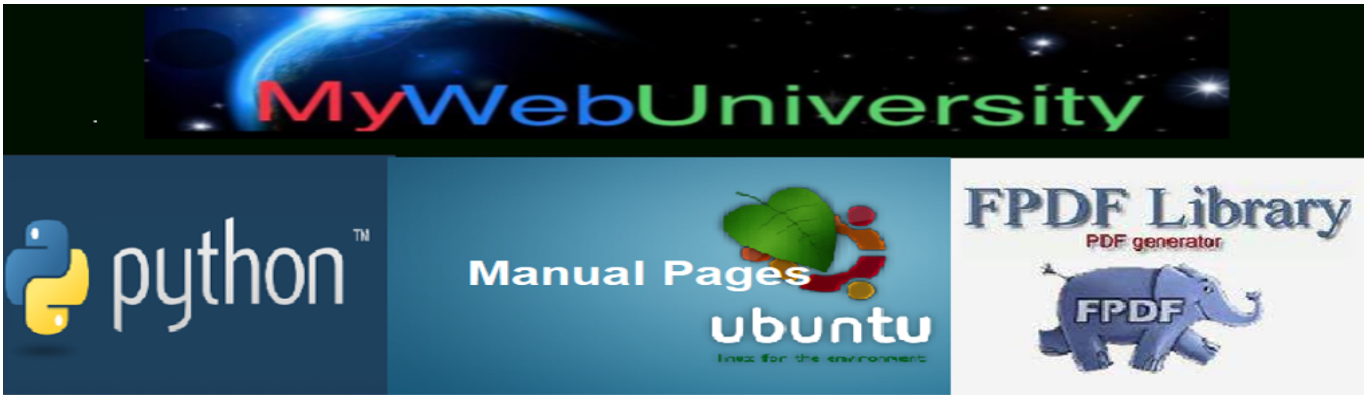

Rocky Enterprise Linux 9.2 Manual Pages on command 'podman-play-kube.1'

$ man podman-play-kube.1

podman-play-kube(1)() podman-play-kube(1)()

NAME

podman-play-kube - Create containers, pods or volumes based on Kubernetes YAML

SYNOPSIS

podman play kube [options] file.yml|-

DESCRIPTION

podman play kube will read in a structured file of Kubernetes YAML. It will then recreate the containers, pods or volumes described in the YAML. Containers within a pod are then started and the ID of the new Pod or the name of the new Volume is output. If the yaml file is specified as "-" then podman play kube will read the YAML file from stdin. Using the --down command line option, it is also capable of tearing down the pods created by a previous run of podman play kube. Ideally the input file would be one created by Podman (see podman-generate-kube(1)). This would guarantee a smooth import and expected results. Currently, the supported Kubernetes kinds are: - Pod - Deployment - PersistentVolumeClaim - ConfigMap

Kubernetes Pods or Deployments

Only two volume types are supported by play kube, the hostPath and persistentVolumeClaim volume types. For the hostPath volume type, only the default (empty), DirectoryOrCreate, Directory, FileOrCreate, File, and Socket subtypes are supported. The CharDevice and BlockDevice subtypes are not supported. Podman interprets the value of hostPath path as a file path when it contains at least one forward slash, otherwise Podman treats the value as the name of a named volume. When using a persistentVolumeClaim, the value for claimName is the name for the Podman named volume.

Note: When playing a kube YAML with init containers, the init container will be created with init type value always.

Note: hostPath volume types created by play kube will be given an SELinux private label (Z)

Note: If the :latest tag is used, Podman will attempt to pull the image from a registry. If the image was built locally with Podman or Buildah, it will have localhost as the do? main, in that case, Podman will use the image from the local store even if it has the :latest tag.

Kubernetes PersistentVolumeClaims

A Kubernetes PersistentVolumeClaim represents a Podman named volume. Only the Persis? tentVolumeClaim name is required by Podman to create a volume. Kubernetes annotations can be used to make use of the available options for Podman volumes.

- ? volume.podman.io/driver
- ? volume.podman.io/device
- ? volume.podman.io/type
- ? volume.podman.io/uid
- ? volume.podman.io/gid
- ? volume.podman.io/mount-options

Play kube is capable of building images on the fly given the correct directory layout and Containerfiles. This option is not available for remote clients yet. Consider the follow? ing excerpt from a YAML file:

```
apiVersion: v1
kind: Pod
metadata:
spec:
  containers:
  - command:
    - top
    - name: container
      value: podman
    image: foobar
```

If there is a directory named foobar in the current working directory with a file named Containerfile or Dockerfile, Podman play kube will build that image and name it foobar.

An example directory structure for this example would look like:

```
|- mykubefiles
   |- myplayfile.yaml
   |- foobar
      |- Containerfile
```

The build will consider foobar to be the context directory for the build. If there is an image in local storage called foobar, the image will not be built unless the --build flag is used.

Kubernetes ConfigMap

Kubernetes ConfigMap can be referred as a source of environment variables in Pods or De? ployments.

