VMware vSphere Integrated Containers Engine Installation

vSphere Integrated Containers Engine 0.5.5

vmware®

Table of Contents

vSphere Integrated Containers Engine Installation

vSphere Integrated Containers Engine Installation provides information about how to install and configure VMware vSphere Integrated Containers Engine.

Product version: 0.5.5

NOTE This book is a work in progress.

For an introduction to vSphere Integrated Containers Engine and descriptions of its main components, see *vSphere Integrated Containers Engine for vSphere Administrators*.

Intended Audience

This information is intended for anyone who wants to install, configure, and get started with using vSphere Integrated Containers Engine. The information is written for experienced VMware vSphere® administrators who are familiar with virtual machine technology and datacenter operations. Knowledge of container technology and Docker is assumed.

Overview of vSphere Integrated Containers Engine

vSphere Integrated Containers Engine provides developers the portability, speed, and agility of using enterprise-class containers, and provide IT Ops the management, security, and visibility they require to run workloads in production.

vSphere Integrated Containers Engine enables IT teams to run traditional and container workloads side-by-side on existing infrastructure seamlessly.

Using constructs from the Open Container Initiative to map Docker containers to vSphere infrastructure, vSphere Integrated Containers Engine containers are provisioned as virtual machines, offering the same security and functionality of virtual machines in ESXi.

A virtual container host is compatible with standard Docker client tools and backed by a pool of resources to accommodate applications.

From a developer's perspective, vSphere Integrated Containers Engine is a seamless Docker interface for containers with a vSphere back end. Developers can deploy, test, and run container processes faster in the same environment as traditional applications.

Comparing vSphere Integrated Containers Engine and Traditional Container Hosts

vSphere Integrated Containers Engine provisions containers as virtual machines, rather than in virtual machines.

Traditional Container Host

A traditional container host is a virtual machine running a Linux OS with the necessary libraries, kernel version, and daemon installed. The container host has a fixed amount of memory and vCPU resource used by the containers provisioned into it.

The hypervisor provides hardware virtualization of the entire container host VM, one or more VMDKs providing local disk for the OS, one or more vNICs to provide network connectivity for the OS and possibly paravirtualization capabilities allowing the containers to directly access hypervisor infrastructure.

vSphere Integrated Containers Engine

vSphere Integrated Containers Engine containers run as virtual machines. The virtual container host is not a VM, but a vApp, which is a kind of resource pool. It is an abstract dynamic resource boundary defined and controlled by vSphere into which you can provision container VMs. The virtual container host can be a subset of a physical host or a subset of a cluster of hosts.

A one to one coupling exists between a container and a virtual machine. A container image is attached to the VM as a disk, the VM is either booted or forked from the kernel ISO, then the containerVM chroots into the container filesystem, effectively becoming the container.

Installing vSphere Integrated Containers Engine

You install vSphere Integrated Containers Engine by using a command line installer, `vic-machine`, that deploys virtual container hosts to VMware ESXi™ hosts or VMware vCenter Server® instances. You connect Docker clients to the virtual container hosts and use the Docker clients to work with containers. You use your vSphere environment to manage the container VMs and container images.

- Environment Prerequisites for vSphere Integrated Containers Engine Installation
- vSphere Integrated Containers Engine Networking Overview
- Contents of the vSphere Integrated Containers Engine Binaries

Environment Prerequisites for vSphere Integrated Containers Engine Installation

Before you install vSphere Integrated Containers Engine, you must ensure that your infrastructure meets certain requirements.

Supported Platforms for `vic-machine`

The vSphere Integrated Containers Engine installation and management utility, `vic-machine`, has been tested and verified on the following 64-bit OS, Windows, Mac OS, and Photon OS systems.

Platform	Supported Versions
Windows	7, 10
Mac OS X	10.11 (El Capitan)
Linux	Ubuntu 16.04 LTS

Other recent 64-bit OS versions should work but are untested.

Supported vSphere Configurations

You can install vSphere Integrated Containers Engine in the following vSphere setups:

- Standalone ESXi 5.5 or 6.0 host that is not managed by a vCenter Server instance.
- vCenter Server 5.5 or 6.0, managing one or more standalone ESXi 5.5 or 6.0 hosts.
- vCenter Server 5.5 or 6.0, managing a cluster of ESXi 5.5 or 6.0 hosts, with DRS enabled.

Deploying vSphere Integrated Containers Engine to a vCenter Server instance that is running in Enhanced Linked Mode is fully supported.

ESXi Host Requirements

To be valid targets for virtual container hosts and container VMs, standalone ESXi hosts and all ESXi hosts in vCenter Server clusters must meet the following criteria:

- In vCenter Server clusters, at least two ESXi hosts must be attached to shared storage for use as container stores, image stores, and volume stores. Using non-shared datastores is possible, but limits the use of vSphere features such as DRS and High Availability. The use of VMware Virtual SAN datastores is fully supported.
- The firewall on all ESXi hosts must be configured to allow connections on the back channel and to allow outbound connections on port 2377. For instruction about how to open port 2377 on ESXi hosts, see VCH Deployment Fails with Firewall Validation Error.
- All ESXi hosts must be attached to the distributed virtual switch for the bridge network in vCenter Server. For more information about distributed virtual switches, see Network Requirements below.
- All ESXi hosts must be attached to any mapped vSphere networks.

During deployment of virtual container hosts, the `vic-machine` utilty checks that the target ESXi hosts meet the requirements, and issues warnings if they do not.

