Running Development Projects as a DevOps Administrator or Developer

VMware vSphere Integrated Containers 1.5.x

vmware

Table of Contents

Running Development Projects as a DevOps Administrator or Developer

Running Development Projects as a DevOps Administrator or Developer provides information about how to use VMware vSphere® Integrated Containers™ Management Portal as a user with the DevOps Administrator or Developer role.

Product version: 1.5

This documentation applies to all 1.5.x releases.

Intended Audience

This information is intended for vSphere Integrated Containers users who have the DevOps Administrator or Developer roles in vSphere Integrated Containers Management Portal. Knowledge of container technology and Docker is assumed.

VMware, Inc. 3401 Hillview Ave. Palo Alto, CA 94304

www.vmware.com

DevOps Administrator and Developer Tasks

DevOps Administrator and Developer Tasks

As a developer, you can perform the following tasks in vSphere Integrated Containers Management Portal:

- Add networks to containers. For more information, see Creating New Networks.
- Add volumes to containers. For more information, see Creating New Volumes.
- Provision containers. For more information, see Provisioning Container VMs.
- Create templates and provision containers from templates. For more information, see Creating Templates.
- Create application templates and deploy applications. For more information, see Creating Applications.
- View the repositories and virtual container hosts for your project. For more information, see Viewing Library and Logs.

As a Devops Administrator, in addition to developer tasks, you can perform the following tasks in vSphere Integrated Containers Management Portal:

- Add developers and viewers to projects and assign other DevOps administrators. For more information, see Add Viewers, Developers, or DevOps Administrators to Projects.
- Change project configurations, such as making the project registry public, changing deployment security settings, and enabling vulnerability scanning. For more information, see Configure Project Settings.

For information about the tasks that the Management Portal Administator can perform such as, creating projects and adding hosts, see vSphere Integrated Containers Management Portal Administration.

What to Do Next

Start Working with Projects.

Working with Projects

In vSphere Integrated Containers, you create different projects to which you assign users, add volumes and networks. You can provision containers, create templates, and applications.

1. Create a Project
2. Assign Users to a Project
3. Assign Projects to a User
4. Add a Container Host
5. Create New Networks
6. Create New Volumes
7. Provision Container VMs
8. Create Templates and Applications
9. View Library and Logs

The first four tasks can only be performed by users with the Management Portal Administrator role. The other tasks can be performed by users with DevOps Administrator or Developer roles.

Adding Container Hosts

As a Management Portal administrator, you can add existing Docker hosts or vSphere Integrated Containers virtual container hosts (VCHs) to projects. After adding the hosts, you can provision containers, view live stats and manage the hosts in the Management Portal.

You must only add a given VCH to one project at a time. Adding the same VCH to multiple projects can lead to conflicts if the registry lists and content trust setttings are different in the different projects.

For more information about adding container hosts, see Add Container Hosts to Projects

What to Do Next

Start Creating New Networks and Creating New Volumes.

Creating New Networks

You can create and attach network configurations to containers, container templates, and applications. You can create a bridge network or a container network.

You can dissociate a network from a container by deleting it.

For more information about container networks and how to configure them, see Configure Container Networks.

Procedure

1. In the management portal, navigate to **Deployments** > **Networks** and click **+Network**.
2. On the Create Network page, select the **Advanced** check box to access all available settings.
3. Configure the following settings:
   - **Name**. Enter a name for the network.
   - **IPAM config**. Enter subnet, IP range, and gateway values that are unique to this network configuration. They must not overlap with any other networks on the same container host.
   - **Custom Properties**. Optionally specify custom properties for the new network configuration.

     For example, you can specify the following properties:

     - `bridge.default_bridge`: `true`,
     - `bridge.enable_icc`: `true`,
     - `bridge.enable_ip_masquerade`: `true`,
     - `bridge.host_binding_ipv4`: `0.0.0.0`,
     - `bridge.name`: `docker0`,
     - `driver.mtu`: `9001`
   - **Hosts**. Specify the virtual container host (VCH) that you want to create the network in.
4. Click **Create**.

Result

The new network is created and you can provision containers on it.

Creating New Volumes

You can create, modify, and attach volume configurations to containers and container templates. When you create a volume, it is added to the volume datastores that exist on the virtual container host.

