The UNIX Accounting System

H. S. McCreary
A. G. Petruccelli

Bell Laboratories
Piscataway, New Jersey 08854

ABSTRACT

The UNIX† Accounting System provides methods to collect per-process resource utilization data, to record connect sessions, to monitor disk utilization, and to charge fees to specific logins. A set of C programs and shell procedures is provided to reduce this accounting data into summary files and reports. This memorandum describes the structure, implementation, and management of this accounting system, as well as a discussion of the reports generated and the meaning of the columnar data.

1. INTRODUCTION

The UNIX Accounting System was originally designed by John Mashey. Several modifications and additions have been made to make the system easier to manage and to make it less susceptible to corrupted data or system errors. The following list is a synopsis of the actions of the accounting system:

- At process termination the UNIX Kernel writes one record per process in **/usr/adm/pacct** in the form of *acct.h*.[1]
- The *login* and *init* programs record connect sessions by writing records into **/usr/adm/wtmp**. Date changes, reboots, and shutdowns are also recorded in this file.
- The disk utilization program *acctdusg*, breaks down disk usage by login.
- Fees for file restores, etc, can be charged to specific logins with the *chargefee* shell procedure.
- Each day the *runacct* shell procedure is executed via *cron* to reduce accounting data, produce summary files and reports.[2]
- The *monacct* procedure can be executed on a monthly or fiscal period basis. It saves and restarts summary files, generates a report, and cleans up the **sum** directory. These saved summary files could be used to charge users for UNIX usage.

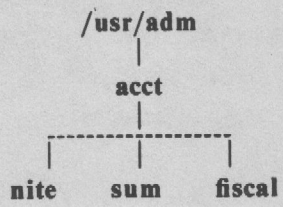
2. FILES AND DIRECTORIES

The **/usr/lib/acct** directory contains all of the C programs and shell procedures necessary to run the accounting system. The **adm** login (currently user ID of 4) is used by the accounting system and has the following directory structure:

---

† UNIX is a trademark of Bell Laboratories.

1. See Attachment 2 for a description of data files.
2. See Attachment 3 for a sample report.



The **/usr/adm** directory contains the active data collection files.[3] The **nite** directory contains files that are re-used daily by the *runacct* procedure. The **sum** directory contains the cumulative summary files updated by *runacct*. The **fiscal** directory contains periodic summary files created by *monacct*.

3. DAILY OPERATION

When UNIX is switched into multi-user mode, **/usr/lib/acct/startup** is executed which does the following:

1. The *acctwtmp* program adds a "boot" record to **/usr/adm/wtmp**. This record is signified by using the system name as the login name in the **wtmp** record.
2. Process accounting is started via *turnacct*. *Turnacct on* executes the *accton* program with the argument **/usr/adm/pacct**.
3. The *remove* shell procedure is executed to clean up the saved **pacct** and **wtmp** files left in the **sum** directory by *runacct*.

The *ckpacct* procedure is run via *cron* every hour of the day to check the size of **/usr/adm/pacct**. If the file grows past 1000 blocks (default), *turnacct switch* is executed. While *ckpacct* is not absolutely necessary, the advantage of having several smaller **pacct** files becomes apparent when trying to restart *runacct* after a failure processing these records.

The *chargefee* program can be used to bill users for file restores, etc. It adds records to **/usr/adm/fee** which are picked up and processed by the next execution of *runacct* and merged into the total accounting records.

Runacct is executed via *cron* each night. It processes the active accounting files, **/usr/adm/pacct?**, **/usr/adm/wtmp**, **/usr/adm/acct/nite/disktacct**, and **/usr/adm/fee**. It produces command summaries and usage summaries by login.

When the system is shut down using *shutdown*, the *shutacct* shell procedure is executed. It writes a shutdown reason record into **/usr/adm/wtmp** and turns process accounting off.

After the first re-boot each morning, the computer operator should execute **/usr/lib/acct/prdaily** to print the previous day's accounting report.

4. SETTING UP THE ACCOUNTING SYSTEM

In order to automate the operation of this accounting system, several things need to be done:

1. If not already present, add this line to the **/etc/rc** file in the state 2 section:

   /bin/su – adm –c /usr/lib/acct/startup

2. If not already present, add this line to */etc/shutdown* to turn off the accounting before the system is brought down:

---

3. For a complete explanation of the files used by the accounting system, see Attachment 1.

/usr/lib/acct/shutacct

3. For most installations, the following three entries should be made in **/usr/lib/crontab** so that *cron* will automatically run the daily accounting:

```
0 4 * * 1-6 /bin/su - adm -c "/usr/lib/acct/runacct 2> /usr/adm/acct/nite/fd2log"
0 2 * * 4 /usr/lib/acct/dodisk
5 * * * * /bin/su - adm -c "/usr/lib/acct/ckpacct"
```

Note that *dodisk* is invoked with super-user privileges of **root** so that directory searching is not road blocked.

To facilitate monthly merging of accounting data, the following entry in **crontab** will allow *monacct* to clean up all daily reports and daily total accounting files and deposit one monthly total report and one monthly total accounting file in the **fiscal** directory:

```
15 5 1 * * /bin/su - adm -c /usr/lib/acct/monacct
```

The above entry takes advantage of the default action of *monacct* that uses the current month's date as the suffix for the file names. Notice that the entry is executed at such a time as to allow *runacct* sufficient time to complete. This will, on the first day of each month, create monthly accounting files with the entire month's data.

4. The **PATH** shell variable should be set in **/usr/adm/.profile** to:

```
PATH=/usr/lib/acct:/bin:/usr/bin
```

5. RUNACCT

Runacct is the main daily accounting shell procedure. It is normally initiated via *cron* during non-prime time hours. *Runacct* processes connect, fee, disk, and process accounting files. It also prepares daily and cumulative summary files for use by *prdaily* or for billing purposes. The following files produced by *runacct* are of particular interest:

nite/lineuse	Produced by *acctcon1*, which reads the **wtmp** file, and produces usage statistics for each terminal line on the system. This report is especially useful for detecting bad lines. If the ratio between the number of logoffs to logins exceeds about 3/1, there is a good possibility that the line is failing.
nite/daytacct	This file is the total accounting file for the previous day in *tacct.h* format.
sum/tacct	This file is the accumulation of each day's **nite/daytacct**, which can be used for billing purposes. It is restarted each month or fiscal by the *monacct* procedure.
sum/daycms	Produced by the *acctcms* program, it contains the daily command summary. The ASCII version of this file is **nite/daycms**.
sum/cms	The accumulation of each day's command summaries. It is restarted by the execution of *monacct*. The ASCII version is **nite/cms**.
sum/loginlog	Produced by the *lastlogin* shell procedure, it maintains a record of the last time each login was used.
sum/rprt.MMDD	Each execution of *runacct* saves a copy of the output of *prdaily*.

