The **LINCKS** Version 2.2 System Administration Manual
Manual version: 1.7 5/27/94

Lab for Intelligent Information Systems
Department of Computer and Information Science
University of Linköping, Sweden

Editor: David Partain

Contributors: Ralph Rönnquist, Lin Padgham, Martin Sjölin

Last updated: May 17, 1994

Abstract

This manual is intended for those responsible for the day-to-day maintenance of the system. It includes a brief organizational description of the system, how to get the database system running, descriptions of the normal tasks which are necessary, as well as solutions to the various problems which might occur.

E-mail: lincks@ida.liu.se
Bug Reports: lincks-bugs@ida.liu.se

Contents

1 Installation of the Software

Installation of **LINCKS** software is described in the **INSTALL** file at the top level of the source tree.

2 Organizational Description

Here we will attempt to give an overview of the parts of the **LINCKS** system included in the distribution and how they are interrelated.

2.1 Parts of the system

The **LINCKS** system is made up of several elements which are all used in some way for accessing or maintenance of the system. Amongst these are:

- Four libraries:
  1. *liblincks.a*: library of database access routines which are linked to applications which use the **LINCKS** database. This will be of interest to application programmers.
  2. *libxl.a*: library of X Windows functions linked to *xlincks*.
  3. *libparser.a*: a parser which is used by several applications in the system.
  4. *libshared.a*: a library of code that is shared by different modules but should not be in liblincks.a.
- Utility software which is used for maintenance:
  1. *cutoff*: a database garbage collection program.
  2. *dbroot*: a program which creates an virtually empty database.
  3. *lincks*: a database server control program used for starting and stopping the database.
  4. *dbstat*: a small program which retrieves a small number of statistics about a database.
  5. *t2lincks*: program which allows you to put appropriately formatted text files into a database.
  6. *dbdump*: program which allows you to produce a textual dump of an entire database which *t2lincks* can load into a different database.
  7. *dbpasswd*: program which allows users to change their password (encrypted or in clear text) for a database.
- Database-side server software:
  1. *monitor*: **LINCKS** database monitor. This program is responsible for management of database files. It starts up the *netserv* process, reads and writes in the various database files, and registers the appropriate functions with the RPC service package. There is one *monitor* per database.

2. *netserv*: Again, there is one such process per database. Upon login, the workspace connects to *netserv*, which then forks a *dbs* process for subsequent communication. Thereafter, *dbs* communicates directly with the *monitor*.
3. *dbs*: the communicator between the workspace (usually *xlincks*) and the database. One *dbs* process runs per user. The user's commands are given to *dbs*, which then passes them along to *monitor* to carry them out.

- An interface, *xlincks*, to the database itself. This is the heart of the system, this program is the interface of choice for manipulation of the database. It is a complete database editing system and allows the user to access and modify database objects. It is built using an X Windows user interface.

2.2 Database Directory Structure

A **LINCKS** database includes several files, which we will briefly describe in this section.

1. The server programs (*monitor*, *netserv*, *dbs*, and *lincks*) must reside in the same directory. In the configuration file, we refer to this directory as *BINARIES*.
2. Each database resides in its own directory that, in addition to the data storage files, contains the following files:
   - *.lincksrc*: the database definition file, a run-time configuration file. Its complete contents are describe in section 3.1.1.
   - User login definitions file *passwd*: users and their passwords. The passwd file contains each user's name, user id, and password. You can select during installation if passwords should be encrypted or not.
     Users can change their passwords by using the *dbpasswd* program (independent of whether they are clear text or encrypted). The *passwd* file should be readable by the database owner only, and *dbpasswd* is installed suid so that the file can be updated.
     The only way to change a username is to edit file.
   - User groups definitions file *groups*: user groups.
     The groups file contains the mapping of user id's to the protection groups of which they are members. The file's syntax is:
     *0 1 2 3 4**
     *2 1 2**
     which means that user 0 has read-write access to objects created by users 1 through 4, and user 2 has read-write access to objects created by users 1 and 2. The asterix marks the end of the definition set for each uid's membership groups.
   - User write groups definitions file *wrgroups*: user wrgroups..
     The *wrgroups* file contains the mapping of user IDs to the protection groups of which they are members and thus which objects they are allowed to *change*. The file's syntax is:
     *0 1 2 3 4**

*2 1 2**
which means that user 0 has *write* access to objects created by users 1 through 4, and user 2 has *write* access to objects created by users 1 and 2. The asterix marks the end of the definition for each uid's membership groups.
Notice that *wrgroups* is only in effect if it is present and then defines the write access for the different users. If the file is not present, only the restriction imposed by the *groups* file is used both for read and write access to objects.

