April 1996
Order Number: 312891-004

Paragon™ System

Multi-User Accounting and Control System

Manual

Intel® Corporation

WARNING

Some of the circuitry inside this system operates at hazardous energy and electric shock voltage levels. To avoid the risk of personal injury due to contact with an energy hazard, or risk of electric shock, do not enter any portion of this system unless it is intended to be accessible without the use of a tool. The areas that are considered accessible are the outer enclosure and the area just inside the front door when all of the front panels are installed, and the front of the diagnostic station. There are no user serviceable areas inside the system. Refer any need for such access only to technical personnel that have been qualified by Intel Corporation.

CAUTION

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

LIMITED RIGHTS

Preface

This manual presents the information that system administrators and users need to know in order to use the Paragon™ system's Multi-User Accounting and Control System (MACS). This manual assumes that you are already familiar with the Paragon system.

Organization

Chapter 1	"Overview of MACS" introduces the MACS system.
Chapter 2	"Basic User Procedures" provides everything a user needs to submit jobs through the MACS system.
Chapter 3	"MACS System Administration" provides MACS account management and system administration procedures for the MACS system administrator.
Chapter 4	"MACS Setup and Configuration" provides information for setting up MACS for specific Paragon system configurations or for specific computing needs.
Chapter 5	"MACS Command Reference" includes reference manual pages for each of the MACS commands.

Notational Conventions

This manual uses the following notational conventions:

Bold Identifies command names and switches, system call names, reserved words, and other items that must be used exactly as shown.

Italic Identifies variables, filenames, directories, processes, user names, and writer annotations in examples. Italic type style is also occasionally used to emphasize a word or phrase.

`Plain-Monospace`
Identifies computer output (prompts and messages), examples, and values of variables. Some examples contain annotations that describe specific parts of the example. These annotations (which are not part of the example code or session) appear in *italic* type style and flush with the right margin.

Bold-Italic-Monospace
Identifies user input (what you enter in response to some prompt).

`Bold-Monospace`
Identifies the names of keyboard keys (which are also enclosed in angle brackets). A dash indicates that the key preceding the dash is to be held down *while* the key following the dash is pressed. For example:

`<Break>` **`<s>`** **`<Ctrl-Alt-Del>`**

`[ ]` (Brackets) Surround optional items.

`...` (Ellipsis dots) Indicate that the preceding item may be repeated.

`|` (Bar) Separates two or more items of which you may select only one.

`{ }` (Braces) Surround two or more items of which you must select one.

Applicable Documents

For more information, refer to the *Paragon™ System Technical Documentation Guide*

Comments and Assistance

Intel Scalable Systems Division is eager to hear of your experiences with our products. Please call us if you need assistance, have questions, or otherwise want to comment on your Paragon system.

U.S.A./Canada Intel Corporation
Phone: 800-421-2823
Internet: support@ssd.intel.com

France Intel Corporation
1 Rue Edison-BP303
78054 St. Quentin-en-Yvelines Cedex
France
0590 8602 (toll free)

United Kingdom Intel Corporation (UK) Ltd.
Scalable Systems Division
Pipers Way
Swindon SN3 IRJ
England
0800 212665 (toll free)
(44) 793 491056
(44) 793 431062
(44) 793 480874
(44) 793 495108

Intel Japan K.K.
Scalable Systems Division
5-6 Tokodai, Tsukuba City
Ibaraki-Ken 300-26
Japan
0298-47-8904

Germany Intel Semiconductor GmbH
Dornacher Strasse 1
85622 Feldkirchen bei Muenchen
Germany
0130 813741 (toll free)

World Headquarters
Intel Corporation
Scalable Systems Division
15201 N.W. Greenbrier Parkway
Beaverton, Oregon 97006
U.S.A.
(503) 677-7600 (Monday through Friday, 8 AM to 5 PM Pacific Time)
Fax: (503) 677-9147

If you have comments about our manuals, please fill out and mail the enclosed Comment Card. You can also send your comments electronically to the following address:

techpubs@ssd.intel.com
(Internet)

Table of Contents

Chapter 1
Overview of MACS

Chapter 2
Basic User Procedures

Chapter 3
MACS System Administration

Chapter 4
MACS Setup and Configuration

Chapter 5
MACS Reference

List of Illustrations

List of Tables

Overview of MACS 1

The MACS system (for *Multi-User Accounting and Control System*) provides accounting and resource management utilities that help control Paragon system operations. These utilities help ensure the effective use of the Paragon system. Most Paragon system users (users not involved with Paragon system administration) do not need to know much about MACS, but they should be aware of the MACS system environment in order to effectively submit their applications.

Basic MACS Concepts

The following discussions briefly describe some basic MACS concepts that are important in understanding the MACS environment.

MACS Accounts

A *MACS account* is a mechanism for identifying and recording group usage of the Paragon system's compute nodes. A MACS account consists of one or more users. The MACS account can limit the total amount of node hours the account users can use, and the maximum number of nodes an application can use. When an application is submitted to the Paragon system, the CPU usage is charged to the MACS account of the user submitting the application.

The user can generate accounting reports (using the **acctrep**, **maclist**, and **jrec** commands) to determine node usage and the amount of node time remaining.

NOTE

MACS will keep track of the CPU usage for *root*, but will not charge the CPU usage to a MACS account unless *root* has been explicitly added to that account as a member.

MACS Modes of Operation

MACS can be configured into one of two modes of operation: *accounting-only* mode, in which the Paragon system usage is monitored and recorded, and *resource control* mode, which keeps all of the functionality of accounting-only mode but adds the ability to set CPU usage limits on MACS accounts and on the individual users of the accounts. The MACS mode of operation is determined by the keywords of the MACDMODE configuration parameter in the *macs.conf* file. (Refer to Table 4-1 on page 4-2 for further explanation of the MACDMODE configuration parameter.)

MACS Accounting-Only Mode

In accounting-only mode, MACS monitors the MACS accounts and the users of the accounts for the following information:

- Node hours used.
- Number of nodes used.
- Application start and end times.
- Application type (either NQS batch request or interactive application).
- Paragon system up/down times.
- Scheduled and unscheduled interrupts.

MACS Resource Control Mode

In resource control mode, MACS can be configured to set CPU usage limits on MACS accounts and on the individual users of the accounts, and to kill applications once the CPU allocation has been reached. The level of resource control (under what circumstances applications are killed) is determined by the ENFORCE configuration parameter and the setting of the **macadmin** *no-kill-flag* (both discussed in Table 4-1 on page 4-2).

In addition to the functionality provided by accounting-only mode, resource control mode allows you to specify the following:

- CPU usage limits for a MACS account.
- CPU usage limits for the users of a MACS account.
- CPU allocation for a user based on a percentage of the MACS account's allocation.
- Whether applications are killed when CPU usage limits are exceeded.

MACS Management

Depending on the site requirements and MACS management strategy, there can be multiple MACS user types associated with MACS: the *system administrator*, the *MACS operator*, the *MACS account manager*, and the *MACS user*.

- The system administrator is someone with *root* access—typically the Paragon system administrator. The system administrator configures MACS for the Paragon system, sets up MACS accounts, and adds users to the MACS accounts.

- The MACS operator receives mail from the MACS system about MACS problems, along with the system administrator. An operator might maintain the system along with the system administrator, possibly during off hours. The operator is defined with the OPERATOR configuration parameter in the *macs.conf* configuration file. Your site may or may not have a MACS operator.

- The MACS account manager controls the CPU resources that have been allocated to a MACS account by the system administrator. The account manager can give users access to a MACS account, set individual user quotas, or give the account's CPU allocation to any other MACS account to which the account manager belongs. Account manager permissions are given by the system administrator using the **macadmin** command. You can find a procedure for doing this in the discussion "Specifying a MACS Account Manager" on page 3-16. Your site may or may not have account managers. Any tasks performed by an account manager can also be performed by the system administrator.

- The MACS user is any Paragon system user who is allowed to execute code on the Paragon system. Each application that the user submits to the Paragon system is billed to one of the user's MACS accounts.

MACS Log Files

The MACS system records Paragon system usage by writing to a daily log file periodically throughout the day. A new log file is created at midnight for the following day. The log files are located in the directory */usr/spool/macs/log.d/macdlog.d* and have the format *macdYYYYMMDD*, where *YYYY* is the year, *MM* is the month, and *DD* is the day. For example, the log file for April 15, 1994 would be named *macd19940415*.

MACS writes to the daily log file at the interval specified by the SYNC_INTERVAL configuration parameter in the *macs.conf* configuration file (the default is 15 minutes).

The MACS account reporting utilities **acctrep** and **jrec** use these daily log files to create their reports.

Applications/Batch Requests

An *application* is any parallel application, interactive or batch; a *batch request* is a batch script containing one or more applications that is submitted to NQS for later execution.

MACS Commands

Most of the MACS commands are provided for the MACS system administrator or account manager. Other commands let all MACS users monitor CPU usage on the Paragon system. The MACS commands are included in Chapters 2 and 3 as part of routine MACS procedures. Manual pages for the MACS commands are included in Chapter 5, and are available on-line using the **man** command. Table 1-1 lists the commands and associated access permissions and provides a brief functional summary.

Table 1-1. Summary of MACS Commands

COMMAND	MODE	USER TYPE	DESCRIPTION
acctrep	accounting-only resource control	system administrator account manager basic user	Generates a usage summary report.
dbconvert	resource control	system administrator	Converts the MACS database file to a newer format.
jrec	accounting-only resource control	system administrator account manager basic user	Processes the accounting log file entries and creates job records.
macadmin	resource control	system administrator	Controls MACS accounts and user access to MACS accounts.
macalloc	resource control	system administrator account manager	Controls MACS accounts and user access to MACS accounts (a subset of the **macadmin** command).
maclist	resource control	system administrator account manager basic user	Reports resource allocation information.
macupdate	resource control	system administrator	Updates or resets MACS account allocation and usage.
si	accounting-only resource control	system administrator	Records the start or end of a scheduled interrupt.

Getting Started

If you want to submit applications to the Paragon system, Chapter 2 shows you how to monitor your MACS account or switch MACS accounts. If you are a system administrator, Chapter 3 provides procedures for routine MACS system administration, account management, and account reporting tasks. If you are a system administrator configuring MACS for a Paragon system, refer to Chapter 4. You can find manual pages for all of the MACS commands in Chapter 5, or on-line using the **man** command.

Basic User Procedures 2

This chapter provides the basic user information required to submit jobs—either interactive jobs or NQS batch requests—to the Paragon system. In general, most Paragon system users (users not involved with Paragon system administration or MACS account management) do not need to know any more about MACS than is presented here.

Some of these procedures depend on the MACS system being configured into *accounting-only* mode or into *resource control* mode. These modes of operation are described in "MACS Modes of Operation" on page 1-2.

Specifying a MACS Account

When you submit a job to the Paragon system, you need to be aware of which *MACS account* will be billed for the CPU usage. A MACS account is a mechanism for identifying and recording usage of the Paragon system's compute nodes.

Finding Your Current MACS Account

Your *current account* is the MACS account that will charged for your CPU usage when you submit a job. Typically, you will have a *default* MACS account when you log into the Paragon system, but this depends on the MACS configuration at your site and is determined by the Paragon system administrator. If you have a default account when you log in, this default account is your current account unless you explicitly change it using the *ACCOUNT* environment variable.

The MACS system determines a user's current MACS account by looking in two locations for a valid account—that is, a MACS account that includes the user as a member. It searches these locations in the following order until it finds a valid account for the user:

1. Your *ACCOUNT* environment variable, if present.

2. The MACS system-level */etc/nx/nx_dflt_accts* file, if present.

If MACS cannot find a valid account for you in those locations, it will look in account 0 (if present) to see if you are a member. Account 0 is a MACS account set up by the system administrator with an account ID of 0.

If you have used the *ACCOUNT* environment variable (described in "Switching Between MACS Accounts" on page 2-4) to specify an account, you can see what your current MACS account is by using the **env** command, which in this case shows that *acct200* is the current account:

```
% env
TERM=xterm
HOME=/home/mikeh
SHELL=/bin/csh
USER=mikeh
LOGNAME=mikeh
PATH=.:/usr/mikeh/bin:/usr/bin
MAIL=/usr/spool/mail/mikeh
TZ=PST8PDT
ACCOUNT=acct200
```

If you have not used the *ACCOUNT* environment variable and the Paragon system administrator has set up an */etc/nx/nx_dflt_accts* file, you can look in the file for your user name. In the following example, the default account for users *rkea*, *boyle*, and *ddh* is *acct100*:

```
% cat /etc/nx/nx_dflt_accts
acct100:rkea,boyle,ddh
acct200:dander,stacy,adam,archer
```

If the system administrator has set up an account 0 and you are a member, account 0 becomes your default (and current) account if you haven't used the *ACCOUNT* environment variable and there isn't a */etc/nx/nx_dflt_accts* file.

You can see if you are a member of account 0 by looking for your user name or an asterisk (*) wildcard in the user list in the */etc/nxaccount* file. For example, either of these entries would include user *dander* as a member of account 0.

```
acct_0:*:0:dander
```

or

```
acct_0:*:0:*
```

If you don't have a default account, you can find what accounts you belong to by looking in the */etc/nxaccount* file and then use the *ACCOUNT* environment variable to select one. Refer to page 4-7 for a description of the */etc/nxaccount* file.

If your site uses resource control mode, the **maclist** command will show your current account along with additional information. The following example shows that the current MACS account for user *mikeh* is *acct100*:

```
% maclist

    name   uid  agid  access percent       allocation            used maxnode
   mikeh 20007     1       U    5.00         16:40:00         0:00:00      64
 acct100           100       N                333:20:00         0:00:00      64
```

Listing all of Your MACS Accounts

If your site uses accounting-only mode, you will need the MACS system administrator at your site to tell you which MACS account (or accounts) you belong to, or look in the file */etc/nxaccount* and search for your user name. Refer to page 4-7 for a description of the */etc/nxaccount* file.

If your site uses resource control mode, you can see what MACS account you are a member of by using the **maclist -u** command. The following example shows all of the MACS accounts for user *mikeh*, as well as additional account information:

```
% maclist -u mikeh

    name   uid  agid  access percent       allocation            used maxnode
   mikeh 20007   100       U    5.00         16:40:00         0:00:00      64
 acct100           100       N                333:20:00         0:00:00      64

    name   uid  agid  access percent       allocation            used maxnode
   mikeh 20007   200       U    0.00          0:00:00         0:00:00      64
 acct200           200       N               1666:40:00         0:00:00      64

    name   uid  agid  access percent       allocation            used maxnode
   mikeh 20007   300       U    0.00          0:00:00         0:00:00      64
 acct300           300       N               1666:40:00         0:00:00      64
```

The output of the **maclist -u** command shows that user *mikeh* belongs to accounts *acct100*, *acct200*, and *acct300*.

Switching Between MACS Accounts

If you belong to more than one MACS account, you can change from one account to another by changing the *ACCOUNT* environment variable. The exact procedure to do this varies, depending on your shell. In the following Bourne shell example, the MACS account *acct200* becomes the current MACS account for the user issuing the commands:

```
$ ACCOUNT=acct200
$ export ACCOUNT
```

In the following C shell example, the MACS account *acct300* becomes the current MACS account for the user issuing the commands:

```
% setenv ACCOUNT acct300
```

You can automate this process by including the *ACCOUNT* settings in a login initialization file such as *.profile* or *.cshrc*.

You can use the *ACCOUNT* environment variable to specify either a MACS account's name or the account's ID. An *ACCOUNT* value that contains only numbers is assumed to be an account ID. For example, the following two lines are equivalent:

```
% ACCOUNT=acct200
% ACCOUNT=200
```

If your site uses NQS, you can use the **qsub -c** command to switch to a specific account for that particular batch request. For example:

```
% qsub -q q64 -c acct200 myapp.sh
request 127.gumshoe submitted to queue: q64
Account = 200
```

Submitting Jobs

Once you belong to a MACS account, you are free to submit interactive jobs or NQS batch requests as you typically would. For example, to submit an interactive job:

```
% myapp -sz 64
```

For more information on submitting interactive jobs, refer to *Paragon™ User's Guide* or the on-line **application** manual page.

Or, to submit an NQS batch request:

```
% qsub -q q64 myapp.sh
request 127.gumshoe submitted to queue: q64
Account = 0
```

To specify a specific MACS account for a batch request:

```
% qsub -q q64 -c acct200 myapp.sh
request 127.gumshoe submitted to queue: q64
Account = 200
```

For more information on submitting an NQS batch request, refer to *Paragon™ System Network Queueing System Manual.*

Getting Your MACS Account Information

Once you have started submitting jobs to the Paragon system, MACS will keep track of your CPU usage in any MACS accounts you belong to. If your site uses MACS accounting-only mode, you can use the **acctrep** command; if your site uses MACS resource control mode, you can also use the **maclist** command.

Using the acctrep Command

You can use the **acctrep** command to monitor your CPU usage. For example:

```
% acctrep -s 04/18/94 -e 04/18/94

account login    queue    size   cpu hours   idle hours   under-used   cpu charge
------- -------- -------- ==== ============ ============ ============ ==========
acct100 mikeh    INTERACT    8        0.312        0.000        0.000        0.312
                            16        0.000        0.000        0.000        0.000
                ------------------------------------------------------------------
                 INTERACT total       0.312        0.000        0.000        0.312

                 BATCH      16        0.342        0.098        0.000        0.440
                ------------------------------------------------------------------
                 BATCH    total       0.342        0.098        0.000        0.440

        ----------------------------------------------------------------------------
         mikeh            total       0.654        0.098        0.000        0.752
```

The **-s** and **-e** options specify the starting and ending dates for the report.

The report will also include usage information for any other users who are members of that MACS account and who have submitted jobs through that account. Refer to "Understanding the Account and User Summary" on page 3-26 for a description of the output produced by **acctrep**.

