Sécurité du Système d'Information

Authentification centralisée et SSO

Nombres de pages : 14	**Version :** 1.0
Auteurs : • HAMROUNI Makram • POISSENOT Thomas • ROUX Nicolas	
Destinataires : • BOMBAL Sébastien	
Remarques : Aucune.	

Tables des matières

1 Introduction

1.1 Présentation de Squid

Squid est un logiciel proxy couramment utilisé dans le monde de l'entreprise. Il peut gérer les protocoles suivant FTP, HTTP, Gopher et HTTPS.

Les logiciels proxy fonctionnent suivant le principe qu'ils acceptent ou refusent une connexion entre le réseau interne et le réseau externe suivant l'entête de celui-ci.

1.2 Présentation du projet

Durant ce projet nous avons mis en place une authentification centralisée sur un Windows serveur 2003 avec Active Directory (AD) gérant les protocoles NTLM et Kerberos.

Nous devons aussi mettre en place un proxy Squid afin de filtrer le trafic HTTP suivant les droits des utilisateurs. Pour ceci, nous devons dans un premier temps tester avec l'authentification NTLM puis Kerberos dans un second temps afin de les comparer.

Le client quand à lui était un poste de travail Windows XP avec Internet Explorer 7 s'authentifiant sur le domaine AD et pouvant ensuite accéder à internet de façon transparente sous IE7 à travers le proxy.

Voir schéma un peu plus loin.

1.3 Pourquoi le SSO ?

Le SSO permet d'avoir un gain de temps et de productivité au niveau des utilisateurs, ils ont ainsi moins de mots de passe à taper lors de leur travail.

SSO est aussi une hausse générale de la sécurité vu que les utilisateurs ont moins de mots de passe à retenir, ils essayent donc de les retenir au lieu de les marquer afin de s'en souvenir.

2 Mise en place du proxy SQUID

2.1 Installation de Squid 2.6 sous Ubuntu

```
$ apt-get install squid
```

2.2 Configuration de Squid pour le HTTP

Le fichier de configuration est squid.conf:

- On a choisi le port 3128 comme port d'écoute pour le serveur :

```
http_port 3128
```

- Configuration de Squid pour les accès HTTP uniquement :

```
# Autorise le protocole HTTP
acl protocole_web proto HTTP
http_access allow protocole_web
# Refuse le protocole FTP
acl protocole_ftp proto FTP
http_access deny protocole_ftp
```

- Configuration du nom de serveur Squid en cas d'erreur :

```
visible_hostname squidServer
```

- Taille en Méga Octets de RAM que peut utiliser Squid pour les transitions d'objets, Hot Objects :

```
cache_mem 32 MB
```

- Spécification sous quel utilisateur le cache va tourner :

```
cache_effective_user makram
```

Important : Il faut que l'utilisateur ait le droit sur les fichiers de cache Squid.

2.2.1 Optimisation de sécurité :

- Sécurisation du fichier de configuration squid.conf

```
$ chown makram squid.conf
$ chmod 400 makram squid.conf
```

- Sécurisation du cache :

On peut envisager de dédier une partition au cache pour éviter que le cache soit sur la même partition que le système.

- Sécuriser la purge du cache en autorisant cette purge uniquement en local :

```
acl PURGE method purge
acl localhost src 127.0.0.1
http_access allow purge localhost
http_access deny purge
```

- Redirection de toutes les connections HTTP vers le proxy :

Il est intéressant pour des raisons de sécurité de rediriger toutes les connections HTTP vers le proxy. D'un côté cela oblige le client à utiliser le proxy pour se connecter sur internet. De l'autre côté cette configuration permet de sécuriser les accès à internet et permet aussi de ne plus configurer le proxy dans les paramètres Internet Explorer 7.

