NAME

collect - Collects data that describes the current system status

SYNOPSIS

/usr/sbin/collect [–] [-a ¦ -F ¦ -h ¦ -i *I*: [*PI*] ¦ -l ¦ -n*Num* ¦ -S ¦ -t ¦ -T ¦ -v ¦ -V]

/usr/sbin/collect [-C *start_time,end_time*]

/usr/sbin/collect [-D *device1,* [*device2, ... deviceN*]]

/usr/sbin/collect [-e ¦ -s [*pmdtlncfqyh*]]

/usr/sbin/collect [-f *file* [options]]

/usr/sbin/collect [-H *h d w m time* [,*how_long*]]

/usr/sbin/collect [-L *group1/vol01,* [*groupX/vol02, ... groupx/vol999*]]

/usr/sbin/collect [-M *suspend_value,resume_value*]

/usr/sbin/collect [-o [*tmfnzlq*]]

/usr/sbin/collect [-p *file1*] [-p *file2*] [-p *filen*]

/usr/sbin/collect [-p *collect_datafile* [-f *output_datafile*]]

/usr/sbin/collect [-P *pid1, ... pidN* ¦ [P *pid1, ... pidN*] ¦ [C *command1, ... commandN*] ¦ [U *user/UID1, ... user/UIDN*]]

/usr/sbin/collect [-R *NumberUnit* ...]

/usr/sbin/collect [-W *NumberUnit* ...]

OPTIONS

`– (dash)`

Directs the output from the `collect` utility to `stdout`, which is usually the screen or window from which the `collect` command was entered. This is the default behavior if no data collection file is specified using the `-f` option.

`-a`

Simultaneously displays `collect` data on the screen (`stdout`) while also recording the data in a file when the `-f` option is specified.

`-C` *`start_time,end_time`*

Extract a series of samples from a file according to the specified start time and end time for the series. The format of the time string is:

`[+]Year:Month:Day:Hour:Minute:Second`

For example: `+2010:11:18:23:30:00`.

Every time string field is optional, except the last (the Second field). If any of the optional fields are not specified, the corresponding values from the Start or End time are used.

The optional + (plus) sign at the beginning of the time string indicates that time is relative to the beginning of the data collection period. If + is not specified, the -C option indicates absolute time.

If the *start_time* argument is omitted, the start of the collection period is used. If the *end_time* argument is omitted, the end of the collection period is used.

-d

Prints debug information to stdout.

-D *device1*, [*device2, ... deviceN*]

Specifies which disks are included for data collection, using the device special filename of the disks, such as dsk3 for SCSI disk number 3. You can obtain a list of disk devices from the device directories under /dev, such as /dev/disk or /dev/tape. Use the hwmgr command to identify devices and obtain device name data. See the hwmgr(8) reference page for information on the command options.

You can also use regular expressions to specify a group of disks. For example dsk* selects all disks. For information on regular expressions, refer to the grep(8) reference page or the Programming Support Tools guide.

-e *pmdtlncfqyh*

Excludes the specified subsystems from the data collection and playback. Do not enter a space between letters when specifying options. For example, the following command specifies that only the CPU and file system data be excluded:

```
# collect -e cf
```

The option letters map to the following subsystems:

p - proc

Specifies that the Process Statistics (RSS & VSZ in MBytes) are excluded from data collection. When included, this data appears similar to the following in the output from the collect command:

```
PID   User   %CPU   RSS   VSZ    UsrTim   SysTim   IBk   OBk   Maj   Min   Command

0     root   2.1    12M   342M   0.00     0.01     0     0     0     0     kern idle

.

.

.
```

m - *m*em

Specifies that the MEMORY STATISTICS are excluded from data collection. When included, this data appears similar to the following in the output from the collect command:

```
# MEMORY STATISTICS=

(<----------- MegaBytes ------------><---------------- Pages/sec ---------------->)

Free   Swap   Act   InAc   Wire   UBC   PI   PO   Zer   Re   COW   SW   HIT   PP   ALL

135    1      44    22     40     36    0    0    7     0    0     0    0     0    0
```

d - *d*isk

Specifies that the DISK Statistics are excluded from data collection. When included, this data appears similar to the following in the output from the collect command:

```
DISK Statistics

DSK   NAME   B/T/L   R/S   RKB/S   W/S   WKB/S   AVS    AVW    ACTQ   WTQ    %BSY

 0    dsk0   0/0/0   53    431     0     5       8.49   0.00   0.46   0.00   43.09
```

t - *t*ape

Specifies that tape device data is excluded from data collection.