Using the maclist Command

If your site uses resource control mode, you can use the **maclist** command to monitor your default MACS account. For example:

```
% maclist

    name   uid  agid  access percent       allocation            used maxnode
   mikeh 20007     1     U      5.00         16:40:00         0:00:00      60
 acct100           1     N                  333:20:00         0:00:00      60
```

If your site uses resource control mode, you can use the **maclist** command with the **-u** option to monitor all your CPU usage. For example:

```
% maclist -u mikeh

    name   uid  agid  access percent       allocation            used maxnode
   mikeh 20007     1     U      5.00         16:40:00         0:00:00      60
 acct100           1     N                  333:20:00         0:00:00      60

    name   uid  agid  access percent       allocation            used maxnode
   mikeh 20007     2     U      0.00          0:00:00         0:00:00      60
 acct200           2     N                 1666:40:00         0:00:00      60

    name   uid  agid  access percent       allocation            used maxnode
   mikeh 20007     3     U      0.00          0:00:00         0:00:00      60
 acct300           3     N                 1666:40:00         0:00:00      60
```

Refer to "Understanding the User Status Display" on page 3-18 for a description of the output produced by **maclist**.

MACS System Administration 3

The MACS system administrator is responsible for several types of tasks. These include MACS account management tasks, in which the system administrator creates MACS accounts and assigns users to the MACS accounts, and account reporting tasks, in which the system administrator creates accounting reports that show the usage of the Paragon system. The system administrator also configures MACS for a particular computing site and particular computing needs by editing the configuration parameters in the */usr/spool/macs/conf/macs.conf* file. Refer to Chapter 4 for MACS configuration information and examples.

Most MACS system administration requires *root* access.

Understanding the MACS Daemons

There are two MACS daemon processes: **macpd** and **macd**. **macpd** is the *persistent daemon*. The **macpd** daemon is started first, which in turn starts **macd**. If **macd** fails for some reason, **macpd** restarts **macd** as soon as possible.

The **macd** daemon controls all MACS functions. At start-up time, **macd** reads the */usr/spool/macs/conf/macs.conf* configuration file. The **macd** daemon must be restarted (see page 3-8) before any modifications in the *macs.conf* file will take effect.

The **macd** daemon logs the following types of events:

- **macd** daemon start-up and shutdown.
- Interactive application start and end status.
- NQS batch request start and end status (if NQS is installed at your site).
- Errors and problems.

This status is written into the log file */usr/spool/macs/log.d/macdlog.d/macdYYYYMMDD*, where *YYYYMMDD* is the year, month, and day at which the log file was created. **macd** also records status information when the output data is switched to a new log file. These log files are used by the accounting utilities to generate accounting reports.

If serious errors occur in the **macd** daemon, they are written to the MACS logfiles and also logged via **syslog**. These syslog messages typically appear in the file */var/adm/syslog/daemon.log*; see the manual page for **syslogd** for more information.

Understanding the MACS Database

The MACS database files are located in the */usr/spool/macs/private* directory. The *macd.data* file is the MACS database file, which is often referred to simply as the "MACS database." The initial *macd.data* file is created during the initial MACS setup. Permissions on the directory and file exclude all users except *root*. The MACS daemon, **macd**, uses the database only in *resource control* mode—that is, when the MACMODE configuration parameter is set to *macwatch*.

The *macd.data* file contains binary information on MACS accounts and MACS users. The **macd** daemon maintains this same information in memory dynamically.

While only the **macd** daemon directly accesses *macd.data*, the **macadmin**, **macalloc**, **maclist**, and **macupdate** commands utilize the database via calls to the **macd** daemon.

The **macd** daemon reads the *macd.data* file once when the daemon starts. Afterwards, the **macd** daemon updates the *macd.data* file at the end of a defined time interval called the *sync interval*. The system administrator controls the sync interval by changing the SYNC_INTERVAL parameter in the */usr/spool/macs/conf/macs.conf* file.

When the system administrator stops the **macd** daemon in a controlled manner (i.e. the **si -shutdown** command, the **/sbin/init.d/macs stop** command, or during a normal Paragon system shutdown procedure) the database is written one last time with current information.

There can be as many as four database files in the */usr/spool/macs/private* directory at one time:

- *macd.data* (the current MACS database).
- *macd.data.new* (a temporary version of the MACS database as explained below).
- *macd.data.old* (a previous MACS database; no more than 15 minutes old).
- *macd.data.bak* (a backup of the MACS database made by the last **macupdate** command).

Whenever the **macd** daemon writes the MACS database to disk, it first writes it to *macd.data.new*. The daemon then checks the current *macd.data* file to be sure it is valid. If the *macd.data* file is valid, the daemon renames *macd.data* to *macd.data.old* and then renames *macd.data.new* to *macd.data*. If the current *macd.data* file is not valid, the **macd** daemon copies the *macd.data.new* file to *macd.data.old* and then renames *macd.data.new* to *macd.data*.

This database file copying sequence ensures that even if the system crashes while writing the database, there is always at least one valid copy of the MACS database on disk. The *macd.data* file begins with a checksum and a database version number. The checksum insures that the database has not been corrupted.

When the **macd** daemon starts up, it checks to be sure that the *macd.data* file is valid. If the *macd.data* file is not valid, the daemon checks *macd.data.new*, *macd.data.old*, and finally *macd.data.bak*. The first valid database found is copied to *macd.data* and opened. If no valid database is found, the **macd** daemon prints an error message and aborts.

NOTE

The system administrator must back up the *macd.data* file frequently and securely. One good way to do this would be to use a **cron** job to copy the database file to another system once an hour.

MACS System Utilities Overview

The MACS system includes the account reporting utilities **acctrep** and **jrec**, and the account management utilities **macadmin**, **macalloc**, **maclist**, **macupdate**, and **si**. You can find manual pages for all of the utilities in Chapter 5, or on-line using the **man** command.

acctrep

The **acctrep** command provides a formatted report summarizing usage by account, user, and job node size. It also reports system uptime, system downtime, the number of scheduled interrupts, and the total number of users.

dbconvert

The **dbconvert** command converts the format of earlier versions of the MACS database file to the current version. The **dbconvert** command is only available to system administrators.

jrec

The **jrec** command processes accounting log file information and sends the resulting text output to standard output or to a specified output file. The command output is available for use by customer-developed accounting reports and by the **acctrep** command output. The **jrec** command output includes a single text line for each interactive application or batch request.

macadmin

The **macadmin** command can be run in either interactive menu-driven mode or in command-line mode. It is only functional when the MACDMODE parameter is set to *macwatch* (resource control mode). **macadmin** allows system administrators to add, delete, or change accounts and users in the MACS database. The changes affect **macd**'s internally maintained database image, the disk database file */usr/spool/macs/private/macd.data*, and the account files */etc/nxaccount* and */etc/nxaccomm*. All requests which change the database contents are logged into the file */usr/spool/macs/log.d/macadmin.log* with information about the requester's login name, the utility name, the time of issuance, the type of request, and the associated parameters.

When **macd** runs under accounting-only mode, the system administrator can edit the */etc/nxaccount* file. But if **macd** runs in resource control mode, the system administrator should always use the **macadmin** utility to add, delete, or change accounts/users in order to maintain consistency between the **macd** database and the account files. The **macadmin** command is only available to system administrators.

macalloc

The **macalloc** utility functions as a subset of **macadmin**. It allows the account manager to distribute allocations among users within an account, and to transfer account allocations between accounts.

maclist

The **maclist** utility displays account and user information in the MACS database.

macupdate

The **macupdate** command is used to periodically update the MACS database. The administrator prepares a formatted input file that lists the new allocation for each account or user, and defines a weight flag for what fraction of the remaining allocation is carried over to the next quota period. The **macupdate** command is only functional when MACS is in resource control mode, and this is the only utility that can change usage data in the database. The **macupdate** command is only available to system administrators.

si

The **si** command is used to mark in the log files the beginning and end of a scheduled interrupt. It also allows the system administrator to specify the type, reason, or cause of an interrupt. MACS accounting report utilities use the log entries to interpret downtime as either scheduled or unscheduled downtime. The **si** command only available to system administrators.

System Administration Procedures

The following discussions provide procedures for routine system administration tasks.They include:

- Recording system downtime.
- Starting/stopping MACS.
- Verifying that MACS is running.

Recording System Downtime

The **si** command writes system downtime status information to the daily log file in the */usr/spool/macs/log.d/macdlog.d* directory. The **si** utility can be run manually by the system administrator before the system is shut down and after the system is booted, or automatically within a script. The MACS daemons must be running before **si** can be used. If **si** is used within a script, the script must ensure that the MACS daemons are running before issuing **si**.

A *scheduled interrupt* is scheduled downtime recorded between an **si -on** and **si -off** pair. Scheduled interrupts are used by the system administrator to record Paragon system downtime that has been anticipated or that is the result of routine maintenance.The amount of scheduled downtime is calculated from the difference in wall clock downtime intervals between the two executions of **si -on** and **si -off**.

Unscheduled downtime is any down time not recorded within a **si -on** and **si -off** pair. Typically, unscheduled down time occurs if the Paragon system crashes or hangs unexpectedly. When the system is rebooted, the MACS daemon writes a time stamp to the log file. The unscheduled down time is calculated from this time stamp back to the last MACS time stamp not associated with the reboot. The system administrator can use **si** after a crash to record the cause of the crash.

The following procedures illustrate some typical uses of the **si** command.

Recording Scheduled Downtime

Follow this procedure to record a scheduled downtime:

1. Mark the start of the downtime using the **si -on** command:

```
# si -on -c "Began scheduled maintenance"
```

2. If you are running NQS, shut down NQS with the **qmgr shutdown** command.

```
# qmgr
Mgr: shutdown
NQS manager[TCML_COMPLETE  ]: Transaction complete at local host.
Mgr: exit
#
```

3. Bring the Paragon system down with the system **shutdown** command and perform the scheduled maintenance. Refer to the *Paragon™ System Administrator's Guide* for details on shutting down the system.

4. Reboot the Paragon system. The system will automatically write a time stamp to the MACS log file. (NQS might not restart during the system reboot, depending on the site NQS configuration. Restart NQS now if it's needed.)

5. Verify that the MACS daemons are running:

```
# ps -ef | grep -i macs
root     724874      1  0.0 13:52:39 p0    0:00.06 /usr/lib/macs/macpd
root     724875 724874  0.0 13:52:39 p0    0:00.64 /usr/lib/macs/macd
root     724879 724438  0.0 13:52:56 p0    0:00.04 grep -i macs
```

6. Record the end of the scheduled downtime with the **si** command:

```
# si -off -c "Scheduled maintenance complete"
```

Recording Unscheduled Downtime

If the system hangs, follow this procedure to record the unscheduled downtime:

1. Reboot the Paragon system. The system will automatically issue an **si -boot** command, which writes a time stamp to the MACS log file.

2. Restart NQS, if needed.

3. Make sure that the MACS daemons are running:

```
# ps -ef | grep -i macs
root     724874      1  0.0 13:52:39 p0    0:00.06 /usr/lib/macs/macpd
root     724875 724874  0.0 13:52:39 p0    0:00.64 /usr/lib/macs/macd
root     724879 724438  0.0 13:52:56 p0    0:00.04 grep -i macs
```

4. Record the reason for the unscheduled downtime using the **si** command:

```
# si -c "System hung unexpectedly"
```

Writing a Comment to the MACS Log File

There might be situations in which you simply want to write a message to the MACS log file. For example, the system administrator may want to record that the Paragon system was brought down to single-user mode, interrupting normal system usage. To write a message to the log file:

```
# si -c "Brought system down to single-user"
```

Starting/Stopping MACS

1. If you are running NQS, shut down NQS before you stop MACS:

```
# /sbin/init.d/nqs stop
```

It may take up to several minutes for NQS to shut down. (The delay can be changed by editing the *GRACE_PERIOD* variable in the */sbin/init.d/nqs* script. The actual delay may be greater than the seconds specified by *GRACE_PERIOD* if the system is very busy.)

2. As *root*, you can stop MACS by entering the following:

```
# /sbin/init.d/macs stop
Shutdown message for MACS sent.
```

3. To restart MACS enter the following:

```
# /sbin/init.d/macs start
MACS services provided.
```

4. Now, restart NQS if you stopped it in Step 1:

```
# /sbin/init.d/nqs start
NQS services provided.
```

Verifying that MACS is Running

You can verify that MACS is running by piping the output from the **ps** command through the **grep** command. If MACS is running, you should see **macpd** and **macd**. For example:

```
# ps -ef | grep -i macs
root     724874      1  0.0 13:52:39 p0    0:00.06 /usr/lib/macs/macpd
root     724875 724874  0.0 13:52:39 p0    0:00.64 /usr/lib/macs/macd
root     724879 724438  0.0 13:52:56 p0    0:00.04 grep -i macs
```

MACS Account Management Procedures

MACS account management is performed by someone with *root* privileges using **macadmin**, or someone with MACS account manager privileges using **macalloc**. The functionality of **macalloc** is a subset of the functionality of **macadmin**. (MACS account manager privileges are given to the user by the system administrator when adding the user to a MACS account. Your site may or may not have account managers.)

NOTE

The MACS account management procedures presented in this chapter are shown using **macadmin** and **macalloc** interactively. You can, however, also use these commands on the command line at the system prompt and within scripts, which might be more applicable to your specific tasks. See the manual pages for **macadmin** and **macalloc** for information on using these commands on the command line.

Keep in mind that the **macadmin** and **macalloc** commands are only available in resource control mode, which is discussed in "MACS Modes of Operation" on page 1-2.

If your site uses accounting-only mode, these account management procedures are accomplished by editing the */etc/nxaccount* file as described on page 4-7.

Figure 3-1 shows you how to invoke **macadmin**.

```
# macadmin

        MACS ADMINISTRATION MENU
        ========================

        0       List Account Information
        1       Add New Accounts
        2       Delete Accounts From Database
        3       Change Account Settings
        4       Transfer Account Allocation

        5       List User Account Information
        6       Add Users To Accounts
        7       Delete Users From Accounts
        8       Change User Account Settings
        9       Exit

Enter menu item number:
```

Figure 3-1. The macadmin Main Menu

Figure 3-2 show you how to invoke **macalloc**.

```
% macalloc

        MACS ACCOUNT MANAGEMENT MENU
        ============================

        0       List Account Information
        1       Change Account Settings
        2       Transfer Account Allocation
        3       List User Account Information
        4       Change User Account Settings

        5       Exit

Enter menu item number:
```

Figure 3-2. The macalloc Main Menu

Creating MACS Accounts

Figure 3-3 contains an example that shows how to use **macadmin** to create MACS accounts. You must be logged in as *root* to invoke **macadmin**.

```
# macadmin

        MACS ADMINISTRATION MENU
        ========================

        0       List Account Information
        1       Add New Accounts
        2       Delete Accounts From Database
        3       Change Account Settings
        4       Transfer Account Allocation

        5       List User Account Information
        6       Add Users To Accounts
        7       Delete Users From Accounts
        8       Change User Account Settings
        9       Exit

Enter menu item number: 1


        Add New Accounts
        ----------------

Enter q<return> or Q<return> to go back to menu

Account Name: sim_acct
Account ID: 300
Account Description: Simulation Group
Weight Flag (1/0, default 1): <Return>
Weight defaulted to 1
No_kill Flag (1/0, default 0): <Return>
No_kill flag defaulted to 0
Lock Flag (1/0, default 0): <Return>
Lock flag defaulted to 0
Allocation node-minutes (default 0, ~ unlimited): 600000
MAX. Nodes Allowed (0 - 64, default ~ system_limit): 64

Input ok (y/n)? y
Adding to comm file sim_acct:300:Simulation Group
Added to comm file sim_acct:300:Simulation Group

continue to Add New Accounts (y/n)? n
```

Figure 3-3. Creating MACS Accounts

Understanding the MACS Account Configuration Options

When you add a MACS account to the MACS system, you are asked several questions that configure the account. The following list describes these options.

Account Name	The name of the MACS account. The account name must begin with a letter and must consist of only letters, digits, and underscores.
Account ID	A unique integer identifying the MACS account.
Account Description	A character string describing the MACS account. The string can be any string of printable characters.
Weight Flag	If set to `0`, all of the account's unused node minutes are automatically carried forward into the next accounting period. If set to `1`, a percentage of unused node minutes can be carried forward into the next accounting period. The percentage carried forward is determined by the *alloc_weight* field in the file used as input to the **macupdate** command. See the **macupdate** manual page for more information.
No_kill Flag	If the No_kill flag is on (set to `1`), MACS will not kill applications if the MACS account has exceeded its CPU allocation. If the No_kill flag is off (set to `0`), the keywords of the ENFORCE configuration parameter determine under what circumstances MACS will kill applications.
Lock Flag	If set to `1`, the No_kill flag cannot be reset. If set to `0`, the No_kill flag can be reset.
Allocation node-minutes	The initial number of node-minutes allocated to this MACS account.
MAX. Nodes Allowed	The maximum number of nodes an application can use when submitted through this MACS account. The default number (`64` in this example) is determined by the NODES entry in the *macs.conf* configuration file.

