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-exec.1' command

$ man podman-exec.1

podman-exec(1) General Commands Manual podman-exec(1)

NAME

podman-exec - Execute a command in a running container

SYNOPSIS

podman exec [options] container [command [arg ...]]

podman container exec [options] container [command [arg ...]]

DESCRIPTION

podman exec executes a command in a running container.

OPTIONS

--detach, -d

Start the exec session, but do not attach to it. The command will run in the background and the exec session will be automatically removed when it completes. The podman exec command will print the ID of the exec session and exit immediately after it starts.

--detach-keys=sequence

Specify the key sequence for detaching a container. Format is a single character [a-Z] or one or more ctrl-<value> characters where <value> is one of: a-z, @, ^, [, , or _. Specifying "" will disable this feature.

The default is ctrl-p,ctrl-q.

This option can also be set in containers.conf(5) file.

--env, -e=env

Set environment variables.

This option allows arbitrary environment variables that are available

for the process to be launched inside of the container. If an environment variable is specified without a value, Podman will check the host environment for a value and set the variable only if it is set on the host. As a special case, if an environment variable ending in * is specified without a value, Podman will search the host environment for variables starting with the prefix and will add those variables to the container.

--env-file=file

Read in a line-delimited file of environment variables.

--interactive, -i

When set to true, keep stdin open even if not attached. The default is false.

--latest, -l

Instead of providing the container name or ID, use the last created container. Note: the last started container could be from other users of Podman on the host machine. (This option is not available with the remote Podman client, including Mac and Windows (excluding WSL2) machines)

--preserve-fds=N

Pass down to the process N additional file descriptors (in addition to 0, 1, 2). The total FDs will be 3+N. (This option is not available with the remote Podman client, including Mac and Windows (excluding WSL2) machines)

--privileged

Give extended privileges to this container. The default is false.

By default, Podman containers are unprivileged (=false) and cannot, for example, modify parts of the operating system. This is because by default a container is only allowed limited access to devices. A "privileged" container is given the same access to devices as the user launching the container, with the exception of virtual consoles (/dev/tty\d+) when running in systemd mode (--systemd=always).

A privileged container turns off the security features that isolate the container from the host. Dropped Capabilities, limited devices, read-

only mount points, Apparmor/SELinux separation, and Seccomp filters are all disabled.

Rootless containers cannot have more privileges than the account that launched them.

--tty, -t

Allocate a pseudo-TTY. The default is false.

When set to true, Podman will allocate a pseudo-tty and attach to the standard input of the container. This can be used, for example, to run a throwaway interactive shell.

NOTE: The --tty flag prevents redirection of standard output. It com? bines STDOUT and STDERR, it can insert control characters, and it can hang pipes. This option should only be used when run interactively in a terminal. When feeding input to Podman, use -i only, not -it.

--user, -u=user[:group]

Sets the username or UID used and, optionally, the groupname or GID for the specified command. Both user and group may be symbolic or numeric.

Without this argument, the command will run as the user specified in the container image. Unless overridden by a USER command in the Con? tainerfile or by a value passed to this option, this user generally de? faults to root.

When a user namespace is not in use, the UID and GID used within the container and on the host will match. When user namespaces are in use, however, the UID and GID in the container may correspond to another UID and GID on the host. In rootless containers, for example, a user name? space is always used, and root in the container will by default corre? spond to the UID and GID of the user invoking Podman.

--workdir, -w=dir

Working directory inside the container.

The default working directory for running binaries within a container is the root directory (/). The image developer can set a different de? fault with the WORKDIR instruction. The operator can override the work? ing directory by using the -w option.

Exit Status

The exit code from podman exec gives information about why the command within the container failed to run or why it exited. When podman exec exits with a non-zero code, the exit codes follow the chroot standard, see below:

125 The error is with Podman itself

$ podman exec --foo ctrID /bin/sh; echo $?

Error: unknown flag: --foo

125

126 The contained command cannot be invoked

$ podman exec ctrID /etc; echo $?

Error: container_linux.go:346: starting container process caused "exec: \"/etc\": permission denied": OCI runtime error

126

127 The contained command cannot be found

$ podman exec ctrID foo; echo $?

Error: container_linux.go:346: starting container process caused "exec: \"foo\": executable file not found in $PATH": OCI runtime error

127

Exit code The contained command exit code

$ podman exec ctrID /bin/sh -c 'exit 3'; echo $?

3

EXAMPLES

$ podman exec -it ctrID ls

$ podman exec -it -w /tmp myCtr pwd

$ podman exec --user root ctrID ls

SEE ALSO

podman(1), podman-run(1)

HISTORY

December 2017, Originally compiled by Brent Baudebbaude@redhat.com ?mailto:bbaude@redhat.com?