User Database (UDB) [6]

The user database (UDB) contains an entry for each user allowed to log into and run jobs on your system. The UDB, which replaces the traditional `/etc/passwd` authorization file, allows faster access to an individual user's information than with the user database previously supplied; it also allows safe use of multiple sources when user information is being changed.

Warning: If your site is using SecurID, there must be consistent definitions in the UDB and the SecurID database. See the preface of this manual for more information on SecurID documentation.

Processes needing information about a user do not need to know the structure or design of the UDB; library interface functions present user information in forms compatible with traditional use. Also, Cray Research continues to support the traditional `/etc/passwd` and `/etc/group` files.

The `/etc/passwd` file is automatically generated during UDB updates and requires no manual modification. The `/etc/group` file must be manually updated to include the proper group IDs and group names, but the group membership lists are automatically maintained when the UDB is updated. The `/etc/acid` file must be manually updated to include the proper account IDs and account names; this file has no membership lists.

The UDB is the core component of the user limits feature, which is discussed in Section 6.2.1, page 131. This section describes the UDB maintenance utilities.

6.1 Login accounts and the UDB

The UDB consists of the files `/etc/udb` and `/etc/udb.public`, plus extension files `udb.index`, `udb.priva`, and `udb.pubva` in the directory `/etc/udb_2`. These extension files support additional fields that could not be added to the existing files. The UDB files `/etc/udb.public`, `/etc/udb_2/udb.index`, and `/etc/udb_2/udb.pubva` have public read permission and reflect all current public information in the database. Sensitive information, such as encrypted passwords and security fields, are present only in the files `/etc/udb` and `/etc/udb_2/udb.priva`.

The UDB has one entry, or record, per user or resource group. Each entry has many fields, which are divided among the UDB files. The access library `libudb`(3) collects all information belonging to a specific record and presents it

in a combined form to the calling process. For a description of the exact content of UDB entries, refer to the udbgen(8) man page.

Warning: For information on setting up accounts on a Cray ML-Safe configuration of the UNICOS operating system, see Chapter 8, page 145.

6.1.1 Providing login accounts

A primary responsibility of any system administrator is to provide users with system login accounts. These logins should allow users to access the system as efficiently as possible, with the smallest possible impact on the system and on other users.

A login needs the following two elements:

- An entry in the UDB
- A home directory for the login

Use the udbgen(8) command to create and maintain the UDB. The udbsee(1) utility converts the database format to an ASCII file that may be used as input to udbgen.

Although the UDB exists in an encoded form, the udbsee utility converts the UDB into a simple text format file. The udbsee utility also provides a number of selection and formatting options that may be used to extract data for reports and other administrative uses. Users can always use udbsee to view their own non-secure control parameters, and, in the form of a text file, the UDB can be transported easily from system to system and manipulated by other UNICOS commands.

Note that *dynamic* user information is updated internally in the UNICOS system but is not written into the UDB except when certain events occur. Therefore, the UDB files may not show the exact state of accumulated information. However, alterations to *fixed* user information are immediately reflected in the UDB because the database editor writes changes as soon as they have been verified by the administrator. The udbpl(8) command produces reports containing information from the kernel tables and the UDB.

6.1.2 Removing login accounts

To remove a login account, use the udbgen(8) delete subcommand to remove the relevant entry from the UDB. However, because the user's files and work in progress may be valuable to other users working with a particular project, it is

more useful initially to disable the user's ability to log in to his or her former account. To do this, use the `udbgen` command to disable the login:

```
udbgen -c 'update:john:permbits+:system-restricted:'
```

After the login has been disabled long enough for authorized users to retrieve any valuable files, you should delete the login's home directory and files.

You may also delete the entry in the UDB at this time; it is more useful, however, simply to keep the entry in the UDB as a record of the login's existence. This may prove useful in cases in which the deleted user owns files in directories other than the home directory and also for accounting purposes.

6.2 User control capabilities

UDB entries can include not only general information users need to log in and establish an initial environment, but also specific information for controlling user limits. This section describes user limits and privileges and their use.