Il faut rajouter les lignes suivantes dans le fichier de configuration de Squid :

```
httpd_accel_host virtual
httdp_accel_port 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
httpd_accel_single_host off
```

On redirige les connections HTTP vers le port 8080 du proxy à l'aide de la règle « iptables » suivante :

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 8080
```

3 Mise en place d'ACLs

Dans le cadre d'une entreprise, il serait utile de mettre en place quelques restrictions sur le contenu des pages web accessibles par la mise en place des règles ACL.

Ces restrictions peuvent être de différentes natures :

1. Limitations des accès aux ports du serveur pour des raisons de sécurité.
2. Authentification NTLM, pour ne pas demander à l'utilisateur de s'authentifier pour les accès aux services internet afin de garantir un service transparent. D'un autre côté, il est inutile de demander l'authentification une fois de plus puisque on est dans un modèle SSO.
3. Mettre des restrictions sur l'accès au service internet pendant les heures hors des horaires de travail habituels.
4. Mettre en place des restrictions sur les sites internet à contenus dangereux ou incohérent avec la politique de sécurité de l'entreprise (ex. les sites warez, les sites à contenu pornographique …).
5. Définir une plage d'IP, afin de garantir l'accès à certains clients au service internet et restreindre l'accès à d'autres clients.
6. Il est intéressant aussi d'interdire la visualisation du contenu multimédia tel que le MP3, vidéos … .
7. Dans une entreprise les employés doivent fournir du travail et non pas se divertir !

- Mise en place des ports utiles :

```
acl SSL_ports port 443 # https
acl SSL_ports port 563 # snews
acl SSL_ports port 873 # rsync
acl Safe_ports port 80 # http
acl Safe_ports port 21 # ftp
acl Safe_ports port 443 # https
acl Safe_ports port 70 # gopher
acl Safe_ports port 210 # wais
acl Safe_ports port 1025-65535 # unregistered ports
acl Safe_ports port 280 # http-mgmt
acl Safe_ports port 488 # gss-http
acl Safe_ports port 591 # filemaker
acl Safe_ports port 777 # multiling http
acl Safe_ports port 631 # cups
acl Safe_ports port 873 # rsync
acl Safe_ports port 901 # SWAT
```

Dans le TAG « http_access » il faut ajouter ceci pour que cela devienne fonctionnel :

```
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
```

- Mise en place des restrictions horaires pour les accès au Web :

```
# Autorise l'accès au cache de 8h à 20h
acl horaire time M-F 08:00-20:00
http_access allow horaire
# Refuse l'accès au cache de 0h à 7h59
acl hors_horaire1 time MTWHF 00:00-07:59
http_access deny hors_horaire1
# Refuse l'accès au cache de 20h01 à 23h59
acl hors_horaire2 time MTWHF 20:01-23:59
http_access deny hors_horaire2
# Refuse l'accès au cache du samedi au dimanche.
acl week_end time A-S
http_access deny week_end
```

- Mise en place des restrictions sur les pages Web:

Pour cela il nous faut deux fichiers : good-sites.txt et bad-sites.txt.

Le fichier good-sites.txt contient les URL's autorisés.

Le fichier bad-sites.txt contient les URL's non autorisés.

Il faut ajouter les lignes suivantes dans le fichier squid.conf :

```
acl GoodSites dstdomain "/etc/squid/good-sites.txt"
acl BadSites  dstdomain "/etc/squid/bad-sites.txt"

http_access deny BadSites
http_access allow GoodSites
```

- Mise en place des restrictions sur les machines pouvant accéder au proxy en spécifiant une plage d'IP:

```
acl mynetwork src <IP de début>-<IP de fin>
http_access allow mynetwork
```

- Mise en place de l'authentification transparente pour l'utilisateur :