In the *.lincksrc* file, we refer to the directory with all these files as *DBDIR*.

3. The server programs writes messages to log files that reside together in a directory. We refer to this directory as *LOGDIR*.

2.3 Database System Files

A **LINCKS** database consists of a number of data files and support files.

The data files are of three types: the index file *index*, one or more molecule header files (*nnnnnnnnnn.mol*), and one or more data storage files (*nnnnnnnnnn.dat*). These are all binary files and therefore not directly readable.

The support files are the files *molecule.names* and *data.names*, the user login definitions file *passwd*, the user group definitions file *groups*, and the optional write user group definitions file *wrgroups*.

2.4 Log Files

Four log files are maintained by the **LINCKS** system, one each for *monitor*, *netserv*, *dbs*, and a catch-all file called *lincks.log* for when logging to the others fails. It is advisable to periodically remove these files to prevent them from growing unchecked. This should only be done after stopping the system (see section 4.1). All of these files are in one directory (*LOGDIR*).

2.4.1 monitor.log

The monitor log file (*monitor.log*) keeps track of all internal monitor errors such as failed RPC calls as well as recording system starts and stops. The file format is as follows:

```
92-12-17  12:16:31  User: (S) 6020  DB: /home/lincks/DB
  monitor 6017 started at obel17
93-02-10  10:44:44  User: (S) 6009  DB: /home/lincks/DB
  monitor 6009 started at obel17    <UNIX: Invalid argument>
```

The <UNIX: X> specifies what UNIX error (if any) that has occurred.

2.4.2 netserv.log

The netserv log file (*netserv.log*) keeps track of network socket problems and logs each login attempt (successful or not) as well as each log out. Also recorded is how the database server processes terminate. The format of the file is as follows:

```
92-12-11  10:35:54  User: (17) 6017  DB: /home/lincks/DB
  NetServer: User partain logged in
92-12-11  10:48:06  User: (17) 6017  DB: /home/lincks/DB
  DBserver: User logged out
92-12-11  11:33:13  User: (U) 6016  DB: /home/lincks/DB
  NetServer: DBserver died, exit 12
```

The *User* entry specifies the uid number of the user causing the message and the socket (*TCPIPNO*) number of the connection. An *U* means that the user is unknown or irrelevant. The entry *DB* tells which database the message is concerning.

2.4.3 dbs.log

The dbs log file (*dbs.log*) records the stop and start of every database server process as well as any internal database server errors. The file format is as follows:

```
92-12-11  11:58:25  User: (17) 6017  DB: /home/lincks/DB
  dbserver 6017 started.
92-12-11  11:58:32  User: (17) 6017  DB: /home/lincks/DB
  dbserver 6017 stopped
92-12-11  11:59:22  User: (17) 6017  DB: /home/lincks/DB
  DBserver: Could not read
```

The format is similar to the other log files.

2.4.4 lincks.log

The *lincks* log file (*lincks.log*) is only used if logging to the standard file fails.

2.4.5 ws.log

The workspace log file (*ws.log*) reports internal workspace errors which are very serious, such as malloc failing because memory ran out. Logging here likely means that the system crashed. The file format is as follows:

```
93-01-25  13:27:34  User: (2) 6017  DB: /home/lincks/DB
  EnsureBuffer: realloc failed: Not enough memory.
```

3 Getting the database running

Note: it is assumed that if you are at this point that all of the software for the system has been installed as per the directions in the **INSTALL** file at the top of the source directory. If not, please do so before proceeding further.

3.1 Creating a database

The process of installing a new database involves three steps. The first is to configure the database. Secondly, an empty bootstrapping database must be created. Thirdly, the base of information sent with the distribution must be loaded into the database. Thereupon, the database system can be started and used.

3.1.1 Configuring the database

LINCKS programs use run-time configuration using information from the definition file, **.lincksrc**, which is specific to each database. The file format is “tag = value”.

