

This document outlines the kube yaml fields that are currently supported by the **podman kube play** command. The notes were cross referenced with the Kubernetes Docs, particularly the [Greppable Reference](#), where printable view make Ctrl-F go vroom.

Not Supported ❌ Not Documented ? Supported ✅ Only with multiple nodes ☑

1. Pod Fields

PodSpec

Field	Field
✅ containers	❌ os.name
✅ initContainers	❌ volumes
❌ imagePullSecrets	
❌ enableServiceLinks	

1.1. Scheduling

[KubeSchedulerConfiguration](#): has many [profiles](#), each identified by a schedulerName. See also section on [Scheduling Profiles](#).

Field	Field	Field
☑ nodeSelector	❌ priority	? preemptionPolicy
☑ nodeName	❌ priorityClassName	? overhead
☑ schedulerName	❌ runtimeClassName	

- [PriorityClass](#)
- [RuntimeClass](#)

1.1.1. Affinity, Tolerations, Taints, Topology Spread Constraints

affinity.*	tolerations.*	topologySpreadConstraints.*
☑ nodeAffinity	☑ key	☑ maxSkew
☑ podAffinity	☑ operator	☑ topologyKey
☑ podAntiAffinity	☑ effect	☑ whenUnsatisfiable
	☑ tolerationSeconds	☑ labelSelector
	☑	☑ minDomains

1.1.2. Lifecycle

Field
✅ restartPolicy
❌ terminationGracePeriod
❌ activeDeadlineSeconds
❌ readinessGates.conditionType

1.2. DNS

Field	hostAliases.*	dnsConfig.*
✅ hostname	✅ hostAliases.hostnames	✅ nameservers
❌ setHostnameAsFQDN	✅ hostAliases.ip	✅ options.name
❌ subdomain		✅ options.value
❌ dnsPolicy		✅ searches

hostAliases
only valid for non-hostNetwork pods.

1.3. Security

Service Accounts

Field
❌ serviceAccountName
❌ automountServiceAccountToken

1.4. Namespaces, sysctl and seccomp profiles

[Emacs also supports seccomp profiles](#), if you're paranoid and for some reason compiling BPF programs isn't an undue time investment.

Field	securityContext.*
✅ hostNetwork	❌ seccompProfile.type
✅ hostPID	❌ seccompProfile.localhostProfile
✅ hostIPC	❌ sysctls.name
✅ shareProcessNamespace	❌ sysctls.value

1.5. Volume/Process Ownership and SELinux

Other securityContext.* fields for PodSpec

securityContext.*	securityContext.*
✅ runAsUser	✅ seLinuxOptions.level
❌ runAsNonRoot	✅ seLinuxOptions.role
✅ runAsGroup	✅ seLinuxOptions.type
✅ supplementalGroups	✅ seLinuxOptions.user
❌ fsGroup	❌ windowsOptions.gmsaCredentialSpec
❌ fsGroupChangePolicy	❌ windowsOptions.hostProcess
	❌ windowsOptions.runAsUserName

fsGroup/ChangePolicy
these pertain to whether a pod can change the ownership of volumes before "being exposed inside Pod." The former is a Group ID and setgid will be set, changing ownership of files created. The latter can only be set to OnRootMismatch Or Always. Not entirely sure, but I wouldn't want to have to fix whatever this solves.

2. Container Fields

Name/Image	Entry Point	Ports	Debugging
✅ name	✅ command	✅ ports.containerPort	❌ stdin
✅ image	✅ args	✅ ports.hostIP	❌ stdinOnce
✅ imagePullPolicy	✅ workingDir	✅ ports.hostPort	❌ tty
		✅ ports.name	
		✅ ports.protocol	

2.1. Env References

Either set env.value or supply a reference source with env.valueFrom, which needs a corresponding envFrom.* source

env.*	env.valueFrom.*	env.valueFrom.*	env.valueFrom.*
✅ name	✅ fieldRef	✅ configMapKeyRef.key	✅ secretKeyRef.key
✅ value	✅ resourceFieldRef	✅ configMapKeyRef.name	✅ secretKeyRef.name
		✅ configMapKeyRef.optional	✅ secretKeyRef.optional

env.valueFrom.fieldRef

- Composed of fieldPath & apiVersion ([ObjectFieldSelector](#))
- Selects a field of the pod
- Only annotations, labels, name and namespace are supported.

env.valueFrom.resourceFieldRef

- Composed of resource, containerName, divisor ([ResourceFieldSelector](#))
- Selects a resource of the container
- Only resources limits and requests (limits.cpu, limits.memory, requests.cpu and requests.memory) are currently supported.

2.2. Env Sources

envFrom.*	envFrom.*	env.*
✗ prefix	✓ configMapRef.name	✓ secretRef.name
	✓ configMapRef.optional	✓ secretRef.optional

2.3. Volumes, Devices and Resources

volumeMounts.*	volumeMounts.*	volumeMounts.*	volumeDevices.*	resources.*
✓ name	✓ mountPath	✗ subPath	✗ devicePath	✓ limits
✓ readOnly	✗ mountPropagation	✗ subPathExpr	✗ name	✓ requests

2.4. Lifecycle, Termination and Probes

LifecycleHandler	TerminationMessage	Probe
✗ lifecycle.postStart	✗ terminationMessagePath	✓ livenessProbe
✗ lifecycle.preStop	✗ terminationMessagePolicy	✗ readinessProbe
		✗ startupProbe

For the termination message, the **path** is the mounted file to which the container’s termination message will be written and the **policy** indicates how the termination message should be populated.

securityContext.*	securityContext.*
✓ securityContext.runAsUser	✓ securityContext.readOnlyRootFilesystem
✗ securityContext.runAsNonRoot	✗ securityContext.procMount
✓ securityContext.runAsGroup	✓ securityContext.privileged
	✓ securityContext.allowPrivilegeEscalation

2.4.1. Capabilities and Seccomp

securityContext.*

sc.capabilities.*	sc.seccompProfile.*
✓ add	✗ type
✓ drop	✗ localhostProfile

2.4.2. SELinux and Windows

securityContext.*

sc.seLinuxOptions.*	sc.windowsOptions.*
✓ level	✗ gmsaCredentialSpec
✓ role	✗ hostProcess
✓ type	✗ runAsUserName
✓ user	

3. PersistentVolumeClaim Fields

Field	resources.*
✗ volumeName	✗ limits
✓ storageClassName	✓ requests
✗ volumeMode	? claims.name
✓ accessModes	
✗ selector	

4. ConfigMap Fields

Field
✓ binaryData
✓ data
✗ immutable

5. Deployment Fields

Field	Field
✓ replicas	✗ minReadySeconds
✓ selector	✗ progressDeadlineSeconds
✓ template	✗ strategy.type
✗ revisionHistoryLimit	✗ strategy.rollingUpdate.maxSurge
✗ paused	✗ strategy.rollingUpdate.maxUnavailable

- (Podman) For *replicas* the actual replica count is ignored and set to 1

Created: 2023-07-14 Fri 05:26

[Validate](#)