RENPAR'16 / CFSE'4 / SympAAA'2005 / Journées Composants
Le Croisic, France, 5 au 8 avril 2005

Authentification Distribuée sur Grille de Grappes basée sur LDAP

Sébastien Varrette[1], Sébastien Georget[2], Jean-Louis Roch[3], Franck Leprévost[1]

[1] Université du Luxembourg, LIASIT/FNR, Luxembourg
[2] INRIA Sophia Antipolis, Service DREAM, Sophia Antipolis, France
[3] Projet APACHE (CNRS/INPG/INRIA/UJF), Laboratoire ID-IMAG, Grenoble, France

---

Résumé
Entre les clusters hautes performances et les grilles de PC se dessine une infrastructure intermédiaire constituée de l'interconnexion de clusters : la grille de grappes. Contrairement aux grilles de PC encore dédiées à des applications spécifiques, les grille de grappes doivent, comme les clusters, permettre à plusieurs utilisateurs d'exécuter des applications de natures différentes. Elles imposent également des contraintes supplémentaires car l'extension géographique pose des problèmes de disponibilité et de sécurité. Dans cet article nous nous intéressons à Grid5000, un projet qui se base sur une topologie de type grille de grappes, et plus particulièrement à son système d'authentification. Ce dernier constitue la brique de base qui doit permettre à un utilisateur d'avoir accès aux ressources de l'ensemble de la grille. Nous montrons dans un premier temps les limites des systèmes d'authentification classiques que sont les fichiers locaux et NIS dans une telle configuration. Nous proposons alors une alternative scalable basée sur le protocole LDAP permettant de répondre aux besoins des grilles de grappes, que ce soit en termes de disponibilité, de sécurité ou de performances.

Mots-clés : Architecture distribuée, authentification sécurisée, LDAP

---

1. Introduction

La rédaction de cet article est motivée par le besoin d'un système d'authentification robuste pour la plate-forme Grid5000 [1]. Ce projet national consiste à mettre en place un environnement pour la recherche dans le domaine du *grid computing*. À terme, ce seront 5000 processeurs qui seront répartis sur un minimum de 8 laboratoires sur 8 sites géographiques différents. Les chercheurs de chacun des laboratoires impliqués pourront déployer des applications sur la grille. Par conséquent, ils devront pouvoir s'authentifier sur chacune des machines connectées. Le système d'identification est donc un élément clé de ce projet et plus généralement dans le domaine des grilles de grappes puisqu'il permet l'attribution des ressources. Celui-ci devra répondre ici à trois contraintes de base :

- Disponibilité : il n'est pas envisageable que l'utilisation de la grille soit entravée par une défaillance ponctuelle du système ;
- Sécurité : même si Grid5000 est un système clos, il ne faut pas pour autant qu'un intrus potentiel puisse avoir accès aux mots de passe des autres utilisateurs ;
- Délégation : chaque laboratoire conserve une liberté d'action et doit pouvoir gérer lui-même ses utilisateurs.

Cet article est organisé comme suit : le §2 précise le contexte en présentant une classification des grilles existantes et en détaillant la notion de grille de grappes à laquelle correspond Grid5000. L'exposé sera principalement orienté vers les systèmes Linux qui constituent des acteurs majeurs dans le domaine du *grid computing*[1]. Nous verrons à travers la présentation des différents systèmes d'authentification disponibles dans ces environnements dans quelle mesure les résultats obtenus peuvent s'étendre à d'autres systèmes.

[1] IBM qui est le constructeur le plus représenté dans la liste des 500 machines les plus puissantes du monde (`http://www.top500.org`) revendique à lui seul 161 clusters Linux dans cette liste, soit trois quarts des systèmes IBM et près d'un tiers tous constructeurs confondus.

Après avoir introduit les services de nommage et d'annuaires (§2.3), nous abordons plus précisément le protocole LDAP (§3). Une large part de cet article est dédiée aux expérimentations (§4) où seront notamment étudiées plusieurs configurations basées sur ce protocole. L'analyse de ces expérimentations conduira à la proposition d'une architecture d'authentification pour la plate-forme Grid5000.

2. Contexte

2.1. Les environnements de type grille

Les grilles d'ordinateurs, comme définies dans [5], sont des infrastructures distribuées, composées de machines géographiquement dispersées, inter-connectées dans le but d'être exploitées conjointement. Ce type de plates-formes, apparues durant les années 90, a connu plusieurs évolutions. Elles peuvent aujourd'hui être classées en deux familles principales de grilles :

1. Les *"Desktop Grid"* [4] utilisent typiquement les ressources inexploitées des ordinateurs connectés à l'Internet. Bien que récemment intégrées dans la problématique générale des grilles de calcul [7], il nous semble pourtant plus correct de continuer à séparer les deux architectures.
2. Les *"Grilles de Calcul"* regroupent plutôt des clusters de machines dédiées. Un cluster (grappe en français) permet de connecter plusieurs machines sur un réseau local pour fournir un ensemble cohérent capable d'exécuter un très grand nombre de calculs. Chaque machine est un nœud du cluster qui est utilisé dans des cadres aussi variés que les calculs parallèles, la répartition de charge ou encore la tolérance aux fautes. Typiquement, l'accès à un cluster est limité à un certains nombre d'utilisateurs qui doivent s'authentifier via un identifiant et un mot de passe. Ces informations sont le plus souvent diffusées par le service **NIS**[2] qui permet de distribuer des données de configuration (utilisateurs, mots de passe...) entre les nœuds du réseau. Pour une grille de calcul, ce système doit être étendu pour supporter le passage à l'échelle.

