

**NAME**

acct – process accounting file

**SYNOPSIS**

```
#include <sys/acct.h>
```

**DESCRIPTION**

If the kernel is built with the process accounting option enabled (**CONFIG\_BSD\_PROCESS\_ACCT**), then calling **acct(2)** starts process accounting, for example:

```
acct("/var/log/pacct");
```

When process accounting is enabled, the kernel writes a record to the accounting file as each process on the system terminates. This record contains information about the terminated process, and is defined in `<sys/acct.h>` as follows:

```
#define ACCT_COMM 16

typedef u_int16_t comp_t;

struct acct {
    char ac_flag;          /* Accounting flags */
    u_int16_t ac_uid;      /* Accounting user ID */
    u_int16_t ac_gid;      /* Accounting group ID */
    u_int16_t ac_tty;      /* Controlling terminal */
    u_int32_t ac_btime;    /* Process creation time
                           (seconds since the Epoch) */
    comp_t ac_untime;     /* User CPU time */
    comp_t ac_stime;      /* System CPU time */
    comp_t ac_etime;      /* Elapsed time */
    comp_t ac_mem;        /* Average memory usage (kB) */
    comp_t ac_io;         /* Characters transferred (unused) */
    comp_t ac_rw;         /* Blocks read or written (unused) */
    comp_t ac_minflt;     /* Minor page faults */
    comp_t ac_majflt;     /* Major page faults */
    comp_t ac_swaps;      /* Number of swaps (unused) */
    u_int32_t ac_exitcode; /* Process termination status
                           (see wait(2)) */
    char ac_comm[ACCT_COMM+1];
                          /* Command name (basename of last
                           executed command; null-terminated) */
    char ac_pad[X];       /* padding bytes */
};

enum {
    /* Bits that may be set in ac_flag field */
    AFORK = 0x01,        /* Has executed fork, but no exec */
    ASU = 0x02,          /* Used superuser privileges */
    ACORE = 0x08,        /* Dumped core */
    AXSIG = 0x10         /* Killed by a signal */
};
```

The `comp_t` data type is a floating-point value consisting of a 3-bit, base-8 exponent, and a 13-bit mantissa. A value, `c`, of this type can be converted to a (long) integer as follows:

```
v = (c & 0x1fff) << (((c >> 13) & 0x7) * 3);
```

The `ac_untime`, `ac_stime`, and `ac_etime` fields measure time in "clock ticks"; divide these values by

`sysconf(_SC_CLK_TCK)` to convert them to seconds.

### Version 3 Accounting File Format

Since kernel 2.6.8, an optional alternative version of the accounting file can be produced if the **CONFIG\_BSD\_PROCESS\_ACCT\_V3** option is set when building the kernel. With this option is set, the records written to the accounting file contain additional fields, and the width of `c_uid` and `ac_gid` fields is widened from 16 to 32 bits (in line with the increased size of UID and GIDs in Linux 2.4 and later). The records are defined as follows:

```
struct acct_v3 {
    char    ac_flag;    /* Flags */
    char    ac_version; /* Always set to ACCT_VERSION (3) */
    u_int16_t ac_tty;   /* Controlling terminal */
    u_int32_t ac_exitcode; /* Process termination status */
    u_int32_t ac_uid;   /* Real user ID */
    u_int32_t ac_gid;   /* Real group ID */
    u_int32_t ac_pid;   /* Process ID */
    u_int32_t ac_ppid;  /* Parent process ID */
    u_int32_t ac_btime; /* Process creation time */
    float    ac_etime;  /* Elapsed time */
    comp_t   ac_utime;  /* User CPU time */
    comp_t   ac_stime;  /* System time */
    comp_t   ac_mem;    /* Average memory usage (kB) */
    comp_t   ac_io;     /* Characters transferred (unused) */
    comp_t   ac_rw;     /* Blocks read or written
                        (unused) */
    comp_t   ac_minflt; /* Minor page faults */
    comp_t   ac_majflt; /* Major page faults */
    comp_t   ac_swaps;  /* Number of swaps (unused) */
    char    ac_comm[ACCT_COMM]; /* Command name */
};
```

## VERSIONS

The `acct_v3` structure is defined in `glibc` since version 2.6.

## CONFORMING TO

Process accounting originated on BSD. Although it is present on most systems, it is not standardized, and the details vary somewhat between systems.

## NOTES

Records in the accounting file are ordered by termination time of the process.

In kernels up to and including 2.6.9, a separate accounting record is written for each thread created using the NPTL threading library; since Linux 2.6.10, a single accounting record is written for the entire process on termination of the last thread in the process.

The `proc/sys/kernel/acct` file, described in **proc(5)**, defines settings that control the behavior of process accounting when disk space runs low.

## SEE ALSO

**lastcomm(1)**, **acct(2)**, **accton(8)**, **sa(8)**

## COLOPHON

This page is part of release 3.19 of the Linux *man-pages* project. A description of the project, and information about reporting bugs, can be found at <http://www.kernel.org/doc/man-pages/>.