You can also configure volume drivers. Volume drivers allow you to store volumes on remote hosts or cloud providers, to encrypt the contents of volumes, or to add other functionality. Configure one of the following volume drivers:

- local - The default built-in `local` driver. Volumes created by this driver have a local scope, which means that they can be accessed only by containers on the same host.
- vsphere - The default driver for vSphere Integrated Containers. `local` is an alias for `vsphere`.
- Third-party plugins - The Management Portal does not support third-party volume plugins officially but it is possible to create and use volumes based on such plugins.

For more information about the volumes, see Virtual Container Host Storage Capacity.

Procedure

1. In the management portal, navigate to **Deployments** > **Volumes** and click **+Volume**.
2. On the Create Volume page, select the **Advanced** check box to access all available settings.
3. Configure the following settings:
   - **Name**. Enter a name for the volume. For example, `pgdata`.
   - **Driver**. The volume driver that you want to use for containers.
   - **Driver Options**. Enter the capacity in megabytes, gigabytes, or terabytes. For example, enter **Option** as `capacity` and **Value** as `10G`.
   - **Custom Properties**. Optionally specify custom properties for the new volume configuration.
   - **Hosts**. Select the host to use the new volume.
4. Click **Create**.

Result

The new volume is created and you can provision containers that access that volume.

Provisioning Container VMs in the Management Portal

You can provision containers or container VMs from the management portal depending on the target host. If your target host is a VCH, you provision container VMs. If your target host is a Docker host, you provision standard containers.

You can customize your deployment by using the available settings. You can either provision your configured container or save it as a template. Saving the configuration as a template allows you deploy multiple containers with the same configuration.

IMPORTANT: vSphere Integrated Containers Management Portal allows you to provision containers from the registries that are included in the lists of global registries that the Management Portal Administrator configures, or project registries that the DevOps administrator configures. However, if the vSphere administrator deployed a VCH with whitelist mode enabled, and if the whitelist on the VCH is more restrictive than the global and project registry lists, you can only provision containers from the registries that the VCH permits in its whitelist, even if the VCH is included in a project that permits other registries. For more information, see VCH Whitelists and Registry Lists in vSphere Integrated Containers Management Portal in *vSphere Integrated Containers for vSphere Administrators*.

You can provision containers and create templates from images.

Procedure

1. In the management portal, navigate to **Deployments** > **Containers** and click **+Container**.
2. On the Provision a Container page, configure the following settings:
    - Basic configuration
    - Network configuration
    - Storage configuration
    - Policy configuration
    - Environment configuration
    - Health configuration
    - Logging configuration
3. Click **PROVISION** to provision the container with the configured settings. Click **SAVE AS TEMPLATE** to save the configured container as a template.

For information about the container configuration, see Container Provisioning Options Reference

Container Provisioning Options Reference

When you create containers, configure the following settings:

Basic Configuration

Configure the basic configuration of the container on the **Basic** tab of the Provision a Container page.

Configure the following settings:

- **Image**. The image that you want to instantiate the container from.
- **Name**. The name of the container in the project.
- **Commands**. The command array that must execute when the container starts.
- **Links**. The link to containers in another service. Specify a service name and a link alias. For example, you can link your container to a database service that runs in another container. You can specify `db` as the **Service** and `database` as the **Alias**.

Network Configuration

Configure the network settings of the container on the **Network** tab of the Provision a Container page.

Configure the following settings:

- **Port Bindings**. A list of the exposed container ports and the host port that they should bind to.
- **Publish All Ports**. Select this option to publish all ports exposed by the container.
- **Hostname**. Specify the host name of the container. Host name is the DNS name of the system.
- **Network mode**. The networking mode of the container. Select one of the following options:
  - **Bridge**. The default network.
  - **None**. Select this option to indicate that the container is a standalone container.
  - **Host**. Select this option if you want the container to use the networking stack of the virtual container host (VCH). In this case, both the container and the VCH will have the same networing stack.

Storage Configuration

Configure the volume settings of the container on the **Storage** tab of the Provision a Container page.

Configure the following settings:

- **Volumes**. The volume name on the VCH and container structure of the volume. You must specify a volume name or an absolute path. The container field is mandatory and must contain an absolute path.

  For example, enter **Host** as `pgdata` and **Container** as `/var/lib/postgresql/data`.

- **Read Only**. Select this option to configure your volume as read only. For example, if you have an application that contains a Web and database service and the Web service shares its volume with the database service, you might want to configure the volume as read only.
- **Volumes From**. A list of volumes to inherit from another container.
- **Working Directory**. The working directory for the commands to run in.

Policy Configuration

You can create container clusters by using Policy settings to specify cluster size.

