

Avoir une identité unique dans un réseau hétérogène

LDAP ET LINUX À LA RESCOURSE!



Pascal.Jermini@epfl.ch, étudiant en Informatique &
Vittoria.Rezzonico@epfl.ch, étudiante en Mathématiques



LE PROBLÈME

Sous ce titre un peu mystérieux se cache le cauchemar de n'importe quel administrateur d'un réseau contenant un bon nombre de machines tournant sur du matériel, et surtout des systèmes d'exploitation hétérogènes. Ce genre de situation apparaît en général dans deux cas:

- premièrement quand un laboratoire entier entre dans une phase de transition d'une plate-forme à une autre. Le cas typique, est le passage des machines Sun tournant sous Solaris vers des PC, tournant sous Windows 2000 ou sous Linux, la tendance actuelle étant celle de passer de stations de travail haut de gamme, provenant de très gros constructeurs, vers des simples PC, beaucoup moins chers et surtout d'un entretien plus aisé. Ces transitions de plates-formes imposent souvent que les deux architectures cohabitent pendant un certain temps: le temps de faire migrer les données vers le nouveau système et surtout le temps de migrer les utilisateurs, qui sont habitués à l'utilisation de leurs vieilles machines.
- le deuxième cas où un réseau hétérogène peut s'imposer est celui d'un laboratoire nécessitant vraiment plusieurs architectures: pour donner un exemple, il y a les incontournables stations SGI, utilisées pour la visualisation, qui côtoient les PC standards, utilisés pour le travail de tous les jours (e-mail, écriture d'une lettre, etc.).

Les deux exemples cités ci-dessus permettent d'avoir une idée sur ce qu'est un réseau hétérogène: une collection de machines, tournant sur un nombre indéterminé de systèmes d'exploitation différents.

Ces deux cas de figure cachent un problème de taille: il est très probable que l'on veuille que chaque utilisateur puisse utiliser l'une ou l'autre machine (tournant sur deux systèmes d'exploitation différents) de manière transparente, c'est-à-dire que l'utilisateur puisse utiliser les mêmes nom d'utilisateur et mot de passe sur toutes les machines, et, si possible, retrouver les mêmes fichiers personnels partout. C'est ce concept de *login identique sur toutes les machines* qui définit le terme *Identité unique*.

A présent que les deux termes du problème ont été éclaircis, essayons de le résoudre! Nous voulons tout de même imposer une contrainte de taille: le tout doit se faire à un coût minimal, et on dispose déjà d'une machine de type PC, pas très puissante, mais qui se défend assez bien (dans notre cas: un AMD K6-2 400, avec 256 Mb de RAM).

Cette contrainte ne nous laisse pas vraiment le choix du système d'exploitation: on doit se porter sur des logiciels

gratuits. Pour notre problème on va utiliser le système d'exploitation qui a fait beaucoup de bruit ces derniers temps, surtout l'année dernière: Linux [1]. Il est vrai que l'on aurait pu utiliser FreeBSD [2], par exemple, mais étant donné que nous étions beaucoup plus familiarisés avec Linux, le problème du choix a été résolu assez rapidement.

PREMIÈRE TENTATIVE: NIS

Comme première tentative d'identité unique nous avons restreint un peu le type de machines et de systèmes d'exploitation utilisés: nous nous sommes orientés uniquement vers des machines tournant sous une variante ou une autre de Unix, en laissant de côté (temporairement) les machines Windows. Déjà rien qu'avec des machines Unix nous obtenions trois systèmes différents: Linux, Irix et Solaris. La solution la plus naturelle avec un environnement Unix est celle d'utiliser NIS ou bien sa variante NIS+. Le problème du choix entre NIS et NIS+ a été très rapidement résolu: il n'existe pas de serveur NIS+ tournant sous Linux. NIS se retrouve donc seul candidat.