l - *L*SM volume

Specifies that LSM volume data is excluded from data collection. When included, this data appears similar to the following in the output from the collect command, one volume at a time:

```
LSM Volume Statistics

#VOL   NAME      R/S   RKB/S   RAVS   W/S   WKB/S   WAVS

  1    rootvol   0     0       0.00   0     12      45.62
```

n - *n*et

Specifies that the Network Statistics are excluded from data collection. When included, this data appears as follows in the output from the collect command:

```
# Network Statistics

#Cnt   NAME   Inpck   InErr   Outpck   OutErr   Coll   IKB   OKB   %BW

  2    tu0    89      0       2        0        0      10    0     0
```

c - *C*PU

Specifies that the CPU SUMMARY and CPU STATISTICS are excluded from data collection. When included, this data appears as follows in the output from the collect command:

```
CPU SUMMARY

USER   SYS    IDLE   WAIT   INTR   SYSC   CS    RUNQ   AVG5   AVG30   AVG60   FORK   VFORK

 13     16     71      0     149    492    725     0    0.13   0.05    0.01    0.30   0.00

SINGLE CPU STATISTICS

CPU    USER   SYS    IDLE   WAIT

0      13     16     71     0
```

f - *f*ile system

Specifies that the FileSystem Statistics are excluded from data collection. When included, this data appears as follows in the output from the collect command:

```
# FileSystem Statistics

#FS   Filesystem         Capacity   Free

 0    root_domain#root     128      90

 1    /proc                 0        0

 2    usr_domain#usr       700      147

 3    usr_domain#var       700      147
```

y - tt*y*

Specifies that the TTY Statistics are excluded from data collection. When included, this data appears as follows in the output from the collect command:

```
# TTY Statistics

#   In   Out    Can   Raw

    3   1489    0    3
```

h - *h*eader

This flag only applies during playback. When included, this data appears as follows in the output from the collect command:

```
#####################################################################
# OSF1 pauli.zso.corp.net V5.1 732 DEC4100 2/531MHz/512MB           #
# HOST.......pauli.zso.cpqcorp.net  Started........Tue Dec  5 11:48:11 2000 #
#                                   Seconds........976045691        #
#                                                                   #
# CPU FAMILY......21164 (EV5 core)  CPU ID.........EV5.6 (21164A)   #
# CPU EXTENSIONS..BYTE                                              #
# PLATFORM NAME...DEC4100      CPU SPEED......531 MHz               #
# SWAP SIZE.......1005 MB      Physical Mem...512 MB                #
# NUM CPUS........2            NUM DISKS......4                     #
# NUM LANS........4            NUM FSYS.......4                     #
# MAX MQUEUES.....64           NUM TAPES......0                     #
# INTERVAL........1.00         PROC_INTERVAL..1.00                  #
# UBCMAXPERCENT...100          UBCMINPERCENT..10                    #
# MAXUSERS........512          MAXUPRC........64                    #
# Delay_WBuffers..0            LSM Volumes....0                     #
#####################################################################

#### RECORD    1 (976045693:45) (Tue Dec  5 11:48:13 2000) ####
```

When the exclusion option is used, only the `RECORD` *N* line appears.

See also the `-s` option, used to specify subsystems that must be included in data collection.

`-f` *file*

Records data in the specified *file*. The argument is a path name to a file such as `/usr/users/collectdata/nov13`. By default, `collect` utility creates a compressed file and appends a `.cgz` extension to the file name that you specify. For example, the file `nov13` will be created as `nov13.cgz` by collect. See the `-o` option if you want to create an uncompressed file.

The `collect -f` command option creates a binary format file. Use the `-p` option if you want to replay the contents of the file.

See also the `-a` option, which enables you to simultaneously direct the output from the `collect` utility to `stdout` (usually the terminal from which the `collect` utility is invoked). You can also specify other data collection options with the `-f` option, such as `-s` or `-n`, to control what information is recorded in the file.

`-F`

Displays or records full process information process lines, which are longer that 80 columns. The process priorities are shown and the `RSS` and `VSZ` values are in kilobytes rather than formatted to fit into four columns. The following is example output, except that here the column widths have been manually adjusted to show the example output:

```
## RECORD 1 (943048211:40) (Fri Nov 19 16:50:11 2000) ##
```

```
## RECORD 1 (943048211:40) (Fri Nov 19 16:50:11 2000) ##

   Process Statistics (RSS & VSZ in KBytes)

PID PPID Usr  %CPU RSS  VSZ  UsrTim SysTim Pri IBk OBk Maj Min Command

 0    0   root  2.4  1984 3744  0.00   0.02   0   0   0   0   0  kern idl

 1    0   root  0.0   96   480   0.00   0.00   44   0   0   0   0    init

.
.
.
```

Compare the preceding output to the example output for `Process Statistics` shown in the entry for the `-e` option.