```
acl ntlm proxy_auth REQUIRED
auth_param ntlm program /usr/bin/ntlm_auth --helper-protocol=squid-2.5-
ntlmssp
auth_param ntlm children 5
auth_param ntlm keep_alive on

auth_param basic program /usr/bin/ntlm_auth --helper-protocol=squid-2.5-basic
auth_param basic children 5
auth_param basic realm Squid proxy-caching web server
auth_param basic credentialsttl 2 hours

http_access allow ntlm
```

4 Authentification utilisateurs

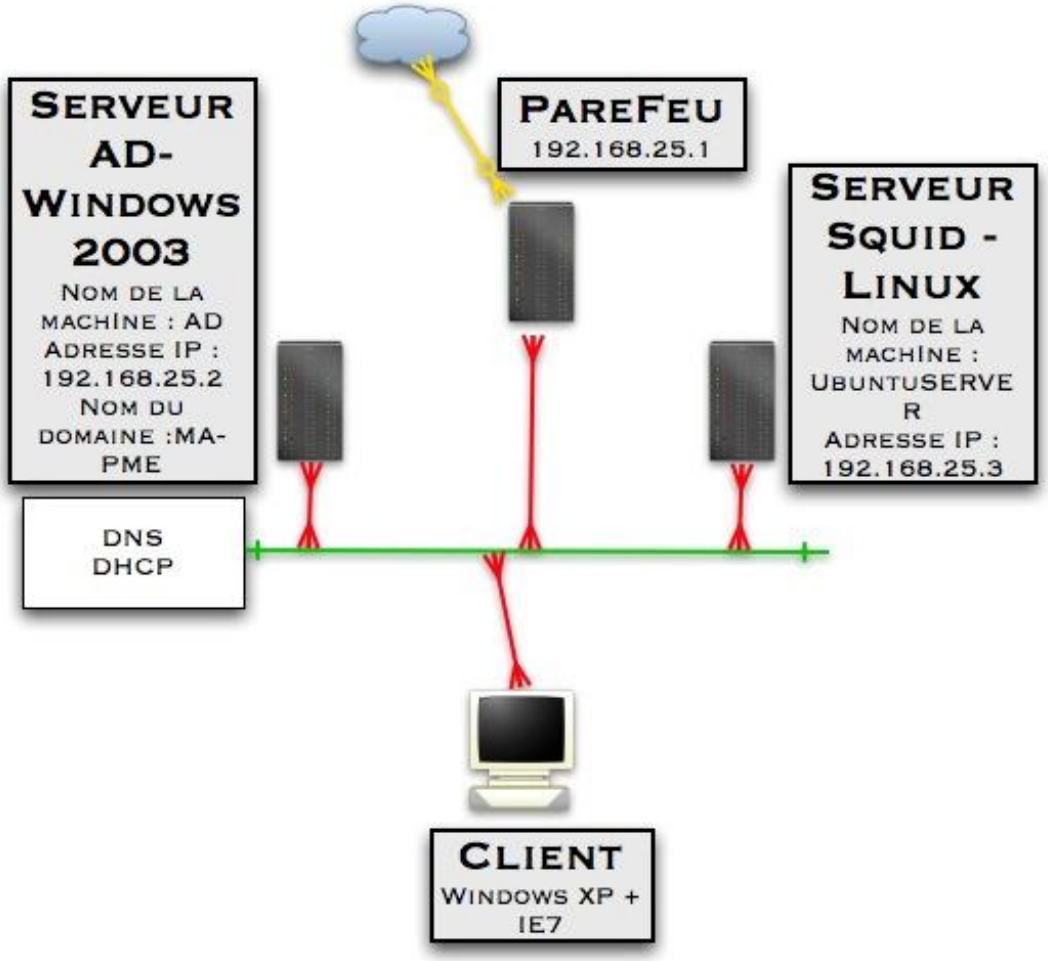
4.1 Scénario d'authentification 1:

4.1.1 Présentation de NTLM

« NTLM (NT Lan Manager) est un protocole d'identification utilisé dans diverses implémentations des protocoles réseau Microsoft et supporté par le « NTLMSSP » (Fournisseur de support de sécurité NT LM). À l'origine utilisé pour une authentification et une négociation sécurisée, NTLM est aussi utilisé partout dans les systèmes de Microsoft comme un mécanisme de signature unique (single sign-on). »

Source Wikipedia (http://fr.wikipedia.org/wiki/NTLM).

4.1.2 Architecture « Full Microsoft »



4.1.3 Fonctionnement de cette architecture

Le protocole NTLM utilise le mécanisme de « challenge-response », les clients doivent prouver au serveur leur identité sans envoyer de mot de passe dans les échanges entre le client et le serveur. Ce mécanisme utilise l'envoi de trois messages : la « «négociation », la « simulation » et l'authentification.

Voici une explication de ce mécanisme :

1. Le client envoie une requête web
2. Squid coupe la connexion avec le code 407 (proxy authentication required)
3. Négociation : Le client envoie à nouveau la requête avec une demande de négociation NTLM et un ensemble de flags contenant les informations sur la configuration de NTLM du client (version utilisée, type de chiffrement, etc.).

4. Stimulation : Squid répond avec le code 407, mais ne coupe pas la connexion. Il envoie ses propres informations NTLM (version etc.) et un Challenge.
5. Authentification : Le client envoie sa réponse au Challenge du serveur, qui est un hash entre le challenge et la "clé" du client.
6. Le client est authentifié auprès du Squid, il peut faire des requêtes

4.1.4 Mise en place de l'architecture

Dans un premier temps nous avons besoin de configurer Winbind dans le but de mapper les utilisateurs et groupes Windows dans l'environnement Linux. C'est à dire associer chaque utilisateur et groupe à un UID et un GID.

Le fichier sert aussi à se connecter à l'Active Directory et joindre le domaine Windows.

L'installation de Winbind se fait par la commande :

