Rocky Enterprise Linux 9.2 Manual Pages on command 'podman-generate-systemd.1'

$ man podman-generate-systemd.1

podman-generate-systemd(1) General Commands Manual podman-generate-systemd(1)

NAME

podman-generate-systemd - Generate systemd unit file(s) for a container or pod

SYNOPSIS

podman generate systemd [options] container|pod

DESCRIPTION

podman generate systemd will create a systemd unit file that can be used to control a container or pod. By default, the command will print the content of the unit files to stdout.

Generating unit files for a pod requires the pod to be created with an infra container (see --infra=true). An infra container runs across the entire lifespan of a pod and is hence required for systemd to manage the life cycle of the pod's main unit.

? Note: When using this command with the remote client, includ? ing Mac and Windows (excluding WSL2) machines, place the gen? erated units on the remote system. Moreover, make sure that the XDG_RUNTIME_DIR environment variable is set. If unset,

set it via export XDG_RUNTIME_DIR=/run/user/$(id -u)._

? Note: The generated podman run command contains an --sdnotify option with the value taken from the container. If the con? tainer does not have any explicitly set value or the value is set to ignore, the value conmon is used. The reason for over? riding the default value container is that almost no container workloads send notify messages. Systemd would wait for a ready message that never comes, if the value container is used for a container that does not send notify messages. The use of the default value might have been unintentional by the user, therefore the overridden default value._

Kubernetes Integration

A Kubernetes YAML can be executed in systemd via the podman-kube@.ser? vice systemd template. The template's argument is the path to the YAML file. Given a workload.yaml file in the home directory, it can be exe? cuted as follows:

$ escaped=$(systemd-escape ~/workload.yaml)

$ systemctl --user start podman-kube@$escaped.service

$ systemctl --user is-active podman-kube@$escaped.service

active

OPTIONS

--after=dependency_name

Add the systemd unit after (After=) option, that ordering dependencies between the list of dependencies and this service. This option may be specified more than once.

User-defined dependencies will be appended to the generated unit file, but any existing options such as needed or defined by default (e.g. on? line.target) will not be removed or overridden.

--container-prefix=prefix

Set the systemd unit name prefix for containers. The default is con? tainer.

--env, -e=env

Set environment variables to the systemd unit files.

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 systemd unit files.

--files, -f

Generate files instead of printing to stdout. The generated files are named {container,pod}-{ID,name}.service and will be placed in the current working directory.

Note: On a system with SELinux enabled, the generated files will inherit contexts from the current working directory. Depending on the SELinux setup, changes to the generated files using restorecon, chcon, or semanage may be required to allow systemd to access these files. Alternatively, use the -Z option when running mv or cp.

--format=format

Print the created units in specified format (json). If --files is specified the paths to the created files will be printed instead of the unit content.

--name, -n

Use the name of the container for the start, stop, and description in the unit file

--new

Using this flag will yield unit files that do not expect containers and pods to exist. Instead, new containers and pods are created based on their configuration files. The unit files are created best effort and may need to be further edited; please review the generated files carefully before using them in production.

Note that --new only works on containers and pods created directly via Podman (i.e., podman [container] {create,run} or podman pod create). It does not work on containers or pods created via the REST API or via podman kube play. For podman kube play, please use the podman-kube@.service systemd template instead.

--no-header

Do not generate the header including meta data such as the Podman ver?
sion and the timestamp.

--pod-prefix=prefix

Set the systemd unit name prefix for pods. The default is pod.

--requires=dependency_name

Set the systemd unit requires (Requires=) option. Similar to wants, but
declares a stronger requirement dependency.

--restart-policy=policy

Set the systemd restart policy. The restart-policy must be one of:
"no", "on-success", "on-failure", "on-abnormal", "on-watchdog", "on-
abort", or "always". The default policy is on-failure unless the con?
tainer was created with a custom restart policy.

Note that generating a unit without --new on a container with a custom
restart policy can lead to issues on shutdown; systemd will attempt to
stop the unit while Podman tries to restart it. It is recommended to
to create the container without --restart and use the --restart-policy
option instead when generating the unit file.

--restart-sec=time

Set the systemd service restartsec value. Configures the time to sleep
before restarting a service (as configured with restart-policy). Takes
a value in seconds.

--separator=separator

Set the systemd unit name separator between the name/id of a con?
tainer/pod and the prefix. The default is -.

--start-timeout=value

Override the default start timeout for the container with the given
value in seconds.

--stop-timeout=value

Override the default stop timeout for the container with the given
value in seconds.

--template

Add template specifiers to run multiple services from the systemd unit

file.