When you configure a cluster, a specified number of containers are provisioned. Requests are load balanced among all containers in the cluster. You can modify the cluster size on a provisioned container or application to increase or decrease the size of the cluster by one. When you modify the cluster size at runtime, all affinity filters and placement rules are considered.

For example, if you require three NGINX containers to serve a web application, specify **Cluster Size** as `3` Three containers are provisioned and a load balancer automatically load balances requests among the three containers.

Configure the following cluster settings on the **Policy** tab of the Provision a Container page:

- **Cluster Size**. The number of nodes that you want to provision.
- **Restart Policy**. The restart behavior that should be applied when the container exits. You can select one of the following options:
  - **None**. Default behavior.
  - **On-failure**. Indicates that the container must restart only when the process running on it fails. If you select this, you must specify the maximum number of restarts.
  - **Always**. Indicates that the container must restart irrespective of the exit code of the process it is running.
- **Max Restarts**. The maximum number of times that the container tries to restart when it fails.
- **CPU shares**. An integer value that specifies the CPU shares for this container in relation to the other container VMs in the VCH resource pool.
- **Memory Limit**. The quantity of memory for use by the VCH resource pool. This limit also applies to the container VMs that run in the VCH resource pool. Specify the memory reservation value in MB.
- **Memory Swap Limit**. The total amount of RAM that the container must use. When the container runs out of RAM, it swaps to disk or physical storage.
- **Affinity Constraints**. Specify VM-Host affinity rules either as a requirement (must/must not rules) or a preference (should/should not rules).

For more information, see Virtual Container Host Compute Capacity.

Environment Configuration

When you configure a container, on the **Environment** tab, you can add environment variables.

Configure the following properties:

- **Environment Variables**. Configure the variables and values that you want to associate with the container. For example, if you are creating a PostgreSQL container, you enter `POSTGRES_PASSWORD` in **Name** and the password in **Value**.
- **Custom Properties**. Specify the attributes of containers that you want to provision.

For information about using Docker environment variables, see Environment variables in Compose in the Docker documentation.

Health Configuration

You can configure a health check method to update the status of a container based on custom criteria. vSphere Integrated Containers uses its own implementation of health checks and not the standard Docker implementation.

You can use HTTP or TCP protocols when executing a command on the container. You can also specify a health check method.

Configure the following health checks settings on the **Health Config** tab of the Provision a Container page:

- **Mode**. Configure one of the following modes:
  - **None**. Default. No health checks are configured.
  - **HTTP**. If you select HTTP, configure the **URL Path** and port for the container. Provide an API to access and an HTTP method and version to use. The API is relative and you do not need to enter the address of the container.

    You can also specify a timeout period for the operation and set health thresholds. For example, a healthy threshold of 2 means that two consecutive successful calls must occur for the container to be considered healthy and in the RUNNING status. An unhealthy threshold of 2 means that two unsuccessful calls must occur for the container to be considered unhealthy and in the ERROR status. For all the states in between the healthy and unhealthy thresholds, the container status is DEGRADED.
  - **TCP connection**. If you select TCP connection, you must only enter a port for the container. The health check attempts to establish a TCP connection with the container on the provided port. You can also specify a timeout value for the operation and set healthy or unhealthy thresholds as with HTTP.
  - **Command**. If you select Command, you must enter a command to be run on the container. The success of the health check is determined by the exit status of the command.
- **Ignore health check on provision**. You can enable a health check as part of the provisioning process for a container. By default, health checks are not performed during provisioning. Deselect this check box to require at least one successful health check before a container can be considered successfully provisioned.
- **Autoredeploy**. When a container returns an ERROR status, you can configure an automated redeploy for that container by selecting the **Autoredeploy** check box.

Logging Configuration

Configure the logging mechanism of the container on the **Log Config** tab of the Provision a Container page.

Configure the following settings:

- **Driver**. The logging driver that you want to use for the container VM. For example, `json-file`.
- **Options**. The options to configure for the logging driver you select. For example, you can set the following names and corresponding values:
  - `max-size`：`10m`,
  - `max-file`：`3`,
  - `labels`：`production_status`,
  - `env`：`os,customer`

Example of Provisioning a Single Container

This section illustrates how you can provision a PostgreSQL container using templates.

Prerequisities

Verify that you have perfomed the following steps:

- Deployed a virtual container host (VCH).
- Have a vCenter Server Single Sign-On user account with vSphere administrator privileges, or a user account that has been granted the Management Portal Administrator role in vSphere Integrated Containers.
- Created a project and assigned users to the project.
- Added the container host to the project.