Une échelle basée sur le nombre d'utilisateurs et l'homogénéité dans l'administration système des ressources de calculs permet de subdiviser les grilles de calculs en sous-catégories :

- Les "Multi-Clusters" qui regroupent plusieurs sites géographiquement proches administrés par une même personne et gérant de l'ordre de 10^2 utilisateurs.
- Les "grilles de grappes" (ou grilles légères [3]) qui agrègent plusieurs sites dispersés grâce à l'Internet. Les sites sont administrés par des personnes différentes mais suffisamment proches pour accepter la mise en place de conventions (par exemples pour la résolution des noms de machines, le système d'authentification, etc.). Une telle topologie (voir fig. 1) gère de l'ordre de 10^3 utilisateurs et correspond à l'architecture de Grid5000.
- Les grilles au sens de Globus[6] regroupent des sites administrativement clos et gèrent un plus grand nombre d'utilisateurs. Les solutions d'authentification proposées dans ce cadre sont évidemment applicables à Grid5000 mais demeurent trop lourdes à mettre en œuvre. Cet article propose une solution plus simple et plus efficace, adaptée aux grilles de grappes.

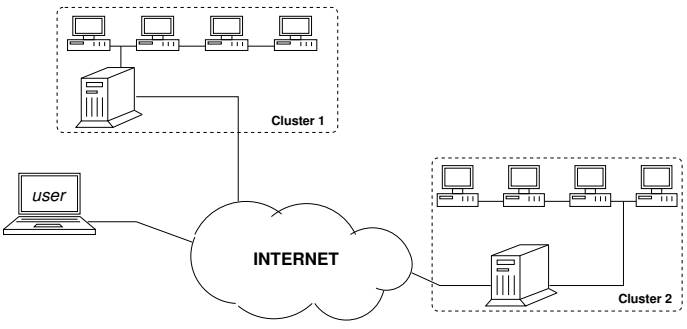

Figure 1: Contexte d'utilisation : la grille de grappe (ou grille légère)

[2] Network Information System, ou Yellow Pages yp. Ce service sera plus amplement détaillé au §2.3.1.

2.2. Authentification des utilisateurs en environnement Linux

Sous Linux, l'authentification d'un utilisateur repose sur deux composants :

1. Le système PAM (Pluggable Authentication Module) qui permet l'intégration transparente de diverses technologies d'authentification (comme le standard UNIX basé sur `crypt`, RSA, DCE, LDAP...) dans des services du système comme `login`, `passwd`, `rlogin`, `su`, `ftp`, `ssh`... sans avoir à les modifier. Si au départ PAM n'était implémenté que sous Solaris, c'est aujourd'hui un standard d'authentification sur de nombreuses distributions Linux (Redhat, Debian...). PAM fournit une API par laquelle les requêtes d'authentification sont mappées sur des actions spécifiques aux technologies (et qui sont implémentées dans des "modules PAM"[3]). Ce mapping est fait par les fichiers de configuration PAM dans lesquels il est spécifié pour chaque service les mécanismes d'authentification utilisables. Ainsi, le module pam_ldap, implémenté dans la librairie partagée `pam_ldap.so`, permet aux utilisateurs et aux groupes de s'authentifier à l'aide du service LDAP (présenté au §3).
2. Le système NSS (Name Service Switch) : une fois l'utilisateur authentifié, de nombreuses applications ont toujours besoin d'accéder aux informations relatives à cet utilisateur. Ces informations sont traditionnellement contenues dans des tables qui sont fournies soit par des fichiers textes locaux (`/etc/passwd`, `/etc/shadow`, `/etc/group`, etc.), soit par un service de nommage[4] comme nous le verrons dans la suite. Lorsqu'un nouveau service de nommage (comme LDAP) est introduit, il peut être implémenté soit dans une librairie C (comme c'est le cas pour NIS ou DNS) ou au sein de l'application qui utilisera ce service de nommage. Mais cela peut être évité en utilisant une API commune et générale dédiée au service de nommage qui réalise une couche intermédiaire entre une application et un ensemble de services de nommage. C'est ainsi que fut implémenté le "Name Service Switch" qui permet d'obtenir des informations de nombreux services de nommage à travers une API commune. NSS utilise un fichier de configuration `/etc/nsswith.conf` dans lequel sont précisés dans des listes ordonnées les services de nommage à utiliser pour récupérer telle ou telle information (comme les bases `passwd`, `shadow` ou `group`)

2.3. Notion de service de nommage et d'annuaire

Pour un ordinateur personnel non connecté en réseau, un utilisateur s'authentifie en général via les tables locales `passwd` et (sous réserve d'existence) `shadow` qui se présentent sous la forme de fichiers stockés dans le répertoire `/etc/`. De même, la traduction d'un nom de machine en adresse IP se fait par la table locale `hosts` stockée sous forme de fichier.

Dans le cas d'un cluster, il est préférable de disposer d'un *service de nommage* qui centralise toutes ces informations sur un ou plusieurs serveurs. Elles pourront alors être diffusées sur un domaine réseau de machines autorisées (typiquement, le cluster). Sans un tel service, chaque nœud du cluster devrait maintenir sa propre copie de la base d'informations, ce qui multiplie d'autant le coût de l'administration du cluster (de plus, comme nous le verrons en §4.2, les recherches dans ce fichier local sont en général séquentielles donc peu optimisées pour une utilisation sur cluster). Ce principe reste évidemment valable pour le cas des grilles de grappes mais il se pose alors le problème du passage à l'échelle. Parmi les exemples de services de nommage, les principaux sont le DNS (Domain Name Service) pour la résolution de nom de machine, NIS et NIS+ (version améliorée de NIS) qui seront détaillés plus loin. C'est la librairie NSS (voir §2.2) qui associe à chaque table la liste des sources disponibles (service de nommage, fichier local, etc.).

2.3.1. NIS

Le Network Information Service fut introduit par SUN en 1985 afin de centraliser l'administration des informations systèmes. Ces informations sont stockées sous forme de maps (tables de correspondance entre une clé et une valeur) dans de simples bases indexées (db, dbm...) accessibles par des appels RPC[5]. NIS fonctionne sur un modèle maître-esclave (les modifications se font sur le maître, elles sont répercutées sur l'(es) esclave(s) du domaine), mais il ne permet pas le traitement de volumes importants

[3] Exemple de modules PAM : `pam_unix.so`, `pam_ldap.so`...

[4] 'Naming Service' en anglais

[5] Remote Procedure Call

de données, et à chaque modification, c'est l'ensemble de la base qui est transférée (réplication totale). De plus, il est particulièrement difficile d'organiser les données sous une forme hiérarchique. Enfin la sécurité d'accès à ce service est très faible. Malgré tous ces inconvénients, NIS reste un système très utilisé au niveau des clusters et des réseaux locaux par sa simplicité d'installation, en particulier au niveau client.

