System Authentication for AIX and Linux using the IBM Directory Server

-A Project Example-

Dr. Stefan Radtke
IBM Server Group

Technique Paper (TP)

Version 1.0 <December 2002>

Revision History

Revision Number	Revision Date	Summary of Changes
0.8	30.11.2002	First Draft Version – not for distribution
0.81	02.12.2002	Changes Subsections of Chapter 5.1
0.9	10.12.2002	- Enhanced ACL privileges, define ACL privileges for discrete attributes: 4.2.4 - Typos
1.0	13.12.2002	- Added some words about user stanzas /etc/security/user in Chapter 5.1

Author:
Dr. Stefan Radtke,
Consulting IT Specialist
eBusiness Architectures
IBM Server Group

Table of Contents

1 Introduction

System- and User- Authentication is common task in every IT-environment. Various requirements such as security, scalability, availability, multi-platform support and shortcomings of other solutions like NIS lead to the fact that LDAP is becoming more and more popular for that task. Althoug LDAP can be used for much more, we'll focus on how LDAP can be used for system authentication in an AIX (Version 5.2 and above) and Linux environment using the IBM Directory Server 4.1 in this paper.

For a better illustration of the required design and implementation steps we'll use a real world scenario derived from a customer project. We'll first describe the server- and user scenario and the customer administration- and security reqierements in Chapter 2. Based on these requirements we'll explain the design decisions (e.g. schema, namespace) for the directoy in Chapter 3. In Chapter 4 and 5 we document the implementation steps for the LDAP-Server on AIX and the Client implementation steps for AIX and Linux respectively.

We do not cover general LDAP basics here. Good material for understanding LDAP can be found in [1] and [2]. General product documentation for the IBM Directory Server can be found in [5] and [6], whereas an excelent LDAP lecture can be enjoyed online in [9].

Availability and Scalability considerations will be added to a future version of this paper (a highly available and scaleable LDAP-Server-Infrastructure can be achieved with the replication features of the IBM Directory Server in combination with server load balaning using for example the IBM Network Dispatcher) as well as other topics (see Chapter 6).

2 Project Scenario and Requirements

For an easier way to understand the design considerations and implementation steps, we'll use a real project example which will be described in this section.

2.1 Server Infrastructure und user Groups

A core component of the customer IT-Infrastructure is a supercombuter based on several clustered IBM eServer pSeries systems, each running AIX as operating system.

In addition to this pSeries cluster, we have multiple workstation groups, consisting of several clustered or non-clustered systems belonging to different departments. Workstations within theses workstation groups run AIX and Linux as operating systems.

Users are members of the groups ***dept1*** (department1), ***dept2*** (department 2) or ***sc*** (supercomputer).

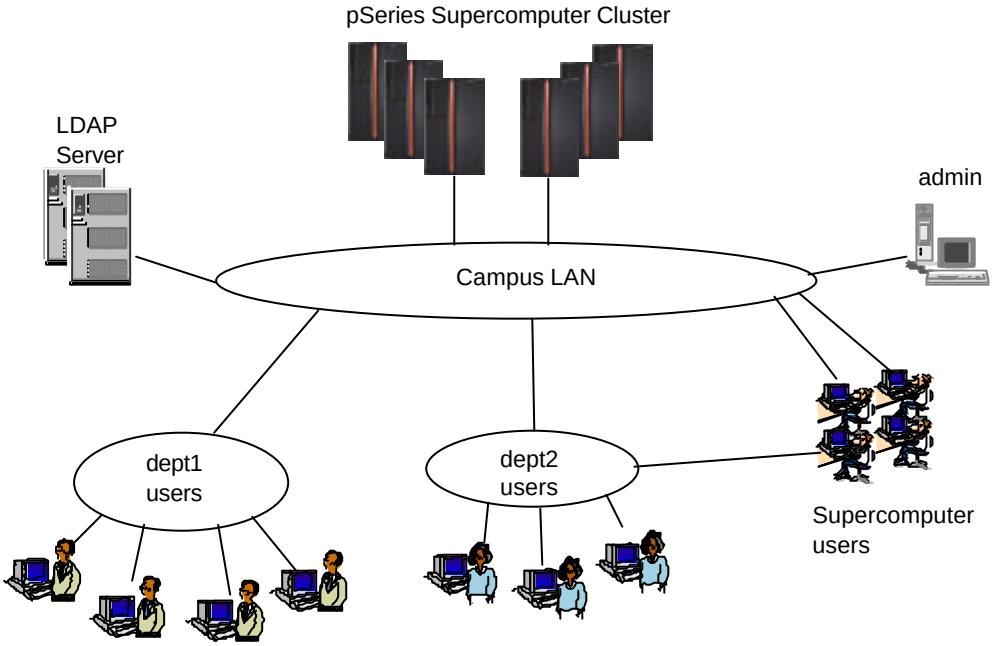

Figure 1: Server Infrastructure and User Groups

2.2 Project Requirements

During a project definition workshop, the following requirements were stated:

I. All users should be authenticated through a central LDAP instance.
II. Users from one group should not be able so retrieve any information about users of another group. (That means for example, a user in the group *dept1* should not be able to retrieve any information from members of *dept2* or *sc*).
III. No root user of any machine should be able to modify LDAP entries (with a few exeptions, see requirement IV). This requirement is caused by the fact that there are a lot of machines in the environment for which the central administrative department does not own the root accounts.

IV. The LDAP clients on the machines should only have very limited write access to the DIT. For example, write privileges are required to update some attribute values like hostLastLogin, hostLastUnsuccessfullLogin, unsuccessfullLoginCount etc.
V. A person can have a user-ID in more than one group.
VI. Some attributes from a user may be different for each group in which this user is a member (i.e. the attribute **homedirectoy** from user stefan can be /sc/home/stefan in the group sc and might be /dept1/home/stefan in group dept1).
VII. Administration will be centralized in the first step. Administration tasks for different user groups might be delegated later on to different admistrators.

3 Directory Design

3.1 Schema

In Version 4.3 and Version 5.1, AIX used a proprietary schema to store the user and group security attributes. In Version 5.2, AIX now supports the following three schemas, AIX, RFC2307, and RFC2307AIX:

> **AIX** ; The AIX schema includes the aixAccount and aixAccessGroup object classes. This schema offers all the AIX user and group attributes. This schema is included to support legacy LDAP installations prior to Version 5.2.
>
> **RFC2307**; The RFC2307 schema includes the posixAccount, posixGroup and other NIS related object classes. This experimental RFC defines a schema that allows NIS maps to be imported into LDAP. RFC2307 only defines a subset of the AIX user and group attributes. This schema supports any RFC2307 compliant platforms and AIX 5L Version 5.2.
>
> **RFC2307AIX;** The RFC2307AIX schema includes the RFC2307 schema plus the AIX specific object classes, aixAuxAccount and aixAuxGroup. The AIX specific object classes provide attributes to store additional attributes not defined by the RFC2307 standard.

For our project we decided to use the RFC2307AIX schema since it is the referred schema for new installations as it supports RFC2307 compliant platforms **and** the extended attributes for AIX (for a list of these extended attributes see [3]. The schema can be selected during the configuration of the IBM Directory Server.

3.2 Namespace

The AIX LDAP client expects the default suffix for user and group attributes to be cn=aixsecdb,cn=aixdata [3]. From our example we can derive the other nodes of the directory tree above this suffix as:

> c=de
> o=mycomany

If we would decide to have only one 'flat' directory tree for all user/group information, this would result in a full userDN for user stefan as follows:

```
uid=stefan, ou=aixuser,cn=aixsecdb,cn=aixdata,o=mycompany,c=de
```

On one hand, this would be easy for setup and administration but on the other hand we have to consider the requirements from Chapter 2.2. Before we can decide whether it's necessary to have several branches in the tree (for example one branch for each department) we need to understand how the AIX client works:

In the actual implementation (AIX V 5.2), each AIX machine has one daemon running which is responsible for binding to the LDAP server. This daemon is configured (etc/security/ldap/ldap.cfg) to bind as a dedicated user to the LDAP server (for example cn=admin). Therefore the LDAP server is not able to determine which user wants to query or modify LDAP entries, since all user requests (for example when the user wants to change his password with the `passwd` command or wants to

see other user information with `lsuser -R LDAP ALL`) are passed through the ldap daemon and therefore the bindDN is always the dedicated bindDN configured in `/etc/security/ldap/ldap.cfg`.

