution qui est retenue.

Comment faire ?

Pour rajouter au royaume Kerberos les utilisateurs qui n'y sont pas déjà

Il faut rajouter au serveur CAS le code nécessaire pour, à chaque fois qu'un utilisateur se connecte avec une authentification différente de Kerberos :

- vérifier si l'utilisateur existe dans le royaume Kerberos
- s'il n'existe pas, le créer avec le mot de passe avec lequel il s'est authentifié auprès d'une autre source (en l'occurrence l'annuaire LDAP).

Voir plus loin pour le code ajouté au serveur CAS.

Pour créer les nouveaux utilisateurs à la fois dans le royaume Kerberos et l'annuaire LDAP

Il faut modifier la procédure de création des comptes des utilisateurs et y rajouter le code nécessaire pour créer les utilisateurs dans le royaume Kerberos.

Selon les procédures en vigueur dans l'établissement, il est également possible de créer le compte dans le royaume Kerberoslors de la première validation du compte à travers une interface web dédiée (comme cela est fait pour l'interface Sésame à l'université de Rennes 1).

Pour maintenir la cohérence des mots de passe LDAP et Kerberos

Il faut pendant la phase de migration s'assurer que les changements de mot de passe soient faits à la fois dans l'annuaire LDAP et le royaume Kerberos.

Pour cela, le changement de mot de passe ne doit pas être possible depuis les postes clients (car il ne serait répercuté dans l'annuaire LDAP) et doit se faire via une interface web centraliséequi répercute les changements à la fois dans le royaume Kerberos et l'annuaire LDAP.

Modifications du serveur CAS

Comme vu précédemment il faut, à chaque fois qu'un utilisateur se connecte avec une authentification différente de Kerberos :

- vérifier si l'utilisateur existe dans le royaume Kerberos
- s'il n'existe pas, le créer avec le mot de passe avec lequel il s'est authentifié auprès d'une autre source (en l'occurrence l'annuaire LDAP).

L'alimentation du royaume Kerberosest faite par un wrapper de AuthenticationHandler ; de cette manière, elle peut être activée pour pour certains handlers seulement.

Modifications des sources

On crée tout d'abord un module supplémentaire nommé **cas-server-integration-kerberosfeed** en installant les sources du zip attaché .

On ajoute le nouveau module dans la liste des modules de **/pom.xml** :

```
<modules>
  <module>cas-server-core</module>
  [...]
  <module>cas-server-webapp</module>
  <module>cas-server-integration-kerberosfeed</module>
</modules>
```

On joute également la propriété **skipTests** au plugin **maven-sunfire** pour éviter de rejouer tous les tests à la compilation :

```
<plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-surefire-plugin</artifactId>
  <configuration>
    <skipTests>true</skipTests>
    <includes>
      <include>**/*Tests.java</include>
    </includes>
    <excludes>
      <exclude>**/Abstract*.java</exclude>
    </excludes>
  </configuration>
</plugin>
```

Pour compiler le nouveau module (nécessaire après tout changement), exécuter :

```
mvn \-pl cas-server-integration-kerberosfeed install
```

Ajouter une dépendance du module **cas-server-webapp** vers le nouveau module dans **/cas-server-webapp/pom.xml** :

```
<dependency>
  <groupId>org.jasig.cas</groupId>
  <artifactId>cas-server-integration-kerberosfeed</artifactId>
  <version>${project.version}</version>
</dependency>
```

A chaque fois que l'on modifie la configuration du module **cas-server-webapp**, exécuter :

```
mvn -pl cas-server-webapp package
```

Cela créee le war **/cas-server-webapp/target/cas.war** qui peut être déployé.

Configuration

Les beans ci-dessous sont dans le fichier **/cas-server-webapp/src/main/webapp/WEB-INF/deployerConfigContext.xml**.

On remplace tout d'abord le bean FastBindLdapAuthenticationHandler par :

```
<bean class="org.esupportail.cas.adaptors.kerberosfeed.KerberosFeedAuthenticationHandlerWrapper" >
  <property name="authenticationHandler">
    <bean class="org.jasig.cas.adaptors.ldap.FastBindLdapAuthenticationHandler" >
      <property name="filter" value="uid=%u,ou=people,dc=univ-rennes1,dc=fr" />
      <property name="contextSource" ref="contextSource" />
    </bean>
  </property>
  <!--
   | The configuration used to feed the Kerberos Realm, mandatory.
  -->
  <property name="config" ref="kerberosFeedConfig" />
  <!--
   | The registry used to store the usernames of the users that have already been added
   | to the realm. Defaults to an 'in memory' implementation where addings will be
   | lost on server startup.
  -->
  <property name="registry" ref="kerberosFeedRegistry" />
</bean>
```

Le bean **kerberosFeedConfig** mutualise la configuration de l'accès au serveur Kerberos :

```
<!--
   | The configuration used to feed the Kerberos Realm.
   | The values below are used to perform a bash kadmin command.
  --><bean id="kerberosFeedConfig"  class=
"org.esupportail.cas.adaptors.kerberosfeed.KerberosFeedConfig">
  <!--
   | The name of the Kerberos realm, mandatory.
  -->
  <property name="realm" value="UNIV-RENNES1.FR" />
  <!--
   | The name of the principal used to authenticate in kadmin, defaults to cas/admin.
  -->
  <property name="principal" value="cas/admin" />
  <!--
   | Set this property to true to use a keytab to authenticate in kadmin (preferred), or false to
use
   | a password. Defaults to true.
  -->
  <property name="useKeytab" value="true" />
  <!--
   | The name of the keytab used when useKeytab is set to true (unused otherwise).
   | Defaults to /etc/admin.keytab.
  -->
  <property name="keytab" value="/etc/admin.keytab" />
  <!--
   | The password used authenticate in kadmin, defaults to secret (you shall probably
   | change it since kadmin authentication will fail with the default value).
  -->
  <!-- property name="password" value="secret" /-->
  <!--
   | A string that contains the chars allowed for the users' passwords (set to the default here).
  -->
  <property
    name="passwordAllowedChars"
    value=
"abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789&amp;~#{([-|`\\_^@)]=+}$%*!:/;.,?&gt;&
/>
</bean>
```

Enfin le bean suivant est le registre utilisé pour mémoriser les noms des utilisateurs qui ont déjà été alimentés dans le royaume Kerberso (pour ne pas rejour deux fois) :

```
<!--
   | The registry used to store the ids of the users that have already been added
   | to the realm. Unlike the default implementation where usernames are backed to
   | memory, the implementation below uses a Berkeley DB and thus is persistent.
  -->
  <bean id="kerberosFeedRegistry"  class=
"org.esupportail.cas.adaptors.kerberosfeed.registry.BerkeleyDbRegistryImpl">
    <!--
     | The path used to store the data (must be writable by the tomcat user).
     | Defaults to /tmp.
    -->
  <property name="dbPath" value="/tmp" />
</bean>
```

x