```
$ apt-get install samba winbind
```

La configuration de Winbind, celui-ci utilise le fichier de configuration de samba. Voici le contenu du fichier smb.conf :

```
[global]
# Le groupe de travail
  workgroup = MA-PME
# Le niveau de sécurité pour l'accès à l'Active Directory
  security = ADS
  winbind separator = /
# oblige samba d'encrypter les échanges avec mot de passe crypté
  encrypt passwords = yes
# Nom NetBios de la machine serveur squid
  netbios name = UBUNTUSERVER
  server string = %h server (Samba %v)
# La plage uid/ gid pour le mappage des comptes windows
  idmap uid = 10000-20000
  idmap gid = 10000-20000
# Directives qui permettent d'énumérer les comptes avec la commande wbinfo
  winbind enum users = yes
  winbind enum groups = yes
  allow trusted domains = yes
  # Le serveur sur lequel le domaine est hébergé
  password server = ad.ma-pme.fr
  # Le nom de domaine AD
  realm = MA-PME.FR
  winbind use default domain = yes
  winbind nested groups = yes
```

4.1.5 Captures du trafic

Résultat de l'exécution des commandes suivantes sur le serveur Squid :

```
$ wbinfo -u
administrateur
invité
krbtgt
```

```
toto

$ wbinfo -g
BUILTIN/administrators
BUILTIN/users
ordinateurs du domaine
contrôleurs de domaine
administrateurs du schéma
administrateurs de l'entreprise
admins du domaine
utilisa. du domaine
invités du domaine
propriétaires créateurs de la stratégie de groupe
dnsupdateproxy

$ wbinfo -m
MA-PME
```

Authentification d'un client avec l'utilisateur « toto » via le proxy :

```
1205350049.426  11915 172.16.171.136 TCP_REFRESH_MISS/200 17694 GET
http://newsrss.bbc.co.uk/rss/newsonline_world_edition/front_page/rss.xml toto
DIRECT/212.58.226.29 application/xml
```

Analyse du trafic HTTP via le proxy en lançant deux fois de suite la commande wget:

```
$ export http_proxy= http://login:password@monproxy.fr:3128
$ time wget -q www.google.com
real    0m0.051s
user    0m0.000s
sys     0m0.012s