Getting MACS Account Status

Figure 3-4 contains an example that shows you how to use **macadmin** to display MACS account status. You must be logged in as *root* to invoke **macadmin**.

```
# macadmin

        MACS ADMINISTRATION MENU
        ========================

        0       List Account Information
        1       Add New Accounts
        2       Delete Accounts From Database
        3       Change Account Settings
        4       Transfer Account Allocation

        5       List User Account Information
        6       Add Users To Accounts
        7       Delete Users From Accounts
        8       Change User Account Settings
        9       Exit

Enter menu item number: 0


        List Account Information
        ------------------------

Maximum of 10 accounts or users may be specified
at one time.  Use comma as delimiter.
Enter q<return> or Q<return> to go back to menu

Account Names or IDs (default is ALL): <Return>

Default to all accounts.

     name   uid  agid  access percent       allocation            used maxnode
 sim_acct         300     N               10000:00:00          0:00:00       64
 res_acct         500     N               12000:00:00          0:00:00       32
continue to List Account Information (y/n)? n
```

Figure 3-4. Getting MACS Account Status

Understanding the MACS Account Status Display

The MACS account status display consists of several fields, which are described below.

name — The name of the MACS account.

uid — Always empty.

agid — The MACS account's ID, a unique integer identifying the account.

access — Indicates certain flag settings:

- **L** — The Lock flag is set to `1`.
- **W** — The Weight flag is set to `1`.
- **I** — The MACS account's allocation of node time has been used up.
- **N** — The No_kill flag is on (set to `1`) and applications won't be killed if they exceed allocations. If the No_kill flag is off (set to `0`) this field is left blank and the keywords of the ENFORCE configuration parameter determine under what circumstances applications are killed. The No_kill flag is set with the **macadmin -N** command when creating a MACS account or when modifying an account's attributes.

percent — Always empty.

allocation — Total amount of node time allocated to the account in the form *hours*:*minutes*:*seconds*.

used — Amount of node time used by the account.

maxnode — The maximum number of nodes an application can use when submitted through this MACS account.

Adding Users to MACS Accounts

Figure 3-5 contains an example that shows you how to use **macadmin** to add users to MACS accounts. To specify multiple MACS accounts and multiple users, separate the account and user names with commas.You must be logged in as *root* to invoke **macadmin**.

```
# macadmin

        MACS ADMINISTRATION MENU
        ========================

        0       List Account Information
        1       Add New Accounts
        2       Delete Accounts From Database
        3       Change Account Settings
        4       Transfer Account Allocation

        5       List User Account Information
        6       Add Users To Accounts
        7       Delete Users From Accounts
        8       Change User Account Settings
        9       Exit

Enter menu item number: 6


        Add Users To Accounts
        ---------------------

Enter q<return> or Q<return> to go back to menu

Account Names or IDs (default is all): sim_acct,res_acct
User Names or IDs: archer,dand
User Permission (M,T,U, default U): <Return>
Permission defaulted to U
Percentage of Account Allocation (0.0 - 100.0 or skip): <Return>
Allocation node-minutes (default 0, ~ unlimited): 6000
MAX. Nodes Allowed (0 - 64, default ~ system_limit): 64

Input ok (y/n)? y

continue to Add Users To Accounts (y/n)? n
```

Figure 3-5. Adding Users to MACS Accounts

Specifying a MACS Account Manager

Depending on the site requirements, the Paragon system administrator can specify an *account manager* for a MACS account. The MACS account manager controls the CPU resources that have been allocated to a MACS account by the system administrator. The account manager can set individual user quotas, or give the account allocation to any other MACS account to which the account manager belongs.

The account manager performs these tasks using the **macalloc** command, which offers a subset of the functionality of the **macadmin** command. All of the functions of the account manager can be performed by the system administrator. A MACS account manager must have the user permissions **M**, **T**, and **U** (described on page 3-18). These permissions can be given by the system administrator when a user is added to a MACS account, or any time later.

Figure 3-6 contains an example that shows how to give account manager privileges to a user using **macadmin**.You must be logged in as *root* to invoke **macadmin**.

```
# macadmin

        MACS ADMINISTRATION MENU
        ========================

        0       List Account Information
        1       Add New Accounts
        2       Delete Accounts From Database
        3       Change Account Settings
        4       Transfer Account Allocation

        5       List User Account Information
        6       Add Users To Accounts
        7       Delete Users From Accounts
        8       Change User Account Settings
        9       Exit

Enter menu item number: 8


        Change User Account Settings
        ----------------------------

Maximum of 10 accounts or users may be specified
at one time.  Use comma as delimiter.
Enter q<return> or Q<return> to go back to menu

Account Names or IDs (default is all): sim_acct
User Names or IDs (Default is all): dand
User Permission (+/-/,M,T,U, skip for unchanged): +MTU
% of Account Allocation (+/-, 0.0 - 100.0, skip for unchanged): <Return>
Allocation node-minutes (+/-/~,integer, skip for unchanged): <Return>
MAX. Nodes Allowed (0 - 60, ~, skip for unchanged): <Return>

Input ok (y/n)? y

continue to Change User Account Settings (y/n)? n
```

Figure 3-6. Creating a MACS Account Manager

Getting User Status

Figure 3-7 shows an example using **macadmin** to get a user's MACS account status. You must be logged in as *root* to invoke **macadmin**.

Understanding the User Status Display

The user status display consists of several fields, which are described below.

name — The name of the user.

uid — The user's ID.

agid — Account ID, a unique integer identifying the MACS account.

access — Indicates access permissions, which are defined as follows:

- **M** — The user has permission to modify MACS account attributes.
- **T** — The user has permission to transfer MACS account allocation.
- **U** — The user has permission to use the MACS account.
- **I** — The user's allocation of node time has been used up.

Basic MACS users will have a **U**; MACS account managers will have **MTU**.

percent — The percentage of a MACS account's node minutes available to the user.

allocation — Total amount of node time allocated to a user.

used — Amount of node time used by the user.

maxnode — The maximum number of nodes an application can use when submitted through this MACS account.

```
# macadmin

        MACS ADMINISTRATION MENU
        ========================

        0       List Account Information
        1       Add New Accounts
        2       Delete Account From Database
        3       Change Account Setting
        4       Transfer Account Allocation

        5       List User Account Information
        6       Add User To Accounts
        7       Delete User From Accounts
        8       Change User Account Setting
        9       Exit

Enter Menu Item Number: 5


        List User Account Information
        -----------------------------

Maximum of 10 accounts or users may be specified
at one time.  Use comma as delimiter.
Enter q<return> or Q<return> to go back to menu

Account Names or IDs (default is ALL): <Return>

Default to all accounts.
User Names or IDs (Default is ALL): <Return>

Default to all users.

     name   uid  agid  access percent       allocation            used maxnode
  ahearns   212   300      UI    0.00          0:00:00         0:00:00 sys.lim
     bend   231   300    MTUI    0.00          0:00:00         0:00:00 sys.lim
    tramp   230   300      UI    0.00          0:00:00         0:00:00 sys.lim
 sim_acct         300       N              10000:00:00         0:00:00      64

     name   uid  agid  access percent       allocation            used maxnode
  ahearns   212   500    MTUI    0.00          0:00:00         0:00:00 sys.lim
 res_acct         500       N                 0:00:00         0:00:00 sys.lim

continue to List User Account Information (y/n)? n
```

Figure 3-7. Getting User Status

Transferring Account Allocations

Figure 3-8 show you how to use **macalloc** to transfer a percentage of one account's CPU allocation to another account. To do this using **macalloc**, you must be an account manager of both accounts. This can also be accomplished as *root* using **macadmin**.

```
% macalloc

        MACS ACCOUNT MANAGEMENT MENU
        ============================

        0       List Account Information
        1       Change Account Settings
        2       Transfer Account Allocation
        3       List User Account Information
        4       Change User Account Settings

        5       Exit

Enter Menu Item Number: 0


        List Account Information
        ------------------------

Maximum of 10 accounts or users may be specified
at one time.  Use comma as delimiter.
Enter q<return> or Q<return> to go back to menu

Account Names or IDs (default is all): <Return>

Default to all accounts.

     name   uid  agid  access percent       allocation            used maxnode
  acct100           1    N                    16:40:00         0:00:00      64
  acct000           0    N                    50:00:00         0:00:00 sys.lim

continue to List Account Information (y/n)? n
```

Figure 3-8. Transferring MACS Account Allocations (1 of 3)

```
        MACS ACCOUNT MANAGEMENT MENU
        ============================

        0       List Account Information
        1       Change Account Settings
        2       Transfer Account Allocation
        3       List User Account Information
        4       Change User Account Settings

        5       Exit

Enter menu item number: 2


        Transfer Account Allocation
        ---------------------------

Enter q<return> or Q<return> to go back to menu

From Account Name or ID: acct100
To Account Name or ID: acct000
% of Account Allocation (0.0 - 100.0, skip for unchanged): 10

Input ok (y/n)? y

continue to Transfer Account Allocation (y/n)? n
```

Figure 3-8. Transferring MACS Account Allocations (2 of 3)

```
        MACS ACCOUNT MANAGEMENT MENU
        ============================

        0       List Account Information
        1       Change Account Settings
        2       Transfer Account Allocation
        3       List User Account Information
        4       Change User Account Settings

        5       Exit

Enter menu item number: 0


        List Account Information
        ------------------------

Maximum of 10 accounts or users may be specified
at one time.  Use comma as delimiter.
Enter q<return> or Q<return> to go back to menu

Account Names or IDs (default is all): <Return>

Default to all accounts.

     name   uid  agid  access percent       allocation            used maxnode
  acct100           1     N                   15:00:00         0:00:00      64
  acct000           0     N                   51:40:00         0:00:00 sys.lim

continue to List Account Information (y/n)? n
```

Figure 3-8. Transferring MACS Account Allocations (3 of 3)

Creating Accounting Reports

The MACS system records Paragon system usage by writing to a daily log file periodically throughout the day, and when applications begin and end. A new log file is created at midnight for the following day. The log files are located in the directory */usr/spool/macs/log.d/macdlog.d* and have names in the format of *macdYYYYMMDD*, where *YYYY* is the year, *MM* is the month, and *DD* is the day. For example, the log file for April 15, 1994 would be named *macd19940415*.

The MACS account reporting utilities **acctrep** and **jrec** use these daily log files to create their reports.

You can track Paragon system usage by:

- MACS account.
- Individual Paragon user.
- Application size by number of nodes.
- System-wide grand totals.

Example Using acctrep

The **acctrep** command generates a usage summary report from the accounting log files, and writes the report output to *stdout*. The report consists of an initial block of system-level information, such as system uptime and downtime and number of users, a subsequent block of MACS account and user allocation information, and a final block of system grand totals. These information blocks are discussed in "Understanding the System-Level Summary" on page 3-24, "Understanding the Account and User Summary" on page 3-26, and "Understanding the System Grand Total Summary" on page 3-27.

Figure 3-9 shows the **acctrep** system-level summary.

```
# acctrep -s 4/18/94 -e 4/19/94

Intel PARAGON Usage Summary Report
- produced on Tue Apr 19 10:30:42 1994



04/18/94 00:00:00 to 04/19/94 10:30:42
System Up 99.97%  ( 1 days 10 hours 29 minutes 59 seconds )
System Down 0.03%  ( 43 seconds )
Scheduled Interrupts:  0
Scheduled Reboots:  0
Scheduled Down Time:
Unscheduled Reboots:  1
Total number of users:  48
```

Figure 3-9. The acctrep System-Level Summary

Understanding the System-Level Summary

The system level summary contains information on the following:

System Up	Amount of system uptime during this report period.
System Down	Amount of system downtime during this report period.
Scheduled Interrupts	Number of scheduled interrupt periods recorded with the **si** command. A scheduled interrupt period is the time lapsed between the **si -on** and **si -off** commands.
Scheduled Reboots	Number of system reboots occurring within scheduled interrupt periods.
Scheduled Down Time	Amount of system downtime occurring within scheduled interrupt periods.
Unscheduled Reboots	Number of system reboots not occurring within scheduled interrupt periods.
Total number of users	The number of users shown in this report.

The system up and system down values are calculated from entries written to the daily log files in the */usr/spool/macs/log.d/macdlog.d* directory. Since the log files are updated once a minute, the downtime calculation can be off by up to one minute.

Figure 3-10 shows the **acctrep** account and user summary.

```
account login    queue     size   cpu hours   idle hours   under-used   cpu charge
------- -------- --------  ---- ------------ ------------ ------------ ------------
ChemSys tramp    INTERACT     4        0.135        0.000        0.000        0.135
                             60        3.240        0.000        0.000        3.240
                 ------------------------------------------------------------------
                 INTERACT total        3.375        0.000        0.000        3.375

                 BATCH        8        0.303        0.007        0.000        0.303
                 ------------------------------------------------------------------
                 BATCH    total        0.303        0.007        0.000        0.303

        ------------------------------------------------------------------------------
        tramp             total        3.678        0.007        0.000        3.678

--------------------------------------------------------------------------------------
ChemSys                   total        3.678        0.007        0.000        3.678

npplabs mikeh    INTERACT     1        0.050        0.000        0.000        0.050
                             2        0.034        0.000        0.000        0.034
                             4        0.135        0.000        0.000        0.135
                             8        0.152        0.000        0.000        0.152
                 ------------------------------------------------------------------
                 INTERACT total        0.371        0.000        0.000        0.371

        ------------------------------------------------------------------------------
        mikeh             total        0.371        0.000        0.000        0.371

        kinglee  INTERACT    16        0.809        0.000        0.000        0.809
                             32        1.095        0.000        0.000        1.095
                             50        3.111        0.000        0.000        3.111
                 ------------------------------------------------------------------
                 INTERACT total        5.015        0.000        0.000        5.015
                 BATCH        8        1.815        0.137        0.000        1.815
                 ------------------------------------------------------------------
                 BATCH    total        1.815        0.137        0.000        1.815

        ------------------------------------------------------------------------------
        kinglee           total        6.830        0.137        0.000        6.830

--------------------------------------------------------------------------------------
npplabs                   total        7.201        0.137        0.000        7.201
```

Figure 3-10. The acctrep Account and User Summary

Understanding the Account and User Summary

In the MACS account and user summary, the parallel applications are sorted into predefined *bins* by the number of nodes used. The number of bins and their node sizes are specified in the */usr/spool/macs/conf/actable* file. The MACS account and user summary contains the following information:

account	The MACS account's name.
login	The login names of the users within a MACS account. For example, an account named *sw_labs* might include the users *archer* and *tracie*.
queue	Either BATCH for NQS batch requests, or INTERACT for interactive applications.
size	The number of nodes (bin size) a parallel application ran on. If an application uses a different number of nodes than those allowed by the bins, the application is placed into the smallest bin that will hold it. The bin sizes are specified in the */usr/spool/macs/conf/actable* file.
cpu hours	Total node hours used by applications in this bin during the period of the report.
idle hours	Idle time of an NQS batch request. Idle time is accrued when a batch request is rolled in but sits idle. No idle time is accrued when an NQS batch request is rolled out.
under-used	Node hours that are available to an NQS batch request but that go unused while the request is running. Under-used time accrues when a batch request uses fewer nodes than the number of nodes NQS has reserved for it.
cpu charge	The total charge for a parallel application.

The cpu charge field is calculated using the following formula:

cpu charge = *rate1* * *cpu hours* + IDLERATE * *idle_node_hours*
+ UNDERUSE * *under-used*

For interactive applications, the value for *rate1* is determined by the DEF_QRATE charge. (DEF_QRATE, IDLERATE, and UNDERUSE are defined in the */usr/spool/macs/conf/macs.conf* configuration file.)

For NQS batch requests, the value for *rate1* is determined by the accounting charge for the queue to which the request was submitted, which is listed in the */usr/spool/macs/conf/nqstable* configuration file. The value for *rate1* defaults to the value set with the DEF_QRATE configuration parameter if there is no specification in the *nqstable* file.

The idle hours and under-used values are only calculated for NQS batch requests and are 0 for interactive jobs.

Figure 3-11 shows the **acctrep** system grand total summary.

```
04/18/94 00:00:00 to 04/19/94 10:30:42

              size   cpu hours    total charge %_distr %_usage
============= ==== ============= ============= ======= =======
GRAND TOTAL      1         0.050         0.050  000.45  000.00
GRAND TOTAL      2         0.034         0.034  000.31  000.00
GRAND TOTAL      4         0.270         0.270  002.48  000.01
GRAND TOTAL      8         2.270         2.270  020.86  000.10
GRAND TOTAL     16         0.809         0.809  009.43  000.03
GRAND TOTAL     32         1.095         1.095  010.06  000.05
GRAND TOTAL     50         3.111         3.111  028.59  000.15
GRAND TOTAL     60         3.240         3.240  029.78  000.15

FINAL TOTAL               10.869        10.869  100.00  000.39
```

Figure 3-11. The acctrep System Grand Total Summary

Understanding the System Grand Total Summary

The system grand total summary contains information on the following:

size	The bin size of the parallel applications. If an application uses a different number of nodes than those allowed by the bins, the application is placed into the smallest bin that will hold it. The bin sizes are specified in the */usr/spool/macs/conf/actable* file.
cpu hours	The total node hours used by all applications in the bin.
total charge	The total charge for all jobs conforming to a particular bin size. This charge includes idle time and under-used time for NQS batch requests.
%_distr	Of the total CPU hours used by applications in this report, the percentage of CPU hours used by each bin.
%_usage	Of the total CPU hours available in this report, the percentage of CPU hours used by each bin.

