NAME

rtprio, **idprio** - execute, examine or modify a utility's or process's realtime or idletime scheduling priority

SYNOPSIS

[id|rt]prio
[id|rt]prio [-]*pid*
[id|rt]prio *priority command* [args]
[id|rt]prio *priority -pid*
[id|rt]prio -t *command* [args]
[id|rt]prio -t *-pid*

DESCRIPTION

The **rtprio** utility is used for controlling realtime process scheduling.

The **idprio** utility is used for controlling idletime process scheduling, and can be called with the same options as **rtprio**.

A process with a realtime priority is not subject to priority degradation, and will only be preempted by another process of equal or higher realtime priority.

A process with an idle priority will run only when no other process is runnable and then only if its idle priority is equal or greater than all other runnable idle priority processes.

Both **rtprio** or **idprio** when called without arguments will return the realtime priority of the current process.

If **rtprio** is called with 1 argument, it will return the realtime priority of the process with the specified *pid*.

If *priority* is specified, the process or program is run at that realtime priority. If **-t** is specified, the process or program is run as a normal (non-realtime) process.

If *-pid* is specified, the process with the process identifier *pid* will be modified, else if *command* is specified, that program is run with its arguments.

Priority is an integer between 0 and RTP_PRIO_MAX (usually 31). 0 is the highest priority

Pid of 0 means "the current process".

Only root is allowed to set realtime or idle priority for a process. Exceptional privileges can be granted through the mac_priority(4) policy and the realtime and idletime user groups. The sysctl(8) variable *security.bsd.unprivileged_idprio* is deprecated. If set to non-zero, it lets any user modify the idle priority of processes they own.

Note that idle priority increases the chance that a deadlock can occur if a process locks a required resource and then does not get to run.

EXIT STATUS

If **rtprio** execute a command, the exit value is that of the command executed. In all other cases, **rtprio** exits 0 on success, and 1 for all other errors.

EXAMPLES

To see which realtime priority the current process is at:

```
rtprio
```

To see which realtime priority of process 1423:

```
rtprio 1423
```

To run cron(8) at the lowest realtime priority:

```
rtprio 31 cron
```

To change the realtime priority of process 1423 to 16:

```
rtprio 16 -1423
```

To run tcpdump(1) without realtime priority:

```
rtprio -t tcpdump
```

To change the realtime priority of process 1423 to RTP_PRIO_NORMAL (non-realtime/normal priority):

```
rtprio -t -1423
```

To make depend while not disturbing other machine usage:

```
idprio 31 make depend
```

SEE ALSO

nice(1), ps(1), rtprio(2), setpriority(2), nice(3), mac_priority(4), renice(8)

HISTORY

The **rtprio** utility appeared in FreeBSD 2.0, but is similar to the HP-UX version.

AUTHORS

Henrik Vestergaard Draboel <*hvd@terry.ping.dk*> is the original author. This implementation in FreeBSD was substantially rewritten by David Greenman.

CAVEATS

You can lock yourself out of the system by placing a cpu-heavy process in a realtime priority.

BUGS

There is no way to set/view the realtime priority of process 0 (swapper) (see ps(1)).

There is in FreeBSD no way to ensure that a process page is present in memory therefore the process may be stopped for pagein (see mprotect(2), madvise(2)).

Under FreeBSD system calls are currently never preempted, therefore non-realtime processes can starve realtime processes, or idletime processes can starve normal priority processes.