Rocky Enterprise Linux 9.2 Manual Pages on command 'crun.1'

$ man crun.1

crun(1) General Commands Manual crun(1)

NAME

crun - a fast and lightweight OCI runtime

SYNOPSIS

crun [global options] command [command options] [arguments...]

DESCRIPTION

crun is a command line program for running Linux containers that follow the Open Container Initiative (OCI) format.

COMMANDS

create Create a container. The runtime detaches from the container process once the container environment is created. It is necessary to successively use start for starting the container.

delete Remove definition for a container.

exec Exec a command in a running container.

list List known containers.

kill Send the specified signal to the container init process. If no signal is specified, SIGTERM is used.

ps Show the processes running in a container.

run Create and immediately start a container.

spec Generate a configuration file.

start Start a container that was previously created. A container can?

not be started multiple times.

state Output the state of a container.

pause Pause all the processes in the container.

resume Resume the processes in the container.

update Update container resource constraints.

checkpoint Checkpoint a running container using CRIU

restore Restore a container from a checkpoint

STATE

By default, when running as root user, crun saves its state under the

/run/crun directory. As unprivileged user, instead the XDG_RUNTIME_DIR

environment variable is honored, and the directory $XDG_RUN?

TIME_DIR/crun is used. The global option --root overrides this set?

ting.

GLOBAL OPTIONS

--debug Produce verbose output.

--log=LOG-DESTINATION Define the destination for the error and warning

messages generated by crun. If the error happens late in the container

init process, when crun already stopped watching it, then it will be

printed to the container stderr.

It is specified in the form BACKEND:SPECIFIER.

These following backends are supported:

? file:PATH

? journald:IDENTIFIER

? syslog:IDENTIFIER

If no backend is specified, then file: is used by default.

--log-format=FORMAT Define the format of the log messages. It can ei?

ther be text, or json. The default is text.

--no-pivot Use chroot(2) instead of pivot_root(2) when creating the

container. This option is not safe, and should be avoided.

--root=DIR Defines where to store the state for crun containers.

--systemd-cgroup Use systemd for configuring cgroups. If not speci?

fied, the cgroup is created directly using the cgroupfs backend.

--cgroup-manager=MANAGER Specify what cgroup manager must be used.

Permitted values are cgroupfs, systemd and disabled.

-?, --help Print a help list.

--usage Print a short usage message.

-V, --version Print program version

CREATE OPTIONS

crun [global options] create [options] CONTAINER

--bundle=PATH Path to the OCI bundle, by default it is the current di?

rectory.

--config=FILE Override the configuration file to use. The default

value is config.json.

--console-socket=SOCKET Path to a UNIX socket that will receive the

ptmx end of the tty for the container.

--no-new-keyring Keep the same session key

--preserve-fds=N Additional number of FDs to pass into the container.

--pid-file=PATH Path to the file that will contain the container

process PID.

RUN OPTIONS

crun [global options] run [options] CONTAINER

--bundle=BUNDLE Path to the OCI bundle, by default it is the current

directory.

--config=FILE Override the configuration file to use. The default

value is config.json.

--console-socket=SOCKET Path to a UNIX socket that will receive the

ptmx end of the tty for the container.

--no-new-keyring Keep the same session key.

--preserve-fds=N Additional number of FDs to pass into the container.

--pid-file=PATH Path to the file that will contain the container

process PID.

--detach Detach the container process from the current session.

DELETE OPTIONS

crun [global options] delete [options] CONTAINER

--force Delete the container even if it is still running.

--regex=REGEX Delete all the containers that satisfy the specified regex.

EXEC OPTIONS

crun [global options] exec [options] CONTAINER CMD

--apparmor=PROFILE Set the apparmor profile for the process.

--console-socket=SOCKET Path to a UNIX socket that will receive the ptmx end of the tty for the container.

--cwd=PATH Set the working directory for the process to PATH.

--cap=CAP Specify an additional capability to add to the process.

--detach Detach the container process from the current session.