Example Using jrec

The **jrec** command lists jobs in chronological order. The **jrec** command accepts most of the same parameters and options as the **acctrep** command. The **jrec** command, however, doesn't provide user, MACS account, or system-level usage totals. Figure 3-12 shows a sample **jrec** report.

```
# jrec -s 1/4/94 -e 1/4/94 -h SSD
SSD Intel PARAGON Job Usage Report
- produced on Tue Jan  4 11:55:54 1994


01/04/94 11:15:00 to 01/04/94 11:55:54

account  login    type size   cpu-hours  idle cpu-hrs under-used   total charge
------- -------- ---- ---- ------------ ------------ ------------ -------------
quarks      adam  1    4         0.334        0.007        0.000         0.334
swengr  kinglee  1    4         0.335        0.003        0.000         0.335
quarks      adam  0    8         0.324        0.000        0.000         0.324
quarks     tramp  1    4         0.334        0.002        0.000         0.334
swengr      chip  0   32         2.686        0.000        0.000         2.686
(0757710900) 01/04/94 11:15:00 to (0757713354) 01/04/94 11:55:54 - down 0 seconds
scheduled down 0 seconds, scheduled 0, reboot 0; unscheduled reboot 0
```

Figure 3-12. The jrec Report

Understanding the jrec Report

In the **jrec** report, the parallel applications are sorted chronologically. The report summary contains the following information:

account	The MACS account's name.
login	The user who submitted the application.
type	The application type. Either 0 for interactive applications, 1 for NQS batch requests, or 2 for dedicated NQS batch requests.
size	The number of nodes a parallel application ran on.
cpu-hours	The total node hours used by the application.
idle cpu-hrs	Idle time of an NQS batch request. Idle time is accrued if a batch request is rolled in but sits idle. No idle time is accrued when a request is rolled-out.
under-used	Node hours that are available to an NQS batch request but that go unused while the request is running. Under-used time accrues when a batch request uses fewer nodes than the number of nodes NQS has reserved for it.
total charge	The total charge for a parallel application.

jrec invoked with the **-j** option includes the following information in the report:

account	The MACS account's name.
login	The user who submitted the job.
partition#	A unique partition identifier that is used by the allocator when communicating with MACS. The same number appears in the MACS logfile entries for jobs running in that partition.
size	The number of nodes a parallel application ran on.
type	The job type; either `0` for interactive jobs, `1` for NQS batch requests, or `2` for dedicated NQS batch requests.
cpu-hours	The total node hours used by the job.
rate	First occurrence: the charge rate for the cpu-hours used.
idle cpu-hrs	Idle time of an NQS batch request. Idle time is accrued if a batch request is rolled in but sits idle. No idle time is accrued when a NQS batch request is rolled-out.
rate	Second occurrence: the rate charged for idle NQS requests.
under-used	Node hours that are available to an NQS batch request but that go unused while the request is running. Under-used time accrues when a batch request uses fewer nodes than the number of nodes NQS has reserved for it.
rate	Third occurrence: the rate charged for under-used node hours.
total charge	The total charge for the job.
job submitted	The date and time the job was submitted to NQS. These are listed as N/A for interactive jobs.
job started	The date and time the job started.
job ended	The date and time the job ended.
st	Job termination status. The value of this field is `1` if the job ran to completion, or `0` if it was terminated by a system crash or hang. In the event of a crash or hang, the application is assumed to have run until the crash or hang.
reqst cpuhrs	Either the time limit requested by the user with the **qsub -lT** option, or the per-request CPU limit for the queue. This is listed as N/A for interactive jobs.
queue name	The name of the queue an NQS request was submitted to. This will be listed as UNKNOWN for interactive jobs.

Updating the MACS Accounting Database

Depending on your site accounting strategy, you may want to update the MACS database on a periodic basis.

The **macupdate** command updates and resets the MACS account allocations and usages in the MACS database. It requires update information as input from a file, one MACS account per line. Each line must be in the following format, with each field separated by spaces or tabs:

acct_name alloc_weight add_alloc usage_remain

The items in the input file are as follows:

acct_name The MACS account name.

alloc_weight A percentage of unused node minutes carried forward, specified as an integer between `0` and `100`. For *alloc_weight* to have an effect, the MACS account's Weight Flag (set with **macadmin**) must be set to `1`; if set to `0`, all unused node minutes are automatically carried forward.

add_alloc An additional allocation of node-minutes.

usage_remain A switch to carry forward all previously used node minutes; `1` carries all used node minutes forward, `0` does not. A `1` also maintains the current setting of the "used" field in the user status display; a `0` resets the "used" field to zero. Refer to "Getting User Status" on page 3-18 for information on the "used" field.

The new allocation is computed by adding together the node-minutes calculated using the *alloc_weight*, *add_alloc*, and *usage_remain* values.

For example, to add 6000 node-minutes to account *acct000*, carrying forward all unused node-minutes and not carrying forward any used node-minutes:

```
acct000 100 6000 0
```

If *acct000* had 500 node-minutes remaining in its allocation, the new allocation would be 6500 node-minutes and there would be no node-minutes used.

To set account *acct100*'s allocation to exactly 6000 node-minutes:

```
acct100 0 6000 0
```

All of the unused node-minutes for *acct100* are thrown away (0% are carried forward). The new allocation is exactly 6000 node-minutes and there would be no node-minutes used.

These examples assume that the weight flag for each account is set to `1` (the default). If the weight flag were set to `0`, all unused node-minutes would be automatically carried forward.

You may want to use **macupdate** as part of a weekly or monthly **cron** job to automatically update the account allocations and usages in the MACS database. See the **cron** manual page for details on doing this.

In Figure 3-13, the MACS account named *acct400* will receive all of its unused node allocation from the previous accounting period, and an additional 1000 hours. The total node hours in the allocation (1999:53:33) is the product of (1000:00:00 - 6:27) + 1000:00:00

```
# macadmin -mlu -a acct400

     name   uid  agid  access percent       allocation            used maxnode
 macsusr2   202     4       U     50.00        500:00:00        00:05:27      64
 macsusr1   201     4       U     50.00        500:00:00        00:00:00      64
 macsusr0   200     4       U     50.00        500:00:00        00:01:00      64
  acct400           4       U                 1000:00:00        00:06:27      64

# cat update-44.txt
acct400 100 60000 0

# macupdate update-44.txt

Start MACD database update

Prepare input file and log file ...... Done!

This utility will ask MACD to stop its internal database-update,
and to start updating the database with information provided
in the input file to this utility.
Continue (y/n)? y

# macadmin -mlu -a acct400

     name   uid  agid  access percent       allocation            used maxnode
 macsusr2   202     4       U     50.00        999:56:46       0:00:00      64
 macsusr1   201     4       U     50.00        999:56:46       0:00:00      64
 macsusr0   200     4       U     50.00        999:56:46       0:00:00      64
  acct400           4       U                 1999:53:33       0:00:00      64
```

Figure 3-13. Updating the MACS Accounting Database

MACS Setup and Configuration 4

Once the Paragon system is installed, you will need to configure MACS for your particular Paragon system configuration and site computing needs. You do this primarily by running the */usr/lib/macs/setup/macs_setup* script, which will ask you specific questions about your site configuration.

Before you run the setup script, you should know if you want to configure MACS for *accounting-only* mode or *resource control* mode. These modes are described in "MACS Modes of Operation" on page 1-2. You should also understand the configuration parameters used to create those modes. Table 4-1 describes all of the MACS configuration parameters you will be asked about during setup.

Finally, you should determine the MACS default account strategy that will be used at your site. Refer to "Understanding Default Accounts" on page 4-6 for more information on default accounts.

Understanding the MACS Configuration Parameters

Table 4-1 describes all of the MACS configuration parameters.

Table 4-1. MACS Configuration Parameters

Parameter	Effect	Default
BINS	Defines the node size (*bin size*) breakdowns that appear in the summary report. If an application uses a different number of nodes than the sizes specified by BINS, the application is placed into the smallest bin that will hold it. For example if `1 256 512` were specified, the report would indicate how many 1-node applications ran, how many 2 to 256-node applications ran, and how many 257 to 512-node applications ran. The node bins should be specified in ascending sequence, and the last node-size should be equal to the total nodes available in the *.compute* partition.	None. You must supply a value.
N_BINS	The number of node bins specified with `BINS`.	None. You must supply a value.
ACT_GRP_LEVEL	This parameter is unused.	Leave default of `3`.
ACT_NAME_SIM	This parameter is unused.	Leave default of `3 3 1`.
ACT_GRP_LABEL	Defines an optional label used in accounting reports.	Leave default of `site account project`.
N_BATCH_QUEUE	Specifies how many NQS batch queues exist.	`0`. You must supply a value if your site uses NQS.
BATCH_QUEUE	Lists all of the NQS batch queues.	None. You must list queues if your site uses NQS.
N_DEDICATE_QUEUE	This parameter is unused.	Leave default of `0`.
DEDICATE_QUEUE	This parameter is unused.	None.
NODES[1]	The total number of nodes available in the *.compute* partition.	None. You must supply a value.
MAILER[1]	The mail program used to send mail to users and to ADMIN in the case of MACS problems. (Typical mailers include */usr/bin/mailx* and */bin/mail*.)	`/bin/mail`

Table 4-1. MACS Configuration Parameters (Continued)

Parameter	Effect	Default
ADMIN[1]	Specifies a MACS administrator. If there is some kind of error (such as an overdrawn account), MACS mails a message to the user and to the MACS administrator (specified with the ADMIN parameter) stating the cause of the problem.	`root`
OPERATOR[1]	Specifies an operator. If there is a MACS system problem, the operator is notified by a broadcast message to the operator's terminal. Your site may not have an operator.	`root`
SWITCHLOG[1]	Because the MACS log files can get rather long, they are switched periodically. `SWITCHLOG` specifies how often the log files are changed, using one of three values: `daily`, `weekly`, or `monthly`.	`daily`
MACDMODE[1]	Allows the administrator to turn off CPU quota monitoring. Allowed values are `acctonly` (provides only accounting reports, referred to as *accounting-only* mode) and `macwatch` (provides both accounting reports and CPU quotas, referred to as *resource control* mode).	`acctonly`
ENFORCE[1]	Specifies the desired level of job control when CPU quotas are exceeded and the *No_kill* flag (set with **macadmin**) is off (set to `0`). The ENFORCE parameter accepts two arguments, which can be used in the following manner: `ENFORCE`. With no arguments, no applications are killed (no control.) (Note that the ENFORCE configuration parameter no longer accepts the invalid entry `nokill`. The correct way to get a "no kill" behavior is to leave the value of ENFORCE blank.) `ENFORCE userkill`. Will kill a user's application when all of the user's allocated time has been used up. When the user allocation is exhausted, this also prevents the user from running a new application. `ENFORCE acctkill`. Will kill all of the applications attached to a MACS account when all of the group's allocated time has been used up. When the MACS account's time allocation is exhausted, no users in that account can submit applications through that account. `ENFORCE userkill acctkill`. The combination of both `userkill` and `acctkill`. See page 5-19 for a discussion of the *No_kill* flag.	No applications are killed.

Table 4-1. MACS Configuration Parameters (Continued)

Parameter	Effect	Default
DEF_QRATE[1]	Specifies the *charge rate* for interactive applications (non-NQS applications) or for applications run in queues that are not defined in the *nqstable* file (see page 4-17). The charge rate is the number of arbitrary units for which each node-hour is charged. Typically each unit is equal to one node-hour, but you might set a higher charge rate for a high-priority queue or a queue with higher-memory nodes. For example, if you use 5 node-hours in a queue with a charge rate of 2.5, you are charged 10 units. These units are the units shown by the **maclist**, **jrec**, and **acctrep** commands.	`1.0`
UNDERUSE[1]	Specifies the charge rate for node hours that are available to an NQS batch request, but that go unused while the request is running. Under-used time occurs when a batch request uses fewer nodes than the number of nodes NQS has reserved for it.	`0.0.`
IDLERATE[1]	Specifies the charge rate for idle node time accrued by NQS batch requests. Idle time is defined as the time when there is no application running in an NQS partition that has been assigned to a batch request.	`0.0`
SYNC_INTERVAL[1]	Specifies the desired time interval (in minutes) that is used for MACS synchronization. At this interval, MACD gets status information from SMD and synchronizes the disk database with the internal tables.	`15` minutes

[1]Since these parameters affect the MACS daemon, MACS must be restarted for new values to take effect. The other parameters only affect the **acctrep** and **jrec** commands, and can be changed anytime prior to issuing the commands.

Enforcing Runtime Limits

Several parameters interact to determine whether or not MACS enforces runtime limits on applications and NQS batch requests. An *application* is any parallel application, interactive or batch; a *batch request* is a batch script containing one or more applications that is submitted to NQS for later execution.

- The USE_MACS parameter in the */etc/nx/allocator.config* file determines whether or not the allocator asks MACS for authorization before running an application. Note that the allocator, and not MACS itself, determines whether or not the application runs. Therefore, if the MACS account has no time left but USE_MACS is set to 0, the allocator allows the application to run anyway since it doesn't check with MACS.

- The MACDMODE parameter in the */usr/spool/macs/conf/macs.conf* file determines how MACS keeps track of the time used by each account.

  - If MACDMODE is set to `macwatch`, MACS records the beginning and end of each application and batch request in the files */usr/spool/macs/log.d/macdlog.d/**, and also deducts the time used by the applications (and idle time used by NQS batch requests) from the MACS account database (*/usr/spool/macs/private/macd.data*). If a MACS account or user runs out of time, MACS instructs the allocator to reject new applications associated with that account or user (and, depending on the setting of the ENFORCE parameter, to kill running applications associated with that account/user).

  - If MACDMODE is set to `acctonly`, MACS records each application and batch request's start and end times in the *macdlog.d* files, but does not maintain the MACS account database.

- The ENFORCE parameter in the *macs.conf* file determines whether or not a running application is killed when a MACS account or user runs out of time.

  - If ENFORCE is null, the application that uses the last node-minute from a user or account is allowed to run to completion.

  - If ENFORCE is set to `userkill` and/or `acctkill`, all applications from a user and/or account are killed immediately.

  If MACDMODE is set to `acctonly`, the ENFORCE parameter is ignored.

  The ENFORCE parameter only affects applications that are running when the MACS CPU allocation for an account or user has been used up. New applications associated with an account or user with no remaining CPU allocation are not allowed to start, regardless of the value of the ENFORCE parameter.

- The macs_flag parameter in the NQS */usr/spool/nqs/conf/sched_param* file determines whether or not NQS thinks MACS is in use at this site.
  - If macs_flag is set to 1, NQS verifies that the MACS account associated with a batch request has some time in it when the request is submitted, and again when the request actually runs. NQS also notifies MACS when a request begins or ends, which MACS uses to determine the idle time used by a request and the name of the queue associated with each request (as seen in the **jrec** report).
  - If macs_flag is set to 0, NQS assumes that MACS does not exist, and neither checks with MACS before running batch requests nor tells MACS anything. MACS can still track the time used by applications within NQS batch requests, but it assumes that all applications are interactive, doesn't know what NQS queue an application ran in, and can't track idle or under-used time.

Understanding Default Accounts

The MACS system determines a user's current MACS account by looking in three locations for a valid account—that is, a MACS account that includes the user as a member. It searches these locations in the following order until it finds a valid account for the user:

1. The user's ACCOUNT environment variable, if present.
2. The MACS system-level */etc/nx/nx_dflt_accts* file, if present.
3. If MACS cannot find a valid account for a user in the above locations, it will look in account 0 (if present) for a valid account. Account 0 is a MACS account set up by the system administrator with an account ID of 0. Creating an account 0, and allowing all users access to it, is one way to ensure all users are members of a MACS account.

The MACS system-level */etc/nx/nx_dflt_accts* file is created by the system administrator as part of MACS configuration. The file lists the user's default MACS account. After logging in, the user can change to another valid account using the ACCOUNT environment variable (see "Switching Between MACS Accounts" on page 2-4 for procedures to do this) or by including the ACCOUNT environment variable in a login initialization file such as *.profile* and *.cshrc* to override the account specified by the */etc/nx/nx_dflt_accts* file.

Depending on site requirements, the system administrator can create an *account 0*, which is a MACS account with an account ID of 0. In accounting-only mode, MACS accounts are created by editing the */etc/nxaccount* file (discussed in Step 1 on page 4-7); in resource control mode, accounts are created using the **macadmin** command.

The entries in the */etc/nx/nx_dflt_accts* file have the following format:

```
account_name:user1,user2,user3
```

For example:

```
acct100:mlearns,doyle,sdo
acct200:bend,tracie,edam,archer
```

Lines must begin with an alpha character (a–z, A–Z) and can contain both alpha and numeric characters (0–9). Comments are allowed and should begin with a (#) sign. Tabs and spaces are not allowed.

If an invalid character, tab or space is encountered, the rest of the line is ignored.

Sample Accounting-Only Configuration

1. Create or edit the file */etc/nxaccount*. Take particular care that every Paragon system user belongs to a valid MACS account. If MACS cannot validate a user or a user's MACS account, all of that user's applications will fail, including any NQS batch requests that are currently queued.

NOTE

Never edit the */etc/nxaccount* file if your site is configured for resource control mode (the MACMODE configuration parameter is set to macwatch). Instead, use the **macadmin** command to add MACS accounts and users.

The format of the file is similar to the Paragon system's */etc/group* file. The following example shows a simple */etc/nxaccount* entry:

```
acct_100:*:100:bend,mikeh,tracie,billops
```

Each field is separated by a colon (`:`). The first field (`acct_100`) is the MACS account's name. The second field is a password field that should always contain an asterisk (`*`). The third field (`100`) is the MACS account's ID. A comma-separated list of MACS users follows the final colon.

You can use the asterisk (`*`) wild card to include all Paragon system users in a MACS account, and the exclamation point (`!`) character to exclude a particular user. For example, this entry would add all Paragon system users to the MACS account `acct_200`:

```
acct_200:*:200:*
```

The following entry would add all Paragon system users except `bend` and `mikeh` to the MACS account `acct_300`:

```
acct_300:*:300:*,!bend,!mikeh
```

The following entry would create account 0 and allow all Paragon system users access to that account.

```
acct_0:*:0:*
```

The */etc/nxaccount* file cannot contain any blank lines or comment lines.

NOTE

The */etc/nxaccount* file must be readable by all users.

2. Determine the default MACS account strategy that best serves your site's computing needs. Refer to "Understanding Default Accounts" on page 4-6 for an overview of default accounts and a description of the */etc/nx/nx_dflt_accts* file. You may want to set up account 0, as discussed in the previous step, or create a */etc/nx/nx_dflt_accts* file. For example:

```
acct_100:bend,mikeh,tracie,billops
```

3. If you are running NQS at your site, ensure that the */usr/spool/nqs/conf/sched_param* file contains a `macs_flag` configuration parameter and that it is set to `1`. Add it if it is missing. The line should look like the following:

```
macs_flag : 1
```

This configuration parameter allows MACS to monitor NQS batch requests and makes NQS verify that users belong to valid MACS accounts.