$ time wget -q www.google.com
real    0m0.008s
user    0m0.000s
sys     0m0.008s
```

On observe l'exécution de la même requête une deuxième fois cela réduit considérablement le temps de réponse (par un facteur égale à 5). On en déduit que le système de cache du proxy joue bien son rôle.

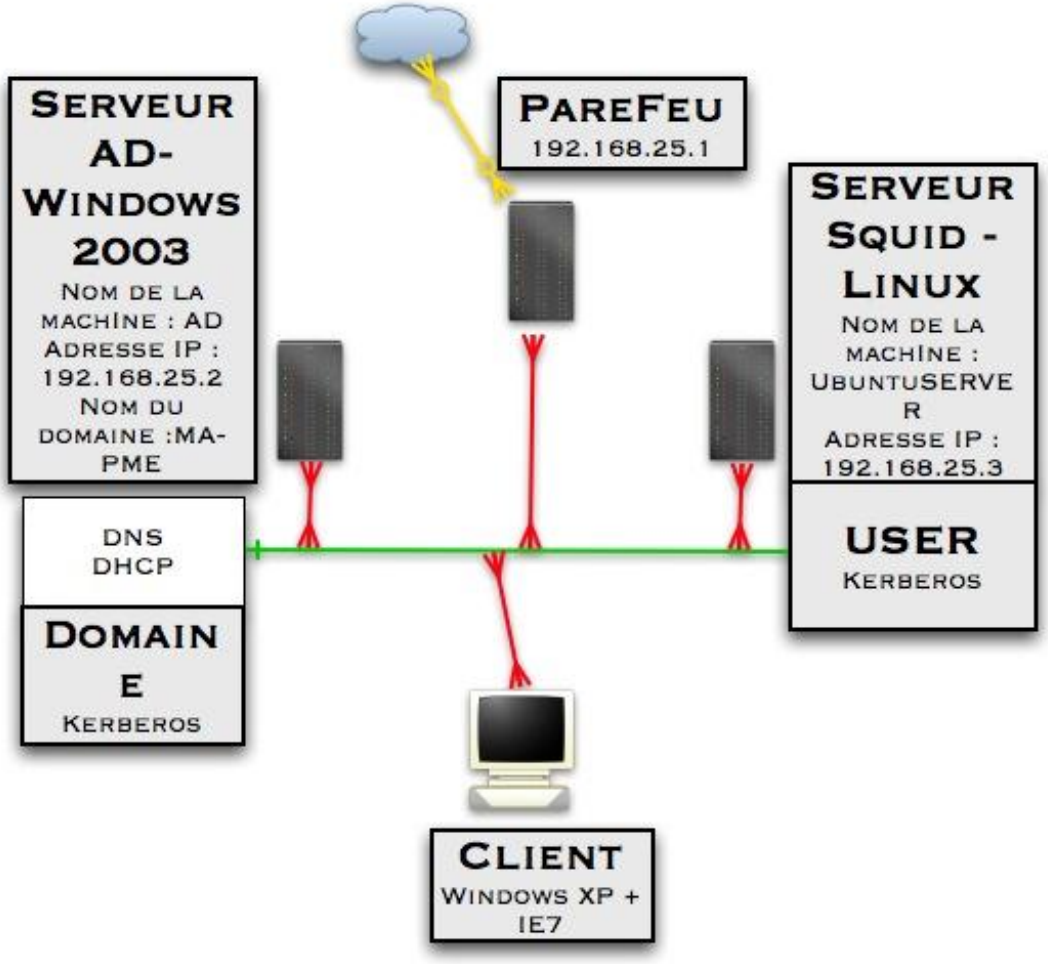
4.2 Scénario d'authentification 2:

4.2.1 Présentation de Kerberos

Le protocole Kerberos est le protocole de sécurité principal par défaut pour les authentifications réalisées dans un domaine Windows 2000/2003.

Il permet aux utilisateurs d'accéder aux ressources réseaux à partir de la même ouverture de session. Dans un réseau, le service Kerberos se trouve dans chaque contrôleur de domaine et les clients Kerberos sur les autres ordinateurs Windows.

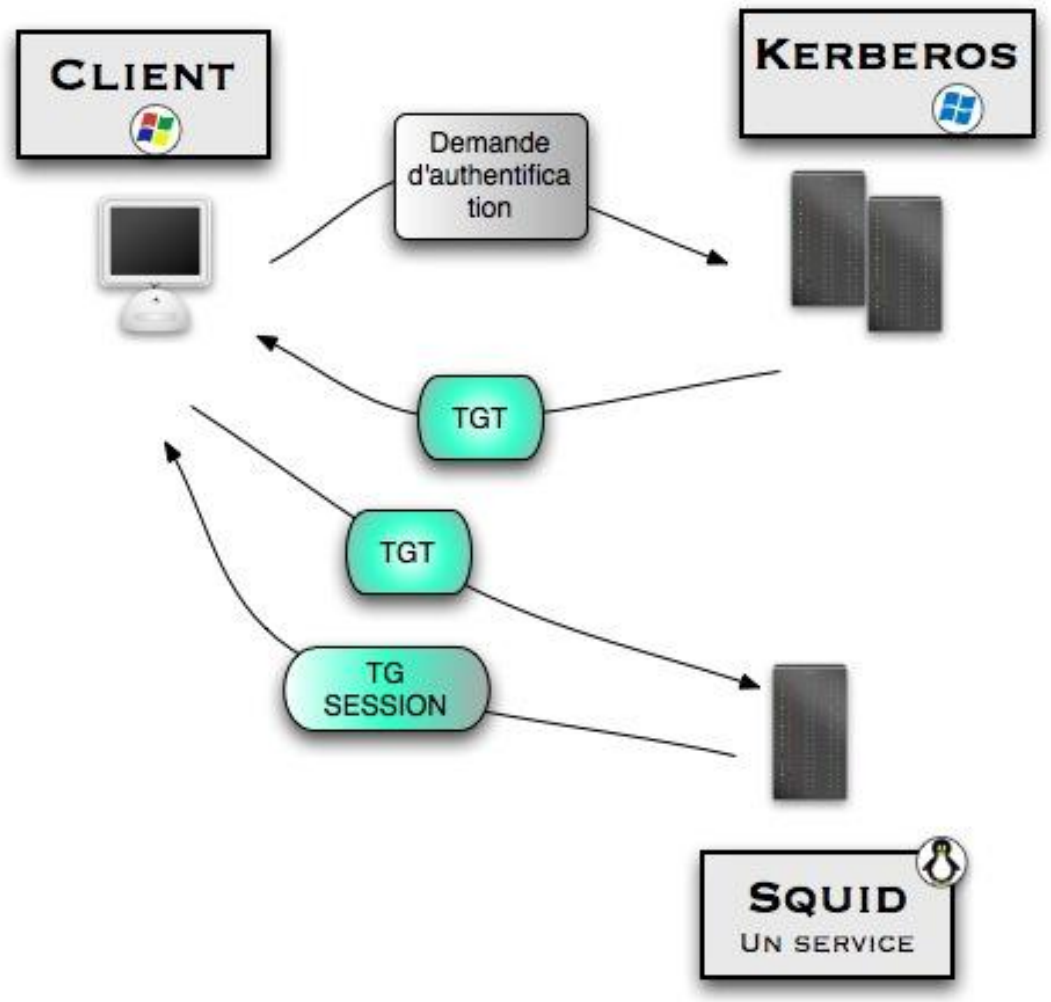
4.2.2 Architecture « Kerberos »



4.2.3 Fonctionnement de cette architecture

Voici les étapes de fonctionnement du protocole Kerberos

- Un client s'authentifie une fois par session auprès d'une autorité de confiance dans notre cas auprès du serveur 2003 : celle-ci lui délivre un ticket nommé TGT (Ticket-Granting Ticket). C'est la seule fois ou l'utilisateur aura à s'authentifier durant sa session de travail, tout services confondus.
- Le client ainsi authentifié souhaite maintenant utiliser un service distant comme le service internet via le proxy, pour la première fois de sa session. Il s'adresse alors à l'autorité de confiance ici dans notre cas au serveur Squid en lui présentant son TGT: celle-ci lui délivre un nouveau ticket, dédié au service.
- Le client s'adresse alors au service, en lui présentant ce nouveau ticket : il est maintenant authentifié auprès de celui-ci. Ce ticket pourra être réutilisé jusqu'à sa péremption.



4.2.4 Mise en place de l'architecture

Pour mettre en place une telle architecture réseau, nous allons garder la configuration du scénario 1 puis nous allons installer Kerberos et le configurer en tant que client sur le serveur Squid.

Installation :

