MyWebUniversity
python
Red Hat Enterprise Linux Manual Pages
FPDF Library PDF generator
Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'podman-update.1' command

$ man podman-update.1

```
podman-update(1)          General Commands Manual          podman-update(1)

NAME

       podman-update - Updates the cgroup configuration of a given container

SYNOPSIS

       podman update [options] container

       podman container update [options] container

DESCRIPTION

       Updates  the cgroup configuration of an already existing container. The

       currently supported options are a subset of the podman  create/run  re?

       source  limits  options.  These new options are non-persistent and only

       last for the current execution of the container; the configuration will

       be  honored  on its next run.  This means that this command can only be

       executed on an already running container and the changes made  will  be

       erased the next time the container is stopped and restarted, this is to

       ensure immutability.  This command takes one argument, a container name

       or ID, alongside the resource flags to modify the cgroup.

OPTIONS

   --blkio-weight=weight

       Block IO relative weight. The weight is a value between 10 and 1000.

       This option is not supported on cgroups V1 rootless systems.

   --blkio-weight-device=device:weight

       Block IO relative device weight.

   --cpu-period=limit
```

Set the CPU period for the Completely Fair Scheduler (CFS), which is a
duration in microseconds. Once the container's CPU quota is used up, it
will not be scheduled to run until the current period ends. Defaults to
100000 microseconds.

On some systems, changing the resource limits may not be allowed for
non-root users. For more details, see https://github.com/contain?
ers/podman/blob/main/troubleshooting.md#26-running-containers-with-re?
source-limits-fails-with-a-permissions-error

This option is not supported on cgroups V1 rootless systems.

--cpu-quota=limit

Limit the CPU Completely Fair Scheduler (CFS) quota.

Limit the container's CPU usage. By default, containers run with the
full CPU resource. The limit is a number in microseconds. If a number
is provided, the container will be allowed to use that much CPU time
until the CPU period ends (controllable via --cpu-period).

On some systems, changing the resource limits may not be allowed for
non-root users. For more details, see https://github.com/contain?
ers/podman/blob/main/troubleshooting.md#26-running-containers-with-re?
source-limits-fails-with-a-permissions-error

This option is not supported on cgroups V1 rootless systems.

--cpu-rt-period=microseconds

Limit the CPU real-time period in microseconds.

Limit the container's Real Time CPU usage. This option tells the kernel
to restrict the container's Real Time CPU usage to the period speci?
fied.

This option is only supported on cgroups V1 rootful systems.

--cpu-rt-runtime=microseconds

Limit the CPU real-time runtime in microseconds.

Limit the containers Real Time CPU usage. This option tells the kernel
to limit the amount of time in a given CPU period Real Time tasks may
consume. Ex: Period of 1,000,000us and Runtime of 950,000us means that
this container could consume 95% of available CPU and leave the remain?
ing 5% to normal priority tasks.

The sum of all runtimes across containers cannot exceed the amount al?
lotted to the parent cgroup.

This option is only supported on cgroups V1 rootful systems.

--cpu-shares, -c=shares

CPU shares (relative weight).

By default, all containers get the same proportion of CPU cycles. This
proportion can be modified by changing the container's CPU share
weighting relative to the combined weight of all the running contain?
ers. Default weight is 1024.

The proportion will only apply when CPU-intensive processes are run?
ning. When tasks in one container are idle, other containers can use
the left-over CPU time. The actual amount of CPU time will vary depend?
ing on the number of containers running on the system.

For example, consider three containers, one has a cpu-share of 1024 and
two others have a cpu-share setting of 512. When processes in all three
containers attempt to use 100% of CPU, the first container would re?
ceive 50% of the total CPU time. If a fourth container is added with a
cpu-share of 1024, the first container only gets 33% of the CPU. The
remaining containers receive 16.5%, 16.5% and 33% of the CPU.

On a multi-core system, the shares of CPU time are distributed over all
CPU cores. Even if a container is limited to less than 100% of CPU
time, it can use 100% of each individual CPU core.

For example, consider a system with more than three cores. If the con?
tainer C0 is started with --cpu-shares=512 running one process, and an?
other container C1 with --cpu-shares=1024 running two processes, this
can result in the following division of CPU shares:

```
????????????????????????????????????????
?PID ? container ? CPU ? CPU share    ?
????????????????????????????????????????
?100 ? C0        ? 0   ? 100% of CPU0 ?
????????????????????????????????????????
?101 ? C1        ? 1   ? 100% of CPU1 ?
????????????????????????????????????????
```

?102 ? C1   ? 2   ? 100% of CPU2 ?

??

On some systems, changing the resource limits may not be allowed for non-root users. For more details, see https://github.com/contain?ers/podman/blob/main/troubleshooting.md#26-running-containers-with-re?source-limits-fails-with-a-permissions-error

This option is not supported on cgroups V1 rootless systems.

--cpus=number

Number of CPUs. The default is 0.0 which means no limit. This is short?hand for --cpu-period and --cpu-quota, therefore the option cannot be specified with --cpu-period or --cpu-quota.

On some systems, changing the CPU limits may not be allowed for non-root users. For more details, see https://github.com/containers/pod?man/blob/main/troubleshooting.md#26-running-containers-with-resource-limits-fails-with-a-permissions-error

This option is not supported on cgroups V1 rootless systems.