--cgroup=PATH Specify a sub-cgroup path inside the container cgroup. The path must already exist in the container cgroup.

--env=ENV Specify an environment variable.

--no-new-privs Set the no new privileges value for the process.

--preserve-fds=N Additional number of FDs to pass into the container.

--process=FILE Path to a file containing the process JSON configura? tion.

--process-label=VALUE Set the asm process label for the process com? monly used with selinux.

--pid-file=PATH Path to the file that will contain the new process PID.

-t --tty Allocate a pseudo TTY.

**-u USERSPEC --user=USERSPEC Specify the user in the form UID[:GID].

LIST OPTIONS

crun [global options] list [options]

-q --quiet Show only the container ID.

KILL OPTIONS

crun [global options] kill [options] CONTAINER SIGNAL

--all Kill all the processes in the container.

--regex=REGEX Kill all the containers that satisfy the specified regex.

PS OPTIONS

crun [global options] ps [options]

--format=FORMAT Specify the output format. It must be either table or

json. By default table is used.

SPEC OPTIONS

crun [global options] spec [options]

-b DIR --bundle=DIR Path to the root of the bundle dir (default ".").

--rootless Generate a config.json file that is usable by an unprivi?

leged user.

UPDATE OPTIONS

crun [global options] update [options] CONTAINER

--blkio-weight=VALUE Specifies per cgroup weight.

--cpu-period=VALUE CPU CFS period to be used for hardcapping.

--cpu-quota=VALUE CPU CFS hardcap limit.

--cpu-rt-period=VALUE CPU realtime period to be used for hardcapping.

--cpu-rt-runtime=VALUE CPU realtime hardcap limit.

--cpu-share=VALUE CPU shares.

--cpuset-cpus=VALUE CPU(s) to use.

--cpuset-mems=VALUE Memory node(s) to use.

--kernel-memory=VALUE Kernel memory limit.

--kernel-memory-tcp=VALUE Kernel memory limit for TCP buffer.

--memory=VALUE Memory limit.

--memory-reservation=VALUE Memory reservation or soft_limit.

--memory-swap=VALUE Total memory usage.

--pids-limit=VALUE Maximum number of pids allowed in the container.

-r, --resources=FILE Path to the file containing the resources to up?

date.

CHECKPOINT OPTIONS

crun [global options] checkpoint [options] CONTAINER

--image-path=DIR Path for saving CRIU image files

--work-path=DIR Path for saving work files and logs

--leave-running Leave the process running after checkpointing

--tcp-established Allow open TCP connections

--ext-unix-sk Allow external UNIX sockets

--shell-job Allow shell jobs

--pre-dump Only checkpoint the container's memory without stopping the container. It is not possible to restore a container from a pre-dump. A pre-dump always needs a final checkpoint (without --pre-dump). It is possible to make as many pre-dumps as necessary. For a second pre-dump or for a final checkpoint it is necessary to use --parent-path to point crun (and thus CRIU) to the pre-dump.

--parent-path=DIR Doing multiple pre-dumps or the final checkpoint af? ter one or multiple pre-dumps requires that crun (and thus CRIU) knows the location of the pre-dump. It is important to use a relative path from the actual checkpoint directory specified via --image-path. It will fail if an absolute path is used.

--manage-cgroups-mode=MODE Specify which CRIU manage cgroup mode should be used. Permitted values are soft, ignore, full or strict. Default is soft.

RESTORE OPTIONS

crun [global options] restore [options] CONTAINER

-b DIR --bundle=DIR Container bundle directory (default ".")

--image-path=DIR Path for saving CRIU image files

--work-path=DIR Path for saving work files and logs

--tcp-established Allow open TCP connections

--ext-unix Allow external UNIX sockets

--shell-job Allow shell jobs

--detach Detach from the container's process

--pid-file=FILE Where to write the PID of the container

--manage-cgroups-mode=MODE Specify which CRIU manage cgroup mode should be used. Permitted values are soft, ignore, full or strict. Default is soft.