```
$ apt-get install kerb5-user
```

Pour la configuration (voir le fichier en annexes), juste un changement dans le fichier Squid, on supprime les authentifications par NTLM en ajoutant les directives suivantes :

```
auth_param negotiate program /usr/local/squid/libexec/squid_kerb_auth -d
auth_param negotiate children 10
auth_param negotiate keep_alive on
acl AUTENTICADO proxy_auth REQUIRED
http_access allow AUTENTICADO
```

De plus cela demande la compilation de Squid avec les flags suivant :

- `–enable-auth="basic negotiate"`
- `–enable-basic-auth-helpers="LDAP"`
- `–enable-negotiate-auth-helpers="squid_kerb_auth"`
- `–enable-external-acl-helpers="ldap_group"`

Ensuite on doit créér une Keytab avec un utilisateur Active directory grâce à la commande ktutil.

On fait un export dans le shell de la variable d'environnement suivante :

```
KRB5_KTNAME=FILE:/etc/squid/squid.keytab
```

Puis on ajoute cette ligne dans le fichier de configuration de Squid :

```
cache_peer ad.ma-pme.fr parent 8080 0 proxy-only no-query login=NEGOTIATE
```

Pour pouvoir avoir des tickets depuis le serveur Kerberos, il faut obligatoirement que le serveur Kerberos soit synchronisé avec le serveur Squid. Pour cela on utilise ntpdate.

4.2.5 Captures du trafic

Pour voir si notre service Squid se fait délivrer des tickets pour les utilisateurs Windows on exécute la commande suivante :

```
$ kinit toto
```

Puis, pour voir les tickets on exécute la commande suivante :

```
$ klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: toto@MA-PME.FR

Valid starting     Expires            Service principal
03/13/08 22:12:08  03/14/08 04:52:08  krbtgt/MA-PME.FR@MA-PME.FR

Kerberos 4 ticket cache: /tmp/tkt0
klist: You have no tickets cached
```

Maintenant nous mettons un répertoire partagé sur le serveur Windows 2003 sous le nom Public ; nous allons accéder à ce répertoire avec smclient :

```
$ smbclient //ad.ma-pme.fr/public -k
OS=[Windows Server 2003 3790 Service Pack 1] Server=[Windows Server 2003 5.2]
smb: \>
```

Puis nous allons exécuter la commande suivante pour lister les tickets Kerberos en cache:

```
$ klist
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: toto@MA-PME.FR

Valid starting     Expires            Service principal
03/13/08 22:21:26  03/14/08 05:01:26  krbtgt/MA-PME.FR@MA-PME.FR
03/13/08 22:22:19  03/14/08 05:01:26  ad$@MA-PME.FR
```

Nous remarquons qu'il y a un nouveau ticket qui est ajouté à liste précédente, cela prouve que la distribution des tickets par Kerberos est fonctionnelle.

5 Conclusion

L'authentification Kerberos est plus sécurisée que NTLM car ce ne sont que des clés qui circule et non les mots de passe des utilisateurs. La configuration de NTLM sur Squid étant nettement plus documentée que celle de Kerberos, elle est donc plus utilisée en pratique et la sécurité convient, elle est parfaitement suffisante dans un environnement interne. Le proxy Squid quand à lui permet de ce protéger légalement envers les utilisations d'internet par ces employés en bloquant le contenu non désiré.

Niveau performances, nous voyons que le proxy Squid a de bonnes performances à la première résolution de noms (< dixième de seconde), et encore de meilleurs performances (< centième de seconde) à la suivante grâce au cache généré par celui-ci.

6 Annexes

6.1 Fichier de configuration de Squid :