If we would have only one 'flat' tree, without different branches for each department and have the same bindDN for all clients (machines), we would violate against requirement 2 from 2.2 (users from one department should not be able so retrieve informations about users of another department) because any user could see user information from **all** other users with `lsuser -R LDAP ALL`. (This behaviour might be changed in future releases, when the bindDN is that of the AIX-user which inititated an LDAP query but not the bindDN from the client-daemon. We would then be able to use the pseudoDN `cn=this` for much more fine grained ACL-permissions).

Furthermore, if the bindDN has given the ACL permission 'write', every root user from from any of the machines could change user attribute values from all LDAP users.

Due to this reasons, we decide to have an own subtree for ervery department. In our example, this would mean that we have three suffixes for the LDAP directory:

cn=aixdata,ou=dept1,o=mycompany,c=de
cn=aixdata,ou=dept2,o=mycompany,c=de
cn=aixdata,ou=sc,o=mycompany,c=de

Also we would decide to create an own bindDN for every department, since we could then assign different ACL rights for each departmentural bindDN. More on that will be covered in Chapter 4.

The resulting directory subtree for our user/group information is illustrated in Figure 2.

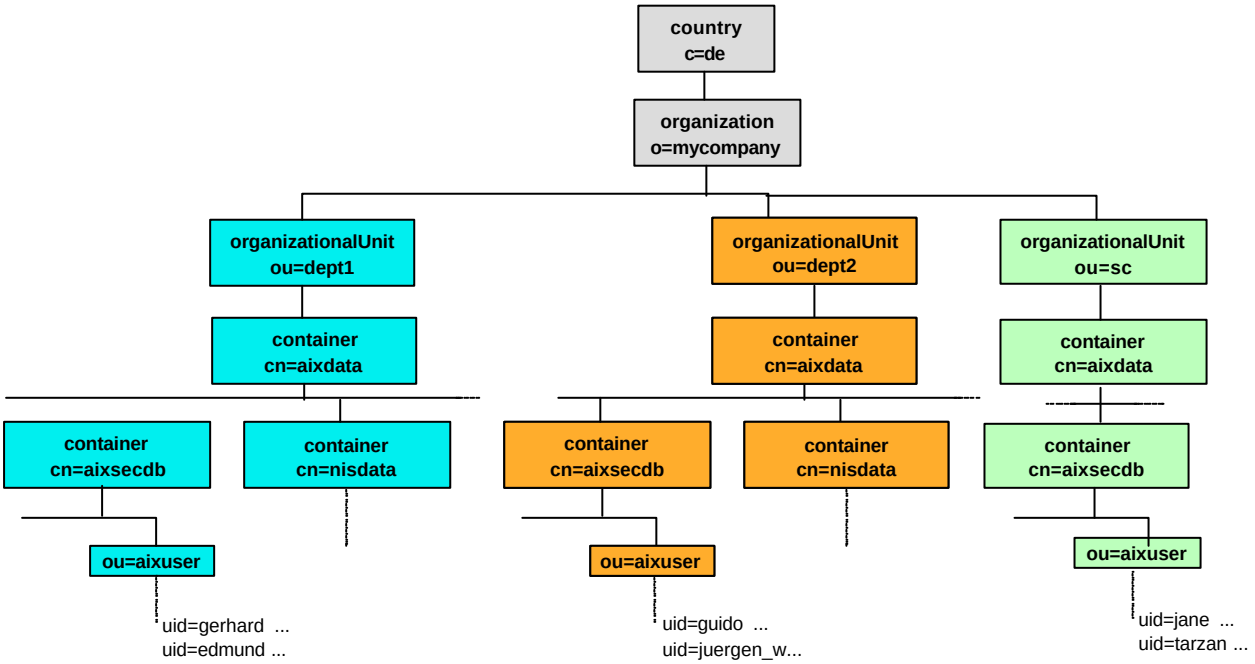

Figure 2: Directory tree with two branches for user information

We can summarize the reasons and advantages for the chosen namespace as follows:

- We can create three different bindDNs, each having ACL permissions only for one subtree (dept1, dept2 or sc)
- Thus, users from dept1 would not be able to see any user attributes or values from users in dept2 and student.
- Assuming that the bindDNs would also have write permittions in the ACL (at least for somme attributes) for their particular subtrees, root users would be able to modify user attribute values only for users within the same department (repsectevely subtree).
- Administration for entries in the subtrees can be deligated to the department administrators
- We would be able to use dedicated LDAP servers for each subtree in the future. Referals or replication could then be used to establish a distributed directory tree while maintaining transparency for users.

One thing to keep in mind about our namespace is the following (we cannot say whether it is an advantage or disadvantage in general since the behaviour can be wanted or not, depending on the organizations need):

- As we have learned before, a client machine can only bind (due to the actual daemon implementation) with one bindDN which is permanently configured for that machine. That means that every machine (client) belongs to one and only to one department. For example, if the the client of the machine host1.mycomany.com is configured to bind as

  uid=admin,ou=aixuser,cn=aixsecdb,cn=aixdata,ou=dept1,o=mycompany,c=de

  a user which is member of ‘dept2’ or ‘sc’ could not logon to this machine. Again, this is intended in our project environment, but migh not be the wanted in other situations.

4 LDAP Server Installation and Configuration Steps

Although the IBM Directory Server is available for multiple platforms, we decided to leverage the strength of an IBM eServer pSeries system and the leading UNIX operating system AIX for the Directory Server.

4.1 Installation

First you need to install the required LPPs. From the command line this would be done with the command

```
installp -acgXd <LPPSOURCE> ldap.server ldap.client ldap.html.en_US
```

where <LPPSOURCE> must be replaced with the appropriate source of the LPP, for example /usr/sys/inst.images or /dev/cd0. Of course could the installation also be done through smit or WebSM. The prerequisites like DB2 will be installed automatically. If SSL is required for the communication between the LDAP server and the client, you need also to install the GSKit.

4.2 Configuration

4.2.1 First configuration Steps

After the successful installation of the LPPs, the server needs to be configured according to the schema- and namespace definitions from Chapter 3. AIX comes along with the nice script `mksecldap` (see [12] for detailed information) which does the main steps for us.

Assume we have the following things at hand:

- AdminDN for the whole directory tree: cn=admin,o=mycompany,c=de
- Password for that AdminDN = mysecret
- Schema = RFC2307AIX
- Suffixes for our Directory:
  - cn=aixdata,ou=**dept1**,o=mycompany,c=de
  - cn=aixdata,ou=**dept2**,o=mycompany,c=de
  - cn=aixdata,ou=**sc**,o=mycompany,c=de