Runacct takes care not to damage files in the event of errors. A series of protection mechanisms are used that attempt to recognize an error, provide intelligent diagnostics, and terminate processing in such a way that *runacct* can be restarted with minimal intervention. It records its

progress by writing descriptive messages into the file **active.**[4] All diagnostic output during the execution of *runacct* is written into **fd2log**. To prevent multiple invocations, in the event of two crons or other problems, *runacct* will complain if the files **lock** and **lock1** exist when invoked. The **lastdate** file contains the month and day *runacct* was last invoked, and is used to prevent more than one execution per day. If *runacct* detects an error, a message is written to **/dev/console**, mail is sent to **root** and **adm**, the locks are removed, diagnostic files are saved, and execution is terminated.

In order to allow *runacct* to be restartable, processing is broken down into separate reentrant states. This is accomplished by using a **case** statement inside an endless **while** loop. Each state is one case of the **case** statement. A file is used to remember the last state completed. When each state completes, **statefile** is updated to reflect the next state. In the next loop through the **while**, **statefile** is read and the **case** falls through to the next state. When *runacct* reaches the **CLEANUP** state, it removes the locks and terminates. *States* are executed as follows:

SETUP	The command **turnacct switch** is executed. The process accounting files, **/usr/adm/pacct***?*, are moved to **/usr/adm/Spacct***?.MMDD*. The **/usr/adm/wtmp** file is moved to **/usr/adm/acct/nite/wtmp.***MMDD* with the current time added on the end.
WTMPFIX	The **wtmp** file in the **nite** directory is checked for correctness by the *wtmpfix* program. Some date changes will cause *acctcon1* to fail, so *wtmpfix* attempts to adjust the time stamps in the **wtmp** file if a date change record appears.
CONNECT1	Connect session records are written to **ctmp** in the form of *ctmp.h*. The **lineuse** file is created, and the **reboots** file is created showing all of the boot records found in the **wtmp** file.
CONNECT2	**Ctmp** is converted to **ctacct.***MMDD*, the connect accounting records.[5]
PROCESS	The *acctprc1* and *acctprc2* programs are used to convert the process accounting files, **/usr/adm/Spacct***?.MMDD*, into total accounting records in **ptacct***?.MMDD*. The **Spacct** and **ptacct** files are correlated by number so that if *runacct* fails, the unnecessary reprocessing of **Spacct** files will not occur. One precaution should be noted; when restarting *runacct* in this state, remove the last ptacct file because it will not be complete.
MERGE	Merge the process accounting records with the connect accounting records to form **daytacct**.
FEES	Merge in any ASCII *tacct* records from the file **fee** into **daytacct**.
DISK	On the day after the *sdisk* procedure runs, merge **disktacct** with **daytacct**.
MERGETACCT	Merge **daytacct** with **sum/tacct**, the cumulative total accounting file. Each day, **daytacct** is saved in **sum/tacct***MMDD*, so that **sum/tacct** can be recreated in the event it becomes corrupted or lost.
CMS	Merge in today's command summary with the cumulative command summary file **sum/cms**. Produce ASCII and internal format command summary files.

4. Files used by *runacct* are assumed to be in the **nite** directory unless otherwise noted.
5. Accounting records are in *tacct.h* format.

USEREXIT — Any installation dependent (local) accounting programs can be included here.

CLEANUP — Clean up temporary files, run *prdaily* and save its output in **sum/rprt***MMDD*, remove the locks, then exit.

6. RECOVERING FROM FAILURE

The *runacct* procedure can fail for a variety of reasons; usually due to a system crash, **/usr** running out of space, or a corrupted **wtmp** file. If the **active***MMDD* file exists, check it first for error messages. If the **active** file and lock files exist, check **fd2log** for any mysterious messages. The following are error messages produced by *runacct*, and the recommended recovery actions:

ERROR: locks found, run aborted
The files **lock** and **lock1** were found. These files must be removed before *runacct* can restart.

ERROR: acctg already run for *date* : check /usr/adm/acct/nite/lastdate
The date in **lastdate** and today's date are the same. Remove **lastdate**.

ERROR: turnacct switch returned rc=*?*
Check the integrity of *turnacct* and *accton*. The *accton* program must be owned by **root**, and have the setuid bit set.

ERROR: Spacct*?*.*MMDD* already exists
file setups probably already run
Check status of files, then run setups manually.

ERROR: /usr/adm/acct/nite/wtmp.*MMDD* already exists, run setup manually
Self-explanatory.

ERROR: wtmpfix errors see /usr/adm/acct/nite/wtmperror
Wtmpfix detected a corrupted **wtmp** file. Use *fwtmp* to correct the corrupted file.

ERROR: connect acctg failed: check /usr/adm/acct/nite/log
The *acctcon1* program encountered a bad **wtmp** file. Use *fwtmp* to correct the bad file.

ERROR: Invalid state, check /usr/adm/acct/nite/active
The file, **statefile**, is probably corrupted. Check **statefile** and read **active** before restarting.

7. RESTARTING RUNACCT

Runacct called without arguments assumes that this is the first invocation of the day. The argument *MMDD* is necessary if *runacct* is being restarted, and specifies the month and day for which *runacct* will rerun the accounting. The entry point for processing is based on the contents of **statefile**. To override **statefile**, include the desired *state* on the command line. For example:

To start *runacct*:

```
nohup runacct  2>  /usr/adm/acct/nite/fd2log&
```

To restart *runacct*:

```
nohup runacct 0601  2>  /usr/adm/acct/nite/fd2log&
```

To restart *runacct* at a specific state:

```
nohup runacct 0601 WTMPFIX  2>  /usr/adm/acct/nite/fd2log&
```

8. FIXING CORRUPTED FILES

Unfortunately, this accounting system is not entirely fool proof. Occasionally a file will become corrupted, or lost. Some of the files can simply be ignored or restored from the file-save backup, but others must be fixed to maintain the integrity of the accounting system.