Note that if --new was not set to true, it is set to true by default.

However, if --new is set to false explicitly the command will fail.

--wants=dependency_name

Add the systemd unit wants (Wants=) option, that this service is (weak) dependent on. This option may be specified more than once. This option does not influence the order in which services are started or stopped. User-defined dependencies will be appended to the generated unit file, but any existing options such as needed or defined by default (e.g. on? line.target) will not be removed or overridden.

EXAMPLES

Generate and print a systemd unit file for a container

Generate a systemd unit file for a container running nginx with an al? ways restart policy and 1-second timeout to stdout. Note that the Re? quiresMountsFor option in the Unit section ensures that the container storage for both the GraphRoot and the RunRoot are mounted prior to starting the service. For systems with container storage on disks like iSCSI or other remote block protocols, this ensures that Podman is not executed prior to any necessary storage operations coming online.

```
$ podman create --name nginx nginx:latest
$ podman generate systemd --restart-policy=always -t 1 nginx
# container-de1e3223b1b888bc02d0962dd6cb5855eb00734061013ffdd3479d225abacdc6.service
# autogenerated by Podman 1.8.0
# Wed Mar 09 09:46:45 CEST 2020
[Unit]
Description=Podman container-de1e3223b1b888bc02d0962dd6cb5855eb00734061013ffdd3479d225abacdc6.service
Documentation=man:podman-generate-systemd(1)
Wants=network-online.target
After=network-online.target
RequiresMountsFor=/var/run/container/storage
[Service]
Restart=always
```

```
        ExecStart=/usr/bin/podman start de1e3223b1b888bc02d0962dd6cb5855eb00734061013ffdd3479d225abacdc6
        ExecStop=/usr/bin/podman stop
              -t 1 de1e3223b1b888bc02d0962dd6cb5855eb00734061013ffdd3479d225abacdc6
        KillMode=none
        Type=forking

PIDFile=/run/user/1000/overlay-containers/de1e3223b1b888bc02d0962dd6cb5855eb00734061013ffdd3479d225abacdc6/u
serdata/conmon.pid
        [Install]
        WantedBy=default.target
  Generate systemd unit file for a container with --new flag
    The --new flag generates systemd unit files that create and remove con?
    tainers  at service start and stop commands (see ExecStartPre and Exec?
    StopPost service actions). Such unit files are not tied to a single ma?
    chine and can easily be shared and used on other machines.
        $ sudo podman generate systemd --new --files --name bb310a0780ae
        # container-busy_moser.service
        # autogenerated by Podman 1.8.3
        # Fri Apr  3 09:40:47 EDT 2020
        [Unit]
        Description=Podman container-busy_moser.service
        Documentation=man:podman-generate-systemd(1)
        Wants=network-online.target
        After=network-online.target
        RequiresMountsFor=/var/run/container/storage
        [Service]
        Environment=PODMAN_SYSTEMD_UNIT=%n
        Restart=on-failure
        ExecStartPre=/bin/rm -f %t/%n-pid %t/%n-cid
        ExecStart=/usr/local/bin/podman run
              --conmon-pidfile %t/%n-pid
            --cidfile %t/%n-cid
            --cgroups=no-conmon
```

```
        -d
        -dit alpine
    ExecStop=/usr/local/bin/podman stop
        --ignore
        --cidfile %t/%n-cid -t 10
    ExecStopPost=/usr/local/bin/podman rm
        --ignore
        -f
        --cidfile %t/%n-cid
    PIDFile=%t/%n-pid
    KillMode=none
    Type=forking
    [Install]
    WantedBy=default.target
```

Generate systemd unit files for a pod with two simple alpine containers

Note systemctl should only be used on the pod unit and one should not start or stop containers individually via systemctl, as they are man? aged by the pod service along with the internal infra-container.

Use systemctl status or journalctl to examine container or pod unit files.

```
    $ podman pod create --name systemd-pod
    $ podman create --pod systemd-pod alpine top
    $ podman create --pod systemd-pod alpine top
    $ podman generate systemd --files --name systemd-pod
    /home/user/pod-systemd-pod.service
    /home/user/container-amazing_chandrasekhar.service
    /home/user/container-jolly_shtern.service
    $ cat pod-systemd-pod.service
    # pod-systemd-pod.service
    # autogenerated by Podman 1.8.0
    # Wed Mar 09 09:52:37 CEST 2020
    [Unit]
    Description=Podman pod-systemd-pod.service
```

Documentation=man:podman-generate-systemd(1)

Requires=container-amazing_chandrasekhar.service container-jolly_shtern.service

Before=container-amazing_chandrasekhar.service container-jolly_shtern.service

Wants=network-online.target

After=network-online.target

RequiresMountsFor=/var/run/container/storage

[Service]

Restart=on-failure

ExecStart=/usr/bin/podman start 77a818221650-infra

ExecStop=/usr/bin/podman stop

-t 10 77a818221650-infra

KillMode=none

Type=forking

PIDFile=/run/user/1000/overlay-containers/ccfd5c71a088768774ca7bd05888d55cc287698dde06f475c8b02f696a25adcd/userdata/conmon.pid

[Install]

WantedBy=default.target

Installation of generated systemd unit files.

