NAME

rctl - display and update resource limits database

SYNOPSIS

rctl [**-h**] [**-n**] [*filter ...*]
rctl -a *rule ...*
rctl -l [**-h**] [**-n**] *filter ...*
rctl -r *filter ...*
rctl -u [**-h**] *filter ...*

DESCRIPTION

When called without options, the **rctl** command writes currently defined RCTL rules to standard output.

If a *filter* argument is specified, only rules matching the filter are displayed. The options are as follows:

-a *rule*
: Add *rule* to the RCTL database.

-l *filter*
: Display rules applicable to the process defined by *filter*. Note that this is different from showing the rules when called without any options, as it shows not just the rules with subject equal to that of process, but also rules for the user, jail, and login class applicable to the process.

-r *filter*
: Remove rules matching *filter* from the RCTL database.

-u *filter*
: Display resource utilization for a subject (**process**, **user**, **loginclass** or **jail**) matching the *filter*.

-h "Human-readable" output. Use unit suffixes: Byte, Kilobyte, Megabyte, Gigabyte, Terabyte and Petabyte.

-n Display user IDs numerically rather than converting them to a user name.

Modifying rules affects all currently running and future processes matching the rule.

RULE SYNTAX

Syntax for a rule is subject:subject-id:resource:action=amount/per.

subject defines the kind of entity the rule applies to. It can be either **process**, **user**, **loginclass**,

	or **jail**.
subject-id	identifies the *subject*. It can be a process ID, user name, numerical user ID, login class name from login.conf(5), or jail name.
resource	identifies the resource the rule controls. See the *RESOURCES* section below for details.
action	defines what will happen when a process exceeds the allowed *amount*. See the *ACTIONS* section below for details.
amount	defines how much of the resource a process can use before the defined *action* triggers. Resources which limit bytes may use prefixes from expand_number(3).
per	defines what entity the *amount* gets accounted for. For example, rule "loginclass:users:vmemoryuse:deny=100M/process" means that each process of any user belonging to login class "users" may allocate up to 100MB of virtual memory. Rule "loginclass:users:vmemoryuse:deny=100M/user" would mean that for each user belonging to the login class "users", the sum of virtual memory allocated by all the processes of that user will not exceed 100MB. Rule "loginclass:users:vmemoryuse:deny=100M/loginclass" would mean that the sum of virtual memory allocated by all processes of all users belonging to that login class will not exceed 100MB.

A valid rule has all those fields specified, except for *per*, which defaults to the value of *subject*.

A filter is a rule for which one of more fields other than *per* is left empty. For example, a filter that matches every rule could be written as ":::=/", or, in short, ":". A filter that matches all the login classes would be "loginclass:". A filter that matches all defined rules for **maxproc** resource would be "::maxproc".

SUBJECTS

process	numerical Process ID
user	user name or numerical User ID
loginclass	login class from login.conf(5)
jail	jail name

RESOURCES

cputime	CPU time, in seconds
datasize	data size, in bytes
stacksize	stack size, in bytes
coredumpsize	core dump size, in bytes
memoryuse	resident set size, in bytes
memorylocked	locked memory, in bytes
maxproc	number of processes

openfiles	file descriptor table size
vmemoryuse	address space limit, in bytes
pseudoterminals	number of PTYs
swapuse	swap space that may be reserved or used, in bytes
nthr	number of threads
msgqqueued	number of queued SysV messages
msgqsize	SysV message queue size, in bytes
nmsgq	number of SysV message queues
nsem	number of SysV semaphores
nsemop	number of SysV semaphores modified in a single semop(2) call
nshm	number of SysV shared memory segments
shmsize	SysV shared memory size, in bytes
wallclock	wallclock time, in seconds
pcpu	%CPU, in percents of a single CPU core
readbps	filesystem reads, in bytes per second
writebps	filesystem writes, in bytes per second
readiops	filesystem reads, in operations per second
writeiops	filesystem writes, in operations per second

ACTIONS

deny	deny the allocation; not supported for **cputime**, **wallclock**, **readbps**, **writebps**, **readiops**, and **writeiops**
log	log a warning to the console
devctl	send notification to devd(8) using **system** = "RCTL", **subsystem** = "rule", **type** = "matched"
sig*	e.g. **sigterm**; send a signal to the offending process. See signal(3) for a list of supported signals
throttle	slow down process execution; only supported for **readbps**, **writebps**, **readiops**, and **writeiops**.

Not all actions are supported for all resources. Attempting to add a rule with an action not supported by a given resource will result in error.

EXIT STATUS

The **rctl** utility exits 0 on success, and >0 if an error occurs.

EXAMPLES

Prevent user "joe" from allocating more than 1GB of virtual memory:

rctl -a *user:joe:vmemoryuse:deny=1g*

Remove all RCTL rules:

rctl -r *:*

Display resource utilization information for jail named "www":

rctl -hu *jail:www*

Display all the rules applicable to process with PID 512:

rctl -l *process:512*

Display all rules:

rctl

Display all rules matching user "joe":

rctl *user:joe*

Display all rules matching login classes:

rctl *loginclass:*

SEE ALSO

cpuset(1), rctl(4), rctl.conf(5)

HISTORY

The **rctl** command appeared in FreeBSD 9.0.

AUTHORS

The **rctl** was developed by Edward Tomasz Napierala <*trasz@FreeBSD.org*> under sponsorship from the FreeBSD Foundation.

BUGS

Limiting **memoryuse** may kill the machine due to thrashing.

The **readiops** and **writeiops** counters are only approximations. Like **readbps** and **writebps**, they are calculated in the filesystem layer, where it is difficult or even impossible to observe actual disk device operations.

The **writebps** and **writeiops** resources generally account for writes to the filesystem cache, not to actual devices.