8.1 Fixing WTMP Errors

The **wtmp** files seem to cause the most problems in the day to day operation of the accounting system. When the date is changed when UNIX is in multi-user mode, a set of date change records is written into **/usr/adm/wtmp**. The *wtmpfix* program is designed to adjust the time stamps in the **wtmp** records when a date change is encountered. Some combinations of date changes and reboots, however, will slip through *wtmpfix* and cause *acctcon1* to fail. The following steps show how to patch up a **wtmp** file.

```
cd /usr/adm/acct/nite
fwtmp < wtmp.MMDD > xwtmp
ed xwtmp
    delete corrupted records or
    delete all records from the beginning up to the date change
fwtmp -ic < xwtmp > wtmp.MMDD
```

If the **wtmp** file is beyond repair, create a null **wtmp** file. This will prevent any charging of connect time. *Acctprc1* won't be able to determine which login owned a particular process, but it will be charged to the login that is first in the password file for that userid.

8.2 Fixing TACCT Errors

If the installation is using the accounting system to charge users for system resources, the integrity of **sum/tacct** is quite important. Occasionally, mysterious **tacct** records will appear with negative numbers, duplicate user IDs, or a user ID of 65535. First check **sum/tacctprev** with *prtacct*. If it looks all right, the latest **sum/tacct.***MMDD* should be patched up, then **sum/tacct** recreated. A simple patchup procedure would be:

```
cd /usr/adm/acct/sum
acctmerg -v < tacct.MMDD > xtacct
ed xtacct
    remove the bad records
    write duplicate uid records to another file
acctmerg -i < xtacct > tacct.MMDD
acctmerg tacctprev < tacct.MMDD > tacct
```

Remember that the *monacct* procedure removes all the **tacct.***MMDD* files; therefore, **sum/tacct** can be recreated by merging these files together.

9. UPDATING PNPSPLIT

The *pnpsplit* subroutine is used by *acctcon1* and *acctprc1* to determine the difference between prime and non-prime time. Prime time is defaulted from 9 a.m. to 5 p.m. Monday through Friday. Non-prime time is considered to be all other hours and the entire day for those days listed in the **holidays** structure in **pnpsplit.c**. The holidays listed are accurate for Bell Laboratories, New Jersey locations for the year the operating system was released. Every year on the day after Christmas (the last holiday of the calendar year), the following message will be printed on the system console terminal and appear in **log**:

```
*** RECOMPILE pnpsplit WITH NEW HOLIDAYS ***
```

This message will continue to be sent each time the accounting is run until *pnpsplit*, *acctcon1*, and *acctprc1* are recompiled. The following steps should be taken to successfully recompile these programs:

1. Edit **pnpsplit.c** to change the **thisyear** variable to the new year. Update the **holidays** structure to reflect the new holidays. The numeric entry in the structure is the day of the year, less one. For example, New Year's Day (January 1) is entered as 0. **Pnpsplit.c** is in **/usr/src/cmd/acct/lib**.

2. Update the accounting library *a.a* and recompile *acctprc1*, and *acctcon1* by:

   super-user to root
   ARGS="acctcon1 acctprc1" /usr/src/:mkcmd acct

10. DAILY REPORTS

Runacct generates 5 basic reports upon each invocation. A sample of these reports are shown in Attachment 3. They cover the areas of connect accounting, usage by person on a daily basis, command usage reported by daily and monthly totals, and a report of the last time users were logged in.

The following paragraphs describe the reports and the meanings of their tabulated data.

10.1 Daily Report

In the first part of the report, the *from/to* banner should alert you to the period reported on. The times are the time the last accounting report was generated until the time the current accounting report was generated. It is followed by a log of system reboots, shutdowns, power fail recoveries, and any other record dumped into **/usr/adm/wtmp** by the *acctwtmp* program (see *acct*(1M)).

The second part of the report is a breakdown of line utilization. The **TOTAL DURATION** tells how long the system was in multi-user state (able to be accessed through the terminal lines). The columns are:

LINE — The terminal line or access port.

MINUTES — The total number of minutes that line was in use during the accounting period.

PERCENT — The total number of MINUTES the line was in use divided into the TOTAL DURATION.

SESS — The number of times this port was accessed for a *login*(1) session.

ON — This column does not have much meaning anymore. It used to give the number of times that the port was used to log a user on, but because *login*(1) can no longer be executed explicitly to log a new user in, this column should be identical with SESS.

OFF — This column reflects not just the number of times a user logged off, but also any interrupts that occur on that line. Generally, interrupts occur on a port when the *getty*(8) is first invoked when the system is brought to multi-user state. These interrupts occur at a rate of about two per event; therefore it is not uncommon to see in excess of twice the amount of OFF than in ON or SESS. Where this column does come into play is when the # OFF exceeds the # ON by a large factor. This usually indicates that the multiplexor, modem or cable is going bad, or there is a bad connection somewhere. The most common cause of this is an unconnected cable dangling from the multiplexor.

During real time, **/usr/adm/wtmp** should be monitored as this is the file that the connect accounting is geared off of. If it grows rapidly, execute *acctcon1* to see which *tty* line is the most noisy. If the interrupting is occurring at a furious rate, you'll be able to feel the effect on general system performance.

10.2 Daily Usage Report

This report gives a by-user breakdown of system resource utilization. Its data consists of:

UID — The user ID.

LOGIN NAME — The login name of the user; there can be more than one login name for a single user ID, this identifies which one.

CPU (MINS) — This represents the amount of time the user's process used the central processing unit. This category is broken down into PRIME and NPRIME (non-prime) utilization. The accounting system's idea of this breakdown is located in the accounting library function *prpsplit* where the **holidays** array, which also determines non-prime time, is also defined. As delivered, prime time is defined to be 0900-1700 hours. The **holidays** array is correct for New Jersey locations of Bell Laboratories for the year of the release.

KCORE-MINS — This represents a cummulative measure of the amount of memory a process uses while running. The amount shown reflects kilo-byte segments of memory used per minute. This measurement is also broken down into PRIME and NPRIME amounts.