4. Obtain a listing of all of the NQS queue names. You will need to supply these names during the MACS setup. For example:

```
# qmgr
Mgr: show queue
==========================================
NQS Version: 2  BATCH  PIPE QUEUES on orbison
==========================================
QUEUE NAME      STATUS   TOTAL   RUNNING  QUEUED   HELD TRANSITION  NODE_GROUP
-------------------------------------------------------------------------------
q4-30           AVAILBL    0       0/1        0       0       0       0
q2-30           AVAILBL    0       0/1        0       0       0       1
q2-60           AVAILBL    0       0/1        0       0       0       2
q2-10           AVAILBL    0       0/5        0       0       0       3
```

The output shows the queues *q4-30*, *q2-30*, *q2-60*, and *q2-10*.

5. Shut down NQS:

```
# qmgr
Mgr: shutdown
NQS manager[TCML_COMPLETE  ]: Transaction complete at local host.
Mgr: exit
#
```

6. Edit the */sbin/init.d/allocator* file and delete the `-MACS` flag, if present.

7. Edit the */etc/nx/allocator.config* file and add the following line:

```
USE_MACS=1
```

This allows MACS to monitor interactive applications and makes the allocator verify that users belong to valid MACS accounts.

8. Ensure that there is an entry for the Paragon system in the */etc/hosts* file, and that the entry includes the host name and address. If these entries are not present, MACS will not start and will issue an error message.

9. Reboot the Paragon system.

10. Restart NQS if it isn't automatically restarted by the previous reboot:

```
# /sbin/init.d/nqs start
NQS services provided
```

11 Next, you will be running the MACS setup script to configure MACS for your site. The script will ask you to supply values for several MACS setup configuration parameters. You can enter a question mark (`?`) when prompted for a value for an explanation of the parameter. All of the MACS configuration parameters are discussed in detail on page 4-2 in Table 4-1.

In most cases, you can just accept the defaults. However, you will need to supply values for the following parameters, as these values will be different for each site.

BINS	The node sizes by which you want MACS to sort applications.
N_BINS	The number of bins created by BINS.
BATCH_QUEUE	The names of the NQS batch queues obtained during Step 4 on page 4-8.
N_BATCH_QUEUE	The number of queues specified by BATCH_QUEUE.
NODES	The number of nodes in the *.compute* partition.

The following example shows a sample MACS setup for a 64-node Paragon system running NQS with queues named *q4-30*, *q2-30*, *q2-60*, and *q2-10*.

```
# /usr/lib/macs/setup/macs_setup
Making MACS support directories.
Please specify or validate the MACS scheduling parameters given in the
following prompts.  The values displayed in brackets are the default
values.  To use the defaults, just press the Return key.  If you want
to change the default or when no default is given, enter the desired
value followed by the Return key.  If you want more information on
a parameter, enter a question mark (?).

N_BINS : [10] 8
BINS : [1 2 4 8 16 32 64 128 256 512 ] 1 2 4 8 16 32 50 64
ACT_GRP_LEVEL : [3] <Return>
ACT_NAME_SIM : [3 3 1] <Return>
ACT_GRP_LABEL : [site account project] <Return>
N_BATCH_QUEUE : [0] 4
BATCH_QUEUE : [] q4-30 q2-30 q2-60 q2-10
N_DEDICATE_QUEUE : [0] <Return>
'DEDICATE_QUEUE' parameter has no effect with 'N_DEDICATE_QUEUE' set to 0
NODES : [] 64
MAILER : [/bin/mail] <Return>
ADMIN : [root] <Return>
OPERATOR : [root] <Return>
SWITCHLOG : [daily] <Return>
MACDMODE : [acctonly] <Return>
ENFORCE : [] <Return>
DEF_QRATE : [1.0] <Return>
UNDERUSE : [0.0] <Return>
IDLERATE : [0.0] <Return>
SYNC_INTERVAL : [15] <Return>

These are the parameter values that have been set:
N_BINS                  : 8
BINS                    : 1 2 4 8 16 32 50 64
ACT_GRP_LEVEL           : 3
ACT_NAME_SIM            : 3 3 1
ACT_GRP_LABEL           : site account project
N_BATCH_QUEUE           : 4
BATCH_QUEUE             : q4-30 q2-30 q2-60 q2-10
N_DEDICATE_QUEUE        : 0
DEDICATE_QUEUE          :
NODES                   : 64
MAILER                  : /bin/mail
ADMIN                   : root
OPERATOR                : root
SWITCHLOG               : daily
MACDMODE                : acctonly
```

```
ENFORCE                 :
DEF_QRATE               : 1.0
UNDERUSE                : 0.0
IDLERATE                : 0.0
SYNC_INTERVAL           : 15
Do you want to change any of these now (y/n)? n
For each batch queue, please assign a charge factor (default 1.0)

q4-30 : [1.0] <Return>
q2-30 : [1.0] <Return>
q2-60 : [1.0] <Return>
q2-10 : [1.0] <Return>

These are the parameter values that have been set:

q4-30 1.0
q2-30 1.0
q2-60 1.0
q2-10 1.0
Do you want to change any of these now (y/n)? n
Commit these parameters to the configuration file (y/n)? y
MACS configuration parameters have been updated

Should MACS be started automatically at system boot (y/n)? y
Do you want to start MACS now (y/n)? y

MACS setup is complete.
```

12. Now, verify that MACS is running:

```
# ps -ef | grep -i macs
root     724874      1  0.0 13:52:39 p0    0:00.06 /usr/lib/macs/macpd
root     724875 724874  0.0 13:52:39 p0    0:00.64 /usr/lib/macs/macd
root     724879 724438  0.0 13:52:56 p0    0:00.04 grep -i macs
```

NOTE

The MACS setup script writes to the *macs.conf*, *nqstable*, and *actable* files located in the */usr/spool/macs/conf* directory. You should edit these files by running the */usr/lib/macs/setup/macs_params* script. If you edit these files by hand, be aware that you can only use a single tab or space as a delimiter.

13. You have configured MACS for the Paragon system. All users listed in the */etc/nxaccount* file can now run parallel applications on the Paragon system.

Sample Resource Control Configuration

1. If you are running NQS at your site, ensure that the */usr/spool/nqs/conf/sched_param* file contains a `macs_flag` configuration parameter and that it is set to `1`. Add it if it is missing. The line should look like the following:

```
macs_flag : 1
```

This configuration parameter allows MACS to monitor NQS batch requests and to verify that users belong to valid MACS accounts.

2. Obtain a listing of all of the NQS queue names. You will need to supply these names during the MACS setup. For example:

```
# qmgr
Mgr: show queue
==========================================
NQS Version: 2  BATCH  PIPE QUEUES on orbison
==========================================
QUEUE NAME      STATUS   TOTAL   RUNNING  QUEUED   HELD TRANSITION  NODE_GROUP
------------------------------------------------------------------------------
q4-30           AVAILBL    0       0/1        0       0       0       0
q2-30           AVAILBL    0       0/1        0       0       0       1
q2-60           AVAILBL    0       0/1        0       0       0       2
q2-10           AVAILBL    0       0/5        0       0       0       3
```

The output shows the queues *q4-30*, *q2-30*, *q2-60*, and *q2-10*.

3. Shut down NQS:

```
# qmgr
Mgr: shutdown
NQS manager[TCML_COMPLETE  ]: Transaction complete at local host.
Mgr: exit
#
```

4. Rename the */etc/nxaccount* file, if it exists:

```
# mv /etc/nxaccount /etc/nxaccount.old
```

NOTE

Configuring MACS for resource control mode will automatically create a */etc/nxaccount* file. Never edit the */etc/nxaccount* file if your site is configured for resource control mode (the MACMODE configuration parameter is set to *macwatch*). Instead, use the **macadmin** command to add MACS accounts and users.

5. Edit the */sbin/init.d/allocator* file and delete the `-MACS` flag, if present.

6. Edit the */etc/nx/allocator.config* file and add the following line:

```
USE_MACS=1
```

7. Ensure that there is an entry for the Paragon system in the */etc/hosts* file, and that the entry includes the host name and address. If these entries are not present, MACS will not start and will issue an error message.

8. Reboot the Paragon system.

9. Next, you will be running the MACS setup script to configure MACS for your site. The script will ask you to supply values for several MACS setup configuration parameters. You can enter a question mark (`?`) when prompted for a value for an explanation of the parameter. All of the MACS configuration parameters are discussed in detail on page 4-2 in Table 4-1.

   In most cases, you can just accept the defaults. However, you will need to supply values for the following parameters, as these values can be different for each site.

BINS	The node sizes by which you want MACS to sort applications.
N_BINS	The number of bins specified by BINS.
BATCH_QUEUE	The names of the NQS batch queues obtained during Step 2 on page 4-8.
N_BATCH_QUEUE	The number of queues specified by BATCH_QUEUE.
NODES	The number of nodes in the *.compute* partition.
MACMODE	Turns on/off CPU quota monitoring.
ENFORCE	Specifies the desired level of job control when CPU quotas are exceeded and the *No_kill* flag (set with **macadmin**) is off (set to `0`).

The following example shows a sample MACS setup for a 64-node Paragon system running NQS with queues named *q4-30*, *q2-30*, *q2-60*, and *q2-10*. Resource control is turned on with MACDMODE set to *macwatch*, and ENFORCE specifies that users' applications will be killed when a MACS account's CPU allocation has been used up.

```
# /usr/lib/macs/setup/macs_setup
Making MACS support directories.
Please specify or validate the MACS scheduling parameters given in the
following prompts.  The values displayed in brackets are the default
values.  To use the defaults, just press the Return key.  If you want
to change the default or when no default is given, enter the desired
value followed by the Return key.  If you want more information on
a parameter, enter a question mark (?).

N_BINS : [10] 8
BINS : [1 2 4 8 16 32 64 128 256 512 ] 1 2 4 8 16 32 50 64
ACT_GRP_LEVEL : [3] <Return>
ACT_NAME_SIM : [3 3 1] <Return>
ACT_GRP_LABEL : [site account project] <Return>
N_BATCH_QUEUE : [0] 4
BATCH_QUEUE : [] q4-30 q2-30 q2-60 q2-10
N_DEDICATE_QUEUE : [0] <Return>
'DEDICATE_QUEUE' parameter has no effect with 'N_DEDICATE_QUEUE' set to 0
NODES : [] 64
MAILER : [/bin/mail] <Return>
ADMIN : [root] <Return>
OPERATOR : [root] <Return>
SWITCHLOG : [daily] <Return>
MACDMODE : [acctonly] macwatch
ENFORCE : [] userkill
DEF_QRATE : [1.0] <Return>
UNDERUSE : [0.0] <Return>
IDLERATE : [0.0] <Return>
SYNC_INTERVAL : [15] <Return>
```

```
These are the parameter values that have been set:
N_BINS                  : 8
BINS                    : 1 2 4 8 16 32 50 64
ACT_GRP_LEVEL           : 3
ACT_NAME_SIM            : 3 3 1
ACT_GRP_LABEL           : site account project
N_BATCH_QUEUE           : 4
BATCH_QUEUE             : q4-30 q2-30 q2-60 q2-10
N_DEDICATE_QUEUE        : 0
DEDICATE_QUEUE          :
NODES                   : 64
MAILER                  : /bin/mail
ADMIN                   : root
OPERATOR                : root
SWITCHLOG               : daily
MACDMODE                : macwatch
ENFORCE                 : userkill
DEF_QRATE               : 1.0
UNDERUSE                : 0.0
IDLERATE                : 0.0
SYNC_INTERVAL           : 15
Do you want to change any of these now (y/n)? n
For each batch queue, please assign a charge factor (default 1.0)

q4-30 : [1.0] <Return>
q2-30 : [1.0] <Return>
q2-60 : [1.0] <Return>
q2-10 : [1.0] <Return>

These are the parameter values that have been set:

q4-30 1.0
q2-30 1.0
q2-60 1.0
q2-10 1.0
Do you want to change any of these now (y/n)? n
Commit these parameters to the configuration file (y/n)? y
MACS configuration parameters have been updated

Should MACS be started automatically at system boot (y/n)? y
MACS is configured in 'macwatch' mode.  In order to prevent users from
running when they have exceeded their allocation, the allocator must be
configured to enforce MACS limits (by adding 'USE_MACS=1' to the
allocator.config file).

Should the allocator be configured to enforce MACS limits (y/n)? y

The allocator has been configured to enforce MACS limits.
The previous /etc/nx/allocator.config file has been saved as
/etc/nx/allocator.config.95.03.20.
```

```
The allocator configuration has been changed.  The allocator, SMD, and
NQS must be restarted for the change to take effect.  THIS WILL KILL
ALL RUNNING APPLICATIONS.  (Running NQS jobs will be restarted from the
beginning.)

Would you like to restart the allocator, SMD, and NQS now (y/n)? y
Partition services provided.
SMD services provided.
NQS services provided.

Do you want to start MACS now (y/n)? y
MACS services provided.

MACS setup is complete.
```

10. Now, verify that MACS is running:

```
# ps -ef | grep -i macs
root     724874      1  0.0 13:52:39 p0    0:00.06 /usr/lib/macs/macpd
root     724875 724874  0.0 13:52:39 p0    0:00.64 /usr/lib/macs/macd
root     724879 724438  0.0 13:52:56 p0    0:00.04 grep -i macs
```

NOTE

The MACS setup script writes to the *macs.conf*, *nqstable*, and *actable* files located in the */usr/spool/macs/conf* directory. You can edit these files by running the */usr/lib/macs/setup/macs_params* script. If you edit these files by hand, be aware that you can only use a single tab or space as a delimiter.

11. Use the **macadmin** command to create MACS accounts and to add users to those accounts. You can use the **macadmin** command interactively or on the system command line. Refer to the discussions "Creating MACS Accounts" on page 3-11 and "Adding Users to MACS Accounts" on page 3-15 for further information. If you are changing your MACS operating mode from accounting-only, you can use the old */etc/nxaccount* file as a guide in recreating the MACS accounts.

12. Determine the MACS default account strategy that best serves your site's computing needs. Refer to "Understanding Default Accounts" on page 4-6 for an overview of default accounts and a description of the */etc/nx/nx_dflt_accts* file. You may want to set up account 0, or create a */etc/nx/nx_dflt_accts* file. For example:

```
acct_100:dand,mikeh,tracy,killops
```

You have configured MACS for the Paragon system.

Adding NQS Queues After Initial Setup

When you run the MACS setup script and supply values for the BATCH_QUEUE and N_BATCH_QUEUE configuration parameters, MACS creates the file */usr/spool/macs/conf/nqstable*. This file begins with a *queue count* (the total number of NQS queues) followed by the names of the queues and the *charge rate* for each queue. For example:

```
# cat /usr/spool/macs/conf/nqstable
4
q4-30 1.0
q2-30 1.0
q2-60 1.0
q2-10 1.0
```

This example shows a queue count of four, and then lists the name of each queue and the queue's charge rate (the number following the queue name, `1.0` in this example).

After the initial setup, you can later edit this file to add additional NQS queues. If you do so, be sure to change the queue count. If the queue count is too large, MACS issues an error message and does not start. If the queue count is too small, MACS reads only the first queues covered by the queue count, and ignores the rest. Also note that this file uses the # character as a comment character.

If you add NQS queues by editing this file, MACS will have to be stopped and restarted before the new queues' charge rates are recognized. Any queues that were not in the file when MACS was started are charged according to the value of the DEF_QRATE configuration parameter.

MACS Reference 5

This chapter provides manual pages for the following MACS commands and files:

acctrep	Command. Compiles and outputs a summary usage report based on data collected from the accounting log file.
actable	File. Contains MACS account and NQS queue information.
dbconvert	Command. Converts the MACS database file.
jrec	Command. Reads process log information in the accounting log file, and outputs a job records listing to *stdout* or to a specified output file.
macadmin	Command. Controls MACS accounts and user access to MACS accounts.
macalloc	Command. Controls MACS accounts and user access to MACS accounts (a subset of the **macadmin** command).
maclist	Command. Reports resource allocation information.
macd.data	File. The MACS database file.
macs.conf	File. The MACS configuration file.
macupdate	Command. Updates or resets MACS account allocation and usage.
nqstable	File. Specifies the number and names of the NQS queues.
nx_dflt_accts	File. Lists MACS default accounts.
nxaccount	File. Account management file.
si	Command. Indicates the duration of a "scheduled interrupt." The MACS accounting report utilities interpret the interval between **si -on** and **si -off** commands as scheduled downtime for the Paragon system.

acctrep acctrep

Generates a usage summary report from the MACS log files.

Synopsis

acctrep -s *start_date* **-e** *end_date*
[**-j**] [**-m**] [**-v**] [**-h** *site_name*] [**-f** *filename*]
[**-d** *filename*] [**-n** *total_nodes*] [**-p** *dir*]

Options

-s *start_date* Specify the starting date of the report.

-e *end_date* Specify the ending date of the report.

Both *start_date* and *end_date* have the following format:

mm/dd/yy[**@***hh*[**:***mm*[**:***ss*]]]

mm is the month, *dd* is the day, and *yy* is the last two digits of the year. The month and day can be specified with a single digit, when appropriate. For example, January 1 1995 can be specified as `1/1/95`.

-j Produce a record for each job in a long listing format. By default, a summary report is produced.

-m Use minutes instead of hours (the default) for the report output.

-v Send status messages about **acctrep** execution to *stderr*.

-h *site_name* Print a report header including *site_name* at the top of the report.

-f *filename* Specify a single input *macdlog* filename for the report output. With the **-f** option, if the **-s** or **-e** options do not state the time, the reported time will be based on the data in the *macdlog* file.

-d *filename* Produce a downtime report in *filename* in addition to the standard report sent to *stdout*.

-n *total_nodes* Specify the number of nodes on which the accounting report is based. Normally, this number represents the total available nodes in the *.compute* partition for the accounting period. If omitted, the default is the number of nodes specified by the NODES parameter in the *macs.conf* file.

