draft-chu-ldap-ldapi-00

LDAP over IPC

Introduction

Bien que LDAP soit un protocole d'accès distribué, il est courant que des clients soient déployés sur la même machine que le serveur. De nombreuses applications embarquent un client et un serveur. Dans ces déploiements intégrés, où il n'y a pas de trafic réseau, l'utilisation de TCP/IP est inutile. Les systèmes type UNIX offrent des mécanismes IPC natives qui fournissent des sémantiques orienté flux d'une session TCP, mais avec une plus grande efficacité.

Depuis Janvier 2000, OpenLDAP fournis la possibilité d'établir des sessions LDAP au travers de sockets Unix aussi bien qu'au travers de TCP/IP. De telles sessions sont aussi sécurisée que les sessions TCP sur la boucle local,, mais elles consomment moins de ressources, sont plus rapide à établir, et fournissent une identification sécurisé du client sans nécessiter de mot de passe additionnel ou autres accréditations.

Motivations

De nombreuses sessions LDAP consistent seulement en une ou 2 requêtes. L'initialisation et la fermeture des connections deviennent une portion significative du temps nécessaire à traiter ces sessions. Également lors de fortes charges, la contrainte de la limite 2MSL dans TCP devient problématique. Par exemple, un processeur moderne AMD64 dual-core qui fait tourner OpenLDAP peut manipuler 32000 authentification par secondes sur un réseau 100Mbps, avec une connection par requête. Connecté via une interface loopback, le taux est plus élevé, mais les connections sont complètement étranglées en moins d'une seconde à cause de tous les numéros de port utilisés et maintenus à l'état TIME_WAIT. Donc même quand le traitement de la charge TCP est insignifiante, la contrainte imposée par la RFC793 crée une limite artificielle. De telles contraintes n'existent pas avec IPC.

Spécification visible à l'utilisateur

Le seul changement nécessaire des clients pour implémenter cette fonctionnalité est d'utiliser une URL spéciale au lieu de **ldap ://** pour spécifier le serveur cible. Également, le serveur a besoin d'inclure cette URL dans le liste des adresses d'écoute.

Schéma d'URL

Le schéma d'URL ldapi : est utilisé pour dénoter une session LDAP sur IPC. La portion d'adresse de l'URL est le nom d'un socket unix, qui est généralement un chemin complet Unix. les "/" dans le chemin doivent être encodés comme décrit dans la rfc3986. exemple : pour un socket nommé : **/var/run/ldapi**, l'url du serveur sera **ldapi ://%26var%26run%26ldapi/**. Tous les autres aspects de l'URL sont conformes à la rfc4516.

Si aucune adresse n'est spécifiée, une adresse par défaut peut être utilisée implicitement. Dans OpenLDAP, l'adresse par défaut est une constante spécifiée à la compilation.

Détails de l'implémentation

Le transport basique utilise un socket unix orienté flux. Les sémantiques de communication dans un tel socket sont essentiellement identiques à utiliser une session TCP. À part l'établissement de la connexion, aucune considération spéciale n'est nécessaire dans le client, les librairies, ou le serveur.

Authentification Client

Depuis leurs introduction dans BSD 4.2, les sockets Unix permettent également de passer des accréditifs d'un processus à un autre. Les systèmes moderne peuvent fournir un serveur avec un moyen plus simple.pour obtenir l'identité du client. L'implémentation d'OpenLDAP exploite plusieurs méthodes pour acquérir l'identité du client. La discussion qui suit est nécessairement spécifique à la plateforme.

- La librairie OpenLDAP fournis getpeereid() pour encapsuler tous les mécanismes utilisé pour acquérir l'identité.
- Sur FreeBSD et MacOsX, getpeereid() natif est utilisé
- Sur les systèmes Solaris modernes, getpeerucred() est utilisé
- Sur les systèmes Linux qui supportent l'option SO_PEERCRED de getsockopt() est utilisé
- Sur les systèmes qui n'ont pas ces méthodes, le passeur de descripteur est utilisé. Dans ce cas, le client doit envoyer un message contenant le descripteur comme toute première action immédiatement après la connction du socket. Le descripteur est attaché à une requête LDAP Abandon avec le message ID 0, sont le paramètre a également le message ID à 0. Cette requête est un pure no-op, et sera simplement ignoré par tout serveur n'implémentant pas ce protocole.

Pour des raisons de sécurité, le descripteur passé doit être contrôlé. Le client crée un pipe et envoie le descripteur de pipe dans le message. Le serveur reçois de descripteur et fait un fstat() dessus pour déterminer l'identité du client. Le descripteur reçu doit être un pipe, et ses permissions doivent permettre l'accès à son propriétaire. L'uid et gid sont ainsi utilisé comme identité du client.

Noter que ces mécanismes sont simplement utilisés pour que l'identité du client sont disponible au serveur. Le serveur n'utilise pas les informations d'identité sauf si le client fait un Bind SASL en utilisant le mécanisme EXTERNAL.

Autres Plateformes

Il est possible d'implémenter une fonctionnalité correspondante sur les systèmes basés sur Microsoft Windows en utilisant les pipes nommés, mais il n'y a pas de demande pour celà, et l'implémentation n'a pas été écrite. Le pipe devrait être créé en mode byte-read, et le client doit spécifier l'accès SECURITY_IMPERSONATION quand il ouvre le pipe. Le serveur peut ainsi retrouver l'identité du client en utilisant la fonction GetNamePipeHandleState(). Vu que les sockets Windows ne sont pas interchangeable avec IPC, un event handler alternatif devra être fournis au lieu d'utiliser select().