slapd-meta

Backend meta

Effectue un proxy LDAP en respectant un jeu de serveurs LDAP distant, appelés target. Les informations contenues par ces serveurs peuvent être présentés comme appartenant à un simple DIT. Il a été conçu comme un amélioration de slapd-ldap. Ces 2 backends partagent de nombreuses fonctionnalités. L'instance proxy doit contenir le schéma pour les attributs et objectClass utilisés dans les filtres. Il doit également avoir le schéma pour les données retournées par les serveurs proxifiés

Directives de configuration spécial

conn-ttl <time> Force la suppression d'une connexion en cache et la recrée après le ttl donné, sans regarder s'il elle est active ou non.

default-target none Force le backend à rejeter toutes les opérations qui doivent être résolues à un simple target dans le cas ou aucun ou plusieurs target sont sélectionnés. Cette directive permet également de marquer un target spécifique comme défaut.

dncache-ttl {DISABLED|forever|<ttl>} Définis le TTL du cache de DN. (0 désactive le cache)

onerr {CONTINUE|report|stop} Mode en cas d'erreur retournée par une target durant un recherche. Continue, continue la recherche et tente de retourner le plus de données possible. Stop, stop la recherche et retourne une erreur. report, la recherche se poursuit mais si une target retourne une erreur, le premier code d'erreur retourné est envoyé au client.

norefs <NO|yes> à yes, ne retourne pas de référence de recherche

noundeffilter <NO|yes> à yes, retourne success au lieu de rechercher si un filtre est indéfinis ou contient des portions indéfinies. Par défaut, la recherche est propagée après avoir remplacée les portions indéfinies avec (!(objectClass=*)), qui correspond à un résultat vide.

protocol-version {0,2,3} Indique quel protocole utiliser pour contacter le serveur distant (0, le protocole utilisé est le même que celui utilisé par le client)

pseudoroot-bind-defer {YES|no} à yes, l'authentification sur le serveur distant avec une identité pseudo-root (idassert-bind) est déférée jusqu'à ce qu'il soit nécessaire. Sinon, tous les binds root sont propagés aux target.

quarantine <interval>,<num>[;<interval>,<num>[...]] Met en quarantaine les URI qui ont retourné LDAP_UNAVAILABLE. retente seulement à interval seconde depuis la dernière tentative, pour exactement num fois ; puis utilise le prochain pattern. si num vaut +, retente indéfiniment.

rebind-as-user {NO|yes} à yes, les crédentials du client sont mémorisés pour rebind, en tentant de re-établir une connexion perdue, ou lors de la poursuite d'aiguillage si chase-referrals est à yes.

session-tracking-request {NO|yes} Ajoute un contrôle de traçage de session pour toutes les requêtes. l'IP/hostname du client et l'identité associée à chaque requête sont envoyés au serveur distant pour information. Est incompatible avec protocol-version 2.

single-conn {NO|yes} Annule toute connexion en cache si le client se rebind

use-temporary-conn {NO|yes} Créé une connexion temporaire en compétition avec d'autres threads pour une connexion partagée. Sinon, attend jusqu'à ce qu'une connexion partagée soit disponible.

Spécification de target

uri <protocol> ://[<host>]/<naming context> [...] naming context est mandatoire pour la première uri, et omis pour les autres

acl-authcDN <administrative DN for access control purposes> DN utilisé pour requêter le serveur distant pour vérifier les ACL. Doit avoir les accès read sur le serveur cible sur les attributs utilisés dans le proxy pour la vérification des acl.

acl-passwd <password> Mot de passe utilisé avec acl-authcDN

bind-timeout <microseconds> Timeout en micro-secondes, utilisé lors du vote pour la réponse après une connexion bind asynchrone. L'appel initial à ldap_result(3) est effectué avec un timeout de 100000us. Si le résultat excède ce timeout, les appels suivants utilisent la valeur de bind-timeout.

chase-referrals YES|no Active/désactive le repérage de référence automatique, qui est déléguée à libldap, avec un rebinding éventuel si la directive rebind-as-user est utilisée. défaut : yes

client-pr {accept-unsolicited|DISABLE|<size>} Permet d'utiliser le contrôle paged result en requêtant un target. Le client n'est pas soumis à ce contrôle.

default-target [<target>] Sans argument, indique que le target courant est le défaut. target est un numéro du target par défaut.

filter <pattern> regex pour indiquer quels termes de filtre de recherche sont servis par le target.

idassert-authzFrom <authz-regexp> Sélectionne quels identités local sont autorisées à exploiter la fonctionnalité d'assertion d'identité. l'expressions suit la règle définis pour authzFrom

idassert-bind bindmethod=none|simple|sasl [binddn=<simpleDN>] [credentials=<simple password>]

[saslmech=<SASL mech>] [secprops=<properties>] [realm=<realm>] [authcId=<authenticationID>]

[authzId=<authorization ID>] [authz=native|proxyauthz] [mode=<mode>] [flags=<flags>] [starttls=no|yes|critical]

[tls_cert=<file>] [tls_key=<file>] [tls_cacert=<file>] [tls_cacertdir=<path>] [tls_reqcert=never|allow|try|demand]

[tls_ciphersuite=<ciphers>] [tls_protocol_min=<version>] [tls_crlcheck=none|peer|all] Permet de définir les paramètres de la méthode d'authentification utilisée en interne par le proxy pour autoriser les connexions qui sont authentifiées par d'autres bases de données. L'identité définie doit avoir l'accès auth sur le serveur cible sur les attributs utilisés et doit avoir le droit d'authoriser d'autres users (proxyAuth sur tout le DN, ex : authzTo=dn.subtree :"") et le serveur distant doit avoir authz-policy mis à to ou both.