We would now launch mksecldap to setup the server:

```
# mksecldap –s –a cn=admin,o=mycompany,c=de –p mysecret –S RFC2307AIX
–d cn=aixdata,ou=dept1,o=mycompany,c=de –u NONE
```

where

-s	is used for Server configuration
-a <…>	configures the adminDN for the server
-p <…>	is used to assign the password for the adminDN
-S <…>	RFC2307AIX assigns the schema
-d <...>	creates the first (of our three) suffixes and subtrees
-u NONE	for not migrating local users into the LDAP tree during this stept

mksecldap will now configure the LDAP server, create appropriate DB2 instaces, create required tables and so on. The output looks like this:

```
# mksecldap -s -a "cn=admin,o=mycompany,c=de" -p mysecret -S
RFC2307AIX -d "cn=aixdata,ou=mydept,o=mycompany,c=de" -u NONE

Creating the directory DB2 default database.
This operation may take a few minutes.
Configuring the database.
Creating database instance: ldapdb2.
Created database instance: ldapdb2.
Starting database manager for instance: ldapdb2.
Started database manager for instance: ldapdb2.
Creating database: ldapdb2.
Created database: ldapdb2.
Updating configuration for database: ldapdb2.
Updated configuration for database: ldapdb2.
Completed configuration of the database.
IBM Directory Server Configuration complete.
Password for administrator DN cn=admin,o=mycompany,c=de has been set.
IBM Directory Server Configuration complete.
Plugin of type EXTENDEDOP is successfully loaded from libevent.a.
Plugin of type EXTENDEDOP is successfully loaded from libtranext.a.
Plugin of type PREOPERATION is successfully loaded from libDSP.a.
Plugin of type EXTENDEDOP is successfully loaded from libevent.a.
Plugin of type EXTENDEDOP is successfully loaded from libtranext.a.
Plugin of type AUDIT is successfully loaded from /lib/libldapaudit.a.
Plugin of type AUDIT is successfully loaded from
/usr/ccs/lib/libsecldapaudit.a(shr.o).
Plugin of type EXTENDEDOP is successfully loaded from libevent.a.
Plugin of type EXTENDEDOP is successfully loaded from libtranext.a.
Plugin of type DATABASE is successfully loaded from /lib/libback-rdbm.a.
Non-SSL port initialized to 389.
Local UNIX socket name initialized to /tmp/s.slapd.
modifying entry cn=schema
...
modifying entry cn=schema
ldif2db: 2 entries have been successfully added out of 2 attempted.
```

We could now start adding (or migrating) users to the first directory tree, but before we would like to complete the setup by adding the remaining two suffixes and subtrees (for dept2 and sc).

4.2.2 Adding other subtrees

At the time of writing, mksecldap only allows to create one branch during the initial execution of mksecldap, which is in our example cn=aixsecdb,cn=aixdata,ou=dept1,o=mycomany,c=de (this was done in the previous step). If we want to create another subtree like cn=aixsecdb,cn=aixdata,ou=dept2,o=mycomany,c=de (see Figure 2), mksecldap would exit since it searches for cn=aixsecdb,cn=aixdata in the whole directoy tree and exits if it finds it anywhere in the tree. Therefore we need to do this manually.

First we have to add the additional suffixes to the directory. We can do this with the Web-Server-Administration [5] . After login click on *Settings*, then *Suffixs* on the navigation bar. Now you can enter the new suffix and click on update. After that, you need to restart the server.

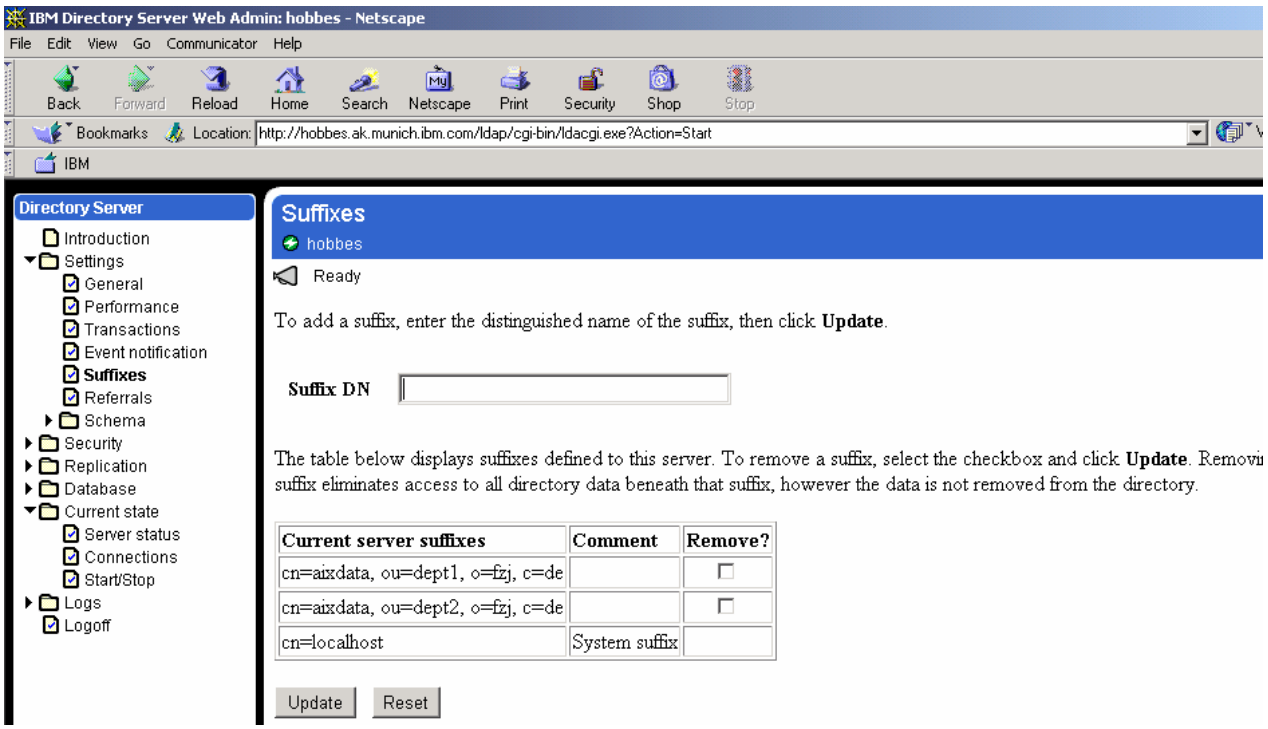

Figure 3: Server Administration: adding suffixes

Alternatevely, you could edit the /etc/slapd32.conf file and insert the following lines with the remaining two suffixes:

```
ibmsldapSuffix: cn=aixdata,ou=dept2,o=mycomany,c=de
ibmsldapSuffix: cn=aixdata,ou=sc,o=mycomany,c=de
```

After saving the file you need to restart the server.

Now we have just added the new suffixes and we now need to add the subtrees to the directory. One way to to this is to create an ldif file which looks like this for our dept2 branch (/tmp/dept2.ldif):

```
dn: ou=dept2,o=mycompany,c=de
ou: dept2
objectClass: organizationalUnit

dn: cn=aixdata,ou=dept2,o=mycompany,c=de
cn: aixdata
objectClass: top
objectClass: container

dn: cn=aixsecdb,cn=aixdata,ou=dept2,o=mycompany,c=de
cn: aixsecdb
objectClass: top
objectClass: container
```

Now, we push this into the directory using the ldif2db tool:

```
# ldif2db –i /tmp/dept2.ldif
```

This adds the three parts of the new branch to the directory and we have to repeat this step for our third subtree ‘ou=sc ...’.

Now we have all three suffixes and directory structures ready and we can start storing users to the directory.

4.2.3 Creating additional administrator users

During the installation of the LDAP-Server, one administrator DN is created (in our example from above it is cn=admin,o=mycompany,c=de). This adminDN is the owner of the whole directory tree, he can *add* and *remove* entries and has *read, search, write* and *compare* rights for the directory. It might not be the best idea to use this adminDN as the bindDN for the clients (see Chapter 5.1) since you may not want to give all client machines the potential priviliges to add, remove or modify all entries in the LDAP directory. Using only this bindDN for all clients would, for example, mean that each root user on any machine could modify all LDAP entries. This is only accectable when the administrator of the LDAP directory owns all root users (i.e. passwords) on all machines but this is unlikely in an environment with a larger number of departments and/or machines.

Let's look to our example: if we would use the directory owner cn=admin,o=mycompay,c=de as the only bindDN for all client machines, a root user of any machine in *dept1* could modify LDAP entries for users in the *dept2* or *sc* directory branch. Therefore we decide to create three different adminDNs, i.e. one for each department.