The *.lincksrc* must be in the main database directory (*DBDIR*) and must configure the following:

Field Name	Description
BINARIES	Directory where the executables for the database server programs (*dbs*, *monitor*, and *netserv*) are. No default.
DBDIR	Directory holding the database files, the password and group files, etc. No default.
LOGDIR	Path to the directory holding the log files. No default.
HOST	Name of the machine running the database servers in either domain format or as an ip number. No default.
OWNER	Name of the user who owns the database files. No default.
TCPIPNO	Port number used for communication. Must be unique for each database you are running on a machine. At our site, we generally use ports above 6000, which should be safe. Contact one of your network administrators if you are unsure. No default.
MONITORPROGNO	The RPC program number used for communication between the *dbs* and *monitor* processes. The default is 572662306, which you probably don’t need to change.

An example setup might look like this:

.lincksrc:

BINARIES	=	/home/lincks/Bin4
DBDIR	=	/home/lincks/DB
LOGDIR	=	/home/lincks/Databases/Log
HOST	=	obel17
LINCKSDIR	=	/home/lincks/DB
OWNER	=	lincks
TCPIPNO	=	6017
MONITORPROGNO	=	572662306

So, first you must set up this file up for your local setting.

3.1.2 Creating the bootstrap database

The second step is to create a bootstrap database using the program *dbroot*. This program creates the root object of an empty database. It is not as automatic as perhaps we would like, so there are several steps that you must take to get the empty database set up.

1. After choosing your **DBDIR** and configuring the database (as detailed in section 3.1.1), create three empty files called *index*, *90091412350.dat*, and *90091412350.mol*:

   `touch index 90091412350.dat 90091412350.mol`

2. Create a file *data.names* with one line in it (ONLY one line):

   `90091412350.dat`

3. Create a file *molecule.names* with one line in it (ONLY one line):

   `90091412350.mol`

4. Create a file *passwd* with one line in it (ONLY one line):

   `lincks   0`

   This is the entry for the user **lincks**, who is user 0. Again, make sure this file is only readable by the database owner.

5. Set the password for user lincks using the *dbpasswd* program by:

   `dbpasswd -d . lincks`

   and answer `hargle` twice. Of course if you are running with clear text password, you can simply add the password `hargle` when creating the file. Note, of course, that you can use whatever password you wish, but **remember it!**

6. Create a file *groups* with one line in it (ONLY one line):

   `0 0*`

   This line means that user 0 is in group 0. The `*` marks the end of the line.

7. Start the database servers (see section 4.1 for full details):

   `lincks -s .`

8. Run *dbroot* on the database, which will create the system root and prepare the database to be loaded properly.

   `dbroot .`

   You will then have to log in as user **lincks** and password **hargle**, at which point the process will be carried out and the program will exit. The database is ready to load.

3.1.3 Running *t2lincks*

The distribution includes several files called something ending with **.t2l*. These files are specially formatted text files which are read by the program *t2lincks* and loaded into the database in the appropriate places. After running *dbroot* to create your empty database, you will now load these *t2lincks* files, which will put in all the information that you need to use the database system. For each *t2lincks* file, issue the following command:

```
t2lincks -f filename.t2l database-dir
```

where database-dir is the path to the database directory.

The output will look something like this:

```
LINCKS login: lincks
password:

Welcome user 17 (lincks).  You're logged in to obel11.
Storing object User Root in database...(1) done.
Storing object System Root in database...(1) done.
Database has sent LOGOUT code

Good-bye!
```

Note that you have to be able to log in to the system in order to use *t2lincks*.

3.2 Security

In general, given the environment in which the system was developed, security has not been given priority. By default, the current system is installed with encrypted passwords, which the user can change by using the *dbpasswd* program. It is up to you, the administrator, to ensure that the *passwd* file is readable only by the owner. Since *dbpasswd* is installed suid database owner, normal users will be able to update their password themselves.

There is no security in the database at the object level. If the user can access an object through his or her user object, there is no protection to keep him or her from reading it. By using the write group user definitions file, *wrgroups*, you can limit certain users' write access to objects *created* by specific users. Notice the write protection is only in effect if the *wrgroups* file exists.