<mode> := {legacy|anonymous|none|self} legacy implique que le proxy va effectuer un simple bind (authcDN) ou SASL (authcID) et assert l'identité du client quand il n'est pas anonyme. Anonymous et self, assert une chaîne vide ou l'identité du client, respectivement.

<flag> := {override|[non-]prescriptive|proxy-authz-[non-]critical} override, l'assertion de l'identité prend place même quand la base est autorisée pour l'identité du client. Quand prescriptive est utilisé, les opérations échouent avec inappropriateAuthentication pour les identités dont l'assertion n'est pas permise par le motif idassert-authzFrom. non-prescriptive, les opérations sont effectuées anonymement pour les identité dont l'assertion n'est pas permise par le pattern idassert-authzFrom. proxy-authz-non-critical, le contrôle proxyAuthz contrôle n'est pas marqué comme critique, en violation de la RFC 4370. il est recommandé d'utiliser proxy-authz-critical

idle-timeout <time> Force une connexion en cache à être supprimée puis recrée après une timeout

keepalive <idle> :<probes> :<interval> Définis les valeurs de idle, probes et interval utilisés pour cérifier si un socket est actif. idle est le nombre de seconde avant qu'une connexion non-active reçoive un keepalive. probes et le nombre maximum de keepalive, et interval est le nombre de seconde entre chaque keepalive.

map {attribute|objectclass <local name> <foreign name>|*} Map des classes d'objet et des attributs comme dans le backend ldap

network-timeout <time> Définis la valeur timeout réseau après que poll(2)/select(2) suivant un retour connect(2) en cas d'inactivité, en seconde

nretries {forever|never|<nretries>} Définis le nombre de fois qu'un bind doit être retenté en cas d'échec temporaire. (défaut : 3)

rewrite*... Permet de réecrire des requêtes

subtree-{exclude|include} <rule> Permet d'indiquer quels subtrees sont actuellement servis par un target. Peut être spécifié plusieurs fois. les syntaxes supportées sont :

<rule> : [dn[.<style>] :]<pattern>

<style> : subtree|children|regex style est soit subtree, children ou un regexp. pattern est un DN qui doit être dans le contexte de nommage servis par le target, ou un regex si style vaut regex. Si dn.<style> est omis, dn.subtree est implicite

Dans la forme subtree-exclude, si le DN demandé match au moins une règle, la cible n'est pas considérée pour remplir la requête.

Dans la forme subtree-include, si le DN demandé match au moins une règle, la cible est considérée basée sur la valeur du request DN.

suffixmassage <virtual naming context> <real naming context> Toutes les directives commençant par "rewrite" obsolète par slapo-rwm.

t-f-support {NO|yes|discover} Permet le support des filtres absolus des serveur distant

timeout [<op>=]<val> [...] Définis un timeout par opération. Les opérations peuvent être : <op> : := bind, add, delete, modrdn, modify, compare, search

tls {[try-]start|[try-]propagate} Exécute StartTls quand la connexion est initialisée. Ne fonctionne qu'avec ldaps ://

scénarios

1. 2 serveurs partagent 2 niveaux de contexte de nommage, "dc=a,dc=foo,dc=com" et "dc=b,dc=foo,dc=com". Un meta annuaire peut être configuré comme :

database meta
suffix "dc=foo,dc=com"
uri "ldap ://a.foo.com/dc=a,dc=foo,dc=com"
uri "ldap ://b.foo.com/dc=b,dc=foo,dc=com"

Les opérations dirigées vers un target spécifique peuvent facilement être résolus parce qu'il n'y pas d'ambiguïté. La seule opération qui peut résoudre plusieurs target est une recherche avec une base "dc=foo,dc=com". qui résulte de 2 recherches dans les targets.

2. 2 serveurs ne partagent aucune portion du contexte de nommage, mais ils se présentent en un simple DIT. "dc=bar,dc=org" et "o=Foo,c=US" qui apparaissent comme des branches de "dc=foo,dc=com", disons "dc=a,dc=foo,dc=com" et "dc=b,dc=foo,dc=com". On doit configurer le backend meta comme suit :

database meta
suffix "dc=foo,dc=com"
uri "ldap ://a.bar.com/dc=a,dc=foo,dc=com"
suffixmassage "dc=a,dc=foo,dc=com" "dc=bar,dc=org"
uri "ldap ://b.foo.com/dc=b,dc=foo,dc=com"
suffixmassage "dc=b,dc=foo,dc=com" "o=Foo,c=US"

De même, les opérations peuvent être résolus sans ambiguïté, bien qu'une réécriture soit requise. Noter que le contexte de nommage de chaque cible est une branche du contexte de nommage de la base. lors d'une recherche avec la base "dc=foo,dc=com" et un scope base, une erreur no such object est générée. Si le scope est one, les 2 targets sont contactés.

3. En utilisant le scénario précédent avec 2 serveurs partageant le même contexte de nommage :

database meta
suffix "dc=foo,dc=com"
uri "ldap ://a.bar.com/dc=foo,dc=com"
suffixmassage "dc=foo,dc=com" "dc=bar,dc=org"
uri "ldap ://b.foo.com/dc=foo,dc=com"
suffixmassage "dc=foo,dc=com" "o=Foo,c=US"

Toutes les considérations précédente sont maintenues, excepté que maintenant il n'y a plus d'ambiguïté pour résoudre un DN

ACL

Vous pouvez ajouter toutes les ACL au backend meta. Cependant, la signification d'un ACL sur un proxy peut nécessiter certaines considérations.

- Le serveur distant dicte les permissions ; le proxy passe simplement ce qu'il reçoit au serveur distant
- Le serveur distant dévoile tout, le proxy est responsable de la protection des accès non autorisés.

rewriting

La syntaxe et le fonctionnement est le même que slapo-rwm