`-h`

Display a usage summary (help) for the `collect` command line options.

`-H` *`h d w m time`*`[,`*`how_long`*`]`

Runs the `collect` utility in historical mode. The *`how_long`* argument defines the length of time that the logs are preserved. The *`how_long`* argument is optional and if you do not specify it the log preservation default is one week.

Time variables are indicated as follows:

MM - Minute, in the range 0-59.

HH - Hour, in the range 0-24.

WD - Weekday, in the range 0-6, with 0 representing Sunday.

MD - Day of the month, in the range 1-31.

The following values for *`time`* can be specified for each argument:

`h` - `hourly at [0-59] minutes`

An hourly rollover at the specified minute.

A value of `-Hh3` will roll over the `collect` log every hour at three minutes past the hour. For example: 0.03, 1:03, 2:03, and 3:03.

`d` - `daily at [00:00-23:59] hours and minutes`

A daily roll-over at the specified hour and minute in 24–hour time format.

For example, a value of `-Hd14:2` will roll over the `collect` log every day at system time 14:02 (2:02 PM).

`w` - `weekly at [0-6@00:00-23:59] days, hours, and minutes`

A weekly roll-over at the specified day, hour and minute in seven–day 24–hour time format. A value of Zero in the day field represents Sunday.

For example, a value of `-Hw1@10:25` will roll over the `collect` log every Monday at 10:25 AM.

m - monthly at [1-31@00:00-23:59] date, hours, and minutes
A monthly roll-over at the specified date, hour and minute in 31–day 24–hour time format.

For example, a value of `-Hm3@21:15` will roll over the `collect` log every third day of the month at 21:15 (9:15 PM).

As for the value of *time*, you can specify day and week values for *how_long*. For example `-Hd14:12,2d5w` will roll over the log every day at 14:12 (2:12 PM) and keep the log for 2 days and 5 weeks.

`-i` *I*`:[`*PI*`]`
Specifies a time value in seconds for the interval (*I*) and, optionally, the process interval (*PI*). This enables you to control the rate at which data is collected from subsystems. Floating-point values are permitted.

When you use this option, the initialization message echoed by the `collect` utility is updated to confirm the value of *I*, as follows:

```
# collect -i 2:8
Initializing (2.0 seconds)(float OK)
# collect -i 5:12
PROC_INTERVAL must be evenly divisible by INTERVAL
```

Note that in the second command, an error message is displayed as the value of *PI* must always be evenly divisible by the value of *I*.

`-l`
Seek to the last valid record and print it. This is primarily used by the graphical interface to get the ending time of the collection period.

`-L` *group1/vol01*
Collect data from one disk group listed in `/dev/vol`.

`-M [suspend_value,resume_value]`
Monitor free disk space. Collect suspends writing to disk when free disk space falls below a declared threshold, and resumes when free space rises above the threshold.

In the following example, Collect suspends disk writes when free disk space falls below 250 MB, and resumes writing when free disk space rises above 300 MB:

```
# collect -M 250,300
```

-n*Num*

Select only top *Num* processes, where *Num* is an integer. This option is useful with the -S sorting option.

-o [*tmfnzlq*]

Options that enable you to control the data collection procedure:

-t (time)

Show absolute system and user time (T in data recorded for the the process subsystem), the way the ps command does. The default is to show a one second normalized delta since the last sample, thus making graphs of these time values more useful.

-m (memory)

Show 8192 byte pages instead of MB (megabytes) for absolute memory values.

-f (force)

Do not prompt before overwriting an existing output file.

-n (nice)

Do not allow the collect utility to set high scheduling priority for itself using the nice command.

-z (zipped)

Do not write a compressed (zipped) output file.

-l (lock)

Prevents the collect utility from locking its pages into memory.

-q (queue)

Causes the collect utility to use instantaneously measured queue lengths, instead of calculated averages.

-p *collect_datafile* [-f *output_datafile*]

When an existing *collect_datafile* is specified alone, the collect utility will play back the contents of the file to stdout (usually the terminal window from which the collect command was entered). You use options such as -e to filter the data read from the *collect_datafile*. As the file contents will be large, you may want to pipe the output to the more command or use the grep command to search for specific data items.