3.3 Network identification numbers

There are two different network identification numbers which are used within the **LINCKS** system. The first of these is the **MONITORPROGNO**, which is the RPC program number used for communication between the *dbs* process and the *monitor* process. The second, **TCPIPNO**, is the port at the server machine on which communication will take place with the workspace program *xlincks*. The first of these likely will never need to be changed. The default is 572662306, and your network administrator should be able to tell you if you must change it. The latter must be unique to the database. Our site uses numbers between 6000 and 6100, which should be safe. Again, your network administrator can confirm or deny this.

4 Normal Administration Tasks

Here we try to cover the administrative tasks that you will have to carry out from time to time. If we have left out something which you feel should be added, please let us know.

4.1 Starting and stopping a database server

The simplest way to start the server is to log in to the database server machine as the user who owns the **LINCKS** databases. This is the user listed as **OWNER** in the *.lincksrc*. By database server machine, we mean **HOST** in the *.lincksrc*. See section 3.1.1 for full details about this file.

To start the database server, type:

```
lincks -s database-directory
```

To stop the database server, type:

```
lincks -k database-directory
```

To restart the netserv process, type:

```
lincks -r database-directory
```

The diagnostics from the program should be adequate to help you in the event of a failure.

4.2 Garbage collection

The **LINCKS** database system can grow quite large. The reason for this is that no objects are overwritten, but rather new versions of objects are created with the appropriate changes. However, some of the space used by the database can be reclaimed using *cutoff*. The steps are:

1. Stop the database server. See section 4.1 for how to do this.

2. In the database directory create a directory called *Archive* (mkdir Archive) and copy all the database files *.dat, *.mol, *.names, and index into *Archive*.
3. Run *cutoff*. It takes only one arguments, the database directory.
   cd ~/mydatabase; cutoff .
   (. is the current directory.)
4. When *cutoff* is finished running, restart the database server.

Be warned that *cutoff* is not exactly speedy, so do not do this at such a time as the database might be needed.

4.3 Checking if the database is running

Upon occasion it is useful to know the status of the database. The program *dbstat* is written for this purpose. Notice that *dbstat* tell us if the database server (netserv) is running with clear text or encrypted passwords, and also if the size of the object identifiers is 32 bits (case below) or 64 bits (possible on an Alpha machine). It produces the following output:

```
obel11> dbstat ~/tmp/test
Configuration of database /home/obelix2/ida/davpa/tmp/test is
     Internet port number: 6099
     Host for servers: obel11
     Database owner: davpa
     Server program directory:
             /home/obelix2/ida/davpa/work/src/bin
     Logging directory:
             /home/obelix2/ida/davpa/tmp/test/log
     The monitor is running.
     The net server responds.
     @(#)netserv.c   1.12 5/3/94, using encrypted passwd, 32 bit label.
     The database is available for workspace connection.
```

4.4 How to Change the LINCKS Server Machine

1. Stop the current server process. See Section 4.1.
2. cd database-directory
3. Edit **.lincksrc**, change the *HOST* value to the name of the machine to which you are moving the server.
4. Verify the *MONITORPROGNO* in **.lincksrc**. It's unlikely that you'll need to change this, but see section 3.3 for information about how to do this if necessary.
5. Restart the server. See Section 4.1.

4.5 How to add a new user

In order to add a new user to the **LINCKS** system, the following steps must be taken.

1. Log in as user who owns the database files and go to the database directory where you want to add the user.

2. Edit the file *passwd*. This file contains each user's name, user number, and password. The format of the file is as follows:

username1	uid1	password1
username2	uid2	password2
username3	uid3	password3
username4	uid4	password4
...	...	...

   Add the new user in the same way. Note that the password is by default encrypted. Leave the password field empty and use *dbpasswd* to add a new password for the new user. However, if LINCKS is installed with clear text passwords, you can add the password by editing the file. Moreover, it is a good idea having the *passwd* file readable only by the owner.

3. Edit the file *groups*. This file outlines the objects to which a user has read-write access. The format of the file is as follows:

uid1	uid1	uid2	$\cdots$	uid$N*$
uid2	uid1	uid2	$\cdots$	uid$N*$
...	...	...	...	...*

   which means that *uid1* has read-write access to all objects created by users 1 though N (where N is the number of the last user). The asterix ($*$) means that this is the last uid for that user id. See figure 1.

