NAME

cpuset **-** configure processor sets

SYNOPSIS

cpuset [**-l** *cpu-list*] [**-n** *policy:domain-list*] [**-s** *setid*] *cmd ...*
cpuset [**-l** *cpu-list*] [**-n** *policy:domain-list*] [**-s** *setid*] **-p** *pid*
cpuset [**-c**] [**-l** *cpu-list*] [**-n** *policy:domain-list*] **-C** **-p** *pid*
cpuset [**-c**] [**-l** *cpu-list*] [**-n** *policy:domain-list*] [**-j** *jail* | **-p** *pid* | **-t** *tid* | **-s** *setid* | **-x** *irq*]
cpuset **-g** [**-cir**] [**-d** *domain* | **-j** *jail* | **-p** *pid* | **-t** *tid* | **-s** *setid* | **-x** *irq*]

DESCRIPTION

The **cpuset** command can be used to assign processor sets to processes, run commands constrained to a given set or list of processors and memory domains, and query information about processor binding, memory binding and policy, sets, and available processors and memory domains in the system.

cpuset requires a target to modify or query. The target may be specified as a command, process id, thread id, a cpuset id, an irq, a jail, or a NUMA domain. Using **-g** the target's set id or mask may be queried. Using **-l** or **-s** the target's CPU mask or set id may be set. If no target is specified, **cpuset** operates on itself. Not all combinations of operations and targets are supported. For example, you may not set the id of an existing set or query and launch a command at the same time.

There are two sets applicable to each process and one private mask per thread. Every process in the system belongs to a cpuset. By default processes are started in set 1. The mask or id may be queried using **-c**. Each thread also has a private mask of CPUs it is allowed to run on that must be a subset of the assigned set. And finally, there is a root set, numbered 0, that is immutable. This last set is the list of all possible CPUs in the system and is queried using **-r**.

Most sets include NUMA memory domain and policy information. This can be inspected with **-g** and set with **-n**. This will specify which NUMA domains are visible to the process and affect where anonymous memory and file pages will be stored on first access. Files accessed first by other processes may specify conflicting policy.

When running a command it may join a set specified with **-s** otherwise a new set is created. In addition, a mask for the command may be specified using **-l**. When used in conjunction with **-c** the mask modifies the supplied or created set rather than the private mask for the thread.

The options are as follows:

-C Create a new cpuset and assign the target process to that set.

-c The requested operation should reference the cpuset available via the target specifier.

-d *domain* Specifies a NUMA domain id as the target of the operation. This can only be used to query the cpus visible in each numberd domain.

-g Causes **cpuset** to print either a list of valid CPUs or, using **-i**, the id of the target.

-i When used with the **-g** option print the id rather than the valid mask of the target.

-j *jail* Specifies a jail id or name as the target of the operation.

-l *cpu-list* Specifies a list of CPUs to apply to a target. Specification may include numbers separated by '**-**' for ranges and commas separating individual numbers. A special list of "all" may be specified in which case the list includes all CPUs from the root set.

-n *policy:domain-list*
Specifies a list of domains and allocation policy to apply to a target. Ranges may be specified as in **-l**. Valid policies include first-touch (ft), round-robin (rr), prefer and interleave (il). First-touch allocates on the local domain when memory is available. Round-robin alternates between every possible domain page at a time. The prefer policy accepts only a single domain in the set. The parent of the set is consulted if the preferred domain is unavailable. Interleave operates like round-robin with an implementation defined stripe width. See domainset(9) for more details on policies.

-p *pid* Specifies a pid as the target of the operation.

-s *setid* Specifies a set id as the target of the operation.

-r The requested operation should reference the root set available via the target specifier.

-t *tid* Specifies a thread id as the target of the operation.

-x *irq* Specifies an irq as the target of the operation.

EXIT STATUS

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

EXAMPLES

Create a new group with CPUs 0-4 inclusive and run */bin/sh* on it:

```
cpuset -c -l 0-4 /bin/sh
```

Query the mask of CPUs the <sh pid> is allowed to run on:

```
cpuset -g -p <sh pid>
```

Restrict */bin/sh* to run on CPUs 0 and 2 while its group is still allowed to run on CPUs 0-4:

```
cpuset -l 0,2 -p <sh pid>
```

Modify the cpuset */bin/sh* belongs to restricting it to CPUs 0 and 2:

```
cpuset -l 0,2 -c -p <sh pid>
```

Modify the cpuset all threads are in by default to contain only the first 4 CPUs, leaving the rest idle:

```
cpuset -l 0-3 -s 1
```

Print the id of the cpuset */bin/sh* is in:

```
cpuset -g -i -p <sh pid>
```

Move the *pid* into the specified cpuset *setid* so it may be managed with other pids in that set:

```
cpuset -s <setid> -p <pid>
```

Create a new cpuset that is restricted to CPUs 0 and 2 and move *pid* into the new set:

```
cpuset -C -c -l 0,2 -p <pid>
```

SEE ALSO

nproc(1), cpuset(2), rctl(8)

HISTORY

The **cpuset** command first appeared in FreeBSD 7.1.

AUTHORS

Jeffrey Roberson <*jeff@FreeBSD.org*>