To convert data files created using previous versions of collect, use the -f option to specify an *output_datafile*.

-P *pid1 ...* ¦ [[P*pid1...*] ¦ [C*command1 ...*] ¦ [U*user/UID1 ...*]]

Specifies process identifiers for which data should be collected. The following process identifiers can be specified:

pid [*pid*,...*pid*]
: Collect data only for processes in list. Specify a percent sign (%) to include the process for the collect command).

P*pid* [*pid*,...*pid*]
: Collect data only for processes whose parent PID (PPID) is specified, or that are members of a process group (PGID) with the same ID.

C*command* [*command*,...*command*]
: Collect data only for processes whose process names contain the specified string. This can be a partial string, but must match exactly. Regular expressions are not allowed.

U*user/UID* [*user/UID*,...*user/UID*]
: Collect data only for processes owned by the specified users. User identifiers (UIDs) can be used in place of the user name. See the /etc/passwd file for a list of user account names and associated UIDs.

-R
: Specify the duration of data collection. Either of the following formats can be specified:

NumberUnit ...
: The value of *Num* is an integer. The value of *Unit* is one of the following:

  w – weeks, such as 4w for four weeks.

  d – days, such as 2d for two days.

  h – hours, such as 12h for twelve hours.

  m – minutes, such as 30m for thirty minutes.

  s – seconds, such as 45s for 45 seconds.

  Any valid combination of times can be entered, such as 4w2d6h45m20s.

[+]Year:Mon:Day:Hour:Min:Sec
: The same time format described for the -C option, except that a plus sign (+) indicates the value is relative to the current time. Without a plus sign, the value is an absolute time at which the data collection period should end.

-s *pmdtlncfqyh*
: Include the specified subsystems in data collection and playback, which can be: Proc, Mem, Disk, Tape, Lsm, Net, Cpu, Filesys, mQueue, ttY, and Header). The option letters (*p m d*...) map to these subsystems and are described under the entry for the -e option.

Do not enter a space between letters when specifying options. For example, the following command specifies that only the CPU and file system data are included:

```
# collect -s cf
```

Note that if you specify a subsystem that is not available on the local system, only a `RECORD` *N* header will be displayed. The following example shows what happens when `t` (tape) is specified, but no tape device exists on the system:

```
# collect -s t
.
.
#### RECORD  4 (943046239:0) (Fri Feb 16 16:17:19 2001) ####
#### RECORD  5 (943046249:0) (Fri Feb 16 16:17:29 2001) ####
.
.
```

`-S`

Sorts processes according to their %CPU usage (percentage of processing time used).

`-t`

Prefixes a tag (or marker) to all data lines to facilitate manipulation of data by scripts.

`-T`

Specifies only that total disk and tape throughput be recorded or displayed as the `Sum MB/sec`. All other subsystems are deselected.

`-v`

Enables verbose mode, listing the devices attached to the system as shown in the following sample output:

```
% collect -v
No objects found of type hardware/tape

    found 4 Disks, 0 Tapes
    found CPU 0 at slot [0]
    found CPU 1 at slot [1]
    max_procs = 16384
    SAMPLE: 0
    Initializing (10.0 seconds) ...
```

`-V`

Displays `collect` executable version, and if used with the `-p` option, also displays the version of the data file.

`-W` *NumberUnit ...*

Declares how often the Collect utility should write data to disk.

The argument is a compound of *number*, an integer representing the number of write instances, and *unit*, which is one or more of the time options (`w`, `d`, `h`, and `m`), such as in the following examples:

collect -W 1h

collect -W 1h30m..

The former command writes data to disk once per hour while the latter writes data to disk every 90 minutes. Writing to disk requires using the file option (f) specifying the file in which to record the data.

DESCRIPTION

The `collect` utility is a system monitoring tool that records or displays specific operating system data. Any set of the *subsystems*, such as file systems, message queue, tty, or header can be included in or excluded from data collection. You can display data at the terminal, or store it in either a compressed or uncompressed data file. Data files can be read and manipulated from the command line, or through use of command scripts.

To ensure that the `collect` utility delivers precise statistics it locks itself into memory using the page locking function `plock()`, and by default cannot be swapped out by the system. It also raises its priority using the priority function `nice()`. However, these measures should not have any impact on a system under normal load, and they should have only a minimal impact on a system under extremely high load. If required, you can disable page locking using the `-ol` command option and disable the `collect` utility's priority setting using the `-on` command option.