CONNECT (MINS) — This identifies "Real Time" used. What this column really identifies, is the amount of time that a user was logged into the system. If this time is rather high and the later column called # OF PROCS is low, this user is what is called a "line hog." That is, this person logs in first thing in the morning and doesn't hardly touch the terminal the rest of the day. Watch out for this kind of critter. This column is also subdivided into PRIME and NPRIME utilization.

DISK BLOCKS — When the disk accounting programs have been run, their output is merged into the total accounting record (**tacct.h**) and shows up in this column. This disk accounting is accomplished by the program *acctdusg*.

OF PROCS — This column reflects the number of processes that was invoked by the user. This is a good column to watch for large numbers indicating that a user may have a shell procedure that runs amuck. The most common example of this is for a **crontab** entry to try to execute a user's **.profile** via **su**— that unfortunately prompts for a terminal type and sits in an endless loop trying to read from the terminal (there isn't one when *cron* is executing a process). Preventive coding is encouraged in the **.profile**.

OF SESS — This is how many times the user logged onto the system.

DISK SAMPLES — This indicates how many times the disk accounting was run to obtain the average number of DISK BLOCKS listed earlier.

FEE — A much often unused field in the total accounting record, the FEE represents the total accumulation of *widgets* charged against the user by the *chargefee* shell procedure (see *acctsh*(1M)). The *chargefee* procedure is used to levy charges against a user for special services performed such as file restores, tape manipulation by operators, etc.

10.3 Daily Command and Monthly Total Command Summaries

These two reports are virtually the same except that the Daily Command Summary only reports on the current accounting period, while the Monthly Total Command Summary tells the story for the start of the fiscal period to the current date. In other words, the monthly report reflects the data accumulated since the last invocation of *monacct*.

The data included in these reports give an administrator an idea as to the heaviest used commands, and based on those commands' characteristics of system resource utilization, a hint as to what to weigh more heavily when system tuning.

These reports are sorted by TOTAL KCOREMIN which is an arbitrary yardstick, but often a good one for calculating "drain" caused on a system.

COMMAND NAME	This is the name of the command. Unfortunately, all shell procedures are lumped together under the name **sh** because only object modules are reported by the process accounting system. The administrator should monitor the frequency of programs called *a.out* or *core* or any other name that doesn't seem quite right. Often people like to work on their favorite version of *backgammon* only they don't want everyone to know about it. *Acctcom* is also a good tool to use for determining who executed a suspiciously named command and also if super-user privileges were used.
NUMBER CMDS	This is the total number of invocations of this particular command.
TOTAL KCOREMIN	The total cummulative measurement of the amount of kilo-byte segments of memory used by a process per minute of run time.
TOTAL CPU-MIN	The total processing time this program has accumulated.
TOTAL REAL-MIN	The total real time (wall-clock) minutes this program has accumulated. This total is the actual "waited for" time as opposed to kicking off a process in the background.
MEAN SIZE-K	This is the mean of the TOTAL KCOREMIN over the number of invocations reflected by NUMBER CMDS.
MEAN CPU-MIN	This is the mean derived between the NUMBER CMDS and TOTAL CPU-MIN.
HOG FACTOR	This is a relative measurement of the ratio of system availability to system utilization. It is computed by the formula (total CPU time) / (elapsed time) This gives a relative measure of the total available CPU time consumed by the process during its execution.
CHARS TRNSFD	This column, which may go negative, is a total count of the number of characters pushed around by the *read*(2) and *write*(2) system calls.
BLOCKS READ	A total count of the physical block reads and writes that a process performed.

10.4 Last Login

This report simply gives the date when a particular login was last used. This could be a good source for finding likely candidates for the tape archives or getting rid of unused logins and login directories.

11. SUMMARY

The UNIX Accounting System was designed from a UNIX system administrator's point of view. Every possible precaution has been taken to ensure that the system will run smoothly and without error. It is important to become familiar with the C programs and shell procedures. The manual entries should be studied, and it is advisable to keep a printed copy of the shell procedures handy. This accounting system should be easy to maintain, provide valuable information for the administrator, and provide accurate breakdowns of the usage of system resources for charging purposes.

Attachment 1

Files in the /usr/adm directory:

diskdiag	diagnostic output during the execution of disk accounting programs
dtmp	output from the *acctdusg* program
fee	output from the *chargefee* program, ASCII tacct records
pacct	active process accounting file
pacct?	process accounting files switched via *turnacct*
Spacct?.MMDD	process accounting files for *MMDD* during execution of *runacct*
wtmp	active wtmp file for recording connect sessions

Files in the /usr/adm/acct/nite directory:

active	used by *runacct* to record progress and print warning and error messages; **active***MMDD* same as **active** after *runacct* detects an error
cms	ASCII total command summary used by *prdaily*
ctacct.MMDD	connect accounting records in *tacct.h* format
ctmp	output of *acctcon1* program, connect session records in *ctmp.h* format
daycms	ASCII daily command summary used by *prdaily*
daytacct	total accounting records for one day in *tacct.h* format
disktacct	disk accounting records in *tacct.h* format, created by *dodisk* procedure
fd2log	diagnostic output during execution of runacct (see *cron* entry)
lastdate	last day *runacct* executed in *date +%m%d* format
lock lock1	used to control serial use of *runacct*
lineuse	tty line usage report used by *prdaily*
log	diagnostic output from *acctcon1*
logMMDD	same as **log** after *runacct* detects an error
reboots	contains beginning and ending dates from **wtmp**, and a listing of reboots
statefile	used to record current state during execution of *runacct*
tmpwtmp	wtmp file corrected by *wtmpfix*
wtmperror	place for *wtmpfix* error messages
wtmperrorMMDD	same as **wtmperror** after *runacct* detects an error
wtmp.MMDD	previous day's wtmp file

Files in the /usr/adm/acct/sum directory:

cms	total command summary file for current fiscal in internal summary format
cmsprev	command summary file without latest update
daycms	command summary file for yesterday in internal summary format
loginlog	created by *lastlogin*
pacct.MMDD	concatenated version of all pacct files for *MMDD*, removed after reboot by *remove* procedure
rprt.MMDD	saved output of *prdaily* program
tacct	cumulative total accounting file for current fiscal
tacctprev	same as **tacct** without latest update
tacct.MMDD	total accounting file for *MMDD*
wtmp.MMDD	saved copy of wtmp file for *MMDD*, removed after reboot by *remove* procedure