For example ConfigMap defined in following YAML:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: foo
data:
   FOO: bar
```

can be referred in a Pod in following way:

```
apiVersion: v1
kind: Pod
metadata:
spec:
  containers:
  - command:
    - top
    name: container-1
    image: foobar
    envFrom:
      - configMapRef:
      name: foo
      optional: false
```

and as a result environment variable FOO will be set to bar for container container-1.

OPTIONS

--authfile=path

Path of the authentication file. Default is ${XDG_RUNTIME_DIR}/containers/auth.json, which is set using podman login. If the authorization state is not found there, $HOME/.docker/config.json is checked, which is set using docker login.

Note: You can also override the default path of the authentication file by setting the REGISTRY_AUTH_FILE environment variable. export REGISTRY_AUTH_FILE=path

--build

Build images even if they are found in the local storage.

--cert-dir=path

Use certificates at path (*.crt, *.cert, *.key) to connect to the registry. (Default: /etc/containers/certs.d) Please refer to containers-certs.d(5) for details. (This option is not available with the remote Podman client)

--configmap=path

Use Kubernetes configmap YAML at path to provide a source for environment variable values within the containers of the pod.

Note: The --configmap option can be used multiple times or a comma-separated list of paths can be used to pass multiple Kubernetes configmap YAMLs.

--creds

The [username[:password]] to use to authenticate with the registry if required. If one or both values are not supplied, a command line prompt will appear and the value can be en? tered. The password is entered without echo.

--down

Tears down the pods that were created by a previous run of play kube. The pods are stopped and then removed. Any volumes created are left intact.

--ip=IP address

Assign a static ip address to the pod. This option can be specified several times when play kube creates more than one pod.

--log-driver=driver

Set logging driver for all created containers.

--mac-address=MAC address

Assign a static mac address to the pod. This option can be specified several times when play kube creates more than one pod.

--network=mode, --net

Change the network mode of the pod. The host and bridge network mode should be configured in the yaml file. Valid mode values are:

? none: Create a network namespace for the container but do not configure network interfaces for it, thus the container has no network connectivity.

? container:id: Reuse another container's network stack.

? network: Connect to a user-defined network, multiple networks should be comma-separated.

? ns:path: Path to a network namespace to join.

? private: Create a new namespace for the container. This will use the bridge mode for rootfull containers and slirp4netns for rootless ones.

? slirp4netns[:OPTIONS,...]: use slirp4netns(1) to create a user network stack. This is the default for rootless containers. It is possible to specify these ad? ditional options:

? allow_host_loopback=true|false: Allow the slirp4netns to reach the host loop? back IP (10.0.2.2, which is added to /etc/hosts as host.containers.internal for your convenience). Default is false.

? mtu=MTU: Specify the MTU to use for this network. (Default is 65520).

? cidr=CIDR: Specify ip range to use for this network. (Default is 10.0.2.0/24).

? enable_ipv6=true|false: Enable IPv6. Default is false. (Required for out? bound_addr6).

? outbound_addr=INTERFACE: Specify the outbound interface slirp should bind to (ipv4 traffic only).

? outbound_addr=IPv4: Specify the outbound ipv4 address slirp should bind to.

? outbound_addr6=INTERFACE: Specify the outbound interface slirp should bind to (ipv6 traffic only).

? outbound_addr6=IPv6: Specify the outbound ipv6 address slirp should bind to.

? port_handler=rootlesskit: Use rootlesskit for port forwarding. Default. Note: Rootlesskit changes the source IP address of incoming packets to a IP address in the container network namespace, usually 10.0.2.100. If your application re? quires the real source IP address, e.g. web server logs, use the slirp4netns port handler. The rootlesskit port handler is also used for rootless containers when connected to user-defined networks.

? port_handler=slirp4netns: Use the slirp4netns port forwarding, it is slower than rootlesskit but preserves the correct source IP address. This port handler cannot be used for user-defined networks.

--quiet, -q

Suppress output information when pulling images

--seccomp-profile-root=path

Directory path for seccomp profiles (default: "/var/lib/kubelet/seccomp"). (This option is not available with the remote Podman client)

--start=true|false

Start the pod after creating it, set to false to only create it.

--tls-verify=true|false

Require HTTPS and verify certificates when contacting registries (default: true). If ex? plicitly set to true, then TLS verification will be used. If set to false, then TLS veri? fication will not be used. If not specified, TLS verification will be used unless the tar? get registry is listed as an insecure registry in registries.conf.

--help, -h

Print usage statement

EXAMPLES

Recreate the pod and containers as described in a file called demo.yml

$ podman play kube demo.yml

52182811df2b1e73f36476003a66ec872101ea59034ac0d4d3a7b40903b955a6

Recreate the pod and containers as described in a file demo.yml sent to stdin

$ cat demo.yml | podman play kube -

52182811df2b1e73f36476003a66ec872101ea59034ac0d4d3a7b40903b955a6

Teardown the pod and containers as described in a file demo.yml

$ podman play kube --down demo.yml

Pods stopped:

52182811df2b1e73f36476003a66ec872101ea59034ac0d4d3a7b40903b955a6

Pods removed:

52182811df2b1e73f36476003a66ec872101ea59034ac0d4d3a7b40903b955a6

Provide configmap-foo.yml and configmap-bar.yml as sources for environment variables within the containers.

$ podman play kube demo.yml --configmap configmap-foo.yml,configmap-bar.yml

52182811df2b1e73f36476003a66ec872101ea59034ac0d4d3a7b40903b955a6

$ podman play kube demo.yml --configmap configmap-foo.yml --configmap configmap-bar.yml

52182811df2b1e73f36476003a66ec872101ea59034ac0d4d3a7b40903b955a6

CNI network(s) can be specified as comma-separated list using --network

$ podman play kube demo.yml --network cni1,cni2

52182811df2b1e73f36476003a66ec872101ea59034ac0d4d3a7b40903b955a6

Please take into account that CNI networks must be created first using podman-network-cre?

ate(1).

SEE ALSO

podman(1), podman-container(1), podman-pod(1), podman-generate-kube(1), podman-play(1),

podman-network-create(1), containers-certs.d(5)

HISTORY

December 2018, Originally compiled by Brent Baude (bbaude at redhat dot com)