Some `collect` operations use kernel data that is only accessible to root. System administration practice should not involve lengthy operations as root, therefore `collect` is installed with permissions set as `04750`. This setting allows group (typically `system`) members to run `collect` with owner `setuid` permissions. If this is inappropriate in your environment, you may reset permissions to fit your needs.

Automatic Starting on a Reboot

You can configure `collect` to start automatically when the system is rebooted. This is particularly useful for continuous monitoring. To do this, use the `rcmgr` command with the `set` operation to configure the following values in `/etc/rc.config*`:

`%` **`rcmgr set COLLECT_AUTORUN 1`**

A value of 1 sets `collect` to automatically start on reboot. A value of 0 (the default) causes `collect` to not start on reboot.

`%` **`rcmgr set COLLECT_ARGS ""`**

A null value causes `collect` to start with the default values (command options) of:

-i60,120 -f /var/adm/collect.dated/collect -H d0:5,1w -W 1h -M 10,15

You can select other values.

`%` **`rcmgr set COLLECT_COMPRESSION 1`**

A value of 1 sets compression on. A value of 0 sets compression off.

See the `rcmgr`(8) reference page for more information.

Playing Back Multiple Data Files

Use the `collect` utility with the `-p` option to read multiple binary data files and play them back as one stream, with monotonically increasing sample numbers. You can also combine multiple binary input files into one binary output file, by using the `-p` option with the input files and the `-f` option with the output file.

The `collect` utility will combine input files in whatever order you specify on the command line. This means that the input files must be in strict chronological order if you want to do further processing of the combined output file. You can also combine binary input files from different systems, made at different times, with differing subsets of subsystems for which data has been collected. Filtering options such as `-e`, `-s`, `-P`, and `-D` can be used with this function.

Normalization of Data

Where appropriate, data is presented in units per second. For example, disk data such as KiloBytes transferred, or the number of transfers, is always normalized for 1 second. This happens no matter what time interval is chosen. The same is true for the following data items:

- CPU interrupts, system calls, and context switches.
- Memory pages out, pages in, pages zeroed, pages reactivated, and pages copied on write.
- Network packets in, packets out, and collisions.
- Process user and system time consumed.

Other data is recorded as a snapshot value. Examples of this are: free memory pages, CPU states, disk queue lengths, and process memory.

The Data Collection Interval

A collection interval can be specified using the `-i` followed by an integer, optionally followed (without spaces) by comma or colon and another integer. If the optional second integer is given, this is a separate time interval which applies only to the process subsystem. The process interval must be a multiple of the regular interval. Collecting process information is more taxing on system resources than are the other subsystems and is not generally needed at the same frequency. Process data also takes up most space in the binary data-file. Generally, specifying a process-interval greater than 1 will significantly decrease the load the collector places on the system being monitored.

Specifying What Data to Collect

Use the `-s` (select) option to select subsystems for inclusion in the data collection, or use the `-e` (exclude) option to exclude subsystems from the data collection.

When you are collecting process data, use the `-S` (sort) and `-n`*`X`* (number) options to sort data by percentage of CPU usage and to save only *`X`* processes. Target specific processes using the `-P`*`list`* option, where *`list`* is a list of process identifiers, comma-separated without blanks.

If there are many (greater than 100) disks connected to the system being monitored, use the `-D` option to monitor a particular set of disks.

Data Compression

The collect utility reads and writes compressed datafiles in gnuzip format. Compressed output is enabled by default but can be disabled using the -oz option. The extension .cgz is appended to the output filename, unless you specify the -oz command option. You can compress older, uncompressed datafiles using the gzip command, and the Collect utility can read the resulting files in their compressed form.

Compression during collection should not generate any additional CPU load. Because compression uses buffers and therefore does not write to disk after every sample, it makes fewer system calls and its overall impact is negligible. However, because the output is buffered there is one possible drawback. If the collect utility terminates abnormally (perhaps due to a system crash) more data samples will be lost than if compression is not used. This should not be an important consideration for most users, as you can specify how often data is written to the disk.

Specifying a Time Range from a Playback File

You can select samples from the total period of the time that data collection ran. Use the -C option to specify a start time and, optionally, an end time. The format is as follows:

[+]Year:Month:Day:Hour:Minute:Second.