Files in the /usr/adm/acct/fiscal directory:

cms?	total command summary file for fiscal *?* in internal summary format
fiscrpt?	report similar to *prdaily* for fiscal *?*
tacct?	total accounting file for fiscal *?*

Attachment 2

Format of wtmp files (utmp.h):

```
/*
 * Format of /etc/utmp and /usr/adm/wtmp
 */
struct utmp {
        char    ut_line[8];             /* tty name */
        char    ut_name[8];             /* user id */
        long    ut_time;                /* time on */
};
```

Definitions (acctdef.h):

```
/*
 *      defines, typedefs, etc. used by acct programs
 *
 *      acct only typedefs
 */
typedef unsigned short  uid_t;

#define LSZ     8       /* sizeof line name */
#define NSZ     8       /* sizeof login name */
#define P       0       /* prime time */
#define NP      1       /* nonprime time */

/*
 *      limits which may have to be increased if systems get larger
 */
#define SSIZE  1000     /* max number of sessions in 1 acct run */
#define TSIZE  100      /* max number of line names in 1 acct run */
#define USIZE 500       /* max number of distinct login names in 1 acct run */

#define EQN(s1, s2)    (strncmp(s1, s2, sizeof(s1)) == 0)
#define CPYN(s1, s2)  strncpy(s1, s2, sizeof(s1))

#define SECS(tics)      ((double) tics)/60.
#define MINS(secs)      ((double) secs)/60.
#define MINT(tics)      ((double) tics)/3600.
#ifdef pdp11
#define KCORE(clicks)           ((double) clicks/16)
#endif
#ifdef vax
#define KCORE(clicks)           ((double) clicks/2)
#endif
#define SECSINDAY   86400L
```

Format of pacct files (acct.h):

```
/*
 * Accounting structures
 */

typedef ushort comp_t;          /* "floating point" */
                /* 13-bit fraction, 3-bit exponent  */

struct  acct
{
        char    ac_flag;        /* Accounting flag */
        char    ac_stat;        /* Exit status */
        ushort  ac_uid;         /* Accounting user ID */
        ushort  ac_gid;         /* Accounting group ID */
        dev_t   ac_tty;         /* control typewriter */
        time_t  ac_btime;       /* Beginning time */
        comp_t  ac_utime;       /* acctng user time in clock ticks */
        comp_t  ac_stime;       /* acctng system time in clock ticks */
        comp_t  ac_etime;       /* acctng elapsed time in clock ticks */
        comp_t  ac_mem;         /* memory usage */
        comp_t  ac_io;          /* chars transferred */
        comp_t  ac_rw;          /* blocks read or written */
        char    ac_comm[8];     /* command name */
};

extern  struct  acct    acctbuf;
extern  struct  inode   *acctp; /* inode of accounting file */

#define AFORK           01      /* has executed fork, but no exec */
#define ASU             02      /* used super-user privileges */
#define ACCTF           0300    /* record type: 00 = acct */
```

Format of tacct files (tacct.h):

```
/*
 *      total accounting (for acct period), also for day
 */

struct  tacct   {
        uid_t           ta_uid;         /* userid */
        char            ta_name[8];     /* login name */
        float           ta_cpu[2];      /* cum. cpu time, p/np (mins) */
        float           ta_kcore[2];    /* cum kcore-minutes, p/np */
        float           ta_con[2];      /* cum. connect time, p/np, mins */
        float           ta_du;          /* cum. disk usage */
        long            ta_pc;          /* count of processes */
        unsigned short  ta_sc;          /* count of login sessions */
        unsigned short  ta_dc;          /* count of disk samples */
        unsigned short  ta_fee;         /* fee for special services */
};
```

Format of ctmp file (ctmp.h):

```
/*
 *      connect time record (various intermediate files)
 */
struct ctmp {
        dev_t   ct_tty;         /* major minor */
        uid_t   ct_uid;         /* userid */
        char    ct_name[8];     /* login name */
        long    ct_con[2];      /* connect time (p/np) secs */
        time_t  ct_start;       /* session start time */
};
```

Attachment 3

Jun 8 04:14 1979 DAILY REPORT FOR pwba Page 1

from Thu Jun 7 06:00:48 1979
to Fri Jun 8 04:00:28 1979
2 shutdown
2 pwba

TOTAL DURATION IS 1320 MINUTES

LINE	MINUTES	PERCENT	# SESS	# ON	# OFF
tty04	479	36	9	9	30
tty47	341	26	4	4	33
tty44	298	23	3	3	29
tty46	336	25	9	9	33
console	1100	83	14	14	21
tty05	448	34	3	3	22
tty06	439	33	9	9	31
tty07	421	32	6	6	24
tty42	53	4	5	5	20
tty09	385	29	11	11	33
tty10	336	25	10	10	31
tty08	464	35	2	2	19
tty26	544	41	6	6	24
tty12	252	19	5	5	25
tty13	258	20	3	3	21
tty14	156	12	6	6	26
tty17	145	11	1	1	16
tty18	39	3	5	5	24
tty15	228	17	5	5	25
tty25	704	53	6	6	25
tty21	0	0	0	0	16
tty19	10	1	1	1	17
tty20	25	2	2	2	18
tty22	0	0	0	0	15
tty23	0	0	0	0	15
tty24	0	0	0	0	16
tty27	481	36	3	3	20
tty28	426	32	5	5	24
tty29	302	23	6	6	25
tty30	257	20	11	11	28
tty40	380	29	5	5	21
tty41	343	26	3	3	21
tty45	0	0	0	0	15
tty11	365	28	7	7	25
tty43	3	0	1	1	17
tty16	213	16	3	3	20
tty31	250	19	4	4	18
tty02	62	5	1	1	3
TOTALS	10544	--	174	174	846

