MyWebUniversity
python
Red Hat Enterprise Linux Manual Pages
FPDF Library PDF generator
FPDF
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-container-checkpoint.1' command

$ man podman-container-checkpoint.1

podman-container-checkpoint(General Commands Manpodman-container-checkpoint(1)

NAME

podman-container-checkpoint - Checkpoints one or more running contain?

ers

SYNOPSIS

podman container checkpoint [options] container [container ...]

DESCRIPTION

podman container checkpoint checkpoints all the processes in one or

more containers. A container can be restored from a checkpoint with

podman-container-restore. The container IDs or names are used as input.

IMPORTANT: If the container is using systemd as entrypoint checkpoint?

ing the container might not be possible.

OPTIONS

--all, -a

Checkpoint all running containers.

The default is false.

IMPORTANT: This OPTION does not need a container name or ID as input

argument.

--compress, -c=zstd | none | gzip

Specify the compression algorithm used for the checkpoint archive cre?

ated with the --export, -e OPTION. Possible algorithms are zstd, none

and gzip.

One possible reason to use none is to enable faster creation of check?

point archives. Not compressing the checkpoint archive can result in
faster checkpoint archive creation.

The default is zstd.

--create-image=image

Create a checkpoint image from a running container. This is a standard
OCI image created in the local image store. It consists of a single
layer that contains all of the checkpoint files. The content of this
image layer is in the same format as a checkpoint created with --ex?
port. A checkpoint image can be pushed to a standard container registry
and pulled on a different system to enable container migration. In ad?
dition, the image can be exported with podman image save and inspected
with podman inspect. Inspecting a checkpoint image would display addi?
tional information, stored as annotations, about the host environment
used to do the checkpoint:

? io.podman.annotations.checkpoint.name: Human-readable name of
the original container.

? io.podman.annotations.checkpoint.rawImageName: Unprocessed
name of the image used to create the original container (as
specified by the user).

? io.podman.annotations.checkpoint.rootfsImageID: ID of the im?
age used to create the original container.

? io.podman.annotations.checkpoint.rootfsImageName: Image name
used to create the original container.

? io.podman.annotations.checkpoint.podman.version: Version of
Podman used to create the checkpoint.

? io.podman.annotations.checkpoint.criu.version: Version of CRIU
used to create the checkpoint.

? io.podman.annotations.checkpoint.runtime.name: Container run?
time (e.g., runc, crun) used to create the checkpoint.

? io.podman.annotations.checkpoint.runtime.version: Version of
the container runtime used to create the checkpoint.

? io.podman.annotations.checkpoint.conmon.version: Version of
conmon used with the original container.

? io.podman.annotations.checkpoint.host.arch: CPU architecture
of the host on which the checkpoint was created.

? io.podman.annotations.checkpoint.host.kernel: Version of Linux
kernel of the host where the checkpoint was created.

? io.podman.annotations.checkpoint.cgroups.version: cgroup ver?
sion used by the host where the checkpoint was created.

? io.podman.annotations.checkpoint.distribution.version: Version
of host distribution on which the checkpoint was created.

? io.podman.annotations.checkpoint.distribution.name: Name of
host distribution on which the checkpoint was created.

--export, -e=archive

Export the checkpoint to a tar.gz file. The exported checkpoint can be
used to import the container on another system and thus enabling con?
tainer live migration. This checkpoint archive also includes all
changes to the container's root file-system, if not explicitly disabled
using --ignore-rootfs.

--file-locks

Checkpoint a container with file locks. If an application running in
the container is using file locks, this OPTION is required during
checkpoint and restore. Otherwise checkpointing containers with file
locks is expected to fail. If file locks are not used, this option is
ignored.

The default is false.

--ignore-rootfs

If a checkpoint is exported to a tar.gz file it is possible with the
help of --ignore-rootfs to explicitly disable including changes to the
root file-system into the checkpoint archive file.

The default is false.

IMPORTANT: This OPTION only works in combination with --export, -e.

--ignore-volumes

This OPTION must be used in combination with the --export, -e OPTION.
When this OPTION is specified, the content of volumes associated with
the container will not be included into the checkpoint tar.gz file.