- a user admin1 for cn=axidata,ou=dept1,o=mycompany,c=de
- a user admin2 for cn=admin2,ou=dept2,o=mycompany,c=de
- a user admin3 for cn=admin3,ou=sc,o=mycompany,c=de

We'll do it through creation of an ldif file with the user information for the three admin users:

/tmp/admins.ldif

```
dn: uid=admin1,ou=aixuser,cn=aixsecdb,cn=aixdata,ou=dept1,o=mycompany,c=de
uid: admin1
objectClass: account
objectClass: posixAccount
objectClass: shadowAccount
objectClass: aixauxaccount
cn: admin1
passwordchar: !
uidNumber: 600
gidNumber: 600
homeDirectory: /home/admin1
loginShell: /usr/bin/ksh
authmethod1: SYSTEM
authmethod2: NONE
isadministrator: false
filepermmask: 22
userPassword: {crypt}cVIyvekXWsIqA
shadowLastChange: 1203755657
passwordflags: NOCHECK

dn: uid=admin,ou=aixuser,cn=aixsecdb,cn=aixdata,ou=dept2,o=mycompany,c=de
uid: admin2
objectClass: account
objectClass: posixAccount
objectClass: shadowAccount
objectClass: aixauxaccount
cn: admin2
passwordchar: !
```

```
uidNumber: 601
gidNumber: 600
homeDirectory: /home/admin2
loginShell: /usr/bin/ksh
authmethod1: SYSTEM
authmethod2: NONE
isadministrator: false
filepermmask: 22
userPassword: {crypt}cVIyvekXWsIqA
shadowLastChange: 1203755657
passwordflags: NOCHECK
dn: uid=admin,ou=aixuser,cn=aixsecdb,cn=aixdata,ou=sc,o=mycompany,c=de
uid: admin3
objectClass: account
objectClass: posixAccount
objectClass: shadowAccount
objectClass: aixauxaccount
cn: admin3
passwordchar: !
uidNumber: 602
gidNumber: 600
homeDirectory: /home/admin3
loginShell: /usr/bin/ksh
authmethod1: SYSTEM
authmethod2: NONE
isadministrator: false
filepermmask: 22
userPassword: {crypt}cVIyvekXWsIqA
shadowLastChange: 1203755657
passwordflags: NOCHECK
```

Now we push this users to the directory with ldif2db (we could also use ldapadd command):

```
# ldif2db –i /tmp/admins.ldif
```

4.2.4 Modify privileges of the administrator users

In the next step we would modify the ACLs for each branch in the directory tree in such a way, that the above adminDNs get the appropriate privileges for that portion of the directory for which they will be responsible (i.e.to which the client will bind). ACLs can be modified also with the LDIF notation, but for a better illustration of the ACL modification we'll walk through the required steps using DMT:

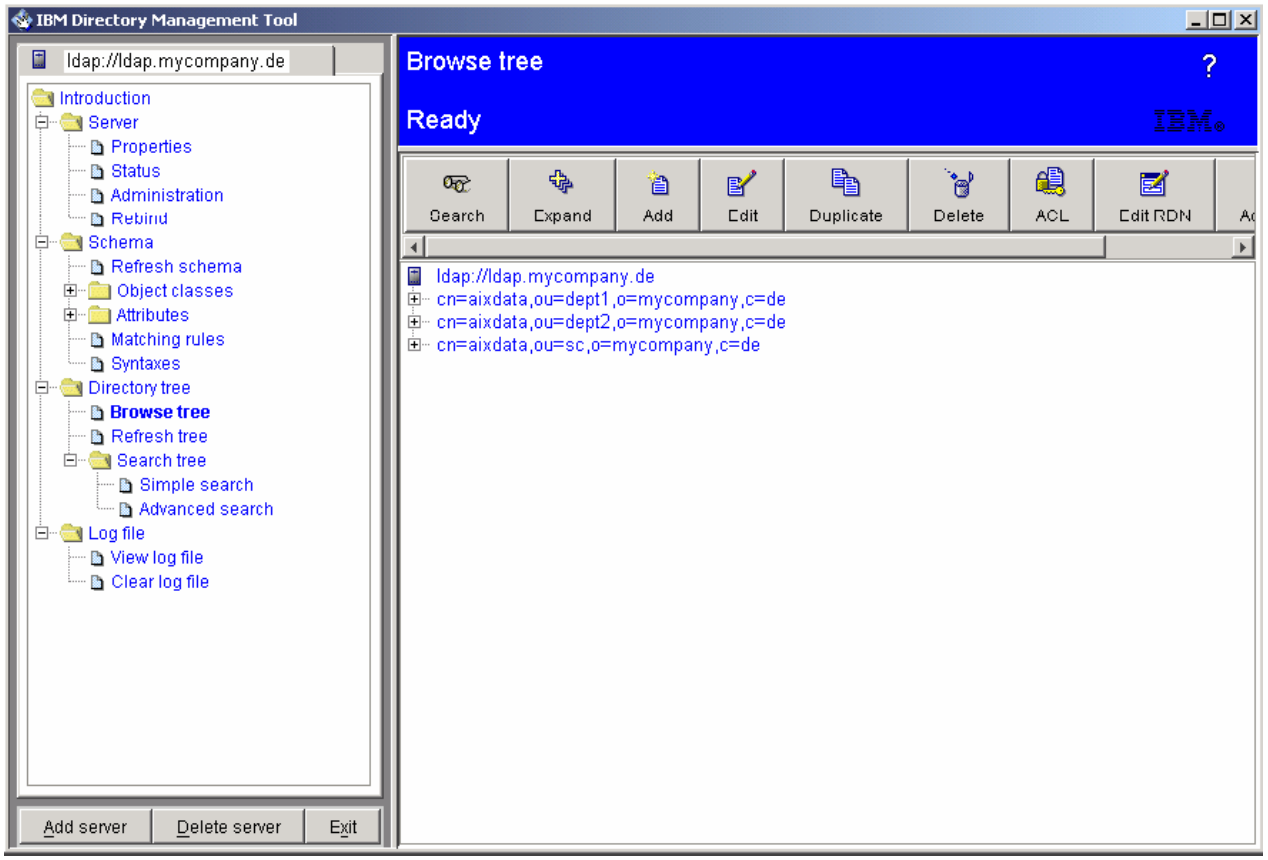

Figure 4: Browse Tree Window of the DMT

First, we launch DMT and click on Browse Tree in the navigation pane (see Figure 4). Then we select cn=aixdata,ou=dept1,o=mycompany,c=de on the directory structure and click on the ACLs icon above the shown directory structure.

The ACL window which comes up has three different sections [13]:

1. The **DN entry** section which shows the directory entry you are working with. Note that the button extending permissions to child directories is checked.

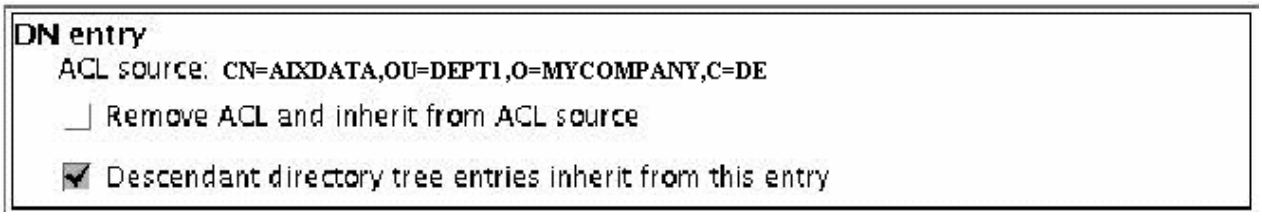

Figure 5: DN entry section of the ACL window

2. The **Subject** area in which you select or type the DN of the user who's ALC privileges have to be modified (you can see all subjects and related privileges by clicking on List all). The default is the value for the **CN=ANYBODY** group DN which everyone belongs to.

