Deploying Kubernetes Workloads

Unit 5.1.1

Deploying Kubernetes Workloads

The deployment process follows a top-down approach to defining the workload parameters, with less-common options hidden behind a link in the bottom right corner.

Basic Options

The deployment type sets the workload to be one of a Deployment, DaemonSet, StatefulSet, CronJob or Job.

The image is the full path to the image, including the external registry, if needed. If no registry is provided, the image will be pulled from Docker Hub. If no tag is provided, Rancher will use the latest tag.

Caveats Around Using "latest"

We recommend that your workloads always specify a specific tag other than latest. When using latest, Rancher sets the imagePullPolicy to Always. This has some unexpected consequences.

1. Kubernetes will always try to pull a copy of the image when the Pod starts, even if a copy of the image is present on the node. It's impossible for Kubernetes to know if there is a more recent version of the latest tag, so it will try to pull it. If the image is hosted in an external registry that is currently unavailable, the Pod will not start.
2. If the Pod restarts for any reason, and if Kubernetes pulls a new version, it's possible that you will be running multiple Pods with different versions of the application.
3. The latest tag will point to the latest build, and that build could be a different major or minor version of the application. This could have different configuration options, different defaults, or any number of other things that can cause your application to fail unexpectedly.

To avoid these issues, ***always*** specify a tag to run. This sets the imagePullPolicy to IfNotPresent, and the behavior of the workload becomes much more predictable.

Adding Ports

Ports that are added in the deployment form automatically deploy Services of the indicated type. This saves time over having to deploy a workload and then go and deploy a service.

The exception to this is with resources that are deployed via kubectl or via direct YAML import. Even if the YAML indicates container ports, Rancher won't automatically create the services.

Environment Variables

You can add environment variables directly, and like other key/value input areas in Rancher, you can paste multiple lines of key=value pairs and they will fill each field.

You can also add environment variables from another source, such as a ConfigMap or Secret, a field from the final Pod manifest, or a resource from one of the containers in the Pod.

When using the field (Pod) or the resource (container) resources, you're using what Kubernetes refers to as the Downward API.

Node Scheduling

Node scheduling allows you to define a set of rules that will choose a node according to the labels that match the rules.

You can require that all provided labels exist on the node, that any of the provided labels exist on the node, or that any of the provided labels ***should*** exist on the node. In the last case, if no node exists with those labels, Kubernetes will pick an available node from any that are present.

An alternative way to configure scheduling is with Taints and Tolerations. If you Taint a host, Kubernetes will not schedule resources to it unless your workload has a Toleration for that Taint.

You can also set a custom scheduler and priority for the workload. Kubernetes will use that scheduler to decide where to run the workload.

Health Check

If you don't configure a health check, Kubernetes will restart the container when it fails. If your application might fail without crashing the container, configure a health check so that Kubernetes will know when the application needs to be restarted.

If you only define a single health check, Rancher will use it for both the liveness and the readiness check. If you wish to define a separate liveness check, you can do so by following the link on the right.

Volumes

One of the best features in the Rancher workload deployment screen is the ease with which you can define and attach volumes to your workloads.

You can add an ephemeral volume, a new or persistent volume claim, or bind-mount a directory from the node. You can also attach a Secret, a ConfigMap, or a certificate as a volume.

Scaling/Upgrade Policy

When upgrading Pods, you can control how Kubernetes performs the rollout. If you're doing a rolling update, you can control how many Pods it starts at a time, and whether it starts new pods before killing old pods or vice versa.

If you choose to do a custom rollout, you can specify valued for maxSurge and maxUnavailable that define how many Pods start over

the desired number and how many Pods can be in an unavailable state during the update process.

Advanced Options

These options are available if you select the link in the bottom right corner, beneath the main options.

Command

The items under command control how the container's main process starts. You can override the entrypoint and command, change the working directory or the effective user id and group id for the process, and set the console and restart options. These map to the corresponding container runtime startup commands.

Changing the console output will affect Rancher's ability to collect stdout/stderr log data from the container.

You can also set the stop timeout for the container. If Kubernetes asks the container to stop, and if this much time passes and the container is still running, Kubernetes will kill the container without waiting for a graceful shutdown.

Networking

The networking options control basic networking functionality within the container. You can set a hostname and domain, configure entries for /etc/hosts and override the DNS configuration. You can also tell the container to use the host’s network space.

Security and Host Config

This section controls security-related aspects of the container. You can define if the image will be fetched on every start, if the container will run in privileged mode, or if it can request more privileges than its parent process. You can define if the process is allowed to run as root or if the root filesystem should be read-only.

This section is also where you'll define resource reservations and limits for CPU and RAM. You can also reserve NVIDIA GPUs, if you're running on a system with a configuration that supports it.

Finally, at the bottom of the section, you can add and remove capabilities to create an environment with the exact set of permissions you require.

Importing YAML Directly

If you have YAML that you'd like to deploy, you can use the **Import YAML** button to load this into Rancher. Rancher will deploy the resources according to the YAML.

When performing the import, you can define the scope as Cluster, Project, or Namespace.

If you use the Cluster scope, Rancher will create namespaces that don't exist, but those namespaces won't be automatically added to the current project. You can also import resources that aren't scoped to a namespace.

If you use the Project scope, you're asked to specify a default namespace. Resources which do not specify a namespace will be added to that default, and if any namespace doesn't exist, Rancher will create it and add it to the current project.

Finally, if you choose the Namespace scope, everything will be imported into the specified namespace. If any resource references a different namespace, the import will fail.

Working With Sidecars

Pods can have more than one container inside of them. The two patterns for this are an initContainer that does environmental setup and runs before the main container, and a standard container that runs alongside the primary container, sharing the same network and storage configuration.

If your pod should have multiple containers, you'll deploy the main container first and then edit it to add a sidecar. This sidecar inherits many of the Pod definitions from the first container's configuration, but it can have its own command, health check, volumes, command and security and host configuration.

When a pod has sidecars, the containers in each pod are collected together when displayed in the UI. You can interact with them individually.

When you edit a workload with sidecars, the UI presents you with a selection of the containers that are available for you to edit. You can edit the primary container, or you can choose to edit or delete sidecar containers.

References

Deploying Workloads - https://rancher.com/docs/rancher/v2.x/en/k8s-in-rancher/workloads/deploy-workloads/

Adding a Sidecar - https://rancher.com/docs/rancher/v2.x/en/k8s-in-rancher/workloads/rollback-workloads/

Kubernetes Downward API - https://kubernetes.io/docs/tasks/inject-data-application/downward-api-volume-expose-pod-information/