The default is false.

--keep, -k

Keep all temporary log and statistics files created by CRIU during

checkpointing. These files are not deleted if checkpointing fails for

further debugging. If checkpointing succeeds these files are theoreti?

cally not needed, but if these files are needed Podman can keep the

files for further analysis.

The default is false.

--latest, -l

Instead of providing the container ID or name, use the last created

container. If other methods than Podman are used to run containers such

as CRI-O, the last started container could be from either of those

methods.

The default is false.

IMPORTANT: This OPTION is not available with the remote Podman client,

including Mac and Windows (excluding WSL2) machines. This OPTION does

not need a container name or ID as input argument.

--leave-running, -R

Leave the container running after checkpointing instead of stopping it.

The default is false.

--pre-checkpoint, -P

Dump the container's memory information only, leaving the container

running. Later operations will supersede prior dumps. It only works on

runc 1.0-rc3 or higher.

The default is false.

The functionality to only checkpoint the memory of the container and in

a second checkpoint only write out the memory pages which have changed

since the first checkpoint relies on the Linux kernel's soft-dirty bit,

which is not available on all systems as it depends on the system ar?

chitecture and the configuration of the Linux kernel. Podman will ver?

ify if the current system supports this functionality and return an er?

ror if the current system does not support it.

--print-stats

Print out statistics about checkpointing the container(s). The output is rendered in a JSON array and contains information about how much time different checkpoint operations required. Many of the checkpoint statistics are created by CRIU and just passed through to Podman. The following information is provided in the JSON array:

? podman_checkpoint_duration: Overall time (in microseconds) needed to create all checkpoints.

? runtime_checkpoint_duration: Time (in microseconds) the con? tainer runtime needed to create the checkpoint.

? freezing_time: Time (in microseconds) CRIU needed to pause (freeze) all processes in the container (measured by CRIU).

? frozen_time: Time (in microseconds) all processes in the con? tainer were paused (measured by CRIU).

? memdump_time: Time (in microseconds) needed to extract all re? quired memory pages from all container processes (measured by CRIU).

? memwrite_time: Time (in microseconds) needed to write all re? quired memory pages to the corresponding checkpoint image files (measured by CRIU).

? pages_scanned: Number of memory pages scanned to determine if they need to be checkpointed (measured by CRIU).

? pages_written: Number of memory pages actually written to the checkpoint image files (measured by CRIU).

The default is false.

--tcp-established

Checkpoint a container with established TCP connections. If the check? point image contains established TCP connections, this OPTION is re? quired during restore. Defaults to not checkpointing containers with established TCP connections.

The default is false.

--with-previous

Check out the container with previous criu image files in pre-dump. It only works on runc 1.0-rc3 or higher.

The default is false.

IMPORTANT: This OPTION is not available with --pre-checkpoint.

This option requires that the option --pre-checkpoint has been used be?

fore on the same container. Without an existing pre-checkpoint, this

option will fail.

Also see --pre-checkpoint for additional information about --pre-check?

point availability on different systems.

EXAMPLES

Make a checkpoint for the container "mywebserver".

```
# podman container checkpoint mywebserver
```

Create a checkpoint image for the container "mywebserver".

```
# podman container checkpoint --create-image mywebserver-checkpoint-1 mywebserver
```

Dumps the container's memory information of the latest container into

an archive.

```
# podman container checkpoint -P -e pre-checkpoint.tar.gz -l
```

Keep the container's memory information from an older dump and add the

new container's memory information.

```
# podman container checkpoint --with-previous -e checkpoint.tar.gz -l
```

Dump the container's memory information of the latest container into an

archive with the specified compress method.

```
# podman container checkpoint -l --compress=none --export=dump.tar

# podman container checkpoint -l --compress=gzip --export=dump.tar.gz
```

SEE ALSO

podman(1), podman-container-restore(1), criu(8)

HISTORY

September 2018, Originally compiled by Adrian Reber areber@redhat.com

?mailto:areber@redhat.com?

podman-container-checkpoint(1)