--cpuset-cpus=number

CPUs in which to allow execution. Can be specified as a comma-separated list (e.g. 0,1), as a range (e.g. 0-3), or any combination thereof (e.g. 0-3,7,11-15).

On some systems, changing the resource limits may not be allowed for non-root users. For more details, see https://github.com/contain?ers/podman/blob/main/troubleshooting.md#26-running-containers-with-re?source-limits-fails-with-a-permissions-error

This option is not supported on cgroups V1 rootless systems.

--cpuset-mems=nodes

Memory nodes (MEMs) in which to allow execution (0-3, 0,1). Only effec?tive on NUMA systems.

If there are four memory nodes on the system (0-3), use --cpuset-mems=0,1 then processes in the container will only use memory from the first two memory nodes.

On some systems, changing the resource limits may not be allowed for non-root users. For more details, see https://github.com/contain?

ers/podman/blob/main/troubleshooting.md#26-running-containers-with-re?
source-limits-fails-with-a-permissions-error

This option is not supported on cgroups V1 rootless systems.

--device-read-bps=path:rate

Limit read rate (in bytes per second) from a device (e.g. --device-
read-bps=/dev/sda:1mb).

On some systems, changing the resource limits may not be allowed for
non-root users. For more details, see https://github.com/contain?
ers/podman/blob/main/troubleshooting.md#26-running-containers-with-re?
source-limits-fails-with-a-permissions-error

This option is not supported on cgroups V1 rootless systems.

--device-read-iops=path:rate

Limit read rate (in IO operations per second) from a device (e.g. --de?
vice-read-iops=/dev/sda:1000).

On some systems, changing the resource limits may not be allowed for
non-root users. For more details, see https://github.com/contain?
ers/podman/blob/main/troubleshooting.md#26-running-containers-with-re?
source-limits-fails-with-a-permissions-error

This option is not supported on cgroups V1 rootless systems.

--device-write-bps=path:rate

Limit write rate (in bytes per second) to a device (e.g. --device-
write-bps=/dev/sda:1mb).

On some systems, changing the resource limits may not be allowed for
non-root users. For more details, see https://github.com/contain?
ers/podman/blob/main/troubleshooting.md#26-running-containers-with-re?
source-limits-fails-with-a-permissions-error

This option is not supported on cgroups V1 rootless systems.

--device-write-iops=path:rate

Limit write rate (in IO operations per second) to a device (e.g. --de?
vice-write-iops=/dev/sda:1000).

On some systems, changing the resource limits may not be allowed for
non-root users. For more details, see https://github.com/contain?
ers/podman/blob/main/troubleshooting.md#26-running-containers-with-re?

source-limits-fails-with-a-permissions-error

This option is not supported on cgroups V1 rootless systems.

--memory, -m=number[unit]

Memory limit. A unit can be b (bytes), k (kibibytes), m (mebibytes), or g (gibibytes).

Allows the memory available to a container to be constrained. If the host supports swap memory, then the -m memory setting can be larger than physical RAM. If a limit of 0 is specified (not using -m), the container's memory is not limited. The actual limit may be rounded up to a multiple of the operating system's page size (the value would be very large, that's millions of trillions).

This option is not supported on cgroups V1 rootless systems.

--memory-reservation=number[unit]

Memory soft limit. A unit can be b (bytes), k (kibibytes), m (mebibytes), or g (gibibytes).

After setting memory reservation, when the system detects memory con?tention or low memory, containers are forced to restrict their consump?tion to their reservation. So always set the value below --memory, oth?erwise the hard limit will take precedence. By default, memory reserva?tion will be the same as memory limit.

This option is not supported on cgroups V1 rootless systems.

--memory-swap=number[unit]

A limit value equal to memory plus swap. A unit can be b (bytes), k (kibibytes), m (mebibytes), or g (gibibytes).

Must be used with the -m (--memory) flag. The argument value should always be larger than that of

-m (--memory) By default, it is set to double the value of --memory.

Set number to -1 to enable unlimited swap.

This option is not supported on cgroups V1 rootless systems.

--memory-swappiness=number

Tune a container's memory swappiness behavior. Accepts an integer be?tween 0 and 100.

This flag is only supported on cgroups V1 rootful systems.

--pids-limit=limit

Tune the container's pids limit. Set to -1 to have unlimited pids for

the container. The default is 2048 on systems that support "pids"

cgroup controller.

EXAMPLEs

update a container with a new cpu quota and period

```
podman update --cpus=5 myCtr
```

update a container with all available options for cgroups v2

```
podman update --cpus 5 --cpuset-cpus 0 --cpu-shares 123 --cpuset-mems 0 --memory 1G --memory-swap 2G
--memory-reservation 2G --blkio-weight-device /dev/zero:123 --blkio-weight 123 --device-read-bps /dev/zero:10mb
--device-write-bps /dev/zero:10mb --device-read-iops /dev/zero:1000 --device-write-iops /dev/zero:1000 --pids-limit 123 ctrID
```

update a container with all available options for cgroups v1

```
podman update --cpus 5 --cpuset-cpus 0 --cpu-shares 123 --cpuset-mems 0 --memory 1G --memory-swap 2G
--memory-reservation 2G --memory-swappiness 50 --pids-limit 123 ctrID
```

SEE ALSO

podman(1), podman-create(1), podman-run(1)

HISTORY

August 2022, Originally written by Charlie Doern cdoern@redhat.com

?mailto:cdoern@redhat.com?