Afin de garantir un minimum de sécurité, nous avons opté pour des *shadow passwords*, qui n'autorisent pas les utilisateurs sans droits d'administrateur à lire le fichier contenant les hachages des mots de passe, typiquement le fichier `/etc/shadow`. Cette décision d'utiliser les *shadow passwords* nous confrontera au premier problème de taille: les machines tournant sous Solaris 8 ne savent pas gérer du NIS avec ce genre de mots de passe. Le problème a été contourné de manière assez brutale: un script synchronise toutes les 15 minutes la base de mots de passe locale à la machine Sun avec celle disponible sur le serveur NIS. Ce genre de manipulation est permis, étant donné que, bizarrement, les outils tels que `ypcat` de Solaris arrivent à lire la liste des mots de passe *shadow* sur un serveur NIS.

En ce qui concerne les machines Windows il y aurait la possibilité d'utiliser un client NIS pour Windows [3]: nous n'avons pas essayé cette alternative, surtout à cause de l'échec partiel dû à Solaris.

LA SOLUTION IDÉALE: LDAP

La solution NIS n'étant pas satisfaisante, nous nous sommes tournés vers ce qui nous semblait être une technologie prometteuse: LDAP (voir à ce sujet, l'article de Claude

Lecommandeur dans le flash info 9/01: <http://sic.epfl.ch/publications/FI01/fi-9-1/9-1-page5.html>. Voilà: nous avons maintenant utilisé tous les éléments du titre de l'article! Il semblerait donc que cette solution marche!

Première étape: un serveur LDAP. Dans le monde OpenSource, le plus fameux est OpenLDAP [4]. Il offre de bons services et de nouvelles versions sont régulièrement disponibles. La configuration du serveur est assez simple, et après avoir fait migrer les utilisateurs dans la base de données LDAP il ne reste plus qu'à configurer les machines clientes. Pour les machines Linux la procédure est rapidement mise en œuvre: le système PAM permet de choisir la source d'authentification. Il existe à ce sujet plusieurs pages Web décrivant la procédure: en voici une parmi tant d'autres <http://www.imaginator.com/~simon/ldap/>.

Irix peut poser des problèmes, surtout avec les anciennes versions (antérieures à la version 6.5.13): le service *nscd* n'était pas en mesure de communiquer correctement avec le serveur LDAP. Une mise à jour résout le problème.

Après cela la marche à suivre est simple: le fichier `/etc/nsswitch.conf` permet de choisir la source d'authentification, et le fichier `/var/ns/ldap.conf` contient les informations telles que l'adresse du serveur LDAP. Dans ce même fichier il y a une modification subtile et importante à faire: la ligne `regsub USERPASSWORD{{CRYPT\}}{}` doit être transformée en `regsub USERPASSWORD{{crypt\}}{}`.

Pour Solaris, l'article se trouvant à l'adresse <http://www.ypass.net/solaris8/openldap/> explique pas à pas la démarche à suivre pour configurer un client tournant sous Solaris 8.

Il reste maintenant à intégrer les machines tournant sous Windows 2000 ou XP. Cela est beaucoup plus laborieux: Windows ne gère pas les *login* directement sur un serveur LDAP. Il faut pour cela passer à travers un serveur de domaine, lequel obtiendra les informations d'authentification sur le serveur LDAP. Le logiciel de référence en matière d'intégration de Windows dans un environnement Unix est Samba [5].