```
http_port 3128
hierarchy_stoplist cgi-bin ?
acl QUERY urlpath_regex cgi-bin \?
cache deny QUERY
acl apache rep_header Server ^Apache
broken_vary_encoding allow apache
cache_mem 32 MB
hosts_file /etc/hosts
auth_param ntlm program /usr/bin/ntlm_auth --helper-protocol=squid-2.5-ntlmssp
auth_param ntlm children 5
auth_param ntlm keep_alive on
auth_param basic program /usr/bin/ntlm_auth --helper-protocol=squid-2.5-basic
auth_param basic children 5
auth_param basic realm Squid proxy-caching web server
auth_param basic credentialsttl 2 hours

refresh_pattern ^ftp:       1440  20%   10080
refresh_pattern ^gopher:    1440  0%    1440
refresh_pattern .           0     20%   4320
#Recommended minimum configuration:
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl to_localhost dst 127.0.0.0/8
acl SSL_ports port 443 # https
acl SSL_ports port 563 # snews
acl SSL_ports port 873 # rsync
acl Safe_ports port 80 # http
acl Safe_ports port 21 # ftp
acl Safe_ports port 443 # https
acl Safe_ports port 70 # gopher
acl Safe_ports port 210 # wais
acl Safe_ports port 1025-65535 # unregistered ports
acl Safe_ports port 280 # http-mgmt
acl Safe_ports port 488 # gss-http
acl Safe_ports port 591 # filemaker
acl Safe_ports port 777 # multiling http
acl Safe_ports port 631 # cups
acl Safe_ports port 873 # rsync
acl Safe_ports port 901 # SWAT
acl purge method PURGE
```

```
acl CONNECT method CONNECT
#acl mynetwork src 172.16.171.0-172.16.171.150
acl ntlm proxy_auth REQUIRED

# Only allow cachemgr access from localhost
#################################
http_access allow manager localhost
http_access deny manager
# Only allow purge requests from localhost
http_access allow purge localhost
http_access deny purge
# Deny requests to unknown ports
http_access deny !Safe_ports
# Deny CONNECT to other than SSL ports
http_access deny CONNECT !SSL_ports

#http_access allow localhost

# And finally deny all other access to this proxy
#http_access allow mynetwork
http_access allow ntlm
http_access deny all
http_reply_access allow all
icp_access allow all
cache_effective_user makram
visible_hostname ubuntuserver
append_domain .ma-pme.fr
forwarded_for off
coredump_dir /var/spool/squid
```

6.2 Fichier de configuration de Samba :

```
[global]
    workgroup = MA-PME
    security = ADS
    winbind separator = /
    encrypt passwords = yes
    netbios name = UBUNTUSERVER
    server string = %h server (Samba %v)
    idmap uid = 10000-20000
    idmap gid = 10000-20000
    winbind enum users = yes
    winbind enum groups = yes
    allow trusted domains = yes
    password server = ad.ma-pme.fr
    realm = MA-PME.FR
```

```
winbind use default domain = yes
winbind nested groups = yes
```

6.3 Fichier de configuration de Kerberos :

```
[libdefaults]
        default_realm = MA-PME.FR
        clock_skew = 300
        ticket_lifetime = 24000
        default_tkt_enctypes = des3-hmac-sha1 des-cbc-crc
        default_tgs_enctypes = des3-hmac-sha1 des-cbc-crc
        dns_lookup_realm = false
        dns_lookup_kdc = true

[realms]
        MA-PME.FR = {
                kdc = ad.ma-pme.fr
                admin_server = ad.ma-pme.fr
                default_domain = MA-PME.FR
                }
[domain_realm]
        ma-pme.fr = MA-PME.FR
        ma-pme.fr = MA-PME.FR
```

6.4 Fichier hosts :

```
127.0.0.1 localhost ubuntuserver
172.16.171.135 ad.ma-pme.fr ad
```

6.5 Fichier resolv.conf :

```
search ma-pme.fr
domain ma-pme.fr
nameserver 192.168.25.2
```