Jun 8 04:14 1979 DAILY USAGE REPORT FOR pwba Page 1

UID	LOGIN NAME	CPU (MINS) PRIME	CPU (MINS) NPRIME	KCORE-MINS PRIME	KCORE-MINS NPRIME	CONNECT (MINS) PRIME	CONNECT (MINS) NPRIME	DISK BLOCKS	# OF PROCS	# OF SESS	# DISK SAMPLES	FEE
0	TOTAL	388	103	12414	2934	9251	1056	0	16164	174	0	0
0	root	47	41	1003	924	67	30	0	2360	8	0	0
4	adm	7	19	48	652	0	0	0	842	0	0	0
19	games	0	0	4	0	0	0	0	28	0	0	0
22	mhb	0	0	1	1	1	1	0	14	2	0	0
37	abs	0	0	4	0	0	0	0	3	0	0	0
37	absjrk	14	0	284	0	423	0	0	1588	4	0	0
68	rje	3	3	24	21	0	0	0	179	0	0	0
71	?	0	0	0	0	0	0	0	12	0	0	0
150	jac	7	0	156	5	281	2	0	510	13	0	0
173	?	0	0	0	0	0	0	0	16	0	0	0
180	?	0	0	0	0	0	0	0	4	0	0	0
185	?	0	0	0	0	0	0	0	2	0	0	0
217	denise	0	0	2	0	31	0	0	32	3	0	0
217	kof	0	0	2	0	1	0	0	7	1	0	0
219	?	0	0	0	0	0	0	0	12	0	0	0
1001	hsm	5	0	189	0	179	0	0	92	2	0	0
2001	systst	0	1	5	28	476	64	0	99	5	0	0
2002	mfp	1	0	7	5	270	62	0	93	3	0	0
2003	als	1	0	23	0	100	0	0	99	3	0	0
2005	eric	0	0	3	0	13	0	0	21	1	0	0
2006	hoot	0	0	2	0	16	0	0	8	1	0	0
2009	agp	47	0	2040	0	444	0	0	492	2	0	0
2009	fsrep1	2	0	60	0	36	0	0	95	1	0	0
2011	pdw	0	0	1	0	4	0	0	11	1	0	0
2012	pwbst	0	0	1	0	28	0	0	9	1	0	0
2014	cath	0	0	1	0	1	0	0	7	1	0	0
2022	rem	32	1	1227	91	576	4	0	226	3	0	0
2025	fld	55	23	2176	862	336	98	0	750	7	0	0
2027	krb	14	2	365	51	547	24	0	372	8	0	0
2028	text	0	0	1	0	3	0	0	13	1	0	0
2030	arf	8	0	288	0	317	0	0	315	3	0	0
2031	dp	12	0	480	3	459	6	0	220	6	0	0
2032	graf	2	0	49	0	23	0	0	118	1	0	0
2033	ecp	3	0	74	0	355	0	0	115	4	0	0
2040	leap	15	0	308	0	513	1	0	505	2	0	0
2041	dan	3	0	93	3	149	2	0	117	8	0	0
2051	ds52	2	2	19	40	375	601	0	611	8	0	0
2055	nuucp	0	0	15	9	17	1	0	10	3	0	0
2057	ech	1	0	28	0	63	0	0	68	2	0	0
2061	jcw	4	3	99	70	37	34	0	869	4	0	0
2064	mjr	18	0	443	0	176	0	0	2065	3	0	0
2065	rrr	0	0	6	0	7	0	0	23	1	0	0
2068	trc	0	0	7	0	10	0	0	29	1	0	0
2075	herb	29	0	1178	1	384	2	0	249	5	0	0
2086	paul	1	0	14	0	152	0	0	28	1	0	0
2087	pris	0	0	0	10	0	2	0	13	1	0	0
2111	pwbcs	2	3	60	85	64	86	0	185	4	0	0
2116	rbj	1	0	16	0	408	0	0	222	1	0	0
2121	teach	0	0	3	0	53	0	0	50	2	0	0
2123	msb	0	0	3	0	5	0	0	24	1	0	0
2124	rnt	2	0	42	0	66	0	0	260	3	0	0
2126	dal	0	0	5	0	121	0	0	17	1	0	0
2127	m2	15	0	495	11	390	2	0	602	10	0	0

Jun 8 04:14 1979 DAILY USAGE REPORT FOR pwba Page 2

2128	jel	14	0	492	9	422	14	0	523	8	0	0
2130	s1	0	0	5	1	16	0	0	42	2	0	0
2130	s3	0	0	0	0	0	2	0	9	1	0	0
2135	jfn	0	1	0	12	0	11	0	33	2	0	0
2136	m2class	0	0	5	0	2	0	0	18	1	0	0
2140	star	4	0	213	12	90	3	0	170	7	0	0
2141	reg	5	0	245	25	470	4	0	181	1	0	0
2199	llc	0	0	1	0	10	0	0	7	1	0	0
2999	stock	0	0	1	0	1	0	0	17	1	0	0
3001	whm	5	0	93	0	253	0	0	414	3	0	0
3332	vjf	0	0	4	0	8	0	0	39	1	0	0