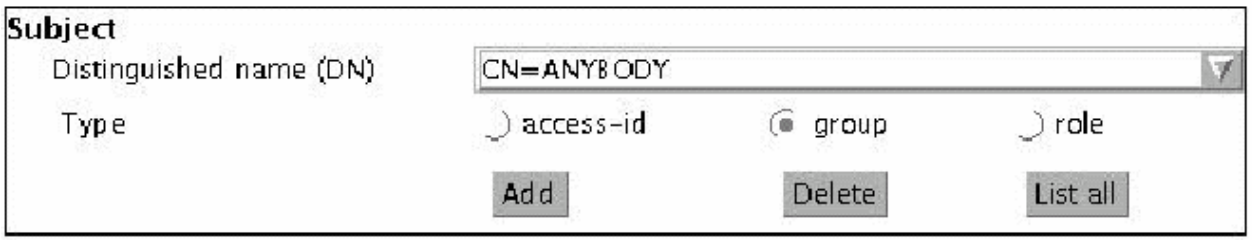

Figure 6: Subject section of the ACL window

3. The **Rights** section which tells whether the currently selected DN can add child entries to this entery or can delete this entry. By default, these are set to **Unspecified** which basically means that **CN=ANYBODY** can't add children or delete.

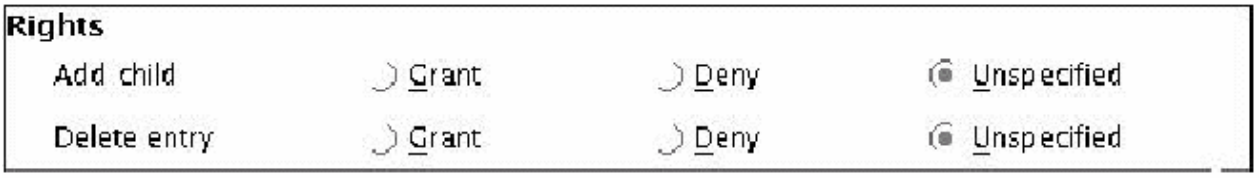

Figure 7: Right section of the ACL window

4. The **Security Class** section which gives the permissions for reading, writing, searching or comparing within different security classes of information (each attribute in the DIT belongs to one of three security classes, which are *normal*, *sensitive* and *critical*). In Figure 8 read, search and compare privileges are granted to attribute values in the security class *normal*. This is information such as the user name, mail and the telephonenumber (depending on the schema). The *sensitive* and *critical* information is set to *unspecified* and therefore not accessible to the CN=ANYBODY DN. You can also set the information for individual attributes (you can select them at the bottom) from this window instead of applying the rules to the security classes. Individual privileges set to single attributes are listed below the Security Class window.

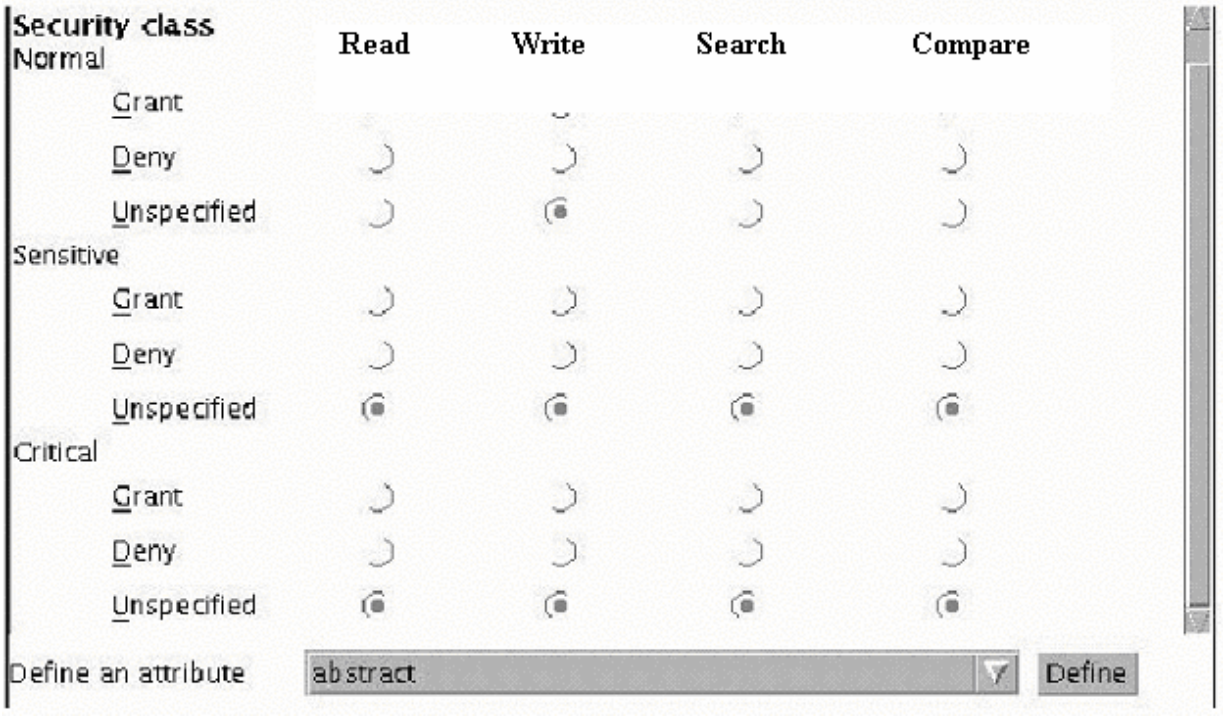

Figure 8: Security class section of the ACL window

Author: Stefan Radtke

Now we will modify the cn=aixdata,ou=dept1,o=mycompany,c=de node so that cn=admin1,ou=dept1,o=mycompany,c=de will be able to read, write, search and compare entries but not add or delete entries (from the given policy in our project scenario, adding/removing entries should only be possible through the general adminDN but not throm the department-admins).

Here are the Steps (see also Figure 9):

1. In the Subject section click the access-id button on the Type line and change the DN to read `uid=admin1,ou=aixuser,cn=aixsecdb,cn=aixdata,ou=dept1,o=mycompany,c=de`
2. Click the Add button.
3. Click the Deny button in the Rights for Add Child and Delete Entry.
4. Due to our requirements III and IV (from Chapter 2.2) we set the privileges in the security class section to allow read, write, search and compare to all data categories but we deny write privileges (see Figure 9).