   Add the new uid (which you added to the *passwd* file in the previous step) to the *groups* file and give the new user access to as many objects created by other users as you feel is appropriate. Since security has not been an major consideration, our general practice has simply been to give each new user access to all other users' objects, thereby giving them access to all information in the database. Compare figure 1. to figure 2 for an example.

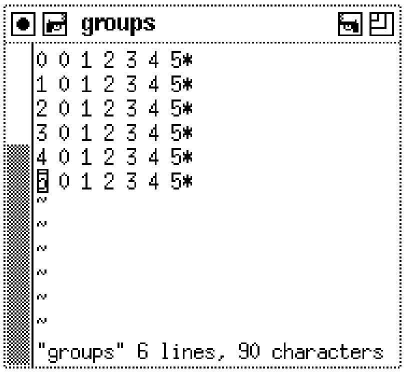

Figure 1: Group file prior to editing

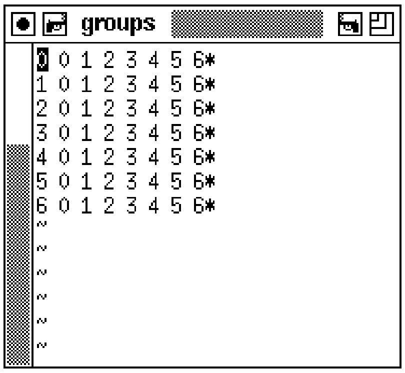

Figure 2: Group file after editing

   Note that both the *passwd* and *groups* as well as the optional *wrgroups* file should be read/writable only by the owner of the database files.

Optional Edit the *wrgroups* file if you are using the write protection provided by this file. It is constructed and used in the same way as the *groups* file, but defines the groups to which the users have write access, in which case the *groups* file only defines read access.

4. Stop and restart the database servers (see section 4.1).

5. Start *xlincks* and log in as user *lincks*. Go to the lincks user object window (see figure 3) and expand the **USERS** field. You will now get

the List of Users window where all users in the database are listed (see figure 4).

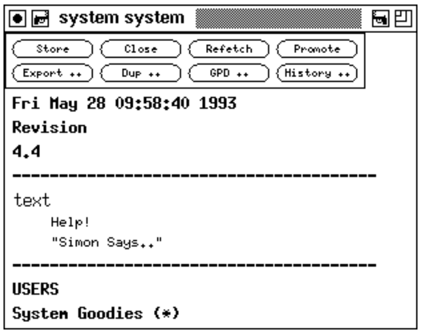

Figure 3: LINCKS system root window

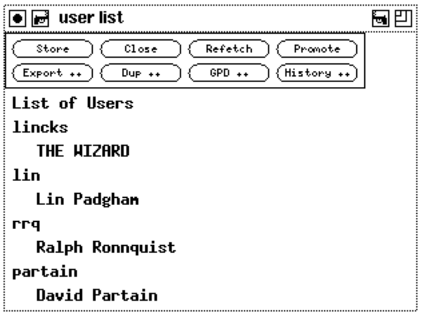

Figure 4: LINCKS list of users

6. Click once at the first user id (in our example, lincks) and then type *meta*-l *meta*-i to create a new user template. (see figure 5).

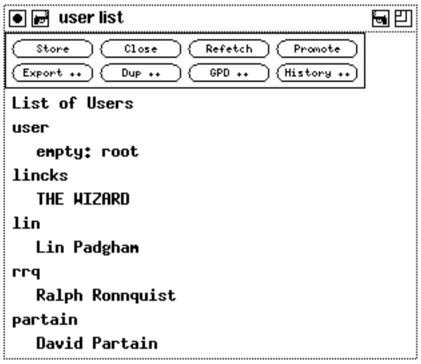

Figure 5: Modified LINCKS list of users

7. Substitute `user` for userid ('tom', for example) and substitute the item `empty: root` with the user's name ('Tom Barth', for example). Store everything.

8. Expand on the user's full name (i.e. 'Tom Barth') which will create a new user object window titled **Tom Barth**. This object is a copy of the default user object and has to be filled in. Expand an old user name field (i.e. David Partain) and you will get a user object window titled **David Partain**. You will use this user window to access and link text objects to the new user object window.

