Unified Authentication, Authorization, and User Administration –
An Open Source Approach

Ted C. Cheng, Howard Chu, Matthew Hardin
Syman Corporation
Outline

Evolution of Related Technologies
Unified AAA Architecture
Provisioning AAA Services
OpenLDAP Name Service Switch (nssov) Overlay
OpenLDAP Proxy Cache Engine
Summary
Evolution of Related Technologies

Linux, Unix-like systems require name services
Name Service Switch (NSS)
  Flat files, e.g., /etc/passwd, /etc/group, and so on
  NIS/NIS+, DNS

/etc/nsswitch.conf
  passwd: files nis
  group: files nis

Pluggable Authentication Modules (PAM)
  Authentication
  Account Management
  Session Management
  Password Management
The PADL Approach

By L. Howard
Directories - IT infrastructure backbone
Two libraries: nss_ldap & pam_ldap
Integrated name services and PAM framework into LDAP directories
Big step forward: performance, scalability, and high-availability
Popular in enterprise deployments
The PADL Approach (cont.)

- PAM-enabled applications
  - ssh, telnet, ftp, rsh, ...
  - sudo

- LDAP Server
  - nss_ldap
  - pam_ldap

LDAP
Opportunities for Improvements

Symbol pollution
Bloated library
Non-reentrancy
Chatty
Limited caching support
No connection sharing
No disconnected operation
Poor performance over high-latency, low-bandwidth networks
The nss-ldapd Daemon

By A. de Jong
The nss-pam-ldapd Daemon

- pam_ldap module developed by H. Chu
- nss-ldapd renamed to nss-pam-ldapd

Diagram:

- PAM-enabled applications
  - ssh
  - telnet
  - ftp
  - rsh
  - sudo

- nss_ldap
- pam_ldap

- nss-pam-ldapd
- NSLCD
- LDAP Server

- LDAP
- Unix domain socket
Unified AAA Architecture

PAM-enabled applications

ssh telnet ftp rsh ... sudo

Unix Domain Socket (IPC)

nssov
SLAPD
SLAPD backend + proxy cache

LDAP

SLAPD + extensions

NSS_LDAP

PAM_LDAP

nsso LDAP protocol

nss LDAP protocol
Unified AAA Architecture (cont.)

Distributed, scalable AAA services
Compatible with existing solutions, e.g., NIS/DNS
No application re-compilation or re-linking
No bloated libraries
LDAP connection sharing/management
Local cache for hiding latency
Support for disconnected operations when LDAP server is not available
Local database can be configured for replication
Flexible in back-mdb integration for performance optimization
Provisioning AAA Services

Hosts with AAA modules
Infrastructure as a service (IaaS)
Virtual machines preconfigured with unified AAA module -> Virtual appliances
Dynamic configuration for flexible client on-boarding
Resources provisioning, e.g., home directory
Provisioning AAA Services (cont.)
Scalability – Horizontal & Vertical

- directory server
- system or virtual appliance provisioned with AAA module
Home Directory Provisioning Overlay

By E. Backes
OpenLDAP overlays – software components stacked together to customize SLAPD behavior
Slapd configuration:

```
  overlay homedir

  homedir-skeleton-path  <pathname>

  homedir-min-uidnumber  <user id number>

  homedir-regexp  <regexp>  <path>

  homedir-delete-style  <IGNORE|DELETE|ARCHIVE>

  homedir-archive-path  <pathname>
```
Provisioning Home Directory

1. Adding a directory user
2. Replicating a directory user
3. Provisioning home directory
4. User pam login
5. SLAPD pam authentication cycle
6. Automounting home directory

- PAM-enabled applications: ssh, telnet, ftp, rsh, ... sudo
- Unix Domain Socket (IPC)
- NSLCD protocol
- nssov
- SLAPD
- ldap backend + proxy cache
- SLAPD + extensions
Name Service Switch Overlay (nssov)

The nssov overlay provides NSLCD communication protocol to SLAPD
Configured with Service Search Descriptors (SSDs)

```
nssov-ssd  <service> <url>

where <service>: aliases, ethers, group, host, netgroup, networks, passwd, protocols, rpc, services, shadow

$url: ldap://[/<basedn>][??[<scope>][?<filter>]]
```
Name Service Switch Overlay: Example