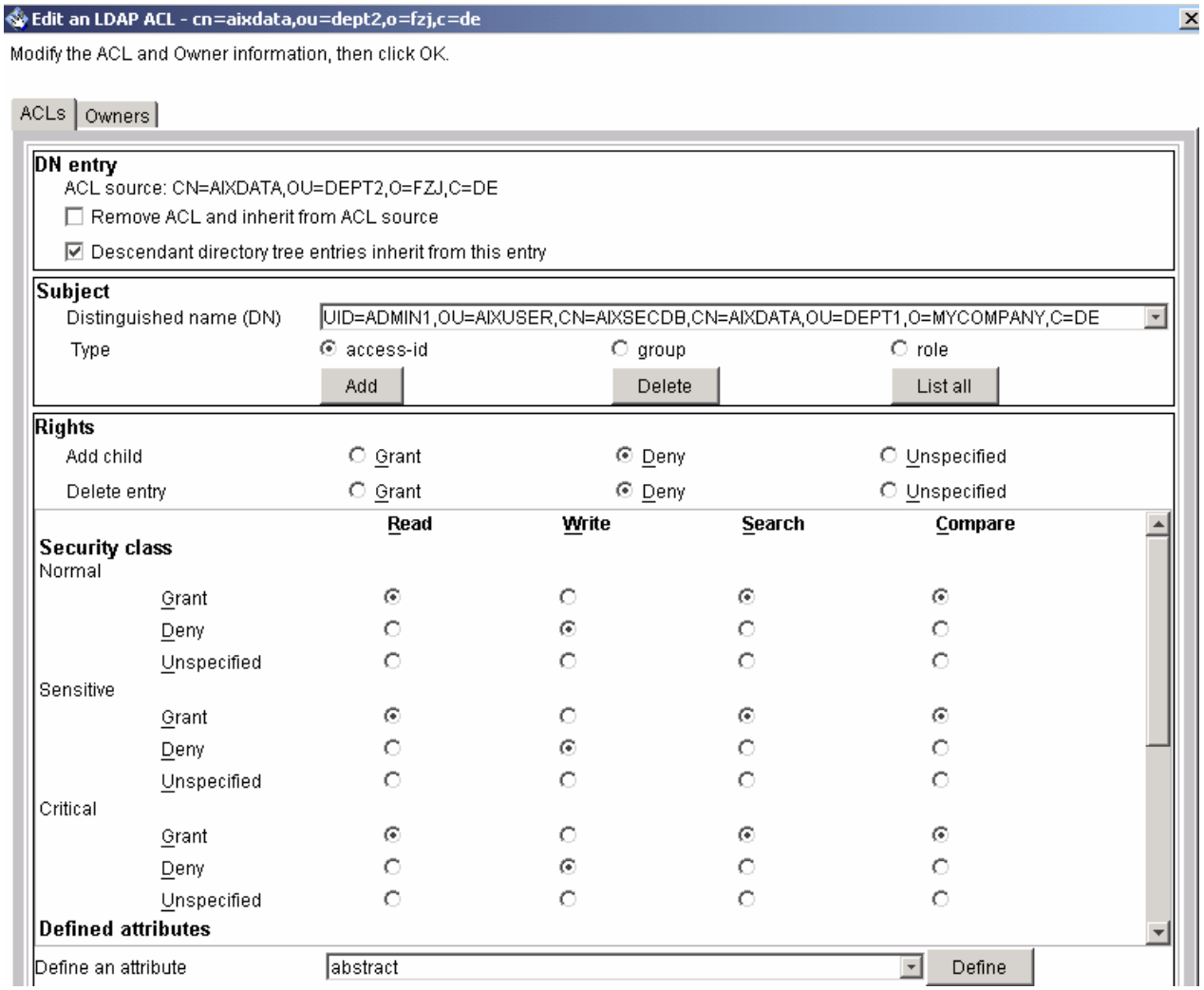

Figure 9: Adding privileges for admin1

5. Requirement IV states that some attribute values such as hostLastLogin, hostLastUnsuccessfullLogin and unsucessfulLoginCount must be modified by the client. Therefore, we need to allow write access for only these attributes. This can be done through selecting the attributes at the bottom of the window and clicking on define (see Figure 10).

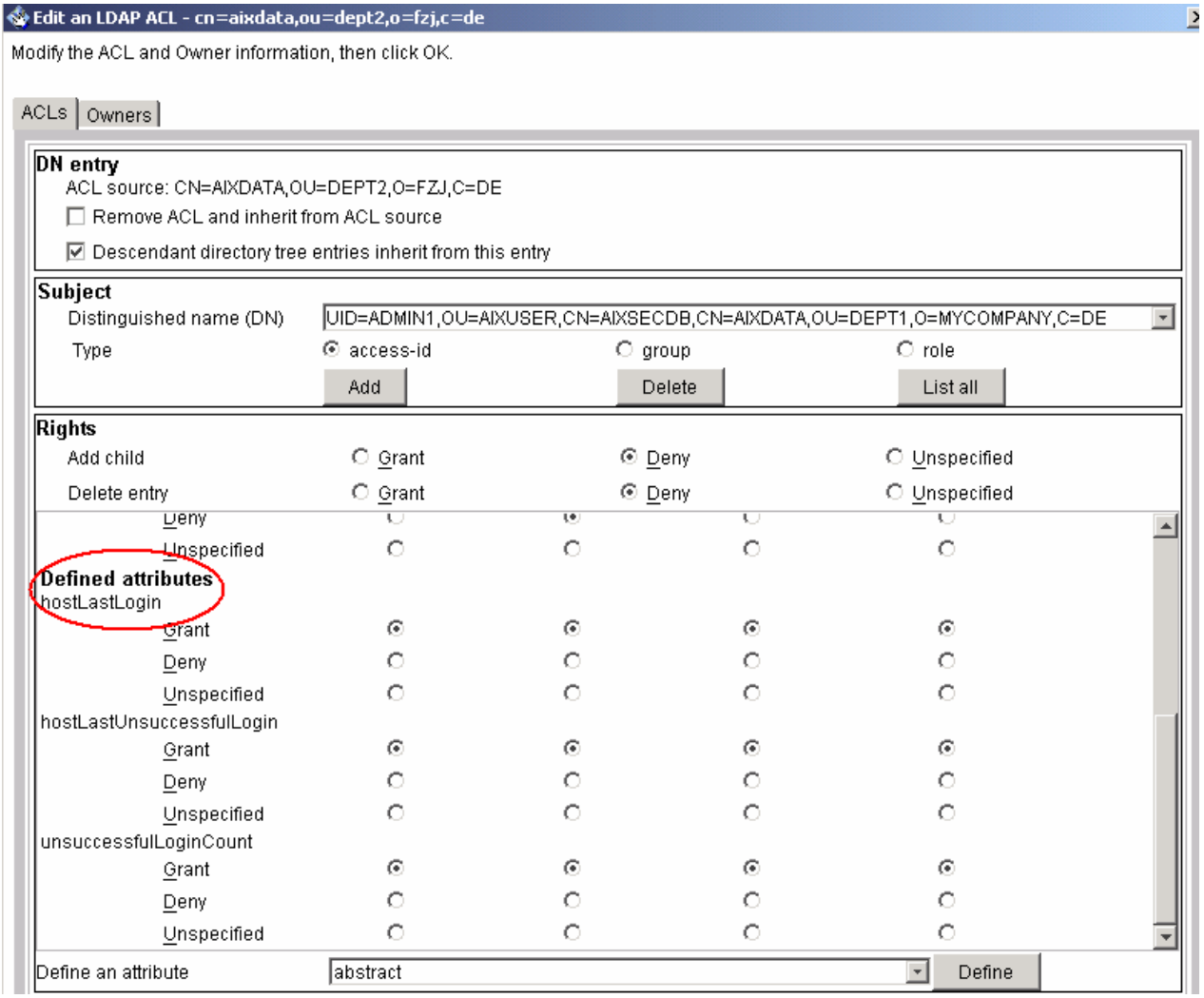

Figure 10: ACL privileges for admin1 to defined attributes

6. Now we need to repeat these steps for the other two branches to give the administrators admin2 and admin3 appropriate privileges.

If you have done steps 1. -5. and click on the List all botton, a window pops up which lists defined ACL privileges for the node cn=aixdata,ou=dept2,o=mycompany,c=de (see Figure 11)

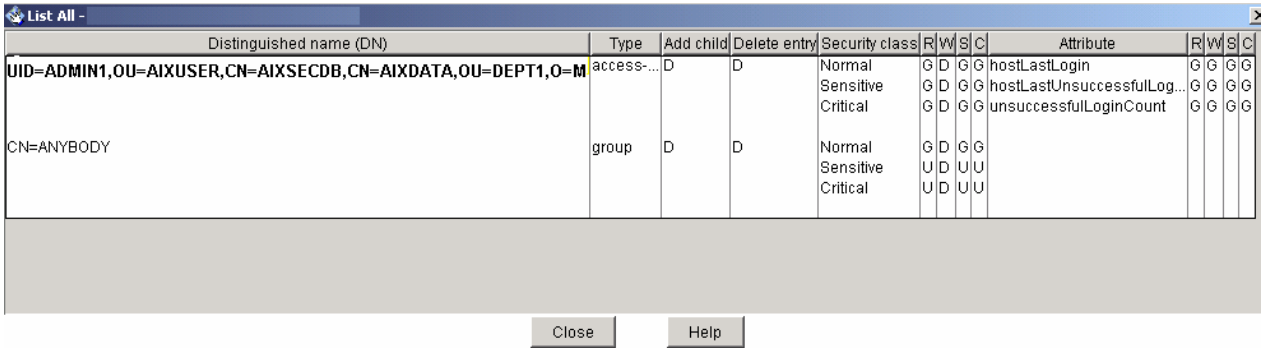

Figure 11: ACL privileges of the subject uid=admin1,...,

4.3 Adding users to the directory

There are several ways in which we can put new user entries to the directory. The easiest way is to create new users with

mkuser -R LDAP SYTEM=LDAP <user>

This assumes that we have already configured the client which is not the case at this point (see Chapter 5). Assuming that we have already local users defined, we can migrate them with the tools sectoldif and ldif2db to the directory.