Extensions to OCI

run.oci.mount_context_type=context

Set the mount context type on volumes mounted with SELinux labels.

Valid context types are:

context (default)

fscontext

defcontext

rootcontext

More information on how the context mount flags works see the mount(8) man page.

run.oci.seccomp.receiver=PATH

If the annotation run.oci.seccomp.receiver=PATH is specified, the seccomp listener is sent to the UNIX socket listening on the specified path. It can also set with the RUN_OCI_SECCOMP_RECEIVER environment variable. It is an experimental feature, and the annotation will be removed once it is supported in the OCI runtime specs. It must be an absolute path.

run.oci.seccomp.plugins=PATH

If the annotation run.oci.seccomp.plugins=PLUGIN1[:PLUGIN2]... is specified, the seccomp listener fd is handled through the specified plugins. The plugin must either be an absolute path or a file name that is looked up by dlopen(3). More information on how the lookup is performed are available on the ld.so(8) man page.

run.oci.seccomp_fail_unknown_syscall=1

If the annotation run.oci.seccomp_fail_unknown_syscall is present, then crun will fail when an unknown syscall is encountered in the seccomp configuration.

run.oci.seccomp_bpf_data=PATH

If the annotation run.oci.seccomp_bpf_data is present, then crun ignores the seccomp section in the OCI configuration file and use the specified data as the raw data to the seccomp(SECCOMP_SET_MODE_FILTER) syscall. The data must be encoded in base64.

It is an experimental feature, and the annotation will be removed once it is supported in the OCI runtime specs.

run.oci.keep_original_groups=1

If the annotation run.oci.keep_original_groups is present, then crun will skip the setgroups syscall that is used to either set the additional groups specified in the OCI configuration, or to reset the list of additional groups if none is specified.

run.oci.pidfd_receiver=PATH

It is an experimental feature and will be removed once the feature is in the OCI runtime specs.

If present, specify the path to the UNIX socket that will receive the pidfd for the container process.

run.oci.systemd.force_cgroup_v1=/PATH

If the annotation run.oci.systemd.force_cgroup_v1=/PATH is present, then crun will override the specified mount point /PATH with a cgroup v1 mount made of a single hierarchy none,name=systemd. It is useful to run on a cgroup v2 system containers using older versions of systemd that lack support for cgroup v2.

Note: Your container host has to have the cgroup v1 mount already present, otherwise this will not work. If you want to run the container rootless, the user it runs under has to have permissions to this mount? point.

For example, as root:

```
mkdir /sys/fs/cgroup/systemd
mount cgroup -t cgroup /sys/fs/cgroup/systemd -o none,name=systemd,xattr
chown -R the_user.the_user /sys/fs/cgroup/systemd
```

run.oci.systemd.subgroup=SUBGROUP

Override the name for the systemd sub cgroup created under the systemd scope, so the final cgroup will be like:

```
/sys/fs/cgroup/$PATH/$SUBGROUP
```

When it is set to the empty string, a sub cgroup is not created.

If not specified, it defaults to container on cgroup v2, and to "" on cgroup v1.

e.g.

```
/sys/fs/cgroup//system.slice/foo-352700.scope/container
```

run.oci.delegate-cgroup=DELEGATED-CGROUP

If the run.oci.systemd.subgroup annotation is specified, yet another sub-cgroup is created and the container process is moved here.

If a cgroup namespace is used, the cgroup namespace is created before moving the container to the delegated cgroup.

/sys/fs/cgroup/$PATH/$SUBGROUP/$DELEGATED-CGROUP

The runtime doesn't apply any limit to the $DELEGATED-CGROUP sub-cgroup, the runtime uses only $PATH/$SUBGROUP.

The container payload fully manages $DELEGATE-CGROUP, the limits ap? plied to $PATH/$SUBGROUP still applies to $DELEGATE-CGROUP.