Slapd configuration:

include <path to> nis.schema
include <path to> nssov.la
database ldap
overlay nssov
nssov-ssd passwd ldap:///ou=users,dc=example,dc=com
nssov-ssd shadow ldap:///ou=users,dc=example,dc=com
nssov-ssd group ldap:///ou=group,dc=example,dc=com
...
nssov-map <service> <original attribute><new attribute>
Dynamic Configuration (nssov): Example

Dynamic configuration under <cn=config>

```
dn: olcOverlay={0}nssov,olcDatabase={1}hdb,cn=config
objectClass: olcOverlayConfig
objectClass: olcNssOvConfig
olcOverlay: {0}nssov
olcNssSdd: passwd ldap://ou=users,dc=example,dc=com
olcNssMap: passwd uid accountName
```
Proxy Cache Engine

By A. Kumar, 2003
Designed to improve the responsiveness of the ldap and meta backends
Cache entries and semantic information corresponding to recently answered queries
Implemented three algorithms:
  Query containment algorithm
  Cache replacement algorithm
  Consistency control algorithm
Proxy Cache: Query Containment Algorithm

Decides whether an incoming search request is semantically contained in any of the recently answered queries.
Example: (shoesize >=9) is contained in (shoesize>=8)
A contained query is answerable from the cache.
The LDAP matching rules and syntaxes are used while comparing assertions for query containment.
Proxy Cache: Query Containment Algorithm (cont.)

Simplified implementation - a list of cacheable templates is specified at configuration time.
A query is cached or answered only if it belongs to one of these templates.
Entries corresponding to cached queries are stored in the proxy cache local database, Berkeley DB or Memory-Mapped Database.
Meta-information (filter, scope, base, attributes) is stored in main memory.
Proxy Cache: Templates

A template is a prototype filter for generating LDAP search requests. The string representation of prototype filters is similar to LDAP filters, except that the assertion values are missing. Search filters are templates associated with their respective list of attribute values.

Example prototype filters: 
(sn=) and (&(sn=)(givenname=))

Corresponding search filters: 
(sn=Doe) and (&(sn=Doe)(givenname=John))
Proxy Cache: Cache Replacement Algorithm

Determines when a query and entries should be removed from the cache
Removes the least recently used (LRU) query and entries belonging to only that query
Proxy Cache: Consistency Control Algorithm

Weak consistency: Queries are allowed a maximum time to live (TTL) in the cache. A background task periodically checks the cache for expired queries and removes them.
Proxy Cache Configuration

proxycache <db> <maxentries><nattrsets><entrylimit><period>

Enable proxy cache and define cache configuration

<db>: underlying database
<brmaxentries>: Maximum cache capacity (entries)
<brnattrsets>: total number of attribute sets that can be defined
<brentrylimit>: maximum number of entries in a cacheable query
<brperiod>: consistency checking period (in seconds)

proxyAttrSet <index> <attributes ...>

Associate a set of attributes to an index

proxyTemplate <prototype filter> <attrset_index> <TTL>
Proxy Cache Configuration: Example

overlay proxycache

proxycache  bdb 100000 11 1000 100
# posixAccount
proxyAttrset 0 cn uid uidNumber gidNumber homeDirectory userPassword loginShell gecos
description objectClass
# shadowAccount
proxyAttrset 1 uid userPassword shadowLastChange shadowMin shadowMax shadowWarning
shadowInactive shadowExpire shadowFlag description objectClass
# posixGroup
proxyAttrset 2 cn gidNumber userPassword memberUid uniqueMember description objectClass
....
# proxy templates
proxyTemplate (&(objectClass=)(uid=)) 0 3600
proxyTemplate (&(objectClass=)(uidNumber=)) 0 3600
proxyTemplate (objectClass=) 0 3600
proxyTemplate (&(objectClass=)(uid=)) 1 3600
proxyTemplate (&(objectClass=)(cn=)) 2 3600
proxyTemplate (objectClass=) 2 3600
proxyTemplate (&(objectClass=)(gidNumber=)) 2 3600
proxyTemplate (&(objectClass=)((memberUid=)(uniqueMember=)) 2 3600
Summary

The unified AAA architecture offers performance, scalability, and high-availability
Compatible with existing IT infrastructure
Modular services provisioning
Name service switch overlay and proxy cache offer client-side caching and disconnected operations
Memory-mapped database improves proxy cache over Berkeley DB
Evolutional - collective efforts of the open source community