Samba: le pont entre deux mondes très différents

La version standard de Samba (2.2.2) a un support très limité du protocole LDAP, ce qui en fait un piètre candidat comme serveur de domaine. Il existe heureusement une branche alternative à Samba, appelée Samba-TNG [6] (Samba The Next Generation). Cette version diffère de celle standard sur plusieurs points: support avancé pour LDAP, contrôleur de domaine Windows assez complet, mais ce n'est qu'une version alpha. Ceci veut dire que le logiciel n'a pas la stabilité de Samba 2.2.2 (version qui a désormais fait ses preuves dans le domaine de la stabilité et des performances) et certaines fonctionnalités ne sont pas implémentées. Malgré la jeunesse de Samba-TNG, nous n'avons constaté aucun problème majeur de stabilité: le serveur a tourné pendant plusieurs semaines d'affilée, sans le moindre pépin. Le service a temporairement été interrompu pour une toute autre raison: un défaut du matériel a imposé l'arrêt de la machine. Un autre problème de Samba-TNG est son incapacité de servir comme serveur de fichiers: ses performances dans ce

domaine sont très limitées. De plus ses performances en tant que serveur WINS sont plutôt décevantes, il vaut mieux se rabattre sur Samba 2.2.2 pour ce service, où il est beaucoup plus mûr et plus testé.

Il est néanmoins possible de contourner le problème en installant les deux versions de Samba en parallèle comme expliqué plus tard.

Lors de la configuration de Samba-TNG il est indispensable d'inclure le support LDAP ainsi que les capacités de contrôleur de domaine. Le premier se définit avant la compilation, tandis que le deuxième est un paramètre du fichier `smb.conf` (paramètre `domain logons = yes`). Pour que Samba-TNG se comporte comme un contrôleur de domaine il ne faut pas oublier de spécifier le partage *Netlogon*, défini toujours dans le fichier `smb.conf` de cette manière, par exemple:

```
[netlogon]
comment = The domain logon service
path = /home/samba/netlogon
writeable = no
locking = no
```

Samba-TNG repose sur une architecture basée sur plus de dix *daemons*, chacun avec sa tâche bien spécifique. Un des utilitaires qui va être très pratique pour l'administrateur du domaine NT est appelé *samedit*(8). C'est un outil de commandes en ligne, mais qui malheureusement montre certains défauts de Samba-TNG: certaines de ses commandes ne marchent pas (le serveur renvoie des messages d'erreur), et l'effacement d'un utilisateur ou d'une machine n'est pas supporté. Cela est un problème relativement mineur: on peut écrire un *script* en *bash* ou en un quelconque autre langage pour l'effacement d'une entrée dans le serveur LDAP.

Mots de passe synchronisés: l'atout majeur de l'identité unique

L'un des atouts de l'identité unique est bien évidemment la possibilité d'avoir un même mot de passe sur toutes les machines, soient-elles Windows ou Unix. Naturellement Unix et Windows n'utilisent pas le même système de hachage des mots de passe, ce qui pose quelques problèmes. Il existe une solution à ce problème, qui malheureusement n'est efficace que partiellement et qui donne un avantage aux machines Unix: on peut remplacer le programme classique *passwd*(1) qui permet le changement des mots de passe par un *script*, qui génère les deux types de hachages et qui les stocke aux bons endroits sur le serveur LDAP [7]. Pourquoi cette méthode n'est-elle pas efficace à 100%? D'une part, il faut installer les outils clients LDAP sur les machines Unix (ou au moins l'outil *ldapmodify*(1), qui permet de modifier des clés de la base de données LDAP) et d'autre part, la synchronisation des mots de passe ne fonctionnera pas lorsque l'on demandera un changement depuis une machine Windows. La raison est qu'en utilisant l'utilitaire fourni avec Windows (celui accessible par **Ctrl+Alt+Del** ➔ *Change password...*) le mot de passe est haché avant d'être envoyé au contrôleur de domaine (sécurité oblige!), et ce dernier ne sera pas capable de revenir en arrière (trouver le mot de passe à partir du hachage, fort heureusement!). Il n'a donc aucun

moyen de connaître le mot de passe Windows qui devrait ensuite être haché en format Unix.

Encore une remarque assez importante: les scripts présentés en [7] sont conçus pour être utilisés avec des mots de passe hachés avec l'algorithme MD5, et non pas avec l'algorithme `crypt()`, encore très diffus. Nous avons dû modifier les scripts afin de tourner avec l'algorithme `crypt()`, essentiellement à cause des machines tournant sous Irix.