Since cgroup delegation is not safe on cgroup v1, this option is sup? ported only on cgroup v2.

run.oci.hooks.stdout=FILE

If the annotation run.oci.hooks.stdout is present, then crun will open the specified file and use it as the stdout for the hook processes.

The file is opened in append mode and it is created if it doesn't al? ready exist.

run.oci.hooks.stderr=FILE

If the annotation run.oci.hooks.stderr is present, then crun will open the specified file and use it as the stderr for the hook processes.

The file is opened in append mode and it is created if it doesn't al? ready exist.

run.oci.handler=HANDLER

It is an experimental feature.

If specified, run the specified handler for execing the container. The only supported values are krun and wasm.

? krun: When krun is specified, the libkrun.so shared object is loaded and it is used to launch the container using libkrun.

? wasm: If specified, run the wasm handler for container. Allows running wasm workload natively. Accepts a .wasm binary as in? put and if .wat is provided it will be automatically compiled into a wasm module. Stdout of wasm module is relayed back via crun.

run.oci.scheduler

The run.oci.scheduler annotation is used to specify the scheduling pol? icy and attributes of a process within a container.

The run.oci.scheduler annotation has the following format: POLICY[#OP? TION:VALUE]#...]

Where POLICY is the scheduling policy to set and OPTION is an optional scheduling attribute to set.

The following scheduling policies are supported:

- ? SCHED_OTHER
- ? SCHED_BATCH
- ? SCHED_IDLE
- ? SCHED_FIFO
- ? SCHED_RR
- ? SCHED_DEADLINE

The following attributes can be set:

- ? runtime=VALUE.
- ? deadline=VALUE.
- ? period=VALUE.
- ? prio=VALUE.
- ? flag_reset_on_fork=1.
- ? flag_reclaim=1.
- ? flag_dl_overrun=1.
- ? flag_keep_policy=1.
- ? flag_keep_params=1.
- ? flag_util_clamp_min=1.
- ? flag_util_clamp_max=1.

The run.oci.scheduler annotation allows you to set the scheduling pol? icy for the container process. The value of the annotation should be in the format POLICY[|OPTION][#PRIORITY], where POLICY is the name of the scheduling policy, OPTION can be SCHED_RESET_ON_FORK and PRIORITY is an optional integer priority value.

It is an experimental feature and will be removed once the feature is in the OCI runtime specs.

Please refer to sched_setattr(2) for more information.

tmpcopyup mount options

If the tmpcopyup option is specified for a tmpfs, then the path that is shadowed by the tmpfs mount is recursively copied up to the tmpfs it? self.

r$FLAG mount options

If a r$FLAG mount option is specified then the flag $FLAG is set recur?

sively for each children mount.

These flags are supported:

? "rro"

? "rrw"

? "rsuid"

? "rnosuid"

? "rdev"

? "rnodev"

? "rexec"

? "rnoexec"

? "rsync"

? "rasync"

? "rdirsync"

? "rmand"

? "rnomand"

? "ratime"

? "rnoatime"

? "rdiratime"

? "rnodiratime"

? "rrelatime"

? "rnorelatime"

? "rstrictatime"

? "rnostrictatime"

idmap mount options

If the idmap option is specified then the mount is ID mapped using the

container target user namespace. This is an experimental feature and

can change at any time without notice.

The idmap option supports a custom mapping that can be different than

the user namespace used by the container.

The mapping can be specified after the idmap option like:

idmap=uids=0-1-10#10-11-10;gids=0-100-10.