9. Go to the **Tom Barth** window and replace the first <<*gpdname*>> with the GPD name *text* and move the mouse cursor out of the window (to save the object). You will now get a field, <<*links*>>, just below text. You can now link in various things at the <<*links*>> position that will automatically be retrieved from the database when Tom Barth logs in. All users should have the 'Help!' and 'Simon Says...' links, so first link these from an old user to the new by first clicking the left mouse button on the item in the old (i.e. 'Help') followed by the item <<*links*>> in the new, then push the "Add Link" button in the command menu. The field <<*links*>> will be replaced by 'Help!'. Repeat the procedure for 'Simon Says...'. It should look something like figure 6.

10. Click once on the item *text* in the **Tom Barth** window and *meta*-l *meta*-i. This will create two new items, *gpdname* and <<*links*>> above *text*.

    Replace *gpdname* with *user* and link in the **Tom Barth** item to where it says <<*links*>>. You can add the link to any item called **Tom Barth**, no matter which window, since they all refer to the same object. This should give you figure 7.

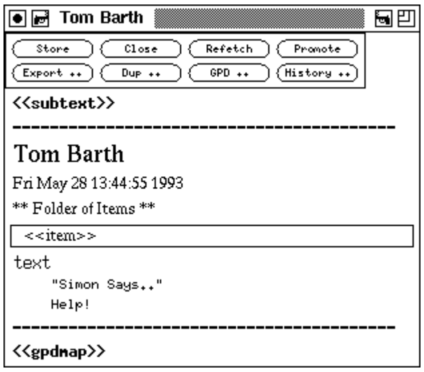

Figure 6: Setting 'text' in user object

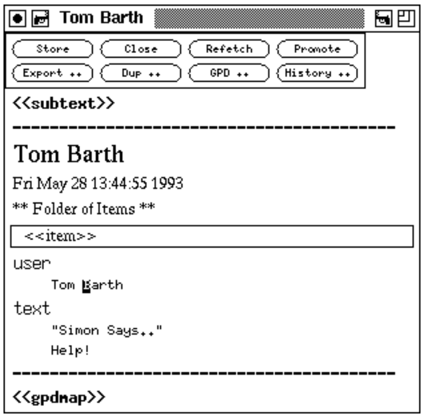

Figure 7: Setting 'user' in user object

11. Fill in the empty slot <<gpdmap>> at the bottom of the window with something like 'Tom's GPD map'. This will be where Tom can store his own GPD's. The completed user object should look like figure 8.

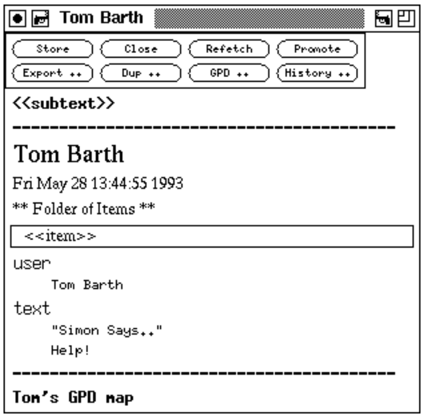

Figure 8: The complete user object

12. Store everything, and you're done. Exit and log in as the new user to verify that all went ok.

5 Problems you might encounter

There are of course things which can go wrong. Here we will try to anticipate some of the problems you might have and offer appropriate solutions.

5.1 When the server machine crashes

Occasionally, the machine where the database servers for the **LINCKS** database system are running will crash. This might cause a problem with a lock file, `.indexfile.lock` left in the database directory by the crashed *monitor* or *cutoff*. When you attempt to start the server(s) after the crash, you might receive the following error:

```
monitor: could not create lockfile ...
```

and the content of the file which tell us the name program (*monitor* or *cutoff*), the host and process id of the program. After verifying that there is no program on the specified host, simply remove the lock file(s) `.lockfile.index` and then start the server(s).

5.2 Adding information to the database

The easiest way to add external data to the database is to use *t2lincks* as outlined in section 3.1.3. Keep in mind that *t2lincks* requires specially formatted files and a good understanding of the database structure. Creating these files can be done by hand, but the simplest way is to use the *t2lincks* dumping facility in *xlincks*, which is fully documented in the user's manual.

Index