Jun 8 04:07 1979 DAILY COMMAND SUMMARY Page 1

COMMAND NAME	NUMBER CMDS	TOTAL KCOREMIN	TOTAL CPU-MIN	TOTAL REAL-MIN	MEAN SIZE-K	MEAN CPU-MIN	HOG FACTOR	CHARS TRNSFD	BLOCKS READ
TOTALS	16164	15332.89	490.72	37463.98	31.25	0.03	0.01	322183844	1097670
nroff	119	3958.68	93.21	569.83	42.47	0.78	0.16	67070052	130284
troff	26	2483.38	51.63	342.70	48.10	1.99	0.15	37869304	48989
xnroff	20	732.03	16.74	111.05	43.73	0.84	0.15	13885248	22659
a.out	31	623.53	10.52	142.77	59.26	0.34	0.07	382435	2758
egrep	185	574.83	13.96	34.53	41.18	0.08	0.40	170625	8249
m2find	232	555.79	9.93	155.11	55.96	0.04	0.06	6155937	30994
c1	150	519.04	13.57	48.89	38.25	0.09	0.28	4285724	16032
c0	165	413.10	9.19	35.16	44.93	0.06	0.26	3827309	12170
m2edit	33	340.92	4.63	148.27	73.62	0.14	0.03	1074914	14492
ld	87	317.38	7.94	38.48	39.97	0.09	0.21	17640896	45797
acctcms	17	294.75	6.49	14.15	45.41	0.38	0.46	2525427	5515
c2	112	289.69	9.13	34.61	31.72	0.08	0.26	3667050	9681
sh	1834	276.98	26.77	20444.24	10.35	0.01	0.00	3496613	71979
ed	524	253.13	14.46	2029.89	17.50	0.03	0.01	18058108	56039
acctprc1	3	231.28	6.67	19.45	34.67	2.22	0.34	2577344	2926
du	145	219.35	19.91	39.08	11.02	0.14	0.51	716389	23695
diff	49	175.53	6.04	25.78	29.05	0.12	0.23	3740887	11351
get	151	152.96	4.28	25.23	35.74	0.03	0.17	3634042	24917
adb	22	148.10	4.07	202.35	36.37	0.19	0.02	2313718	9813
tbl	24	143.43	2.44	210.65	58.71	0.10	0.01	1536210	3433
dd	9	139.24	10.15	51.05	13.72	1.13	0.20	26006848	294
as2	155	129.33	9.82	42.25	13.17	0.06	0.23	10500835	30165
sed	597	115.46	4.19	36.23	27.57	0.01	0.12	783825	24497
ps	51	109.69	5.92	41.55	18.54	0.12	0.14	2278056	8310
make	89	102.94	2.87	203.32	35.81	0.03	0.01	1018461	8664
delta	25	90.23	2.27	17.80	39.70	0.09	0.13	2909269	9321
cpp	172	89.37	2.69	11.32	33.19	0.02	0.24	3519054	12155
fsck	16	86.94	1.30	10.57	66.85	0.08	0.12	27671849	2927
find	52	86.64	5.05	63.87	17.15	0.10	0.08	565125	11161
ls	706	82.47	5.78	62.85	14.26	0.01	0.09	1811882	29659
xck	2	79.44	10.49	47.89	7.57	5.25	0.22	198016	21995
awk	22	78.83	1.37	5.24	57.72	0.06	0.26	355466	3769
uucico	60	75.55	1.42	632.50	53.27	0.02	0.00	398693	6377
acctcom	9	75.21	2.81	11.49	26.75	0.31	0.24	1283776	3771
echo	2814	66.10	7.08	91.80	9.33	0.00	0.08	168651	24253
ged	3	57.27	0.82	7.51	70.16	0.27	0.11	51832	426
dc	284	56.92	2.42	9.43	23.48	0.01	0.26	15283	20329
450	7	48.03	6.80	84.45	7.06	0.97	0.08	279451	1700
cat	749	45.49	5.69	478.54	8.00	0.01	0.01	8959500	27903
ntd	6	41.52	1.55	7.55	26.87	0.26	0.20	59888	478
mail	202	39.95	2.05	532.98	19.53	0.01	0.00	427217	14377
acctprc2	3	38.95	1.43	19.45	27.24	0.48	0.07	587336	87
sort	94	38.72	1.09	9.73	35.41	0.01	0.11	375876	4433
pr	104	34.89	2.47	214.50	14.10	0.02	0.01	1060989	6572
haspmain	7	33.20	5.28	1244.54	6.29	0.75	0.00	63064	36635
ex	17	31.69	0.62	41.04	50.97	0.04	0.02	514624	3593
grep	213	28.73	2.98	21.01	9.64	0.01	0.14	2100229	14297

Jun 8 04:07 1979 MONTHLY TOTAL COMMAND SUMMARY Page 1

COMMAND NAME	NUMBER CMDS	TOTAL KCOREMIN	TOTAL CPU-MIN	TOTAL REAL-MIN	MEAN SIZE-K	MEAN CPU-MIN	HOG FACTOR	CHARS TRNSFD	BLOCKS READ
TOTALS	553286	297698.78	10916.09	742924.94	27.27	0.02	0.01	820472546	26253312
nroff	1687	44681.55	995.92	5737.25	44.86	0.59	0.17	613403153	1089180
troff	1351	25692.15	583.69	4356.05	44.02	0.43	0.13	413163589	646243
spellpro	6466	17298.41	294.16	1893.79	58.81	0.05	0.16	334572640	853901
m2edit	654	13526.69	164.62	4238.58	82.17	0.25	0.04	54940426	427924
xnroff	397	10408.44	203.72	1496.32	51.09	0.51	0.14	215221419	301967
sort	7983	9292.34	226.01	2298.05	41.11	0.03	0.10	80108304	355963
c1	6139	8949.86	236.45	861.09	37.85	0.04	0.27	79897995	489661
ld	3244	8852.96	223.19	1128.09	39.67	0.07	0.20	493701995	1278119
sed	53134	8126.71	313.85	2241.78	25.89	0.01	0.14	23035033	1692990
m2find	2982	7984.45	140.18	1698.25	56.96	0.05	0.08	111330040	449604
c0	6586	7866.42	185.16	725.47	42.49	0.03	0.26	72595655	389426
ed	20083	7822.78	425.90	41898.18	18.37	0.02	0.01	483425634	1541326
tbl	660	7766.69	113.95	2458.55	68.16	0.17	0.05	50760094	83887
sh	40476	7499.67	635.00	383786.53	11.81	0.02	0.00	70525236	1421194
du	1941	6730.54	553.04	1128.44	12.17	0.28	0.49	20848359	628324
a.out	1483	5658.46	126.87	1868.87	44.60	0.09	0.07	16158675	80260
egrep	4801	5573.51	139.86	460.25	39.85	0.03	0.30	6823696	237298
lint1	793	5325.66	71.23	425.67	74.76	0.09	0.17	9599001	131592
cat	21170	4657.53	236.59	4354.24	19.69	0.01	0.05	239180412	1023965
acctprc1	42	3837.84	110.88	291.34	34.61	2.64	0.38	43954136	61123
c2	4067	3807.25	144.86	477.28	26.28	0.04	0.30	57519376	213521
grep	21212	3204.86	300.44	2727.87	10.67	0.01	0.11	139340583	899415
cpp	7469	3060.72	94.12	647.79	32.52	0.01	0.15	91471956	459882
getty	35556	2948.71	853.53	101107.45	3.45	0.02	0.01	34704751	263866
m2editD	83	2707.27	28.79	361.84	94.02	0.35	0.08	2852202	33949
as2	6454	2698.74	218.96	910.59	12.33	0.03	0.24	213336016	705690
make	1858	2449.10	64.69	4388.86	37.86	0.03	0.01	24116259	175544
ps	1034	2384.14	128.29	1207.87	18.58	0.12	0.11	54873792	204172
acctcms	294	2288.36	51.99	116.06	44.01	0.18	0.45	36124940	80523
uucico	815	2226.75	40.42	11729.01	55.08	0.05	0.00	11086105	162558
ls	18876	2170.01	152.76	1538.09	14.20	0.01	0.10	32418106	691028
find	1705	2114.18	114.35	920.75	18.49	0.07	0.12	94631199	338600
ged	72	2026.43	28.54	317.21	71.01	0.40	0.09	1648636	10374
echo	84710	2018.23	190.14	1138.49	10.61	0.00	0.17	2926992	649200
cpio	127	1956.60	77.03	391.45	25.40	0.61	0.20	190822346	296302
maze	8	1620.42	44.80	128.25	36.17	5.60	0.35	120399	212
mail	4735	1474.38	76.92	14262.62	19.17	0.02	0.01	25719618	463748
get	1085	1358.03	37.59	234.97	36.13	0.03	0.16	31540008	178623
acctcom	165	1253.99	47.06	339.34	26.64	0.29	0.14	57405662	68949
yacc	58	1187.17	15.36	36.90	77.31	0.26	0.42	4096070	12093
col	638	1064.40	49.01	2199.00	21.72	0.08	0.02	23835395	16903
line	27184	1036.03	93.14	1941.33	11.12	0.00	0.05	925447	296142
nroff1.2	29	909.83	17.71	56.97	51.38	0.61	0.31	11459920	18802
delta	264	904.54	23.07	254.06	39.21	0.09	0.09	24219141	87164
td	175	886.19	25.74	159.73	34.43	0.15	0.16	1990177	15792
ar	1434	872.65	61.87	309.07	14.11	0.04	0.20	189858731	428871
m2findD	144	864.29	12.54	344.13	68.94	0.09	0.04	1184947	28576
rm	15319	857.97	85.65	754.20	10.02	0.01	0.11	453479	433903
acctdusg	1	819.77	39.30	170.10	20.86	39.30	0.23	1812480	39744
f77pass1	155	779.13	7.97	29.09	97.70	0.05	0.27	990027	34702
diff	786	767.31	32.77	260.27	23.41	0.04	0.13	22940094	97214