The plus sign (+) indicates that the time should be interpreted as relative to the beginning of the collection period. If any of the fields are excluded from the string, the corresponding values from the start time are used in their place as the time value is parsed from right to left. Thus, field one is interpreted as Second, field two (if there is one), as Minute, and so on. For example, if the collection period is from February 16th, 2001, 16:44:03 to February 16th, 2001, 16:54:55, and you wish to extract one minute, all but minutes and seconds can be omitted from the command option: -C46:00,47:00 (from 16:46:00 to 16:47:00). However, if the collection ran overnight, it is necessary to specify the day as well. For example, when the period is February 16th, 16:44 to February 17th, 9:30, enter the following command to specify a time range from 23:00 to 1:00:

```
# -C 21:23:00:00,22:1:00:00
```

General Command Options

The following command options are useful:

Use the -a option to display simultaneous text (ascii) output to the screen while collecting to a file.

Use the -t option to prefix each data line with a unique tag. This makes it easier for your scripts to find and to extract data. Tags are superfluous if you use the perl script cfilt.

Use the -T option to shut off collection for all subsystems except disk, and only display a total MB/sec across all disks in system. Use the -s option with the -T option to override this behavior and collect data for other subsystems.

Use the -R option to terminate data collection after a specified amount of time.

All flags that can reasonably be applied to both collection and playback will work. The -P*list* filter option used during collection will collect data only for the processes you specify. During playback it will only display data for the corresponding processes. To save space in the binary data file, you can limit your collection to specific processes, specific disks, or specific subsystems. However, if you want to look at volumes of data and select different chunks

at a time, you should collect everything and later use the filter options to select data items during playback.

Disk Statistics

Note that under certain circumstances the data provided under the Disk Statistics section of the report might be only approximate. For older releases of `collect`, some data fields were zero and data in some fields could be inaccurate under certain circumstances.

Data Conversion and Filtering

The `collect` utility automatically reads older datafile versions when playing back files.

You can convert an older `collect` version datafile to the current version using the `-p` *`collect_datafile`* option with the `-f` *`file`*. During conversion you can use most command options to extract specific data from the input *`collect_datafile`*. For example:

- Use the `-s` and `-e` options to select data only from particular subsystems.
- Use the `-n`*`X`* and `-S` options to take only *`X`* processes and sort them by CPU usage.
- Use the `-D` option to select disks and the `-L` option to select LSM volumes.
- Use the `-P`, `-PC`, `-PU`, `-PP` options to select processes based on their identifiers.
- Use the `-C` option to extract data according to specified start and stop times.

Data Fields

The following table provides definitions for the data fields that you might see in any output from the `collect` utility.

Data Field	Description
Process Section	
PID	The process ID.
User	The username.
%CPU	The percent of the CPU(s) the process is currently (more or less) using.
RSS	Resident Set Size. Physical memory used by process; includes shared memory. When the -F flag is used, this value is in kilobytes, otherwise it is displayed in a compact format using 4 columns. The suffixes K, M, and G are decimal multipliers. That is, K means x 1000, M x 1000000, and G x 1000000000.
VSZ	Virtual memory used by process. The format is the same as described above for RSS.
UsrTim	The user-mode CPU time being consumed by the process. It has two modes, depending on whether the -ot option was specified. In the default mode, the value is a normalized delta, that is, how much user time has been consumed since the last sample, normalized over 1 second. If the -ot option is specified, the value is the absolute amount of user time the process has accumulated since it started, in the form Minutes:Seconds.
SysTim	The CPU time in kernel-mode being consumed by the process (see the description of UsrTime above).

Data Field	Description
Pri	The UNIX priority of the process. This is only shown when the -F option is used.
IBk	Input Block Operations. Actual file system blocks read or written.
OBk	Output Block Operations.
Maj	Major faults. Faults that were satisfied by doing I/O (going to disk).
Min	Minor faults. Faults that were satisfied from cache.
Command	The name of the running program. Arguments specified when the program was invoked are not retrieved.

Disks Section	
DSK	An index into the table that collect outputs, used for scripting.
NAME	The name of the device, specified as dsk*instance*, such as dsk23, and found in the system's /dev directory.
B/T/L	If this is a SCSI disk it contains the Bus/Target/Lun identifier, otherwise a - (dash). Use the hwmgr command to identify devices, as described in the hwmgr(8) reference page.
R/S	Reads per second.
RKB/S	KiloBytes read per second.
W/S	Writes per second.
WKB/S	KiloBytes written per second.
AVS	Average service time. The time spent actually servicing the request -no wait time (in milliseconds).
AVW	Average wait time; time spent in the wait queue. (In milliseconds).
ACTQ	The number of requests in the active queue (that is, being serviced by the disk).
WTQ	The number of requests in the wait queue (have not yet been submitted to disk).
%BSY	Percent Busy. Time spent servicing requests in interval divided by the interval.