2.3.2. NIS+

NIS+ fut la réponse de SUN aux inconvénients de NIS évoqués au paragraphe précédent. NIS+ introduit la propagation des données entre maître-esclave de manière incrémentale, en ajoutant la notion d'arbre hiérarchique pour les données. Enfin, l'utilisation de certificats permet de combler le problème de sécurité. L'obligation d'une architecture hiérarchique de haut en bas dans NIS+, qui implique une coopération entre les administrateurs systèmes des différents départements d'une organisation, ainsi que la gestion des paires clé privée/public fut finalement un frein au passage de NIS à NIS+. Pourtant, cette approche préfigure beaucoup les concepts utilisés par la suite dans LDAP.

2.3.3. Les annuaires électroniques et leur utilisation comme service de nommage

Un annuaire électronique est une base de données spécialisée qui permet de partager une base d'informations sur un réseau. Cette base peut contenir toutes sortes de données, comme des coordonnées téléphoniques ou des données systèmes. Les annuaires peuvent également être utilisés comme services de nommage. La normalisation ISO dédiée aux annuaires électroniques intervient dans le cadre de la norme X.500. Cette norme reste cependant trop complète et n'est pas réellement adaptée aux réseaux TCP/IP. LDAP [10] (Lightweight Directory Acces Protocol, voir §3) est une version allégée le la norme X.500 exploitable sur TCP/IP et utilisant les mêmes concepts que le DNS. Dans la suite, et bien que des solutions propriétaires existent, nous utilisons l'implémentation éprouvée et Open Source de LDAP développée à l'université du Michigan : OpenLDAP (`http://www.openldap.org`).
Pour finir, il est important de noter qu'il existe d'autres alternatives aux services de nommage pour assurer une authentification sécurisée des utilisateurs sur une grille [9], mais cela dépasse le cadre de cet article.

3. LDAP

LDAP, comme tout annuaire électronique, fonctionne de façon similaire à une base de données, même si quelques différences subsistent comme nous le verrons au §3.3. LDAP apporte également de nombreuses garanties en terme de sécurité grâce à des mécanismes de chiffrement (SSL/TLS) et d'authentification (SASL[6]). Couplés à des mécanismes de règles d'accès (ACL[7]), ils permettent de protéger les transactions et l'accès aux données. Par tous ces avantages, LDAP est un support de choix pour remplacer NIS dans la gestion de l'authentification des utilisateurs. Nous nous intéressons dans cet article à la version 3 de LDAP, référencée sous le nom LDAPv3 [10].

3.1. Fonctionnement de LDAP

Le service d'annuaire LDAP est basé sur un modèle client/serveur. Un ou plusieurs serveurs LDAP contiennent les données. Un client LDAP se connecte à un serveur et soumet sa requête. Il reçoit en retour une réponse ou un pointeur (on parle de *referral*) sur l'endroit (typiquement un autre serveur) où le client pourra trouver plus d'informations. Quelque soit le serveur auquel le client se connectera, il aura la même vue de l'annuaire : un nom référera la même entrée quelque soit le serveur accédé, comme pour DNS. Il existe plusieurs configurations possibles :

1. *Serveur local* : où un serveur unique est capable de traiter toutes les requêtes des clients.
2. *Serveur local avec referrals* : le serveur est configuré pour pouvoir traiter les requêtes du domaine local et renvoyer un "referral" (une référence, un pointeur) vers un serveur supérieur capable de répondre aux requêtes en dehors du domaine local.
3. *Serveur répliqué* : LDAP permet en effet de propager les changements effectués sur un serveur maître vers un ou plusieurs serveurs esclaves. La réplication peut n'affecter qu'une partie de la

[6] Simple Authentication and Security Layer
[7] Access Control List

base : le réplication partielle est donc possible.

4. *Serveur distribué* : La base est partitionnée en plusieurs sous-bases, répliquées ou non, qui sont accessibles via un ensemble de referrals vers des serveurs supérieurs ou inférieurs.

Ces deux dernier modes vont particulièrement nous intéresser dans le cadre de l'authentification sur grille de grappes. Les données LDAP sont structurées dans une arborescence hiérarchique (comparable à celle des systèmes de fichiers UNIX) appelée Directory Information Tree (DIT). Chaque nœud de l'arbre correspond à une *entrée*[8] de l'annuaire. Au sommet de cet arbre (appelé Directory Information Tree-DIT) se trouve la racine ou suffixe. Un exemple de DIT est fournit en figure 2.

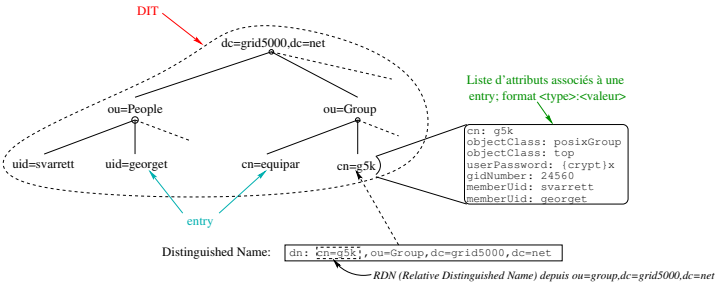

Figure 2: Exemple de DIT : cas de la gestion des utilisateurs

DC	domain components
OU	organizational unit
O	organization
CN	common name
SN	surname
UID	user ID

Figure 3: Principales abréviations utilisées dans le champ DN

Chaque entrée est référencée de manière unique dans le DIT par son *distinguished name (DN)*. Cette unicité est obtenue par la combinaison des attributs listés dans le tableau 3.

3.2. La sécurité dans LDAP

Le succès de LDAP (et son intérêt dans le cadre de Grid5000) vient de sa capacité à adresser les problèmes de sécurité suivants :

- les accès non autorisés (authentification)
- les droits d'accès aux données (autorisation)
- la confidentialité et l'intégrité des communications avec le serveur.