6.2.1 User limits

User limits are generally applied to processes or jobs and establish the maximum amount of a resource that can be requested. Limit information may be separated for job (J) and process (P) usage. Most limits are actually two values; one for batch and the other for interactive. This means that you can provide restricted resources for interactive use, for example, without limiting a user's batch resources to the same degree. The following limits are available:

Resource	Control
Number of processes	J
CPU time	J, P
Memory	J, P
Secondary data segments (SDS) (Cray PVP systems only)	J, P
Tapes	J
File allocation	J, P
PE limit	J
MPP time	J, P

MPP barrier	J
Core file limit	P
Open file limit	P
Shared memory segments (CRAY T90 series systems only)	J
Shared memory size (CRAY T90 series systems only)	J

Limits can be disabled by assigning a 0 value, except for CPU time, memory, file allocation (where a zero value means unlimited), and the open file limit (which can be set only within a range of values). When a new UDB record is created with a minimum set of parameters, for example, as with the following command, default values are assigned to all limit fields.

```
udbgen -c 'create:buck:uid:230:gids:10:
```

The default values are as follows:

- Allow unlimited CPU, memory, file, and core file limits
- Deny access to all other resources (such as tapes and SDS)
- Set the open file limit to 255
- Set the process limit to the configured kernel process limit

The defaults are a property of the UDB; the administrator can change them to site-specific defaults as desired.

6.2.2 Privileges

Privileges are enforced in several ways, depending on what they are and how the privilege is handled in the system. Various mechanisms exist to identify privilege violations that are affected by this control. Some privileges, such as interactive authorization, should be made clear to the user.

There are a number of controls for establishing privileges. The following list shows those controls necessary ("Type" can be categorized as an L or an A. The L represents a single value privilege, such as true, false, or an integer, and an A represents an array of privileges):

Privilege	Type
Account IDs	A

Compartments	A
Default compartments	A
Minimum compartments	A
Default security level	L
Default integrity class	L
Default integrity category	A
Group IDs	A
Maximum security level	L
Minimum security level	L
Maximum integrity class	L
Categories	A
Permissions	A
MLS permissions	A
Site-specific	A

Note that there are two permission fields in the UDB: *permbits* for user permissions and *permits* for MLS permissions.

Some arrays may also be bit lists, as used for security controls. Some privileges, such as access to tapes and SDS, are implied by nonzero values in the relevant limit control field and do not require an explicit privilege in this category.

Space for 32 site-specific privilege bits is provided. There is no use of these bits in the released system.

6.2.3 Quota fields

The UDB includes three fields for controlling quotas:

Field	Description
CPU quotas	Tenths of seconds allotted
CPU quotas used	Accumulated CPU time in tenths of a second

Login failures	Quotas for UNICOS MLS feature

6.2.4 Other UDB information

There is a collection of items that do not fit neatly into the limit, quota, or privilege categories previously described. They mostly involve the areas of data migration and the fair-share scheduler, as follows:

Control	Category
Authorized shares	Share
Decaying accumulated costs	Share
Last decay time	Share
Nice increment	System
Online thresholds	Data migration
Password aging	System
Media selection	Data migration

An example of fair-share scheduling using UDB entries can be found in *UNICOS Resource Administration*, Cray Research publication SG–2302.

6.3 The `/etc/passwd` and `/etc/group` files

The `/etc/passwd` and `/etc/group` files exist and can be used with programs. They are automatically updated when the UDB is updated. You do not need to update these files, except to add group names and group IDs to the `/etc/group` file when a new group is added.

6.3.1 The `/etc/passwd` file

The `/etc/passwd` file (see `passwd`(5)) is maintained by the `udbgen`(8) command and is provided for compatibility with previous systems (and will never be removed because it is an integral part of the UNIX system). This file is not capable of performing UNICOS user validation.