-p *dir* Specify an alternate log file directory. This option supercedes the **-f** option.

acctrep *(cont.)*

acctrep *(cont.)*

Description

The **acctrep** command generates a usage summary report from the accounting log files, and writes the report output to *stdout*. The report consists of an initial block of system-level information, such as system uptime and downtime, a subsequent block of MACS account and user allocation information, and a final block containing the system grand totals.

The system level summary contains information on the following:

System Up	Amount of system uptime during this report period.
System Down	Amount of system downtime during this report period.
Scheduled Interrupts	Number of scheduled interrupt periods recorded with the **si** command. A scheduled interrupt period is the time lapsed between the **si -on** and **si -off** commands.
Scheduled Reboots	Number of system reboots occurring within scheduled interrupt periods.
Scheduled Down Time	Amount of system down time occurring within scheduled interrupt periods.
Unscheduled Reboots	Number of system reboots not occurring within scheduled interrupt periods.
Total number of users	The number of users shown in this report.

The system up and system down values are calculated from entries written to the daily log files in the */usr/spool/macs/log.d/macdlog.d* directory. Since the log files are updated once a minute, the down time calculation can be off by up to one minute.

acctrep *(cont.)*

acctrep *(cont.)*

In the MACS account and user summary, the parallel applications are sorted into predefined bins by the number of nodes used. The number of bins and their node sizes are specified in the */usr/spool/macs/conf/actable* file. The MACS account and user summary contains the following information:

account	The MACS account's name.
login	The login names of the users within a MACS account. For example, an account named *sw_labs* might include the users *archer* and *tracie*.
queue	Either BATCH for NQS batch requests, or INTERACT for interactive applications.
size	The number of nodes (bin size) a parallel application ran on. If an application uses a different number of nodes than those allowed by the bins, the application is placed into the smallest bin that will hold it. The bin sizes are specified in the */usr/spool/macs/conf/actable* file.
cpu hours	The total node hours used by all applications in the bin.
idle hours	Idle time of an NQS batch request. Idle time is accrued when a batch request is rolled in but sits idle. No idle time is accrued when a NQS batch request is rolled out.
under-used	Node hours that are available to an NQS batch request but that go unused while the request is running. Under-used time accrues when a batch request uses fewer nodes than the number of nodes NQS has reserved for it.
cpu charge	The total charge for a parallel application.

acctrep *(cont.)*

acctrep *(cont.)*

The cpu charge field is calculated using the following formula:

cpu charge = *rate1* * *cpu hours* + IDLERATE * *idle_node_hours* + UNDERUSE * *under-used*

For interactive applications, the value for *rate1* is determined by the DEF_QRATE charge. (DEF_QRATE, IDLERATE, and UNDERUSE are defined in the */usr/spool/macs/conf/macs.conf* configuration file.)

For NQS batch requests, the value for *rate1* is determined by the accounting charge for the queue to which the request was submitted, which is listed in the */usr/spool/macs/conf/nqstable* configuration file. The value for *rate1* defaults to the value set with the DEF_QRATE configuration parameter if there is no specification in the *nqstable* file.

The idle hours and under-used values are only calculated for NQS batch requests and are 0 for interactive jobs.

The system grand total summary contains information on the following:

size	The bin size of the parallel applications. If an application uses a different number of nodes than those allowed by the bins, the application is placed into the smallest bin that will hold it. The bin sizes are specified in the */usr/spool/macs/conf/actable* file.
cpu hours	The total node hours used by all applications in the bin.
total charge	The total charge for all jobs conforming to a particular bin size. This charge includes idle time and under-used time for NQS batch requests (idle time is time during which the partition is idle; under-used time occurs when a request uses fewer nodes than the number of nodes reserved by NQS). No idle time is accrued when a user's job is rolled out; however, if a job is rolled in but does not use any of the nodes in the partition, idle time is accrued.
%_distr	Of the total CPU hours used by applications in this report, the percentage of CPU hours used by each bin.
%_usage	Of the total CPU hours available in this report, the percentage of CPU hours used by each bin.

Exit Status

`0`	If the command completes successfully.
`1`	If the command fails; the command will also print an error message to *stderr*.

acctrep *(cont.)* acctrep *(cont.)*

Example

In the following example, the **acctrep** command is issued with only the minimum required arguments (start date and end date). The report consists of an initial block of system-level information, a subsequent block of MACS account and user allocation information, and a final block consisting of the system grand totals.

```
# acctrep -s 11/18/93 -e 11/18/93


Intel PARAGON Usage Summary Report
- produced on Thu Nov 18 14:17:47 1993


11/18/93 00:00:00 to 11/18/93 14:17:47
System Up  100.00%  ( 14 hours 9 minutes 19 seconds )
System Down 0.00%
Scheduled Interrupts:  0
Scheduled Reboots:  0
Scheduled Down Time:
Unscheduled Reboots:  0
Total number of users:  43
```

acctrep *(cont.)*

acctrep *(cont.)*

```
account login    queue     size   cpu hours   idle hours   under-used   cpu charge
------- -------- --------- ---- ------------ ------------ ------------ ------------
acct100 mikeh    INTERACT     8        0.312        0.000        0.000        0.312
                             16        0.000        0.000        0.000        0.000
                 ------------------------------------------------------------------
                 INTERACT total        0.312        0.000        0.000        0.312

                 BATCH       16        0.342        0.098        0.000        0.440
                 ------------------------------------------------------------------
                 BATCH     total       0.342        0.098        0.000        0.440

         ---------------------------------------------------------------------------
         mikeh               total       0.654        0.098        0.000        0.752

         pinglee  INTERACT     1        0.024        0.000        0.000        0.024
                              2        0.025        0.000        0.000        0.025
                              4        0.034        0.000        0.000        0.034
                             16        0.000        0.000        0.000        0.000
                 ------------------------------------------------------------------
                 INTERACT total        0.083        0.000        0.000        0.083

                 BATCH       16        0.607        0.049        0.000        0.657
                 ------------------------------------------------------------------
                 BATCH     total       0.607        0.049        0.000        0.657

         ---------------------------------------------------------------------------
         pinglee             total       0.690        0.049        0.000        0.740

-------------------------------------------------------------------------------------
acct100                      total       1.344        0.147        0.000        1.492
acct200 pinglee  INTERACT    16        1.046        0.000        0.000        1.046
                 ------------------------------------------------------------------
                 INTERACT total        1.046        0.000        0.000        1.046

                 BATCH       16        0.606        0.049        0.000        0.655
                 ------------------------------------------------------------------
                 BATCH     total       0.606        0.049        0.000        0.655

         ---------------------------------------------------------------------------
         pinglee             total       1.652        0.049        0.000        1.701

-------------------------------------------------------------------------------------
acct200                      total       1.652        0.049        0.000        1.701

11/18/93 00:00:00 to 11/18/93 14:17:47
```

acctrep *(cont.)*

acctrep *(cont.)*

	size	cpu hours	total charge	%_distr	%_usage
GRAND TOTAL	1	0.024	0.024	000.80	000.00
GRAND TOTAL	2	0.025	0.025	000.83	000.01
GRAND TOTAL	4	0.034	0.034	001.13	000.01
GRAND TOTAL	8	0.312	0.312	010.41	000.12
GRAND TOTAL	16	2.601	2.798	086.81	001.06
FINAL TOTAL		2.996	3.193	100.00	001.22

Files

/usr/spool/macs/conf/actable	Accounting table information.

See Also

jrec, si

actable actable

MACS file that contains MACS account and NQS queue information.

Synopsis

/usr/spool/macs/conf/actable

Description

When you run the MACS setup script, MACS creates the file */usr/spool/macs/conf/actable* which contains MACS account and NQS queue informations. After the initial setup, you can later edit this file to change account information. If you do so, note that you can only use a single tab or space as a delimiter. The following example shows a sample *actable* file:

```
# File created Tue May 24 09:09:00 PDT 1994
#
N_BINS 8
BINS 1 2 4 8 16 32 50 66
ACT_GRP_LEVEL 3
ACT_NAME_SIM 3 3 1
ACT_GRP_LABEL site account project
N_BATCH_QUEUE 6
BATCH_QUEUE q16m q20m q21m q22m q23m q24m
N_DEDICATE_QUEUE 0
DEDICATE_QUEUE
```

actable *(cont.)*

actable *(cont.)*

The entries in the *actable* file do the following:

N_BINS	The number of node bins specified with `BINS`. There is no default. You must supply a value.
BINS	Defines the node size (*bin size*) breakdowns that appear in the summary report. If an application uses a different number of nodes than one of the sizes specified, the application is placed into the smallest bin that will hold it. For example if `1 256 512` were specified, the report would indicate how many 1-node jobs ran, how many 2 to 256-node jobs ran, and how many 257 to 512-node jobs ran. The node bins should be specified in ascending sequence, and the last node-size should be equal to the total nodes available in the *.compute* partition. There is no default. You must supply a value.
ACT_GRP_LEVEL	This parameter is unused. Leave default of `3`.
ACT_NAME_SIM	This parameter is unused. Leave default of `3 3 1`.
ACT_GRP_LABEL	Defines an optional label used in accounting reports. Leave the default of `site account project`.
N_BATCH_QUEUE	Specifies how many NQS batch queues exist. The default is `0`. You must supply a value if your site uses NQS.
BATCH_QUEUE	Lists all of the NQS batch queues. There is no default. You must list queues if your site uses NQS.
N_DEDICATE_QUEUE	This parameter is unused. Leave default of `0`.
DEDICATE_QUEUE	This parameter is unused. There is no default.

See Also

nqstable, nxaccount, macd.data, macs.conf, nx_dflt_accts

dbconvert dbconvert

Converts the MACS database file to the current format. This command is only used by the Paragon system administrator when installing the Paragon system operating system Version R1.3 on systems running earlier versions of the operating system.

Synopsis

dbconvert *oldfile newfile*

Arguments

oldfile	Specifies the pathname of the existing database file to be converted.
newfile	Specifies the pathname of the new MACS database file.

Description

The **dbconvert** command reads a MACS database file *(/usr/spool/macs/private/macd.data)* and creates a new file in the format required by the current version of MACS.

NOTE

Do not name files *macd.data.new*, *macd.data.old*, or *macd.data.bak* as these names are used by MACS itself.

Exit Status

0	If the command completes successfully.
1	If the command fails.

In either case, the command will print status and/or error messages to *stderr*.

Example

```
# cd /usr/spool/macs/private
# cp macd.data macd.data.r1.2
# dbconvert macd.data.r1.2 macd.data.r1.3
# cp macd.data.r1.3 macd.data
```

jrec jrec

Processes the accounting log file entries and writes job records to *stdout*.

Synopsis

jrec -s *start_date* **-e** *end_date*
[**-j**] [**-v**] [**-h** *site_name*] [**-n** *total_nodes*]
[**-f** *file*] [**-d** *file*] [**-p** *dir*]

Options

-j Produce a record for each job in a long listing format. By default, a summary report is produced.

-v Produce status messages about **jrec** execution on the standard error output.

-h *site_name* Specify the *site_name* used in the report output header. If omitted, the output will not include the system name or the column headings.

-n *total_nodes* Specify the number of nodes on which the accounting report is based. Normally, this number represents the total available nodes in the *.compute* partition for the accounting period. If omitted, the default is the number of nodes on the current system, which is the largest bin specified in the *actable* file.

-s *start_date* Specify the starting date of the report.

-e *end_date* Specify the ending date of the report.

Both *start_date* and *end_date* have the following format:

mm/dd/yy[*@hh*[*:mm*[*:ss*]]]

mm is the month, *dd* is the day, and *yy* is the last two digits of the year. The month and day can be specified with a single digit, when appropriate. For example, January 1 1995 can be specified as `1/1/95`.

jrec *(cont.)*

jrec *(cont.)*

-f *file* Specify a single input *macdlog* file for the report output. With the **-f** option, if the **-s** or **-e** options do not state the time, the reported time will be based on the data in the *macdlog* file.

-d *file* Produce a downtime report in *file* in addition to the standard report sent to *stdout*.

-p *dir* Specify an alternate log file directory. This option supercedes the **-f** option.

Description

The **jrec** command processes the log files located in the */usr/spool/macs/log.d/macdlog.d* directory and writes job records to *stdout*.

The **jrec** command maintains an internal job table. As it reads the records from the log files, it creates a new job entry and records the job start time for every job initiation and writes a job record out for every job completion. If the job completion does not match any entry in the job table, the start time of the report or the last system uptime (whichever comes last) is used as the job start time.

When a system-down record is encountered, a job record is written out for every job entry remaining in the job table. The system-down time is used as the job end time, and all job entries are deleted from the job table.

In normal operation, the **jrec** command is invoked by the **acctrep** command. When invoked by itself, the **jrec** command outputs a text file that can be sorted and formatted to produce a report customized to match individual site requirements. If the command is invoked using the **-h** argument, the **jrec** output includes a user-specified *site_name* and the output also includes column headings that indicate column content.

If the **-j** argument is used, **jrec** outputs the job records in a long format. Otherwise, **jrec** produces the report in a format that is easier for **acctrep** to process.

jrec *(cont.)*

jrec *(cont.)*

In the **jrec** report, the parallel applications are sorted chronologically. The report summary contains the following information:

account	The MACS account's name.
login	The user who submitted the job.
type	The job type; either `0` for interactive jobs, `1` for NQS batch requests, or `2` for dedicated NQS batch requests.
size	The number of nodes (bin size) a parallel application ran on. If an application uses a different number of nodes than those allowed by the bins, the application is placed into the smallest bin that will hold it. The bin sizes are specified in the */usr/spool/macs/conf/actable* file.
cpu-hours	The total node hours used by the job.
idle cpu-hrs	Idle time of an NQS batch request. Idle time is accrued if a batch request is rolled in but sits idle. No idle time is accrued when a NQS batch request is rolled-out.
under-used	Node hours that are available to an NQS batch request but that go unused while the request is running. Under-used time accrues when a batch request uses fewer nodes than the number of nodes NQS has reserved for it.
total charge	The total charge for the job.

jrec *(cont.)*

jrec *(cont.)*

jrec invoked with the **-j** option includes the following information in the report:

account	The MACS account's name.
login	The user who submitted the job.
partition#	A unique partition identifier that is used by the allocator when communicating with MACS. The same number appears in the MACS logfile entries for jobs running in that partition.
size	The number of nodes a parallel application ran on.typeThe job type; either `0` for interactive jobs, `1` for NQS batch requests, or `2` for dedicated NQS batch requests.
cpu-hours	The total node hours used by the job.
rate	First occurrence: the charge rate for the cpu-hours used.
idle cpu-hrs	Idle time of an NQS batch request. Idle time is accrued if a batch request is rolled in but sits idle. No idle time is accrued when a NQS batch request is rolled-out.
rate	Second occurrence: the rate charged for idle NQS requests.
under-used	Node hours that are available to an NQS batch request but that go unused while the request is running. Under-used time accrues when a batch request uses fewer nodes than the number of nodes NQS has reserved for it.
rate	Third occurrence: the rate charged for under used node requests.
total charge	The total charge for the job.
job submitted	The date and time a job was submitted to NQS. Shown as N/A for interactive jobs.
job started	The date and time an NQS job started. Shown as N/A for interactive jobs.
job ended	The date and time an NQS job ended. Shown as N/A for interactive jobs.
st	Job termination status. The value of this field is `1` if the job ran to completion, or `0` if it was terminated by a system crash or hang. In the event of a crash or hang, the application is assumed to have run until the crash or hang.
reqst cpuhrs	Either the time limit requested by the user with the **qsub -lT** option, or the per-request CPU limit for the queue. This is listed as N/A for interactive jobs.
queue name	The name of the queue an NQS request was submitted to. This will be listed as UNKNOWN for interactive jobs.

jrec *(cont.)* jrec *(cont.)*

Exit Status

0	If the command completes successfully.
1	If the command fails.

Files

/usr/spool/macs/conf/actable	Accounting table information.
*/usr/spool/macs/log.d/macdlog.d/macd**	The MACS log files.

Example

```
# jrec -s 6/21/94 -e 6/21/94 -h SSD
SSD Intel PARAGON Job Usage Report
- produced on Tue Jun 21 09:28:22 1994


06/21/94 00:00:00 to 06/21/94 09:28:22

account  login   type size   cpu-hours  idle cpu-hrs under-used   total charge
------- -------- ---- ---- ------------ ------------ ------------ -------------
swengr     lynng 1    6        0.101        0.005        0.000         0.101
swengr     stanh 1    6        0.101        0.010        0.000         0.101
quarks    sydney 0    2        0.034        0.000        0.000         0.034
proj94       mmh 0    2        0.033        0.000        0.000         0.033
quarks    sydney 1    6        0.034        0.207        0.000         0.034
quarks     tonyc 1    6        0.102        0.005        0.000         0.102
(0772182000) 06/21/94 00:00:00 to (0772216102) 06/21/94 09:28:22 - down 0
seconds, scheduled down 0 seconds, scheduled 0, reboot 0; unscheduled reboot 0
```

See Also

acctrep

macadmin macadmin

Controls MACS accounts and user access to MACS accounts. This command is only used by the Paragon system administrator.

Synopsis

macadmin
macadmin -m la [**-a** *acct-name-or-ID*]
macadmin -m aa -A *new-acct-name, ID, description* [*acct-flags*]
macadmin -m da [**-a** *acct-name-or-ID*]
macadmin -m ca [**-a** *acct-name-or-ID*] [*acct-flags*]
macadmin -m au [**-a** *acct-name-or-ID*] **-u** *username-or-ID* [*user-flags*]
macadmin -m du [**-a** *acct-name-or-ID*] [**-u** *username-or-ID*]
macadmin -m cu [**-a** *acct-name-or-ID*] [**-u** *username-or-ID*] [*user-flags*]
macadmin -m lu [**-a** *acct-name-or-ID*] [**-u** *username-or-ID*]
macadmin -m ta -F *account* **-T** *account* { **-t** *minutes* | **-%** *percent* }

Description

The **macadmin** command allows system administrators to add or delete MACS account and user data in the MACS database, to change or set MACS account attributes, and to modify or transfer MACS account allocations. When invoked without command line options, the **macadmin** command enters into a menu-driven interactive user interface.