Le gros du travail est de déterminer les règles d'accès aux données. Pour cela, LDAP utilise les ACLs (Access Control Lists). Le serveur peut être de type read-only ou read-write. Dans les deux cas il faut déterminer pour chaque attribut quel est son niveau de confidentialité (un mot de passe est plus sensible qu'une adresse mail) et quel utilisateur ou quelle application pourra y accéder en lecture (tout le monde, certains utilisateurs, uniquement les administrateurs...) ou en écriture (utilisateur, manager, administrateur).
Pour pouvoir accéder à l'annuaire LDAP, le client LDAP doit d'abord s'authentifier. LDAPv3 propose plusieurs mécanismes d'authentification (anonyme, simple par mot de passe en clair même si c'est à éviter, simple encapsulé dans SSL/TLS, SASL), dont certains sont particulièrement adaptés au cas des grilles de grappes (les deux derniers). Si l'authentification réussit, alors le client peut envoyer une requête au serveur qui vérifiera si ce dernier est autorisé ou non à effectuer la requête (via les ACLs).

3.3. LDAP vs. DB

Comme nous l'avons vu : le fonctionnement interne de LDAP reste très proche de celui d'une base de données. Néanmoins, il subsiste quelques différences qui sont résumées dans le tableau 1.

3.4. LDAP vs. NIS

L'objectif de cet article est de comparer une solution d'authentification basée sur LDAP par rapport à NIS. Le tableau 2 dresse un premier bilan des caractéristiques de chaque solution. Les expérimentations du §4 compléteront cet aperçu.

[8] 'entry' où 'directory service entry' (DSE) dans la littérature anglaise.

Critère	LDAP	Base de Données
Rapport lecture/écriture	optimisé en lecture	lecture/écriture
Extensibilité	facile (schéma LDAP)	difficile (via schéma entité-association)
Distribution des tables	inhérente	rare [8]
Réplication	possible	possible
Modèle transactionnel	simple	avancé
Standard	oui	non (spécifique à un SGBD)

Tableau 1: Avantages/inconvénients de LDAP sur les bases de données

Critère	LDAP	NIS
Port	spécifique (389/636 par défaut)	arbitraire (appel RPC)
Chiffrement des données	possible	impossible
Mécanisme de contrôle d'accès	oui	non
Distribution des tables	oui	non
Réplication	oui (réplication partielle possible)	oui (uniquement totale)
Sémantique des recherches	avancée	simple

Tableau 2: Avantages/inconvénients de LDAP sur NIS

4. Expérimentations

4.1. Préambule

La comparaison des différents systèmes d'authentification a été réalisée en se plaçant du côté du client. Ainsi, les mesures prennent en compte les trois éléments de la chaîne d'identification : le module PAM, la couche de transport entre client et serveur et le temps de réponse du serveur.
Cette approche permet d'obtenir des résultats qui correspondent à la réalité de l'utilisation des systèmes et non aux performances théoriques que peuvent atteindre les serveurs.

4.2. Modèle local

En plus de poser des difficultés d'administration importantes, la réplication en local sur chaque nœud des fichiers `passwd`, `shadow` et `group` n'est pas la solution la plus efficace. Comme nous pouvons le constater dans les courbes de la figure 4, elle pose des problèmes de performances.

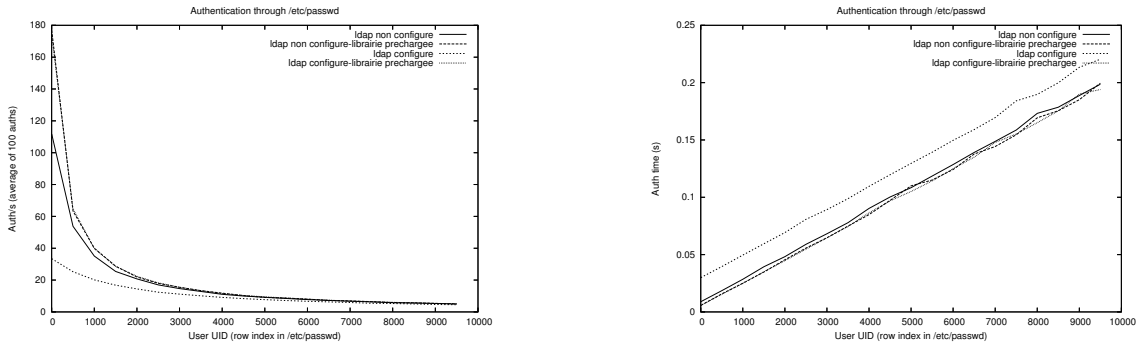

Figure 4: Modèle par tables locales : impact du positionnement dans la table sur le temps d'authentification.

Lors d'une authentification, ces fichiers sont lus linéairement pour rechercher une entrée demandée. Le temps nécessaire à l'identification dépend donc directement du nombre d'entrées dans ces fichiers. Dans le cadre d'un système avec un nombre important d'utilisateurs, comme c'est le cas avec les clusters

ou les grilles, il s'agit de la solution ayant les performances les plus faibles. Nous pouvons également remarquer sur les courbes présentées que le fait que les librairies utilisées dans l'authentification soient préchargées ou non influence les performances de l'authentification.

4.3. Comparaison NIS / gestion locale

Nous avons vu que le modèle local n'est pas la solution la plus efficace. En particulier, le temps d'authentification dépend directement du nombre d'utilisateurs enregistrés dans les tables. Dans le cas d'une gestion par un serveur NIS, les courbes de la figure 5 montrent que le temps d'authentification reste globalement indépendant du nombre d'entrées dans la table. Nous verrons que ce comportement est également obtenu avec un serveur LDAP dont les données sont correctement indexées (§4.4.1).