4.3.1 Exporting local security repository into LDAP

To export users from the local files to LDAP we use **sectoldif** (for more details see [4]).

```
# sectoldif -d cn=aixsecdb,cn=aixdata,ou=dept1,o=mycompany,c=de \
-S RFC2307AIX –u Gerhard, Edmund > dept1.ldif
```

This would produce an ldif file for the users Gerhard and Edmund. The following is an excerpt of the LDIF file created by the previous `sectoldif` command. The first entry is the user information for gerhard and the second entry is the group information for that account.

```
dn: uid=gerhard,ou=aixuser,cn=aixsecdb,cn=aixdata,ou=dept1,o=mycompany,c=de
uid: gerhard
objectClass: account
objectClass: posixAccount
objectClass: shadowAccount
objectClass: aixauxaccount
cn: gerhard
passwordchar: !
uidNumber: 500
gidNumber: 500
homeDirectory: /home/gerhard
loginShell: /usr/bin/ksh
authmethod1: SYSTEM
authmethod2: NONE
isadministrator: false
filepermmask: 22
userPassword: {crypt}cVIyvekXWsIqA
shadowLastChange: 1203755657
passwordflags: NOCHECK
ixtimelastlogin: 1032794759
hostlastlogin: server3
unsuccessfullogincount: 0
```

```
...
dn: cn=staff,ou=aixgroup,cn=aixsecdb,cn=aixdata,ou=dept1,o=mycompany,c=de
cn: staff
objectClass: posixGroup
objectClass: aixauxgroup
gidNumber: 300
memberUid: gerhard
isadministrator: false
…
```

After the local user informations are exported into the LDIF files, you must run the ldif2db command to import them into the LDAP directory. Please note that it's not nessesary to have a client configured to run ldif2db or sectoldif, but need to run this commands on the server machine. The following example imports the user and group credentials from /tmp/dept1.ldif

```
# ldif2db -i /tmp/dept1.ldif
```

We have to repeat the steps for those local users who shoul be migrated to the ‘dept2’ and/or ‘sc’ subtree of the directory.

After using the ldif2db command to import the user and group data, you must restart the IBM directory server.

5 Client Configuration

Before we can to configure the client, it is important that we have imported at least one user and group into the LDAP directory (see Chapter 4.3) since the mksecldap command –which is also used for client configuration- requires the ou=aixuser, and the ou=aixgroup subtrees to exist before configuring the client.

5.1 AIX-Clients

The LDAP client in AIX is implemented as an AIX Loadable Authentication Module just like authentication modules exist for DCE, Kerberos, NIS , PAM etc. Therefore, configuration options are very flexible which means for example, different users might be pointed to different authentication modules or more than one authentication method can be applied during login. To which authentication mechanism a user is pointed is determined in the user's stanza in the `/etc/security/user` file. If a user is pointed to the LDAP loadable authentication mode, the stanza looks like this:

```
stefan:
      SYSTEM = "LDAP"
      registry = LDAP
      …
```

where

`-SYSTEM` defines the system authentication mechanism for the user and

`-registry` defines the authentication registry where the user is administered.
It is used to resolve a remotely administered user to the local administered domain.

To stack two authentication mechanisms, the SYSTEM value can have multiple options:

```
stefan:
      SYSTEM = "LDAP or compat"
      …
```

If a user which tries to authenticate to the system has no stanza entry in `/etc/security/user` (which is what we want because if we decide to use a centralized authentication mechanism such as LDAP, we do not want to have any specific user information on local systems anymore), the default stanza applies:

```
default:
      SYSTEM = "LDAP or compat or DCE"
      registry = LDAP

      …
```

I strongly recommend to have at least one user with root privileges pointing to the local authentication mechanism – at least if you are starting to play with LDAP. If everything points only to LDAP and the LDAP service fails, youll not able to login to the machine anymore (if this happens, you have to boot in maintenance mode and modify the `/etc/security/user` stanza manually).

The stanzas for default and dedicated users can be modified by editing them manually or one can use the chuser command:

```
# chuser -R LDAP SYSTEM=LDAP registry=LDAP stefan
```

Further details of the LDAP implementation and the AIX authentication modules can be found in [3].

Using the mksecladap command with the –c flag configures the AIX client to use LDAP as a possible authentication method. In our example we would launch:

```
# mksecldap -c -h ldap.mycompany.de -a  cn=admin,o=mycompany,c=de
-p mysecret -d cn=aixdata,ou=dept1,o=mycompany,c=de -u NONE
```

where -h <...> sets the LDAP server
-d <…> configures the base DN for this client of the AIX data subtree
-u NONE prevents any users from being migrated to LDAP

The above `command` enables the LDAP authentication load module by inserting the following stanza into the /usr/lib/security/methods.cfg file.

```
LDAP:
   program = /usr/lib/security/LDAP
   program_64 =/usr/lib/security/LDAP64
```

The mksecldap client setup also starts the secldapclntd daemon. The secldapclntd daemon manages connections and transactions from the LDAP authentication load module to the remote LDAP security information servers. The secldapclntd daemon caches LDAP queries to order to improve performance. It is configured using the /etc/security/ldap/ldap.cfg file. The following excerpt from the ldap.cfg file shows the client configuration generated from the previous mksecldap command (some comments and the prolog is left out here).

```
# secldapclntd LDAP client daemon configuration file

# Comma separated list of ldap servers this client talks to
ldapservers:9.23.4.67

# LDAP server bindDN
ldapadmin:cn=admin,o=mycompany,c=de

# LDAP server bindDN password
ldapadmpwd:mysecret

useSSL:no

# SSL key file path and key password
#ldapsslkeyf:/tmp/key.kdb
#ldapsslkeypwd:mykeypwd

# AIX-LDAP attribute map path.
userattrmappath:/etc/security/ldap/2307aixuser.map
groupattrmappath:/etc/security/ldap/2307aixgroup.map
idattrmappath:/etc/security/ldap/aixid.map

# Base DN where the user and group data are stored in the LDAP server.
userbasedn:ou=aixuser,cn=aixsecdb,cn=aixdata,ou=dept1,o=mycompany,c=de
groupbasedn:ou=aixgroup,cn=aixsecdb,cn=aixdata,ou=dept1,o= mycompany,c=de
idbasedn:cn=aixid,ou=system,cn=aixsecdb,cn=aixdata,ou=dept1,o= mycompany,c=de

# LDAP class definitions.
userclasses:account,posixaccount,shadowaccount,aixauxaccount
groupclasses:posixgroup,aixauxgroup

# LDAP server version. Valid values are 2 and 3. Default is 3.
#ldapversion:3
```

```
# LDAP server port. Default to 389 for non-SSL connection and
# 636 for SSL connection
ldapport:389

# Follow aliases. Valid values are NEVER, SEARCHING, FINDING, and
# ALWAYS. Default is NEVER.
#followaliase:NEVER

# Number of user cache entries. Valid value is 100 - 10000 entries.
# Default is 1000.
#usercachesize: 1000

# Number of group cache entries. Valid value is 10 - 1000 entries.
# Default is 100.
#groupcachesize: 100

# Cache timeout value in seconds. Valid value is 60 - 60*60 seconds.
# Default is 300. Set to 0 to disable caching
#cachetimeout: 300

# Time interval in seconds that the secldapclntd daemon contact the
# LDAP server for server status. Valid value is 60 - 60*60 seconds.
# Default is 300.
#heartbeatinterval: 300

# Number of threads the secldapclntd daemon uses to to process jobs.
#Valid value is 1 - 1000. Default is 10
#numberofthread: 10
```

The following entry is added to the /etc/inittab file to start the secldapclntd daemon during the system boot.

```
ldapclntd:2:once: /usr/sbin/secldapclntd > /dev/console 2>&1
```

5.1.1 Client tools

Several commands are available to control and monitor the secldapclntd daemon. The `flush-secldapclntd` and `ls-secldapclntd` commands flush the LDAP client cache and display LDAP client statistics. The `restart-secldapclntd`, `start-secldapclntd`, and `stop-secldapclntd` commands restart, start and stop the secldapclntd daemon.