Options

-m *function* Specifies a function. The *functions* are:

la	List MACS accounts.
aa	Add a new MACS account.
da	Delete MACS accounts.
ca	Change MACS account attributes.
ta	Transfer allocation between two MACS accounts.
lu	List user's allocations.
au	Add user to MACS accounts.

macadmin *(cont.)* macadmin *(cont.)*

du Delete user from MACS accounts.

cu Change user's MACS account attributes.

-a *acct-name-or-ID*
MACS account name or account ID, or multiple MACS account names or IDs delimited by commas. If this option is not used, the MACS account specification defaults to all MACS accounts.

-A *new_acct_name, ID, description*
Specifies a new MACS account name, the associated MACS account ID, and description. The MACS account name is an alphanumeric string with an alphabetic leading character. The MACS account ID is an integer, and the MACS account description is a printable character string delimited by single quotes. These fields are delimited by commas. No space is allowed except in the description field. This option is only used in *function* **aa** (add a new MACS account).

-u *user-name-or-ID*
User name or user ID, or multiple user names or IDs delimited by commas. If this option is not used, the user specification defaults to all users. This option generates an error message if used with *function*s that end in an "**a**" character (e.g., **la**, **aa**, etc.).

-F *account* Specifies a "from" MACS account name or account ID. This is the MACS account from which the allocation (specified by **-%** or **-t**) is transferred from to another MACS account. This option is only used in *function* **ta** (transfer allocation between two MACS accounts).

-T *account* Specifies a "to" MACS account name or account ID. This is the MACS account to which the allocation (specified by **-%** or **-t**) is transferred. This option is only used in *function* **ta** (transfer allocation between two MACS accounts).

acct-flags [**-t** *minutes* | **-r**] [**-w** *weight-flag*] [**-N** *no-kill-flag*] [**-l** *lock-flag*] [**-n** *max#-of-nodes*]

user-flags [**-p** [**M**] [**T**] [**U**]] [**-t** *minutes* | **-r** | **-%** *percent*] [**-n** *max-nodes*]

The *acct-flags* and *user-flags* options are described below.

-t *minutes* Specifies an allocation in node-minutes. The argument is an integer with or without prefixes + or -. With a + prefix, the specified time will be added to the existing allocations. With a - prefix, the specified time allocation will be deducted from the existing allocations. Without any prefix, the allocation is set to the specified value.

macadmin *(cont.)*

macadmin *(cont.)*

This option is only used in *functions* **aa** (add new MACS account), **ca** (change MACS account attributes), **au** (add user to MACS accounts), **cu** (change user's MACS account attributes), or **ta** (transfer allocation). If this option is not specified in *functions* **aa** or **au**, the allocation defaults to `0`. If this option is not specified in *functions* **ca**, **ta**, or **cu**, the allocation remains unchanged.

-r Specifies unlimited allocation. This option is only used in *functions* **aa**, **ca**, **au**, and **cu**.

-w *weight-flag* The weight flag is used when the MACS system does periodic allocation updates. During each update, a new allocation is added to each MACS account. The amount of unused allocation from the previous period that can be carried over to the current period is determined by the weight-flag settings, and the retained percent value defined in the **macupdate** input file for each MACS account.

The weight flag can be either `0` or `1`. If the weight flag is set to`1`, the unused time from the previous period will be multiplied by the retained percent value. If the weight flag is set to `0`, the remaining time from the previous allocation period is carried over totally to the next period.

This option is only used in *function* **aa** (add a new MACS account) or **ca** (change MACS account attributes).

If this option is not specified in *function* **aa** (add new MACS account), the weight flag defaults to `1`. If this option is not specified in *function* **ca**, the weight flag remains unchanged.

-N *no-kill-flag* The no-kill flag value for the specified MACS accounts. The no-kill flag can be either `0` or `1`. The no-kill flag is used to determine whether a job should be killed when its allocation has run out. The no-kill flag can only be changed when the lock-flag (set by the system administrator) is off (set to `0`).

If the no-kill flag is on (set to `1`), MACS will not kill jobs if the MACS account has exceeded its CPU allocation.

If the no-kill flag is off (set to `0`), the keywords of the ENFORCE configuration parameter determine under what circumstances MACS will kill jobs.

This option is only used in either *function* **aa** (add a new MACS account) or **ca** (change MACS account attributes).

If this option is not specified in *function* **aa**, the no-kill flag defaults to `0`. If this option is not specified in *function* **ca**, the no-kill flag remains unchanged.

macadmin *(cont.)* macadmin *(cont.)*

-l *lock-flag* The lock flag value for the specified MACS accounts. The lock flag can be either `0` or `1`. When the lock flag of an account is `1`, the *no-kill-flag* cannot be changed. This option is only used in either *function* **aa** (add a new MACS account) or **ca** (change MACS account attributes).

If this option is not specified in *function* **aa**, the lock flag defaults to `0`. If this option is not specified in *function* **ca**, the lock flag is unchanged.

-n *max-nodes* Specifies the maximum number of nodes permissible for a user's job. The argument must be an integer. This option is only used in *functions* **aa** (add a new MACS account), **ca** (change MACS account attributes), **au** (add user to MACS accounts), or **cu** (change user's MACS account attributes).

If this option is not specified in *functions* **aa** or **au**, the maximum number of nodes default to the maximum number of nodes in the system. If this option is not specified in *functions* **ca** or **cu**, the maximum number of nodes remain unchanged.

-p *user-access-permissions*
Specifies the user access permissions for the specified MACS account. Valid arguments are any combination of the following character codes, with or without the prefixes + or -:

M	permission to modify MACS account attributes
T	permission to transfer MACS account allocation
U	permission to use the MACS account

With a + prefix, the specified permissions are granted. With a - prefix, the specified permissions are removed. Without any prefix, the user access permission is set to the specified value. This option is only used in either *function* **au** (add user to MACS accounts) or **cu** (change user's MACS account attributes).

If this option is not specified in *function* **au**, the user permission defaults to **U**. If this option is not specified in *function* **cu**, the user access permission remains unchanged.

-% *percentage* Specifies the user allocation as a percent of the MACS account allocation. The argument must be a number between `0` and `100`.

This option is only used in *functions* **au** (add user to MACS accounts), **cu** (change user's MACS account attributes), and **ta** (transfer allocation).

If this option is not specified in *functions* **au** or **ta**, the user percent allocation defaults to `0`. If this option is not specified in *function* **cu**, the user percent allocation will remain unchanged.

macadmin *(cont.)* macadmin *(cont.)*

Exit Status

0	If the command completes successfully.
-1	If the command fails; the command will also print an error message to *stderr*.
>0	Could be a partial success; exit code is the number of errors encountered and reported to *stderr*.

Files

/usr/sbin/macadmin	Location of the **macadmin** program.
/usr/spool/macs/log.d/macadmin.log	The **macadmin** usage log.
/etc/nxaccount	MACS account names and IDs.
/etc/nxaccomm	MACS account descriptions.
/usr/spool/macs/conf/macs.conf	Configuration for MACS.
/usr/spool/macs/private/macd.data	The MACS database file.

Examples

To enter the interactive **macadmin** menu interface:

```
macadmin
```

To list all account information:

```
macadmin -mla
```

To add a new account named *nan000* with a time allocation of 10000 node-minutes:

```
macadmin -maa -Anan000,100,'new account' -t100000
```

To delete an account named *old001*, and an account with an account ID of *200*:

```
macadmin -mda -aold001,200
```

To change the no-kill flag account attribute for accounts with IDs *100*, *200*, and *300*:

```
macadmin -mca -a100,200,300 -N0
```

macadmin *(cont.)* macadmin *(cont.)*

To transfer an allocation from account *ops150* to account *app350*:

```
macadmin -mta -Fops150 -Tapp350 -%20
```

To list all user accounts in account *app350*:

```
macadmin -mlu -aapp350
```

To add users to the accounts *ops150* and *app350*:

```
macadmin -mau -aops150,app350 -usmithd,1187 -t180000
```

To delete user *dand* from accounts *100* and *300*:

```
macadmin -mdu -a100,300 -dand
```

To give user *1187* account manager permissions:

```
macadmin -mcu -aops150-u1187 -pMTU
```

See Also

maclist, macalloc, macupdate

macalloc macalloc

Controls MACS accounts and user access to MACS accounts. This command is only used by MACS account managers.

Synopsis

macalloc
macalloc -m la [**-a** *acct-name-or-ID*]
macalloc -m lu [**-a** *acct-name-or-ID*] [**-u** *username-or-ID*]
macalloc -m cu [**-a** *acct-name-or-ID*] [**-u** *username-or-ID*] [*user-flags*]
macalloc -m ta -F *account* **-T** *account* { **-t** *minutes* | **-%** *percent* }
macalloc -m ca -a *acct-name-or-ID* **-N** *no-kill-flag*

Description

The **macalloc** command allows users with MACS account management privileges to control access to the MACS accounts. The **macalloc** command's functionality is a subset of the **macadmin** command's functionality. The **macalloc** command enters into a menu-driven interactive user interface when invoked without command line options.

Options

-m *function* Specifies a function mode. The *functions* are:

la List MACS accounts.

lu List user's allocations.

cu Change user's MACS account attributes.

ta Transfer allocation between two MACS accounts.

ca -N *no-kill-flag*
Sets the no-kill flag value for the specified MACS accounts.

-a *acct-name-or-ID*
MACS account names or account IDs, or multiple account names or IDs delimited by commas. If this option is not used, the MACS account specification defaults to all MACS accounts.

macalloc *(cont.)* macalloc *(cont.)*

-u *user-name-or-ID*

User names or user IDs, or multiple user names or IDs delimited by commas. This option is only used in *functions* **au** (add user to MACS accounts) and **cu** (change user's MACS account attributes). If this option is not used, the user specification defaults to all users.

user-flags [**-p** [**M**] [**T**] [**U**]] [**-t** *minutes* | **-r** | **-%** *percent*] [**-n** *max#-of-nodes*]

-p *user-access-permissions*

Specifies the user access permissions for the specified MACS account. Valid arguments are any combination of the following character codes, with or without the prefixes + or -:

M permission to modify MACS account attributes

T permission to transfer MACS account allocation

U permission to use the MACS account

With a + prefix, the specified permissions are granted. With a - prefix, the specified permissions are removed. Without any prefix, the user access permission is set to the specified value. This option is only used in either *function* **au** (add user to MACS accounts) or **cu** (change user account attributes).

If this option is not specified in *function* **au**, the user permission defaults to **U**. If this option is not specified in *function* **cu**, the user access permission remains unchanged.

-t *minutes* Specifies the allocation in node-minutes. The argument must be an integer with or without prefixes + or -. With a + prefix, the specified time will be added to the existing allocations. With a - prefix, the specified time allocation will be deducted from the existing allocations. Without any prefix, the allocation is set to the specified value.

This option is only used in *functions* **cu** (change user's MACS account attributes) and **ta** (transfer allocation between two MACS accounts). If this option is not specified, the allocation remains unchanged.

-r Specifies unlimited allocation. This option is only used in *function* **cu**.

-% *percentage* Specifies the user allocation as a percent of the MACS account allocation. The argument must be a number between `0` and `100`.

This option is only used in *functions* **cu** (change user's MACS account attributes) and **ta** (transfer allocation between two MACS accounts). If both the **-%** and the **-t** options are specified, the **-t** option takes precedence.

macalloc *(cont.)*

macalloc *(cont.)*

If this option is not specified in *function* **ta**, the user percent allocation defaults to `0`. If this option is not specified in *function* **cu**, the user percent allocation will remain unchanged if this option is not specified.

-F *account*
Specifies a "from" MACS account name or account ID. This is the MACS account from which the allocation (specified by **-%** or **-t**) is transferred from to another MACS account. This option is only used in *function* **ta** (transfer allocation between two MACS accounts).

-T *account*
Specifies a "to" MACS account name or account ID. This is the MACS account to which the allocation (specified by **-%** or **-t**) is transferred. This option is only used in *function* **ta** (transfer allocation between two MACS accounts).

-n *max-nodes*
Specifies the maximum number of nodes permissible for a user job. The argument must be an integer. This option is only used in function mode **cu** (change user's MACS account attributes). If this option is not used, the maximum number of nodes remains unchanged.

-N *no-kill-flag*
Sets the no-kill flag value for the specified MACS accounts. The no-kill flag can be either `0` or `1`. The no-kill flag is used to determine whether a job should be killed when its allocation has run out. The no-kill flag can only be changed when the lock-flag (set by the system administrator) is off (set to `0`).

If the no-kill flag is on (set to `1`), MACS will not kill jobs if the MACS account has exceeded its CPU allocation.

If the no-kill flag is off (set to `0`), the keywords of the ENFORCE configuration parameter determine under what circumstances MACS will kill jobs.

In either case, once an account has exceeded its allocation, the ENFORCE parameter determines whether or not a user can start a new application.

Exit Status

`0`
If the command completes successfully.

`-1`
If the command fails; the command will also print an error message to *stderr*.

>0
Could be a partial success; exit code is the number of errors encountered and reported to *stderr*.

macalloc *(cont.)* macalloc *(cont.)*

Files

/usr/bin/macalloc	Location of the **macalloc** program.
/usr/sbin/macadmin	Location of the **macadmin** program.
/usr/spool/macs/log.d/macadmin.log	The **macadmin** usage log.
/etc/nxaccount	MACS account names and IDs.
/etc/nxaccomm	MACS account descriptions.
/usr/spool/macs/conf/macs.conf	Configuration for MACS.
/usr/spool/macs/private/macd.data	The MACS database file.

Examples

To enter the interactive **macalloc** menu interface:

```
macalloc
```

To list all MACS account information:

```
macalloc -mla
```

To transfer an allocation from MACS account *ops150* to MACS account *app350*:

```
macalloc -mta -Fops150 -Tapp350 -%20
```

To list all user's in MACS account *app350*:

```
macalloc -mlu -aapp350
```

See Also

maclist, macadmin, macupdate

macd.data macd.data

The MACS database file.

Synopsis

/usr/spool/macs/private/macd.data

Description

The *macd.data* file is the MACS database file, which is often referred to simply as the "MACS database." The initial *macd.data* file is created during the initial MACS setup. Permissions on the directory and file exclude all users except *root*. The MACS daemon, **macd**, uses the database only in *resource control* mode—that is, when the MACMODE configuration parameter is set to *macwatch*.

The *macd.data* file contains binary information on MACS accounts and MACS users. The **macd** daemon maintains this same information in memory dynamically.

While only the **macd** daemon directly accesses *macd.data*, the **macadmin**, **macalloc**, **maclist**, and **macupdate** commands utilize the database via calls to the **macd** daemon.

The **macd** daemon reads the *macd.data* file once when the daemon starts. Afterwards, the **macd** daemon updates the *macd.data* file at the end of a defined time interval called the *sync interval*. The system administrator controls the sync interval by changing the SYNC_INTERVAL parameter in the */usr/spool/macs/conf/macs.conf* file.

When the system administrator stops the **macd** daemon in a controlled manner (i.e. the **si -shutdown** command, the **/sbin/init.d/macs stop** command, or during a normal Paragon system shutdown procedure) the database is written one last time with current information.

There can be as many as four database files in the */usr/spool/macs/private* directory at one time:

- *macd.data* (the current MACS database).
- *macd.data.new* (a temporary version of the MACS database as explained below).
- *macd.data.old* (a previous MACS database; no more than 15 minutes old).
- *macd.data.bak* (a backup of the MACS database made by the last **macupdate** command.

Whenever the **macd** daemon writes the MACS database to disk, it first writes it to *macd.data.new*. The daemon then checks the current *macd.data* file to be sure it is valid. If the *macd.data* file is valid, the daemon renames *macd.data* to *macd.data.old* and then renames *macd.data.new* to *macd.data*. If the current *macd.data* file is not valid, the **macd** daemon copies the *macd.data.new* file to *macd.data.old* and then renames *macd.data.new* to *macd.data*.

macd.data *(cont.)*

macd.data *(cont.)*

This database file copying sequence ensures that even if the system crashes while writing the database, there is always at least one valid copy of the MACS database on disk. The *macd.data* file begins with a checksum and a database version number. The checksum insures that the database has not been corrupted.

When the **macd** daemon starts up, it checks to be sure that the *macd.data* file is valid. If the *macd.data* file is not valid, the daemon checks *macd.data.new*, *macd.data.old*, and finally *macd.data.bak*. The first valid database found is copied to *macd.data* and opened. If no valid database is found, the **macd** daemon prints an error message and aborts.

NOTE

The system administrator must back up the *macd.data* file frequently and securely. One good way to do this would be to use a **cron** job to copy the database file to another system once an hour.

See Also

dbconvert, macadmin, macalloc, maclist, macupdate

maclist maclist

Lists resource allocation information from the resource database.

Synopsis

maclist
maclist [**-A**]
maclist [**-U**]
maclist [**-a** *accts*] [**-u** *users*]

Options

-A Lists account information about all accounts to which the user belongs. When invoked by *root*, **maclist -A** lists account information for all accounts on the system. The **-A** option cannot be used with the **-U**, **-a**, or **-u** options.

-U Lists user and account information about all users in all accounts to which the user belongs. When invoked by *root*, **maclist -U** lists account and user information for all accounts on the system. The **-U** option cannot be used with the **-A**, **-a**, or **-u** options.

-a *accts* Lists user and account information about all users in all accounts specified by *accts*. *accts* is a list of one or more MACS account names or IDs, separated by commas. If the user invoking **maclist** does not belong to one or more of the specified accounts, that account is skipped and an error message is displayed. The **-a** option cannot be used with the **-A** or **-U** options.