LA MACHINE WINDOWS DIT BONJOUR À UN SERVEUR UNIX!

Pour qu'une machine Windows 2000 ou XP rejoigne le domaine il faut d'abord créer un compte-machine sur le contrôleur de domaine, et ensuite utiliser sur le client la méthode habituelle (click droit sur *My Computer* → *Propriétés* → Onglet *Computer Name* → *Network ID*). Il est possible qu'il faille attendre une minute ou deux avant de voir la boîte de dialogue *Welcome to domain MYDOMAIN*: ceci est bon signe!

Nous n'avons pas eu l'occasion de tester cette procédure sur des machines Windows NT 3.51 ou 4.0, mais à notre connaissance il n'y a aucun problème spécifique à ces deux versions.

Il y a tout de même quelques subtilités particulières à certaines versions de Windows. Le premier cas est Windows 2000 SP2 (et ultérieurs?): cette version n'est pas en mesure de charger et sauvegarder correctement les informations des profils utilisateurs (message *Access denied*). On contourne ceci en spécifiant l'option: `nt acl support = no` pour le partage contenant les profils des utilisateurs.

Le deuxième cas problématique est Windows XP: les machines tournant sous ce système d'exploitation joignent sans problème le domaine mais les utilisateurs ne peuvent pas faire le *login* (message disant qu'aucun contrôleur de domaine n'a été trouvé). Ici aussi une solution existe mais elle doit être mise en œuvre sur la machine cliente elle-même. Il faut modifier la clé suivante du registre: `HKLM\System\CurrentControlSet\Services\Netlogon\parameters\requiresignorseal` doit valoir 0 au lieu de 1. Une fois la modification effectuée, les utilisateurs peuvent faire le *login*.

GARDER SA PROPRE PERSONNALITÉ: RÉPERTOIRES PERSONNELS CENTRALISÉS

Si l'on désire partager des fichiers (ou bien avoir des répertoires personnels centralisés) vers des machines Windows, il est nécessaire d'avoir les deux versions de Samba en parallèle. La mise en place d'un tel système n'est pas triviale: il faut créer une interface réseau virtuelle et instruire un des deux serveurs Samba à se mettre à l'écoute sur l'interface virtuelle (par exemple en utilisant dans le fichier `smb.conf` une directive de configuration similaire à celle-ci: `interfaces = 192.168.100.254`). Deux serveurs Samba signifient deux fichiers de configuration bien distincts: le premier contenant la configuration du serveur de fichiers et le deuxième celle du contrôleur de domaine. Les paramètres-clés du fichier `smb.conf` de Samba 2.2.2 qui permet l'échange des

données de sécurité entre le serveur de fichiers et le contrôleur de domaine sont les suivants:

```
security = domain
password server = SERVEUR_SAMBA_TNG
```

Voir la référence [8] pour une marche à suivre détaillée sur la mise en œuvre de deux serveurs Samba en parallèle.

Il est important de remarquer une finesse que permet un serveur de fichier central utilisé par Windows et par Unix: l'utilisateur a accès à exactement les mêmes fichiers, qu'il travaille sous Windows ou sous Unix. Ceci est particulièrement intéressant, étant donné que cela évite des va-et-vient avec des disquettes, des sessions FTP interminables, etc.

La mise en place des répertoires personnels centralisés sous Unix ne diffère guère de la méthode utilisée dans un environnement NIS: il suffit d'utiliser des utilitaires tels que *automount* ou *autofs*. Il est possible de stocker les *maps* de *automount* dans la base de données LDAP. Nous n'avons pas essayé cette possibilité en raison de l'ancienneté de la version de *automount* dont nous disposons. Dans notre cas nous avons utilisé NFS comme protocole de distribution de fichiers.

Sous Windows par contre, cette opération est simple: il faut juste spécifier le chemin réseau vers le répertoire de l'utilisateur dans son *profile* stocké sur le serveur.