Jun 8 04:07 1979 LAST LOGIN Page 1

00-00-00	dii	00-00-00	rudd	79-06-08	adm
00-00-00	absadm	00-00-00	s10	79-06-08	agp
00-00-00	absafr	00-00-00	s2	79-06-08	als
00-00-00	abscas	00-00-00	s4	79-06-08	arf
00-00-00	absjcw	00-00-00	s5	79-06-08	cath
00-00-00	abspvg	00-00-00	s6	79-06-08	dal
00-00-00	abstbm	00-00-00	s8	79-06-08	dan
00-00-00	adm94	00-00-00	s9	79-06-08	denise
00-00-00	apb	00-00-00	scbsa	79-06-08	dp
00-00-00	archive	00-00-00	sjm	79-06-08	ds52
00-00-00	asc	00-00-00	srb	79-06-08	ech
00-00-00	badt	00-00-00	sys	79-06-08	ecp
00-00-00	btb	00-00-00	tgp	79-06-08	eric
00-00-00	bvl	00-00-00	tld	79-06-08	fld
00-00-00	bwk	00-00-00	ussc	79-06-08	fsrep1
00-00-00	chicken	00-00-00	uucpa	79-06-08	games
00-00-00	class	00-00-00	uvac	79-06-08	graf
00-00-00	cleary	00-00-00	vav	79-06-08	herb
00-00-00	cs	00-00-00	wdr	79-06-08	hoot
00-00-00	dbs	00-00-00	willa	79-06-08	hsm
00-00-00	deby	00-00-00	zooma	79-06-08	jac
00-00-00	dec	79-06-04	dws	79-06-08	jcw
00-00-00	demo	79-06-04	ewb	79-06-08	jel
00-00-00	dlt	79-06-04	kas	79-06-08	jfn
00-00-00	dmr	79-06-04	satz	79-06-08	kof
00-00-00	docs	79-06-04	uucp	79-06-08	krb
00-00-00	dug	79-06-05	bcm	79-06-08	leap
00-00-00	ellie	79-06-05	lprem	79-06-08	llc
00-00-00	fsrep2	79-06-05	s7	79-06-08	m2
00-00-00	gas	79-06-05	sccs	79-06-08	m2class
00-00-00	graphics	79-06-06	conv	79-06-08	mfp
00-00-00	hjg	79-06-06	dck	79-06-08	mhb
00-00-00	hlb	79-06-06	dmt	79-06-08	mjr
00-00-00	inst	79-06-06	emp	79-06-08	msb
00-00-00	jfm	79-06-06	pah	79-06-08	nuucp
00-00-00	jrh	79-06-06	sync	79-06-08	paul
00-00-00	ken	79-06-06	tad	79-06-08	pdw
00-00-00	lco	79-06-07	ams	79-06-08	pris
00-00-00	learn	79-06-07	bin	79-06-08	pwbcs
00-00-00	lppdw	79-06-07	dgd	79-06-08	pwbst
00-00-00	lrbb	79-06-07	haight	79-06-08	rbj
00-00-00	maj	79-06-07	hasp	79-06-08	reg
00-00-00	mar	79-06-07	jgw	79-06-08	rem
00-00-00	mash	79-06-07	leb	79-06-08	rje
00-00-00	meq	79-06-07	ljk	79-06-08	rnt
00-00-00	mifi	79-06-07	mep	79-06-08	root
00-00-00	mlc	79-06-07	nhg	79-06-08	rrr
00-00-00	mmr	79-06-07	nws	79-06-08	s1
00-00-00	mpf	79-06-07	qtroff	79-06-08	s3
00-00-00	plan	79-06-07	tbm	79-06-08	star
00-00-00	plum	79-06-07	train	79-06-08	stock
00-00-00	pvg	79-06-07	whr	79-06-08	systst
00-00-00	rakesh	79-06-07	wwe	79-06-08	teach
00-00-00	rfg	79-06-08	?	79-06-08	text
00-00-00	rlc	79-06-08	abs	79-06-08	trc
00-00-00	rrc	79-06-08	absjrk	79-06-08	vjf
				79-06-08	whm