Tapes Section	
NUM	An index for scripting.
NAME	The device name, tape*instance*, where *instance* is an integer in the range 0-256 and can be found in the /dev/tape directory. The hwmgr command can also be used to find devices. See the hwmgr(8) reference page for information on the command options.
B/T/L	The Bus/Target/Lun IDs (identifiers).
R/S	Reads per second.
RKB/S	KiloBytes read per second.
W/S	Writes per second.
WKB/S	KiloBytes written per second.

LSM Volumes Section	
VOL	Index for scripting.
NAME	Name in the form Diskgroup/Volume to ensure uniqueness.
R/S	Reads per second.
RKB/S	KiloBytes read per second.
RAVS	Average service time for reads with respect to LSM driver. (This includes disk driver wait time).
W/S	Writes per second.
WKB/S	KiloBytes written per second.
WAVS	Average service time for writes with respect to LSM driver. (Includes disk driver wait time).

CPU Summary Section	
USER...WAIT	CPU states, averaged over all CPUs.
INTR	Interrupts per second.
SYSC	System calls per second.
CS	Context switches per second.
RUNQ	Number of processes in the run queue.
AVG5,30,60	Load average over the last 5, 30, and 60 seconds.
FORK	Number of forks per second.
VFORK	Number of vforks per second.

Single CPU Section	
CPU#	Index for scripts.
USER	Percent time (ticks) spent in user-level code. This includes nice ticks.
SYS	Percent time (ticks) spent in kernel.
IDLE	Percent time (ticks) spent doing nothing.
WAIT	Idle ticks while waiting for I/O to happen.

Memory	
Free	Number of megabytes available. This is reported as pages available if you specify the -om option.
Swap	Number of megabytes (or pages) available on swap device(s).
Act	Amount of active memory in megabytes (or pages).
InAc	Amount of inactive memory in megabytes (or pages) allocated to a process, but marked as not used in greater than X seconds.
Wire	Nonswappable kernel memory in megabytes (or pages).
UBC	Megabytes (or pages) of memory used by Bufcache.
PI	Pages paged in per second.
PO	Pages paged out per second.

Zer	Pages zeroed per second (overwritten with zeroes before handing to a process).
Re	Pages reactivated (status changed from inactive to active).
COW	Copies-on-write per second.
SW	Processes swapped per second.
HIT	UBC (unified buffer cache) hits per second.
PP	UBC pages pushed (written to disk) per second.
ALL	Pages allocated by UBC per second.

Filesystem Section	
FS	Index for scripting.
Filesystem	Name of file system, or the Domain#Fileset in the case of and AdvFS file system. See the /etc/fstab file for a list of file systems present on the system.
Capacity	In megabytes.
Free	In megabytes.

Network Section	
Cnt	Index for scripting.
Name	Name of Network adaptor.
Inpck	Packets received per second.
InErr	Input error packets per second.
Outpck	Packets sent per second.
OutErr	Output error packets per second.
Coll	Collisions per second.
IKB	KiloBytes received per second.
OKB	KiloBytes sent per second.
%BW	Percent of theoretical bandwidth being used (Ethernet = 10Mbits/sec).

Message Queues Section	
ID	This is the ID according to ipcs.
Key	The key according to ipcs.
OUID	The owner UID (user identifier) of the message queue.
BYTES	The number of bytes in use for all messages in this queue.
Cnt	The number of messages in queue.
SPID	The PID (process identifier) of the last process to send a message on this queue.
RPID	The PID (process identifier) of the last process to read a message from this queue.
STIME	The time (in epoch seconds) of the last send.
RTIME	The time of the last receive.

CTIME	The creation time of this queue.

Terminal I/O Section	
In	The number of characters input.
Out	The number of characters output.
Can	Portion of input characters on the CANNON queue.
Raw	Portion of input characters on the RAW queue.

RESTRICTIONS

The following restrictions apply when using `collect` in this release:

- The average service time for raid devices is not available.
- The `collect` utility cannot dynamically recognize new devices or hardware added while collect is running. If you run collect, and then install a new disk or tape device and start using that device, `collect` cannot gather data on the newly-installed device. The same is true of any LSM volumes created on newly-installed disks. There is one exception: `collect` (data file version 15 and above) will recognize the addition or removal of CPUs.