-u *users* Lists user and account information about the users specified by *users* in all accounts to which the user invoking **maclist** and the specified user both belong. *users* is a list of one or more user names or IDs, separated by commas. The **-u** option cannot be used with the **-A** or **-U** options.

If **-a** and **-u** are used together, only the specified users are listed in the specified accounts.

If no options are used, **maclist** lists the user and default account information for the invoking user.

maclist *(cont.)* maclist *(cont.)*

Description

The **maclist** command reports resource allocation information about individual MACS account users, account access, and other allocation information contained in the resource database. The report, sent to *stdout*, consists of a header, followed by an individual line of output for each database entry selected. This command allows any Paragon system user to view the allocation information that the MACS system maintains in its resource allocation database.

Exit Status

If there is an error, the **maclist** command returns one of the following nonzero exit codes and displays a message indicating the cause of the error:

Code	Meaning
1	Usage error. You have entered something incorrectly.
2	Internal error. There is a problem with the program. Contact SSD Customer Support.
3	MACS account entry not found for the prior selections. You have entered a non-existent MACS account name.
4	User access entry not found for the prior selections. You have entered a non-existent user name or user ID (UID).

Files

/usr/bin/maclist	Location of the **maclist** program.
/usr/spool/macs/private/macd.data	The MACS database

maclist *(cont.)* maclist *(cont.)*

Examples

In the following example, *root* uses **maclist -A** to get a list of all MACS accounts on the system:

```
# maclist -A

     name   uid  agid  access percent       allocation            used maxnode
    acct1           1  W                      0:00:00         0:00:00 sys.lim
    acct2           2  W                      0:00:00         0:00:00 sys.lim
    acct3           3  W                      0:00:00         0:00:00 sys.lim
    acct4           4  W                      0:00:00         0:00:00 sys.lim
```

In the following example, *root* uses **maclist -U** to get a list of all MACS accounts and the users of the accounts:

```
# maclist -U

     name   uid  agid  access percent       allocation            used maxnode
 macsusr2 20002     1     UI     0.00         0:00:00         0:00:00 sys.lim
 macsusr1 20001     1     UI     0.00         0:00:00         0:00:00 sys.lim
 macsusr0 20000     1     UI     0.00         0:00:00         0:00:00 sys.lim
    acct1           1  W                      0:00:00         0:00:00 sys.lim

     name   uid  agid  access percent       allocation            used maxnode
 macsusr2 20002     2     UI     0.00         0:00:00         0:00:00 sys.lim
 macsusr0 20000     2     UI     0.00         0:00:00         0:00:00 sys.lim
    acct2           2  W                      0:00:00         0:00:00 sys.lim

     name   uid  agid  access percent       allocation            used maxnode
 macsusr1 20001     3     UI     0.00         0:00:00         0:00:00 sys.lim
 macsusr0 20000     3     UI     0.00         0:00:00         0:00:00 sys.lim
    acct3           3  W                      0:00:00         0:00:00 sys.lim

     name   uid  agid  access percent       allocation            used maxnode
 macsusr2 20002     4     UI     0.00         0:00:00         0:00:00 sys.lim
 macsusr1 20001     4     UI     0.00         0:00:00         0:00:00 sys.lim
    acct4           4  W                      0:00:00         0:00:00 sys.lim
```

maclist *(cont.)* maclist *(cont.)*

In the following example, *root* uses **maclist -a** to get a list of all the users of MACS account *acct1*:

```
# maclist -a acct1

     name   uid  agid  access percent       allocation            used maxnode
 macsusr2 20002     1      UI    0.00          0:00:00         0:00:00 sys.lim
 macsusr1 20001     1      UI    0.00          0:00:00         0:00:00 sys.lim
 macsusr0 20000     1      UI    0.00          0:00:00         0:00:00 sys.lim
    acct1           1  W                     0:00:00         0:00:00 sys.lim
```

In the following example, user *macsuser1* uses **maclist -A** to get a list of all MACS accounts to which she belongs:

```
% maclist -A

     name   uid  agid  access percent       allocation            used maxnode
    acct1           1  W                     0:00:00         0:00:00 sys.lim
    acct3           3  W                     0:00:00         0:00:00 sys.lim
    acct4           4  W                     0:00:00         0:00:00 sys.lim
```

See Also

macalloc, **macadmin**

macs.conf macs.conf

The MACS configuration file.

Synopsis

/usr/spool/macs/conf/macs.conf

Description

When you run the MACS setup script, MACS creates the */usr/spool/macs/conf/macs.conf* configuration file, which contains several configuration parameters that control MACS setup and operation. If a parameter appears more than once in the file, the last occurrence takes precedence.

After the initial setup, you can later edit this file to change values for the configuration parameters. Since these parameters affect the MACS daemon, MACS must be restarted for new values to take effect. If you edit this file, note that this file does not allow any comment characters, and that you can only use a single tab or space as a delimiter. The *macs.conf* configuration file contains the following configuration parameters:

NODES	The total number of nodes available in the *.compute* partition. There is no default. You must supply a value.
MAILER	The mailer used to send mail to users and to `ADMIN` in the case of MACS problems. (Typical mailers include */usr/bin/mailx* and */bin/mail*.) The default is */bin/mail*.
ADMIN	Specifies a MACS administrator. If there is some kind of error (such as an overdrawn account), MACS mails a message to the user and to the MACS administrator stating the cause of the problem. The default is `root`.
OPERATOR	Specifies an operator. If there is a MACS system problem, the operator is notified by a broadcast message to the operator's console. Your site may not have an operator. The default is `root`.
SWITCHLOG	Because the MACS log files can get rather long, they are switched periodically. `SWITCHLOG` specifies how often the log files are changed, using one of three values: `daily`, `weekly`, or `monthly`. The default is `daily`.

macs.conf *(cont.)*

macs.conf *(cont.)*

MACDMODE

Allows the administrator to turn off CPU quota monitoring. Allowed values are `acctonly` (provides only accounting reports, referred to as *accounting-only* mode) and `macwatch` (provides both accounting reports and CPU quotas, referred to as *resource control* mode). The default is `acctonly`.

ENFORCE

Specifies the desired level of job control when CPU quotas are exceeded, depending on the setting of the *no-kill-flag* (set with **macadmin -N**).

No jobs are killed by default.

The ENFORCE parameter accepts two arguments, which can be used in the following manner.

When an account's *no-kill-flag* is off (set to `0`):

`ENFORCE`. With no arguments, no jobs are killed when allocations are exceeded (no control.)

`ENFORCE userkill`. Will kill a user's job when all of the user's allocated time has been used up.

`ENFORCE acctkill`. Will kill running jobs when all of the group's allocated time has been used up.

`ENFORCE userkill acctkill`. Will kill jobs if either the user allocation or account allocation is used up.

macs.conf *(cont.)* macs.conf *(cont.)*

When an account's *no-kill-flag* is on (set to `1`):

`ENFORCE`. With no arguments, no jobs are killed when allocations are exceeded (no control.)

`ENFORCE userkill`. Will not kill a user's job when all of the user's allocated time has been used up.

`ENFORCE acctkill`. Will not kill running jobs when all of the group's allocated time has been used up.

`ENFORCE userkill acctkill`. Will not kill jobs if either the user allocation or account allocation is used up.

DEF_QRATE

Specifies the charge rate for interactive jobs (non-NQS jobs) or for jobs run in queues that are not defined in the *nqstable* file. The default is `1.0`.

The charge rate is the number of arbitrary units for which each node-hour is charged. Typically each unit is equal to one node-hour, but you might set a higher charge rate for a high-priority queue or a lower charge rate for a queue with lower-memory nodes.

For example, if you use 5 node-hours in a queue with a charge rate of 2.5, you are charged 10 units. These units are the units shown by the **maclist**, **jrec**, and **acctrep** commands.

UNDERUSE

Specifies the charge rate for node hours that are available to an NQS batch request, but that go unused while the request is running. Under-used time occurs when a batch request uses fewer nodes than the number of nodes NQS has reserved for it. The default is `0.0`.

IDLERATE

Specifies the charge rate for idle node time accrued by NQS jobs. Idle time is defined as the time when there is no application running in an NQS partition that has been assigned to a job. The default is `0.0`.

SYNC_INTERVAL

Specifies the desired time interval (in minutes) that is used for MACS synchronization. At this interval, MACD gets status information from SMD and synchronizes the disk database with the internal tables. Default is `15` minutes.

macs.conf *(cont.)*

macs.conf *(cont.)*

See Also

actable, macd.data, nqstable, nxaccount, nx_dflt_accts

macupdate macupdate

Updates the MACS account allocation and usage in the MACS database. This command is only used by the Paragon system administrator.

Synopsis

macupdate *input_filename*

Options

input_filename Specifies the path name of the input file.

Description

The **macupdate** command updates and resets the MACS account allocations and usages in the MACS database. It requires update information as input from a file, one MACS account per line. Each line must be in the following format, with each field separated by tabs or spaces:

acct_name alloc_weight add_alloc usage_remain

The items in the input file are as follows:

acct_name	The MACS account name.
alloc_weight	A percentage of unused node minutes carried forward, specified as an integer between `0` and `100`. For *alloc_weight* to have an effect, the MACS account's Weight Flag (set with **macadmin**) must be set to `1`; if set to `0`, all unused node minutes are automatically carried forward.
add_alloc	An additional allocation of node-minutes.
usage_remain	A switch to carry forward all previously used node minutes; `1` carries all used node minutes forward, `0` does not.

The new allocation is computed by adding together the node-minutes calculated using the *alloc_weight*, *add_alloc*, and *usage_remain* values.

macupdate *(cont.)* macupdate *(cont.)*

For example, to add 6000 node-minutes to account *acct000*, carrying forward all unused node-minutes and not carrying forward any used node-minutes:

```
acct000 100 6000 0
```

If *acct000* had 500 node-minutes remaining in its allocation, the new allocation would be 6500 node-minutes and there would be no node-minutes used.

To set account *acct100*'s allocation to exactly 6000 node-minutes:

```
acct100 0 6000 0
```

All of the unused node-minutes for *acct100* are thrown away (0% are carried forward). The new allocation is exactly 6000 node-minutes and there would be no node-minutes used.

These examples assume that the weight flag for each account is set to `1` (the default). If the weight flag were set to `0`, all unused node-minutes would be automatically carried forward.

You may want to use **macupdate** as part of a weekly or monthly **cron** job to automatically update the account allocations and usages in the MACS database. Refer to the **cron** manual page for details on doing this.

Exit Status

`0`	If the command completes successfully.
`1`	If the command fails; the command will also print an error message to *stderr*.
>1	Could be a partial success; the exit code is the number of records in the input file that were skipped because of an error. One error message is written to *stderr* for each skipped record.

Files

/usr/spool/macs/conf/macs.conf	The MACS configuration file.

Example

The following file, when used as input for **macupdate**, will update the MACS database for two MACS accounts. The MACS account *res_acct* will get half of its unused node minutes added to 300,000 more node minutes. The MACS account *sim_acct* will get the same number of node minutes as the last accounting period, since all of the unused node minutes are added to all of the used node minutes.

```
% cat upd.infile
res_acct 50 300000 0
sim_acct 100 0 1
```

See Also

macadmin, macalloc

nqstable nqstable

MACS file that specifies the number and names of the NQS queues.

Synopsis

/usr/spool/macs/conf/nqstable

Description

When you run the MACS setup script and supply values for the BATCH_QUEUE and N_BATCH_QUEUE configuration parameters, MACS creates the file */usr/spool/macs/conf/nqstable*. This file begins with a *queue count* (the total number of NQS queues) followed by the names of the queues and the *charge rate* for each queue. For example:

```
# cat /usr/spool/macs/conf/nqstable
4
q4-30 1.0
q2-30 1.0
q2-60 1.0
q2-10 1.0
```

This example shows a queue count of four, and then lists the name of each queue and the queue's charge rate (the integer following the queue name, `1.0` in this example).

After the initial setup, you can later edit this file to add additional NQS queues. If you do so, be sure to change the queue count. If this integer does not match the number of queues, MACS will not start and will not issue any messages indicating why.

This file uses the # character as a comment character.

If you add NQS queues by editing this file, MACS will have to be stopped and restarted before the new queues are recognized.

See Also

actable, macd.data, nxaccount, macs.conf, nx_dflt_accts

nx_dflt_accts nx_dflt_accts

Lists MACS default accounts.

Synopsis

/etc/nx/nx_dflt_accts

Description

The MACS system determines a user's current MACS account by looking in three locations for a valid account—that is, a MACS account that includes the user as a member. It searches these locations in the following order until it finds a valid account for the user:

1. The user's *ACCOUNT* environment variable, if present.

2. The MACS system-level */etc/nx/nx_dflt_accts* file, if present.

3. If MACS cannot find a valid account for a user in the above locations, it will look in account 0 (if present) to see if the user is a member. Account 0 is a MACS account set up by the system administrator with an account ID of 0. Creating an account 0, and allowing all users access to it, is one way to ensure all users are members of a MACS account.

The MACS system-level */etc/nx/nx_dflt_accts* file is created by the system administrator during MACS configuration. The file lists the user's default MACS account. After logging in, the user can change to another valid account using the *ACCOUNT* environment variable or by including the *ACCOUNT* environment variable in a login initialization file such as *.profile* and *.cshrc* to override the account specified by the */etc/nx/nx_dflt_accts* file.

In accounting-only mode, MACS accounts are created by editing the */etc/nxaccount* file; in resource control mode, accounts are created using the **macadmin** command.

nx_dflt_accts *(cont.)*

nx_dflt_accts *(cont.)*

The entries in the */etc/nx/nx_dflt_accts* file have the following format:

account_name:*user1*,*user2*,*user3*

For example:

```
acct100:archer,connor,ddh
acct200:sandy,tracie,adam,archer
```

Lines must begin with an alpha character (a–z, A–Z) and can contain both alpha and numeric characters (0–9). Comments are allowed and should begin with a (#) sign.

Tabs and spaces are not allowed. If an invalid character, tab or space is encountered, the rest of the line is ignored.

See Also

actable, macd.data, nqstable, macs.conf, nxaccount

nxaccount nxaccount

MACS account management file.

Synopsis

/etc/nxaccount

Description

In resource control mode, MACS account management is performed by someone with *root* privileges using **macadmin**, or someone with MACS account manager privileges using **macalloc**. These utilities write to the *nxaccount* file. If your site uses accounting-only mode, these management procedures are accomplished by editing the */etc/nxaccount* file directly.

NOTE

Never edit the */etc/nxaccount* file if your site is configured for resource control mode (the MACMODE configuration parameter is set to *macwatch*). Instead, use the **macadmin** command to add and remove MACS accounts and users.

The format of the file is similar to the Paragon system's */etc/group* file. The following example shows a simple */etc/nxaccount* entry:

```
acct_100:*:100:sandy,mikez,tracie,billops
```

Each field is separated by a colon (`:`). The first field (`acct_100`) is the MACS account's name. The second field is a password field that should always contain an asterisk (`*`). The third field (`100`) is the MACS account's ID. A comma-separated list of MACS users follows the final colon.

You can use the asterisk (`*`) wild card to include all Paragon system users in a MACS account, and the exclamation point (`!`) character to exclude a particular user. For example, this entry would add all Paragon system users to the MACS account `acct_200`:

```
acct_200:*:200:*
```

nxaccount *(cont.)* nxaccount *(cont.)*

The following entry would add all Paragon system users except `sandy` and `mikez` to the MACS account `acct_300`:

```
acct_300:*:300:*,!sandy,!mikez
```

The following entry would create account 0 and allow all Paragon system users access to that account.

```
acct_0:*:0:*
```

The */etc/nxaccount* file cannot contain any blank lines or comment lines.

See Also

actable, macd.data, nqstable, macs.conf, nx_dflt_accts

si si

Reports the start or end of a scheduled interrupt time period. This command is used by the Paragon system administrator.

Synopsis

si -boot | -shutdown
si [**-on** | **-off**] [**-t** "*type*"] **-c** "*comment*"

Options

-boot Writes a line beginning with `PARABOOT` to the MACS log file.

-shutdown Shuts down the MACS daemon and writes a line beginning with `MACDOWN` to the MACS log file.

-on Reports the start of a scheduled interrupt time period.

-off Reports the end of a scheduled interrupt time period.

-t "*type*" A text string used to describe the type of interrupt. The text string must begin with one of the following:

crash	System panic
hangs	System hung
maint	Scheduled maintenance
softw	Software interrupt
hardw	Hardware interrupt
testi	Scheduled system testing
other	Miscellaneous downtime
exter	External cause

In absence of this option, *type* is reported as **unkno** (for unknown).

-c "*comment*" Used to describe the reason for the interrupt.

si *(cont.)* si *(cont.)*

Description

The **si** command writes the status of a scheduled interrupt time period into the daily log file located in the */usr/spool/macs/log.d/macdlog.d* directory. The daily log files have the format *macdYYYYMMDD*, where *YYYY* is the year, *MM* is the month, and *DD* is the day. For example, the log file for April 15, 1994 would be named *macd19940415*.The **si** command writes the time, status type, and reason for the interrupt. You define the start of a scheduled interrupt by issuing an **si -on** command. When the scheduled interrupt is over, issue the **si -off** command. Except when used with the **-shutdown** switch, the **si** command has no effect on Paragon system operations other than to cause down time to be interpreted as scheduled (between **si -on** and **si -off**) in the accounting reports.

Exit Status

`0`	If the command completes successfully.
`-1`	If the command fails; the command will also print an error message to *stderr*.

Files

/usr/spool/macs/conf/macs.conf	Configuration for MACS.
*/usr/spool/macs/log.d/macdlog.d/macd**	The MACS log files.

Index

A

B

C

D

E

F

H

I

J

L

M

N

O

R

S

U

W