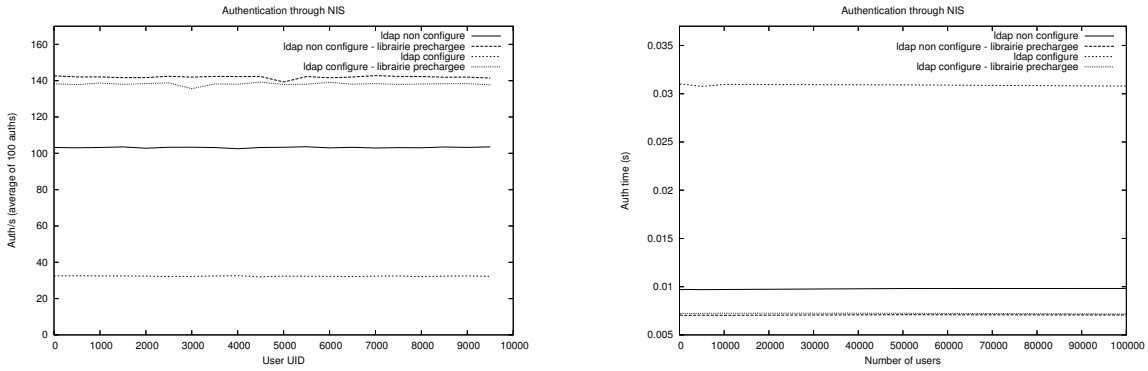

Figure 5: Cas de NIS : indépendance de l'uid authentifié et de la taille de la base sur le temps d'authentification.

Nous nous intéressons maintenant à diverses configurations du (des) serveur(s) LDAP afin d'élaborer la proposition la plus adaptée au cas des grilles de grappes (et de Grid5000 en particulier).

4.4. Modèle centralisé client/serveur

Dans ce modèle, le serveur LDAP est configuré de façon similaire à un serveur NIS : le contenu de toutes les tables se trouve sur un même serveur accessible par tous les nœuds de la grille.

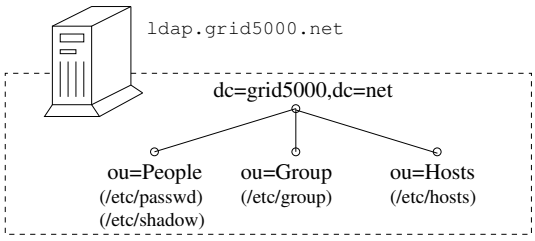

Figure 6: Modèle centralisé client/serveur

La figure 6 illustre la structure des tables utilisées sur le serveur LDAP. Avant de comparer cette solution avec NIS, nous avons tenu a évaluer l'impact de la configuration du serveur LDAP sur ses performances. Dans ce but, nous montrons l'importance d'un indexage optimisé des données (§4.4.1) et l'impact de l'utilisation d'un journal d'événement (§4.4.2). Le type de backend utilisé en revanche ne semble pas atténuer les performances du serveur (§4.4.3).

Dans tous les cas, nous comparons l'impact de SSL sur les performances générales du serveur.

4.4.1. Influence de l'indexage des données dans LDAP

Lors de l'installation d'un serveur LDAP, un indexage basique est proposé. Nous montrons qu'il *ne faut pas* utiliser cet indexage directement. Un indexage approprié[9] doit être utilisé. Les courbes de la figure 7 comparent l'impact de ces deux indexages sur les performances du serveur. Nous pouvons remarquer qu'un indexage optimisé garantit des performances constantes avec le nombre d'entrées dans la base LDAP. Il est également important de noter le coût important de la sécurité puisque l'utilisation de SSL divise les performances par 9. C'est le prix à payer pour l'authentification des ressources et le chiffrement des communications entre les nœuds de la grille et le serveur. Ce rapport se confirmera dans les expériences suivantes (voir §4.4.5).

[9] Voir `http://www.openldap.org/faq/data/cache/42.html` par exemple.

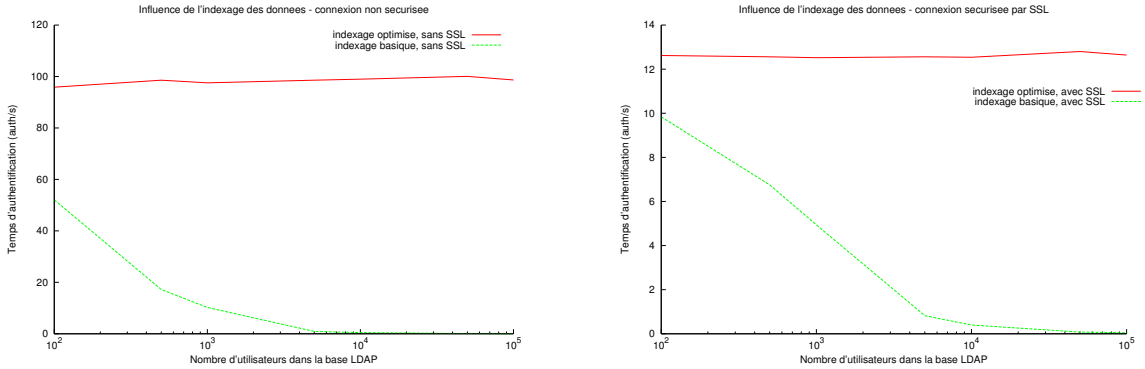

Figure 7: Influence de l'indexage des données sur les performances du serveur LDAP.

4.4.2. Impact du niveau de logs d'un serveur LDAP

Avec OpenLDAP, il est possible de configurer très précisément le journal des événements (on parle de *logs*) du serveur. Cette configuration de divise en niveaux, chacun responsable du log d'un événement particulier. Les niveaux possibles et les événements associés sont listés dans le tableau 8.