For each triplet, the first value is the start of the backing file sys?
tem IDs that are mapped to the second value on the host. The length of this mapping is given in the third value.
Multiple ranges are separated with #.
These values are written to the /proc/$PID/uid_map and /proc/$PID/gid_map files to create the user namespace for the idmapped mount.
The only two options that are currently supported after idmap are uids and gids.
When a custom mapping is specified, a new user namespace is created for the idmapped mount.
If no option is specified, then the container user namespace is used.
If the specified mapping is prepended with a '@' then the mapping is considered relative to the container user namespace. The host ID for the mapping is changed to account for the relative position of the con?
tainer user in the container user namespace.
For example, the mapping: uids=@1-3-10, given a configuration like

```
"uidMappings": [
    {
      "containerID": 0,
      "hostID": 0,
      "size": 1
    },
    {
      "containerID": 1,
      "hostID": 2,
      "size": 1000
    }
  ]
```

will be converted to the absolute value uids=1-4-10, where 4 is calcu?
lated by adding 3 (container ID in the uids= mapping) and 1 (hostID - containerID for the user namespace mapping where containerID = 1 is found).

The current implementation doesn't take into account multiple user namespace ranges, so it is the caller's responsibility to split a map? ping if it overlaps multiple ranges in the user namespace. In such a case, there won't be any error reported.

Automatically create user namespace

When running as user different than root, an user namespace is automat? ically created even if it is not specified in the config file. The current user is mapped to the ID 0 in the container, and any additional id specified in the files /etc/subuid and /etc/subgid is automatically added starting with ID 1.

CGROUP v2

Note: cgroup v2 does not yet support control of realtime processes and the cpu controller can only be enabled when all RT processes are in the root cgroup. This will make crun fail while running alongside RT pro? cesses.

If the cgroup configuration found is for cgroup v1, crun attempts a conversion when running on a cgroup v2 system.

These are the OCI resources currently supported with cgroup v2 and how they are converted when needed from the cgroup v1 configuration.

Memory controller

?OCI (x)	? cgroup 2 value (y)	? conversion	? comment ?
?limit	? memory.max	? y = x	? ?
?swap	? memory.swap.max	? y = x - memory_limit	? the swap limit ? ? ? ? ? on cgroup v1 in? ? ? ? ? cludes the mem? ? ? ? ? ory usage too ?
?reservation	? memory.low	? y = x	? ?

PIDs controller

??

?OCI (x) ? cgroup 2 value (y) ? conversion ? comment ?

??

?limit ? pids.max ? y = x ? ?

??

CPU controller

??

?OCI (x) ? cgroup 2 value (y) ? conversion ? comment ?

??

?shares ? cpu.weight ? y = (1 + ((x - ? ?

? ? ? 2) * 9999) / ? ?

? ? ? 262142) ? ?

??

? ? convert from ? ? ?

? ? [2-262144] to ? ? ?

? ? [1-10000] ? ? ?

??

?period ? cpu.max ? y = x ? period and quota ?

? ? ? ? ? are written to? ?

? ? ? ? ? gether ?

??

?quota ? cpu.max ? y = x ? period and quota ?

? ? ? ? ? are written to? ?

? ? ? ? ? gether ?

??

blkio controller

??

?OCI (x) ? cgroup 2 value (y) ? conversion ? comment ?

??

?weight ? io.bfq.weight ? y = x ? ?

??

?weight_device ? io.bfq.weight ? y = x ? ?

??

?weight ? io.weight (fallback) ? y = 1 + (x-10)*9999/990 ? convert linearly ?

? ? ? ? from [10-1000] ?

? ? ? ? to [1-10000] ?

??

?weight_device ? io.weight (fallback) ? y = 1 + (x-10)*9999/990 ? convert linearly ?

? ? ? ? from [10-1000] ?

? ? ? ? to [1-10000] ?

??

?rbps ? io.max ? y=x ? ?

??

?wbps ? io.max ? y=x ? ?

??

?riops ? io.max ? y=x ? ?

??

?wiops ? io.max ? y=x ? ?

??

cpuset controller

??

?OCI (x) ? cgroup 2 value (y) ? conversion ? comment ?

??

?cpus ? cpuset.cpus ? y = x ? ?

??

?mems ? cpuset.mems ? y = x ? ?

??

hugetlb controller

??

?OCI (x) ? cgroup 2 value (y) ? conversion ? comment ?

??

?.limit_in_bytes ? hugetlb..max ? y = x ? ?

??