SI ON MET TOUTES CES INFORMATIONS ENSEMBLE...

La figure 1 montre les dépendances entre les services et les machines clientes. La communication entre les deux serveurs Samba est due au fait que Samba 2.2.2 obtient les informations sur les utilisateurs ainsi que leurs droits d'accès depuis le contrôleur de domaine (Samba TNG). Il faut remarquer que le serveur Samba 2.2.2 est enregistré sur le domaine comme étant une machine normale: ceci est nécessaire pour une communication entre les deux serveurs Samba.

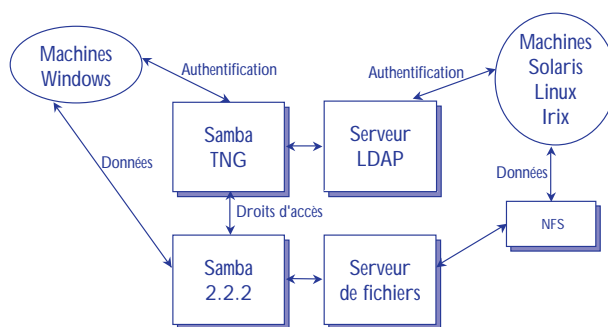


Figure 1: résumé des dépendances entre les différents services présents sur le serveur.

LIMITATIONS

Actuellement ce système d'authentification présente plusieurs problèmes: la synchronisation des mots de passe se fait dans une seule direction. En changeant son propre mot

Avoir une identité unique dans un réseau hétérogène: LDAP et Linux à la rescousse!

de passe depuis Windows, on n'aura pas changé le mot de passe sous Unix, alors qu'en effectuant la même opération sous Unix le mot de passe Windows sera lui aussi changé. Ceci ne pose pas de problèmes si la plate-forme principale est Unix.

Et les Mac dans tout ça? Malheureusement nous ne disposons d'aucune machine Mac ou MacOS X pour tester notre implémentation, donc nous ne pouvons pas émettre d'avis.

Il est donc tout à fait possible de faire cohabiter plusieurs systèmes d'exploitation et d'avoir une identité unique sur tout le réseau, sans devoir déboursier de grosses sommes d'argent et sans mettre en œuvre du matériel complexe. Il est vrai que Samba-TNG n'est pas ce qu'il y a de plus performant en matière de contrôleur de domaine, mais pour un réseau relativement petit il se défend assez bien. Il s'améliore de version en version, et son but est de pouvoir remplacer complètement une machine Windows 2000 Server pour la gestion d'un domaine.

Le système LDAP s'est avéré globalement un excellent choix pour la gestion centralisée des utilisateurs et des machines: plus besoin de se rappeler d'un mot de passe différent sur chaque machine et, surtout, il a permis l'intégration de stations de travail Windows 2000 / XP dans un réseau Unix.

Quelques jours avant la clôture de la rédaction de cet article une nouvelle version de Samba (branche *classique*, version 2.2.3) a été publiée. Parmi les nouveautés remarquables il faut citer le support LDAP amélioré. Nous n'avons pas encore eu le temps de l'essayer mais ceci est prévu sous peu. Pour ce qui concerne la version TNG, il n'y a pas de nouvelles *releases* mais on peut suivre le développement via CVS.

RÉFÉRENCES

- [1] <http://www.linux.org>
- [2] <http://www.freebsd.org>
- [3] Le client s'appelle Nisgina (<http://www.ldv.ei.tum.de/software/nisgina/>). Nous ne savons pas quelles sont ses performances, ni si il marche vraiment!
- [4] <http://www.openldap.org>
- [5] <http://www.samba.org> ou <http://samba.epfl.ch>
- [6] <http://www.samba-tng.org>
- [7] <http://www.mami.net/univr/tng-ldap/howto/>
- [8] http://www.deschner.de/gd/dual_samba.html ■