Niv.	Description	Impact
-1	enable all debugging	++++
0	no debugging	–
1	trace function calls	+++
2	debug packet handling	0
4	heavy trace debugging	+
8	connection management	++
16	print out packets sent/received	0
32	search filter processing	+
64	configuration file processing	0
128	access control list processing	+++
256	stats log connections/op/results	+
512	stats log entries sent	≃0
1024	print coms with shell backends	0
2048	print entry parsing debugging	0

Figure 8: Les niveaux de log dans OpenLDAP (source :[2])

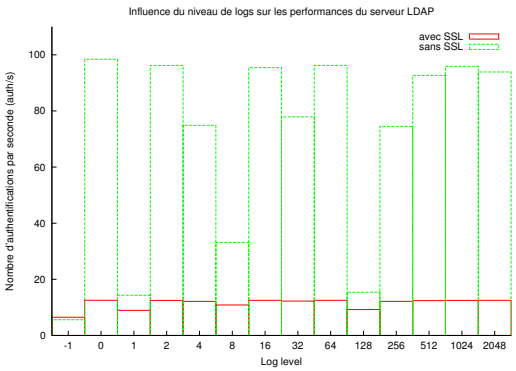

Figure 9: Impact des logs sur les performances

La combinaison de plusieurs niveaux se fait en les ajoutant. La figure 9 montre que chaque niveau à un impact différent sur les performances du serveur. En particulier, le niveau "-1" est inutile car il dispense trop d'informations qui ne sont pas forcément pertinentes et diminue les performances du serveur de 95%. Nous constatons également que l'utilisation de SSL atténue l'impact des logs sur les performances.

4.4.3. Influence du backend utilisé

La manière dont sont stockées les données d'un serveur OpenLDAP dépend d'un *backend*. Il existe plusieurs backend possibles ("`ldbm`" par défaut). Les premières mesures effectuées pour comparer les deux backend les plus utilisés (`ldbm` et `bdb`) n'ont pas montré de réel avantage d'une solution par rapport à l'autre sur les performances générales du serveur.
Cependant, il faudrait des analyses plus poussées pour confirmer ou infirmer cette hypothèse. De plus, tous les backend possibles n'ont pas été testés. Ces comparaisons feront l'objet de prochains travaux.
Enfin, on verra en §4.5.2 l'utilisation d'un backend particulier ("`meta`") pour l'intégration de la réplication des branches LDAP

4.4.4. Influence du chiffrement symétrique et du protocole d'échange de clé dans SSL

Nous avons également tenu à savoir si dans le cas de l'utilisation de SSL, un algorithme de chiffrement symétrique et/ou un protocole d'échange de clés devait être privilégié ou non. A cet effet, nous avons testé les niveaux de sécurité HIGH[10] (proposant d'utiliser les algorithmes AES ou 3DES , avec des clés de 256 ou 168 bits) et MEDIUM (où les algorithmes négociables sont AES ou RC4 avec des clés de 128bits). En ce qui concerne les protocole d'échange de clé (Key Agreement) et toujours en utilisant la nomenclature OpenSSL, nous avons étudié l'impact de kDH (Diffie Hellman) et kRSA (RSA)

Aucune différence sensible n'a été constatée. Cependant les tests utilisés pour la mesure du temps d'authentification n'échangeaient peut-être pas assez de données pour que la différence entre les algorithmes apparaisse. Il était légitime de penser qu'un algorithme AES soit plus performant qu'un 3DES. Des test plus approfondis devront être menés dans ce sens.

4.4.5. Influence de SSL sur LDAP

Comme nous l'avons vu au §2.2, l'authentification d'utilisateurs dans un environnement Linux repose sur deux composants : la librairie PAM et la librairie NSS. Dans le cadre d'une authentification via LDAP, une interface spécifique doit être utilisée pour chacune de ces librairies. Elles peuvent être configurées pour interroger le serveur LDAP en clair ou via SSL. Pour analyser finement l'impact de SSL sur les performances du serveur, il convient donc de distinguer l'utilisation ou non de SSL pour chaque librairie. Le tableau 3 fournit le résultat de cette analyse en ajoutant l'impact du pré-chargement des autres librairies utilisées dans l'authentification sur les performances du serveur (caractérisées ici par le nombre d'authentifications par seconde réalisées).

PAM		**NSS**		**#authentifications/s**		
`ldap`	`ldaps`	`ldap`	`ldaps`	intervalle	moyenne	avec preload
X		X		entre 94 et 97	95.99	145.9
X			X	entre 92 et 95	94.15	144.4
	X	X		entre 12 et 13	12.46	14.28
	X		X	entre 12 et 13	12.51	14.27

Tableau 3: Impact de l'utilisation de SSL pour les librairies PAM et NSS sur les performances - Mesures intra-cluster

Ces résultats ont été obtenus pour un serveur et un client appartenant à un même cluster (donc avec une latence faible entre les deux). De façon générale, nous pouvons constater l'utilisation de LDAP avec SSL (ce qui correspond au protocole `ldaps`) à un fort impact sur les performances du serveur. Ce résultat peut être affiné en remarquant que la librairie NSS en mode SSL a une influence beaucoup plus faible sur les performances que la librairie PAM configurée dans le même mode.

L'influence de SSL sur les performances s'expliquent facilement lorsqu'on analyse le nombre de messages échangés lors d'une authentification par `ldap` ou par `ldaps`. En effet, dans le cadre d'une authentification par `ldap` (sans SSL), 45 messages LDAP sont échangés avec 35 messages TCP. En revanche, dans le cadre d'une authentification par `ldaps`, 70 messages TLS (LDAPS) sont nécessaires, auxquels s'ajoutent 47 messages de négociation TCP. Les communications sont donc plus importantes avec `ldaps` et il faut également prendre en compte le temps nécessaire au chiffrement/déchiffrement des paquets.

4.4.6. Influence de la latence inter-cluster entre client/serveur LDAP

Nous avons réalisé la même expérience que celle décrite au §4.4.5 mais en mesurant l'impact de la latence sur les performances. Les mesures présentées dans le tableau 4 ont été réalisées pour un client LDAP présent sur un autre cluster que celui du serveur[11]. Le premier constat est que par rapport aux mesures intra-cluster, la latence influe énormément sur les performances de l'authentification (on est ici environ 4 fois plus lent). Il est important de noter également que l'écart type entre les mesures effectuées est nettement plus important . Cette diminution de performances s'explique par la latence et les perturbations du réseau.

[10] selon la nomenclature OpenSSL `http://www.openssl.org`

[11] Plus précisément, le serveur était localisé à Sophia Antipolis et le client à Grenoble

PAM		NSS		#authentifications/s	
ldap	ldaps	ldap	ldaps	intervalle	moyenne
X		X		entre 15 et 22	20.1
X			X	entre 7 et 23	17.6
	X	X		entre 6 et 7	6.89
	X		X	entre 5 et 7	6.79

Tableau 4: Impact de la latence (inter-cluster) pour les librairies PAM et NSS sur les performances

4.4.7. Influence de la latence entre client/serveur NIS

Nous avons réalisé une expérience dans des conditions similaires à celles des paragraphe 4.4.5 et 4.4.6 mais avec un serveur NIS. Les mesures effectuées sont résumées dans le tableau 5.