5.1.2 Configure another bindDN for a client

At the time of writing this document, an AIX client runs a daemon (secldapclntd) which binds to the directoy only with one configured bindDN. Therefore, an AIX machine can access only one of our three branches and we define this with the configuration of one of our three administratorDNs as the bindDN for the client.

Assuming that we have already configured the client (with the default bindDN as shown at the beginning of Chapter 5.1) we just need to modifiy the client configuration file /etc/security/ldap/ldap.cfg as follows:

We replace:

```
ldapadmin:cn=admin,o=mycompany,c=de
ldapadmpwd:mysecret
```

with:

```
ldapadmin:uid=admin1,ou=aixuser,cn=aixsecdb,cn=aixdata,ou=dept1,o=mycompany,c=de
ldapadmpwd:secret_from_admin1
```

For machines which belong to *dept2* or *sc*, we would also have to modify the userbasedn, groupbasedn and idbasedn:

```
# Base DN where the user and group data are stored in the LDAP server.
userbasedn:ou=aixuser,cn=aixsecdb,cn=aixdata,ou=dept2,o=mycompany,c=de
groupbasedn:ou=aixgroup,cn=aixsecdb,cn=aixdata,ou=dept2,o= mycompany,c=de
idbasedn:cn=aixid,ou=system,cn=aixsecdb,cn=aixdata,ou=dept2,o= mycompany,c=de
```

After saving the file we need to restart the client. From now on, it will bind with the configured bindDN and has the privileges configured in the ACLs of the directory.

5.2 Linux-Clients

To configure Linux to be a LDAP client, we need to touch three things, which are the configuration of the Plugable Authentication Modules (PAM) [9], the Name Service Switch (NSS) and the LDAP configuration files. Since a huge amount of documentation for the different Linux distributions and LDAP can be found in the Internet, we cover just the basic settings for the LDAP client here.

5.2.1 RPM Installtation

On Linux the following he RPMs must be installed (depending on which documentation you read, there might be some other RPMs, depending on the distribution):

openladp-client-2.0.23-53
nss_ldap-183-65
pam_ldap-137-67

5.2.2 LDAP-Client Konfiguration

The configuration file /etc/openldap/ldap.conf configures the client. The most parameters are self explaining. In our example, the file would look like this:

```
host  10.10.10.1
base  cn=aixdata,ou=dept1,o=mycompany,c=de
ldap_version     3

binddn uid=admin1,ou=aixuser,cn=aixsecdb,cn=aixdata,ou=dept1,o=mycompany,c=de
bindpw  secret_of_admin1
scope sub
```

Other optional parameters are

DEREF wich determins how references are handled. Possible values are always, finding, never, searching whereas never is the default.

With SIZELIMIT and TIMELIMIT you can limit the size and time for queries.

5.2.3 NSS configuration to use LDAP

The next step is to configure NSS to use LDAP. NSS stands for Name Service Switch and it is used to tell the system what sources you want referenced for certain information (for the AIX gurus, we do configure these things in /etc/netsvc.conf in AIX). The configuration file is /etc/nsswitch.conf and probably looks something like this:

```
passwd: files nisplus nis
shadow: files nisplus nis
group: files nisplus nis
hosts: files nisplus nis dns
```

For having our user and group information in LDAP, we change the first two lines to something like this:

```
passwd: files ldap
shadow: files ldap
```

Lookups for user/group information would now query local information and then LDAP.

5.2.4 PAM configuration

Finally, we need to configure PAM to put LDAP in the authentication stack. PAM is very flexible in such a way that authentication modules can be stacked in various ways (for more information about PAM see [9]). A separate configuration file is available for each service like ssh, telnet, ftp (where the filename is the same as the servervice) and even others in the directory `/etc/pam.d/`. Let's examine the file `/etc/pam.d/ssh` which is used by the system for ssh logons.

```
#%PAM-1.0
auth sufficient /lib/security/pam_ldap.so
auth required /lib/security/pam_pwdb.so shadow try_first_pass
auth required /lib/security/pam_nologin.so
account sufficient /lib/security/pam_ldap.so
account required /lib/security/pam_pwdb.so
password required /lib/security/pam_cracklib.so
password sufficient /lib/security/pam_ldap.so
password required /lib/security/pam_pwdb.so shadow nullok use_authtok
session sufficient /lib/security/pam_ldap.so
session required /lib/security/pam_pwdb.so
```

To bring LDAP into the authentication stack, I added the lines which are printed in bold faces. I'll not explain details here since these have been documented thousand times elsewhere. The keyword `sufficient` means, that if a user is authenticated through LDAP, the access is granted and the other modules are not queried anymore. If pam_ldap.so returns unsuccessfully, the next module will then be queried. If a module with a `required` keyword returns unsuccessfully, the authentication fails and further modules would not be queried.

6 Future Work

This paper based on some experience in a customer project using the IBM Directory Server Version 4.1 for LDAP authentication of AIX 5.2 and Linux Clients, but it is by far not complete.

I think the follwing topics are important and need further discussion in later versions of this paper:

- Availability and Scalability Configuration (Load Balancing, Replication etc.)
- Security considerations, e.g. the use of SSL between client and server
- Configuration of other clients like Solaris machines to authenticate against the IBM Directory Server with the RFC2307AIX schema.
- New functionality of the brand new IBM Directory Server Version 5.1
- Enhance the description of the Linux client configuration

7 References

[1] Heinz Johner, Larry Brown, Franz-Stefan Hinner, Wolfgang Reis, Johan Westman, Understanding LDAP
IBM Redbook, IBM Corporation, 1998
http://ibm.com/redbooks

[2] Heinz Johner, Michel Melot, Harri Stranden, Permana Widhiasta
LDAP Implementation Cookbook
IBM Redbook, IBM Corporation, 1999
http://ibm.com/redbooks

[3] AIX 5L Version 5.2 Security Guide
IBM Corporation, 2002
http://ibm.com/aix

[4] Scott Vetter, Marc Bouzigues, Ed Geraghty, Damon Roberts, Ralf Volmer
AIX 5.2 Differences Guide,
IBM Redbook, IBM Corporation, 2002
http://ibm.com/redbooks

[5] IBM Directory Server Version 4.1: Installation and Configuration Guide for Multiplatforms
IBM Corporation, 2002
http://ibm.com/software/network/directory/server

[6] IBM Directory Server Version 4.1: Administration Guide
IBM Corporation, 2002
http://ibm.com/software/network/directory/server

[7] IBM Directory Server Version 4.1: Tuning Guide
IBM Corporation, 2002
http://ibm.com/software/network/directory/server

[8] IBM Directory Server Version 4.1 Installation and Configuration Guide for Multiplatforms
IBM Corporation, 2002
http://ibm.com/software/network/directory/server

[9] The Linux-PAM System Administrator's Guide: The Linux-PAM configuration file
http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/pam-4.html

[10] L. Howard
An Approach for using LDAP as a Network Information Service
RFC 2307, The Internet Society, March 1998
http://www.ietf.org

[11] Veronika Megler
A Highly Available & Scalable LDAP Cluster in an IBM AIX Environment
White Paper, revised 9.November 2001

[12] AIX 5L Version 5.2 Commands Reference
IBM Corporation, 2002
http://ibm.com/aix

Author: Stefan Radtke

[13] Tesch's tips: LDAP access and ACL's
IBM Intranet
http://tesch.aix.dfw.ibm.com/aixldap/access.html