Podman-generated unit files include an [Install] section, which carries installation information for the unit. It is used by the enable and disable commands of systemctl(1) during installation.

Once the systemd unit file is generated, install it to /etc/sys? temd/system to be run by the root user or to $HOME/.config/systemd/user for installing it as a non-root user. Enable the copied unit file or files using systemctl enable.

Note: Copying unit files to /etc/systemd/system and enabling it marks the unit file to be automatically started at boot. And similarly, copy? ing a unit file to $HOME/.config/systemd/user and enabling it marks the unit file to be automatically started on user login.

Generated systemd files.

$ podman pod create --name systemd-pod

$ podman create --pod systemd-pod alpine top

$ podman generate systemd --files --name systemd-pod

Copy all the generated files.

$ sudo cp pod-systemd-pod.service container-great_payne.service /etc/systemd/system

$ systemctl enable pod-systemd-pod.service

Created symlink /etc/systemd/system/multi-user.target.wants/pod-systemd-pod.service ? /etc/systemd/system/pod-systemd-pod.service.

Created symlink /etc/systemd/system/default.target.wants/pod-systemd-pod.service ? /etc/systemd/system/pod-systemd-pod.service.

$ systemctl is-enabled pod-systemd-pod.service

enabled

To run the user services placed in $HOME/.config/systemd/user on first login of that user, enable the service with --user flag.

$ systemctl --user enable <.service>

The systemd user instance is killed after the last session for the user is closed. The systemd user instance can be started at boot and kept running even after the user logs out by enabling lingering using

$ loginctl enable-linger <username>

Use systemctl to perform operations on generated installed unit files.

Create and enable systemd unit files for a pod using the above examples as reference and use systemctl to perform operations.

Since systemctl defaults to using the root user, all the changes using the systemctl can be seen by appending sudo to the podman cli commands.

To perform systemctl actions as a non-root user use the --user flag when interacting with systemctl.

Note: If the previously created containers or pods are using shared re? sources, such as ports, make sure to remove them before starting the generated systemd units.

$ systemctl --user start pod-systemd-pod.service

$ podman pod ps

POD ID NAME STATUS CREATED # OF CONTAINERS INFRA ID

0815c7b8e7f5 systemd-pod Running 29 minutes ago 2 6c5d116f4bbe

$ sudo podman ps # 0 Number of pods on root.

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

```
$ systemctl stop pod-systemd-pod.service
$ podman pod ps
POD ID        NAME         STATUS   CREATED         # OF CONTAINERS   INFRA ID
272d2813c798  systemd-pod  Exited   29 minutes ago  2                 6c5d116f4bbe
```

Create a simple alpine container and generate the systemd unit file with --new flag. Enable the service and control operations using the systemctl commands.

Note: When starting the container using systemctl start rather than al? tering the already running container it spins up a "new" container with similar configuration.

```
# Enable the service.
$ sudo podman ps -a
CONTAINER ID  IMAGE                          COMMAND  CREATED       STATUS    PORTS  NAMES
bb310a0780ae  docker.io/library/alpine:latest  /bin/sh  2 minutes ago  Created         busy_moser
$ sudo systemctl start container-busy_moser.service
$ sudo podman ps -a
CONTAINER ID  IMAGE                          COMMAND  CREATED       STATUS       PORTS      NAMES
772df2f8cf3b  docker.io/library/alpine:latest  /bin/sh  1 second ago  Up 1 second             distracted_albattani
bb310a0780ae  docker.io/library/alpine:latest  /bin/sh  3 minutes ago  Created                busy_moser
```

SEE ALSO

podman(1), podman-container(1), systemctl(1), systemd.unit(5), sys? temd.service(5), conmon(8)

HISTORY

April 2020, Updated details and added use case to use generated .ser? vice files as root and non-root, by Sujil Shah (sushah at redhat dot com)

August 2019, Updated with pod support by Valentin Rothberg (rothberg at redhat dot com)

April 2019, Originally compiled by Brent Baude (bbaude at redhat dot com)