Type latence	#authentifications/s			
	intervalle	moyenne	intervalle avec preload	moy. avec preload
Intra-Cluster	entre 230 et 310	263.0	entre 266 et 338	290
Inter-Cluster	entre 32 et 37	35.1	entre 31 et 38	35.3

Tableau 5: Impact de la latence sur les performances d'une authentification basée sur NIS

Généralement, une authentification basée sur NIS reste donc plus performante car il y a moins de communications. La différence entre LDAP et NIS se fera sur un autre aspect intéressant de LDAP : la possibilité de distribuer et de répliquer partiellement le contenu des tables. La section suivante aborde ces configurations et leurs impacts sur les performances de l'authentification.

4.5. Modèle distribué "plat"

L'un des gros avantages de LDAP est qu'il permet de distribuer les tables sur plusieurs serveurs. Cette distribution est envisageable sur deux niveaux :

- Soit en utilisant le mécanisme de réplication partielle de LDAP qui permet de répliquer le contenu de certaines branches de la base (par exemple, l'arbre `o=grenoble,dc=grid5000,dc=net`) administré par un serveur maître A dans la base d'un serveur B esclave. Un démon permet alors de diffuser les modifications de la branche maître sur les serveurs esclaves. Il est important de noter que NIS ne propose qu'un mécanisme de réplication totale
- Soit en utilisant le mécanisme de *referral* qui permet de "pointer" vers un serveur hébergeant une branche spécifique de la base. Dans ce cas, les données ne sont pas dupliquées et le serveur renvoie simplement au client l'adresse du serveur administrant la branche demandée. A charge ensuite au client de contacter ce serveur pour obtenir les entrées recherchées.

Une illustration du modèle de distribution pour Grid5000 est proposée dans la figure 10.

4.5.1. Impact du mécanisme de "referral"

Dans un premier temps, nous avions envisagé l'utilisation de referrals pour distribuer au vrai sens du terme la base LDAP. Après quelques tests, il semble que l'utilisation des referrals (un vers chaque site) ne soit pas une solution efficace. Cela est dû au fonctionnement par défaut des referrals qui est illustré dans la figure 11 : à partir du moment où un referral existe, le serveur renvoie ce referral vers le client qui envoie une nouvelle requête au serveur pointé. Ainsi, supposons vouloir effectuer l'authentification de l'utilisateur `svarrett` géré par le serveur A (voir fig.11). Même si la machine cliente appartient au même cluster que le serveur A, une nouvelle requête sera générée vers le serveur B. Une piste possible pour éviter cela serait l'utilisation d'un

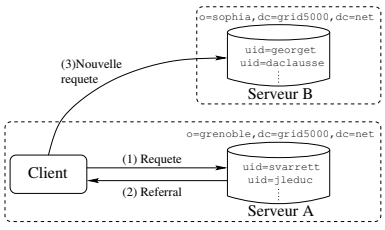

Figure 11: Fonctionnement d'un referral

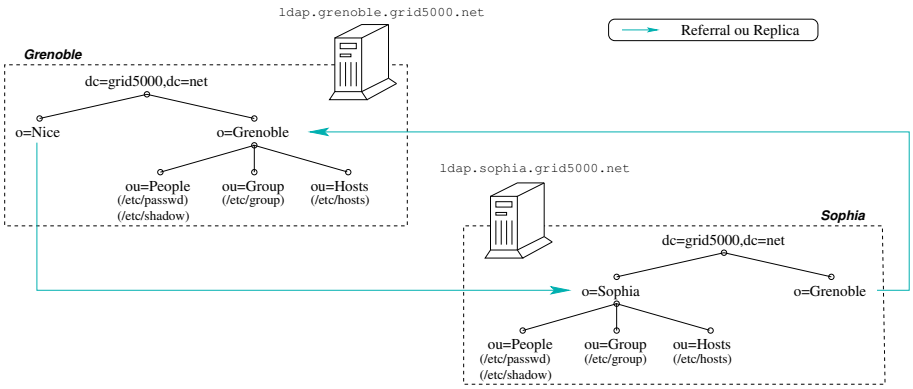

Figure 10: Modèle de distribution plate pour l'authentification par LDAP

proxy de cache. Cette alternative fera l'objet de travaux futurs. Dans l'immédiat, les pertes de performance engendrées par l'utilisation des referrals nous ont amenés à utiliser le mécanisme de réplication partielle qui est maintenant présenté.

4.5.2. Expérimentations sur le mécanisme de réplication partiel

Les mesures présentées ici ont été effectuées en utilisant la version 2.2 de OpenLDAP dans la configuration illustrée par la figure 10 en mode réplica. La seule réplication des branches entre les serveurs ne suffit pas : il faut une interface qui intègre ce contenu dans la hiérarchie LDAP. Le backend `meta` permet de réaliser cet interfaçage en fusionnant virtuellement plusieurs branches locales ou distantes. Dans ce dernier cas, l'architecture obtenue est proche de celle des referrals ce qui aura quelques inconvénients comme nous le verrons dans la suite. Les expérimentations ont été menées à partir d'un client appartenant au cluster de Grenoble. Les performances du mode réplica avec un backend `meta`

	backend `meta`		**#auth./s** (`ldap`)			**#auth./s** (`ldaps`)			
Mode	**Grenoble**	**Sophia**	**min.**	**moy.**	**max.**	**min.**	**moy.**	**max.**	**Commentaires**
centralisé	désact.	désact.	84.9	89.0	107.3	16.7	17.4	17.9	mode centralisé (voir §4.4)
réplica	désact.	désact.	82.9	96.6	108.9	17.2	17.4	17.8	Impact de la configuration des réplicas
réplica	local	local	82.0	87.9	97.1	16.9	17.1	17.5	Utilisateur géré à Grenoble
réplica	local	local	81.3	88.9	97.6	16.8	17.2	17.5	Utilisateur géré à Sophia
réplica	local	distant	35.8	36.8	38.3	13.3	13.4	13.7	Utilisateur géré à Grenoble impact du backend distant
réplica	local	distant	16.1	16.5	17.0	9.2	9.2	9.4	Utilisateur géré à Sophia impact du backend distant