An entry in `/etc/passwd` has the following format:

```
login:passwd:uid:gid:comment:home:shell
```

login The identifier for the login. The user uses this as the name under which to log in to the system. It is often helpful to the administrator (and to the system's users) if there is some evident logic behind the assignment of login names to users; typical examples include the user's initials, last name (plus a first initial), or employee identification number. Alternatively, because the `who`(1) utility lists login names of those logged in to the system, login names can be arbitrary strings of letters or numbers if the users of those logins require anonymity when using the system.

passwd The encrypted string representing the login's password. This string is always an asterisk (*) in a system using the UDB.

uid The numeric user identification number given to the user when he or she logs in to the login account.

gid The numeric group identification number given to the user when he or she logs in to the login account. The *gid* should be a valid group listed in the `/etc/group` file (see `group`(5) for details on the file format).

comment Traditionally used to store the name of the user for whom the login has been created.

home The path name of the login's home directory, to which the user's current directory is set when the user logs in. The home directory must exist, and its user and group ownership should be set to the user's uid and gid numbers. The directory's permissions should be set so that the owner has read, write, and execute (search) permission for the directory; the group and other permissions may be set according to local security considerations.

shell The program executed as the user's login shell program. This field may be left blank, in which case the shell program is the standard shell `/bin/ksh` (see `ksh`(1)).

6.3.2 The `/etc/group` file

The `/etc/group` file (see `group`(5)) is provided to translate group names to group IDs and group IDs to names. It is not used for user validation.

An entry in `/etc/group` has the following format:

```
group:passwd:gid:logins
```

group The group name. The name can be up to 8 characters long, the first of which must be alphabetic; the remaining characters are alphanumeric.

passwd The encrypted password for the group. This field is always set to an asterisk (*) (this field is unused) in a system using the UDB.

gid The numeric group ID. The *gid* is an integer with 0 through 99 reserved for system use and 100 through 65535 available for user group identification.

logins A list of login names as defined in the UDB. The names are separated by commas.

If the list of login names is longer than approximately 400 characters, additional lines are created in `/etc/group` to hold the remainder of the membership list. All the lines for a single group are adjacent in the file, and the *group*, *passwd*, and *gid* fields in each line for each group member are identical. This accommodates groups with an arbitrarily large membership, while keeping the line length in `/etc/group` within reasonable bounds and imposing minimal impact on existing usage.

You can edit this file in order to add a new group name. To do so, add a line similar to the following:

group_name`:*:137:`

Otherwise, a group name cannot be used with the `udbgen gid` field directive, only group ID numbers. When a new group ID number is introduced through the `udbgen gid` field directive, a default name, `G-`*nnnnn*, is created (where *nnnnn* is the gid number). You may later change this name in `/etc/group` to something more meaningful.

6.4 The nu command

You can use the `nu`(8) command to create, modify, delete, and eliminate login accounts. If you prefer to use a graphical interface to manage user logins, you can use the `xadmin`(8) command, which has all the functionality of `nu`. For additional information about the `xadmin` command, see the `xadmin`(8) man page or the online tutorial in `xadmin`.

The `nu` command handles file locking and syntax checking, and it permits the use of configurable defaults, making the process of maintaining large numbers of logins easier.

When adding new logins, the nu program prompts for the login ID, password, name, and other information for each new user. The program facility then creates the login, creates its directories, initializes the directory contents, and makes an entry in a log file.

When modifying logins, the nu program asks repeatedly for login names and instructions for the changes that are to be made to those logins. When the changes are completed, it sorts the updated login records and merges them simultaneously into the UDB.

When logins are deleted, an entry exists in the UDB for the deleted logins. This prevents those specific UIDs from being reused, and it permits accounting data to be meaningful after the accounts are deleted. The program repeatedly asks for the names of logins to be deleted and verifies the deleting of files within those logins.

The nu program can also eliminate logins. In this case, nu deletes almost all information pertaining to a specified login ID.

A complete description of the use of nu and the setup of the configuration file are presented in the *UNICOS Administrator Commands Reference Manual*, Cray Research publication SR–2022.