  To resolve this problem, restart the `collect` utility after adding hardware to the system.
- Statistics for ISDN PPP connections are not available.

EXAMPLES

1. The following example shows how to run a full data collection and display the output at the terminal using the standard interval of 10 seconds:

   ```
   # collect
   ```

   This command is similar to the output monitoring commands such as `vmstat`, `iostat`, or `netstat` and the command `volstat`.

2. The following command uses the `-s` option to collect only process information in the file `sys.data`. The `-S` option specifies that the data is sorted by CPU usage, and the `-n` option specifies that the top ten processes are saved:

   ```
   # collect -sp -S -n10 -f sys.data
   Initializing (10.0 seconds)
   ```

   The message `Initializing (10.0 seconds)` indicates that data collection will be performed at 10–second intervals.

3. The following command displays the data collected in the preceding example by piping the output to the `more` command:

```
# collect -p sys.data.cgz | more
################################################################
OSF1 glop.ytx.tog.com T5.0 77.11 DEC1000 1/266MHz/256MB
HOST............glop.ytx.tog.cm  Started.<DY:MM:DT:HH:MM:SS:YR>
                                  Seconds........943298217

 CPU FAMILY......21064 (EV4 core) CPU ID.........EV4.5 (21064)
 CPU EXTENSIONS..
 PLATFORM NAME...DEC1000          CPU SPEED......266 MHz
 SWAP SIZE.......196 MB           Physical Mem...256 MB
 NUM CPUS........1                NUM DISKS......3
 NUM LANS........3                NUM FSYS.......4
 MAX MQUEUES.....64               NUM TAPES......0
 INTERVAL........10.00            PROC_INTERVAL..10.00
 UBCMAXPERCENT...100              UBCMINPERCENT..10
 MAXUSERS........256              MAXUPRC........64
 Delay_WBuffers..0                LSM Volumes....0
################################################################

#### RECORD    1 (943298227:10) (Mon Nov 22 14:17:07 2000) ####

Process Statistics (RSS & VSZ in KBytes)
PID  User %CPU RSS  VSZ UsrTim SysTim IBk OBk Maj Min   Command
   0 root 1.7  12M  342M  0.00  0.00  0   0   0   0 kernel idle
3275 root 0.3  3.3M 5.6M  0.00  0.00  0   0   0   8     collect
 482 root 0.0  2.6M 6.3M  0.00  0.00  0   0   0   0    insightd
 360 root 0.0  2.0M 4.4M  0.00  0.00  0   0   0   0   automount
.
.
.
```

Note that the preceding sample report is modified and compressed for ease of reference. It might appear wider on your terminal or in a printed report.

4. The following command uses the `-e` option to exclude filesystem data and collects data every second, except for process data, which is collected every 5 seconds. The times are set using the `-i` option.

   ```
   # collect -ef -i1,5 -f sys.data
   Initializing (1.0 seconds) ... done.
   ```

   Note that the time has changed in the initialization message.

5. The following command prints only the header section of a data file. That is the information bordered by the hash (or pound) symbol, # as shown in the sample output in Example 3:

   ```
   # collect -sh -p sys.data
   ```

6. The following command selects only the data from the network subsystem and displays it at the command prompt:

   ```
   # collect -sn
   Initializing (10.0 seconds) ... done.

   ### RECORD 1 (943045470:0) (Fri Nov 19 16:04:30 2000) ###

   #  Network Statistics
   #Cnt    Name Inpck InErr Outpck OutErr Coll  IKB  OKB %BW
      0     lo0     0     0      0      0    0    0    0   0
      1     sl0     0     0      0      0    0    0    0   0
      2     tu0    75     0      0      0    0    8    0   0
   ```

7. The following command specifies only data from the disk subsystem, and then only from specific disks identified as dsk0, dsk1, and dsk8. The disk names are determined by their device special file names in the /dev/disk directory.

   ```
   # collect -sd -Ddsk0,dsk1,dsk8
   Initializing (1.0 seconds) ... done.
   ```

   The hwmgr command can also be used to find devices. See the hwmgr(8) reference page for information on the command options.

8. The following command shows how to use the -p option to convert data files created using a previous version of the collect utility:

   ```
   # collect -p /tmp/olddata.col -f \
     /tmp/oldconverted.col
   Initializing (1.0 seconds) ... done.
   ```

FILES

/usr/sbin/collect
: The executable image.

SEE ALSO

Commands: sys_check(8), hwmgr(8)

Manuals: *System Configuration and Tuning Guide, System Administration Guide*