Tableau 6: Expériences de réplication partielle des branches `o=grenoble` et `o=sophia`

utilisant des branches locales restent tout à fait correctes puisqu'aucune perte n'est observée par rapport au mode centralisé. La configuration d'un backend `meta` avec des branches distantes pose de nombreux problèmes de disponibilité tout en introduisant une plus grande latence dans les temps d'authentification. De plus, si le serveur distant devient indisponible, alors aucun utilisateur (d'aucune branche) ne pourra être authentifié. Nous déconseillons donc l'utilisation du backend `meta` avec des serveurs distants si ceux-ci risquent d'être indisponibles ou injoignables.

4.5.3. Impact de chaque mode sur la sécurité et la disponibilité sur la grille

Chacune des approches proposées (par réplication partielle ou par referral) a un impact similaire sur la sécurité générale de la grille. Dans chaque cas, la corruption d'un site (en fait du serveur LDAP) permet à l'attaquant de récupérer les mots de passes locaux sous forme hashée et/ou de créer un nouvel utilisateur et ainsi accéder à tous les sites. Il est impératifs que ces serveurs soient des bastions. En terme

de disponibilité, la disparition d'un site est bloquante pour l'authentification sur les autres sites avec l'approche par referral. L'utilisation d'un proxy cache devrait permettre de pallier à ce problème mais le coût reste à évaluer et ce problème n'apparaît pas en configuration réplication partielle avec backend `meta`. C'est pourquoi nous préconisons cette dernière approche dans le cas de Grid5000.

5. Conclusion

Les expériences menées ont confirmé les atouts du système NIS. Il s'agit d'une solution rapide à mettre en œuvre et performante. Elle trouve parfaitement sa place dans un cluster pour lequel il n'existe pas ou peu de contraintes en terme de sécurité. Le contexte des grilles fait passer ces contraintes au premier plan. La distance géographique qui sépare les sites ne permet plus d'utiliser un réseau dédié et contrôlé. Il est alors nécessaire d'assurer la confidentialité des informations échangées au niveau des protocoles utilisés. Comme nous l'avons vu dans le cadre du modèle centralisé, c'est ce que permet de faire le protocole `ldaps`.

Le passage des clusters aux grilles pose également des problèmes d'organisation entre plusieurs entités administratives distinctes. Il n'est plus possible de s'appuyer sur un modèle centralisé car chaque intervenant doit pouvoir gérer ses utilisateurs. Là encore nous avons vu que LDAP proposait plusieurs solutions pour répondre à ce problème, que ce soit grâce aux *referrals* ou à la réplication partielle. Cette dernière solution offre par ailleurs l'avantage de rendre chaque site indépendant ; en cas de défaillance au niveau d'un cluster, les autres sites ne sont pas pénalisés.

L'étude préliminaire des systèmes d'authentification distribués nous aura permis de proposer une solution robuste pour répondre aux besoins de Grid5000. L'infrastructure à base de réplications partielles est actuellement en cours de déploiement afin de procéder à des tests en situation réelle. Les prochains mois devraient permettre de la valider ou de faire apparaître de nouveaux besoins pour Grid5000.

Des évolutions sont déjà envisagées avec l'intégration d'informations complémentaires à l'annuaire LDAP comme l'insertion de données relatives au matériel installé. L'objectif est de créer un référentiel sur lequel s'appuieraient les différents services de la grille tels que le DNS[12], le *monitoring* ou encore le système de réservation.

Les auteurs tiennent à remercier Olivier Richard, Nicolas Capit et Julien Leduc, du laboratoire ID-IMAG, pour leurs contributions scientifiques et techniques.

Bibliographie

1. The Grid5000 project. `http://www.grid5000.org`.
2. OpenLDAP 2.2 Administrator's Guide, Feb 2004. `http://www.openldap.org/doc/admin22/`
3. Capit (N.). – *Cigri, Ciment Grid: une grille légère*. – Rapport technique, ACI GRID, 2004.
4. Fedak (G.), Germain (C.), N'eri (V.) et Cappello (F.). – Xtremweb: A generic global computing system. *In : In IEEE Int. Symp. on Cluster Computing and the Grid*. – 2001.
5. Foster (I.) et Kesselman (C.). – Globus: A metacomputing infrastructure toolkit. *International J. of Supercomputer Applications and High Performance Computing*, vol. 11, n° 2, Summer 1997, pp. 115–128.
6. Foster (I.), Kesselman (C.), Tsudik (G.) et Tuecke (S.). – A Security Architecture for Computational Grids. *In : Fifth ACM Conference on Computer and Communications Security Conference*, pp. 83–92. – San Francisco, California, 3–5 Novembre 1998.
7. Foster (Ian). – The anatomy of the Grid: Enabling scalable virtual organizations. *Lecture Notes in Computer Science*, vol. 2150, 2001.
8. Stonebraker (M.), Aoki (P. M.), Devine (R.), Litwin (W.) et Olson (M. A.). – Mariposa: A new architecture for distributed data. *In : International Conference on Data Engineering (ICDE)*, pp. 54–65. – 1994.
9. Varrette (S.), Roch (J.-L.), Denneulin (Y.) et Leprevost (F.). – Secure Architecture for Clusters and Grids. *In : Proceedings of the 2ème Conférence Internationale sur les Infrastructures Critiques CRIS 2004)*, éd. par IEEE. – Grenoble, France, October 2004.
10. Wahl (M.), Howes (T.) et Kille (S.). – *RFC 2251 - Lightweight Directory Access Protocol (v3)*. – Rapport technique, IETF, Dec. 1997. `http://www.ietf.org/rfc/rfc2251.txt`

[12] Domain Name Server : service de résolution de noms en adresse IP