Create a Volume

Create a volume called `pgdata`.

1. In the management portal, navigate to **Deployments** > **Volumes** and click **+Volume**.
2. On the Create Volume page, select the **Advanced** check box to access all available settings.
3. Configure the following settings:
   - **Name**. Enter `pgdata` as the volume name.
   - **Hosts**. Select the host from the list.
4. Click **Create**.

Create a Network

1. In the management portal, navigate to **Deployments** > **Networks** and click **+Network**.
2. On the Create Network page, configure the following settings:
   - **Name**. Enter `datanet` as the network name.
   - **Hosts**. Select the host from the list.
3. Click **Create**.

Create a Template

Create a template and add the `postgres` container to it.

1. In the management portal, navigate to **Library** > **Templates** and click **+Template**.
2. On the Create a Template page, enter the container name, for example, `Postgres-container` and click **Proceed**.
3. In the Edit Template page, click **Add Container**.
4. In the Add Container Definition page, select the `library/postgres` container and click **Continue**.

Configure the Template

1. In the Edit Container Defintion page, configure the basic details:
    i. **Image**. The image that you want to instantiate the container from. This displays `registry.hub.docker.com/library/postgres`. Select the version. For example, `9.6`.
    ii. **Name**. Displays the name that you entered, `Postgres-container`.
2. On the Network tab, configure the following:
    i. Select **Publish All Ports**.
    ii. In the list under **Networks**, select **Add Network**.
    iii. Select the **Existing** checkbox and click in the search field under **Name** to see a list of added networks. Select `datanet` from the list
3. On the Storage tab, in **Volumes**, enter `pgdata` as **Host** and `/var/lib/postgresql/data` as **Container**.
4. On the Policy tab, configure the following:
    i. Select `Always` under **Restart Policy**.
    ii. Enter `2` for **CPU Shares**.
    iii. Enter `4 GB` for **Memory Limit**.
5. On the Environment tab, configure **Environment Variables**. Enter `POSTGRES_PASSWORD` in **Name** and the password in **Value**.
6. Click **Add** to add the container.

Provision the Template

Once you configure the template, the container and the network and volume that you have configured appear in the Edit Template page.

Click the  icon on the right hand top corner of the page to provision `Postgres-container`.

Creating Templates and Applications

You can use templates to provision individual vSphere Integrated Container VMs or standard containers, and multi-container application deployments in the management portal.

Viewing Library and Logs

The vSphere Integrated Containers Management Portal allows you to view the list of available repositories and activity logs.

- Built-in Repositories
- Repositories
- Recent Activity

Built-in Repositories

The Management Portal allows you to provision containers from project registries that the DevOps administrator configures.

You can browse repositories to see the different tags applied to images in the repository. You can also delete a repository or a tag in a repository.

To view a list of built-in images that are available in the project, navigate to **Library** > **Built-in Repositories**.

Select a repository to perform the following tasks:

- Edit description. Edit the repository description in the **Info** tab.
- Scan. vSphere Integrated Containers uses the open source project Clair to scan images for known vulnerabilities. You can run a vulnerability scan on all images, on a per-project level, or on individual images. For more information, see Vulnerability Scanning
- Copy Digest. You can pull images via image digest. Digest is a sha256 content-addressable identifier of an image. It represents a layer or group of layers of the iamge of the image. When pulling an image by digest, you specify the version of an image to pull. You can see the digest of an image in the ouput after pulling it. For example, `sha256:5b7ecd9d3e7ae1923ad7a1861fbeecccc23ddeb209cea69ae5d823ff90f6a2c2`. When you click `Copy Digest`, you copy the layer representation of an image that you build and can reuse it while building another image.
- Copy Pull command. You can copy the `docker pull` command for the image in the **Images** tab.
- View Vulnerability Log. Consists of a list of vulnerabilities that Clair has found in your image while scanning it.

Repositories

The Management Portal allows you to provision containers from the registries that are included in the lists of global registries that the Management Portal Administrator configures.

To view a the list navigate to **Library** > **Repositories**.

You can provision container or container VMs from these registries.

Recent Activity

You can view the following logs under the **Recent Activity**:

- Recent Activity. A list of all the operations performed on the Managament Portal such as, creation or deletion of containers, networks, and so on.
- Event logs. The list of API calls and responses when any of the requests fail.