License Requirements

The type of license that vSphere Integrated Containers Engine requires depends on the way in which you deploy the software.

Type of Installation	vSphere Feature Used	Required License
ESXi host	Network Serial Port	vSphere Enterprise
vCenter Server	Distributed Virtual Switch	vSphere Enterprise Plus

All of the ESXi hosts in a cluster require an appropriate license. Installation fails if your environment includes one or more ESXi hosts that have inadequate licenses.

Role and Permissions Requirements

You must use an account with the vSphere Administrator role when you install vSphere Integrated Containers Engine.

Network Requirements

- Use a trusted network for the deployment and use of vSphere Integrated Containers Engine.
- Use a trusted network for connections between Docker clients and the virtual container hosts.
- Each virtual container host requires the following network configuration:
  - An IP address that is obtained by DHCP. Virtual container hosts do not currently support static IP addresses.
  - A network with DHCP for use as the external network. You can share this network between multiple virtual container hosts.
  - One VLAN, if you use VLAN for network isolation.
  - One IP address for each container that you run with the `docker run --net=host` option.
- In vCenter Server environment, before you deploy a virtual container host, you must create a bridge network for use by container VMs.
  - Create a distributed virtual switch with a distributed port group for each virtual container host, for use as the bridge network. You can create multiple port groups on the same distributed virtual switch, but each virtual container host requires its own port group. For information about how to create a distributed virtual switch and a distributed port group, see Create a vSphere Distributed Switch in the vSphere 6.0 documentation.
  - Add the target ESXi host or hosts to the distributed virtual switch. For information about how to add hosts to a distributed virtual switch, see Add Hosts to a vSphere Distributed Switch in the vSphere 6.0 documentation.
  - Assign a VLAN ID to the port group, to ensure that the bridge network is isolated. For information about how to assign a VLAN ID to a port group, see VMware KB 1003825.

vSphere Integrated Containers Engine Networking Overview

Virtual container hosts connect to different types of network.
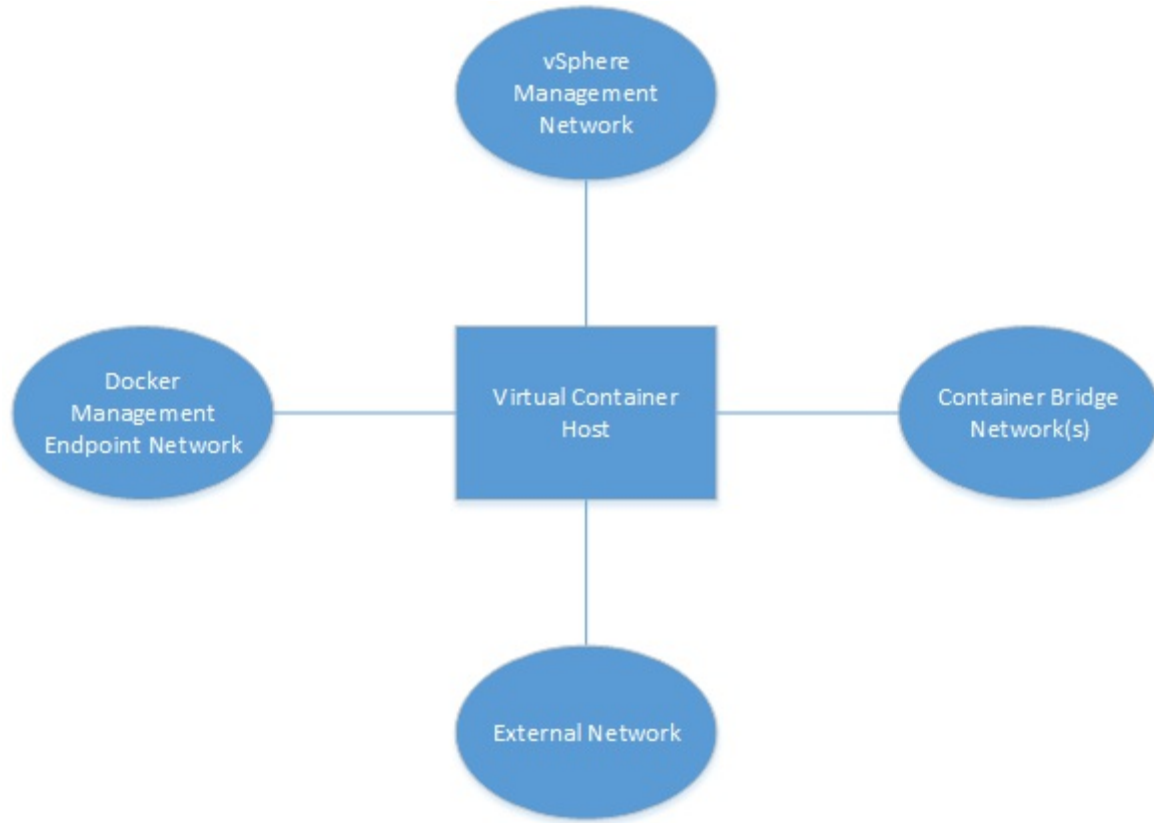
This topic provides an overview of the different network types.

Container Bridge Networks

The network or networks that container VMs use to communicate with each other. Each virtual container host requires a unique bridge network.

You define the bridge networks by setting the `bridge-network` option when you run `vic-machine create`. For more detailed information about bridge networks, see the section on the `bridge-network` option in Virtual Container Host Deployment Options.

Container application developers can also use `docker network create` to create additional bridge networks. You can define a range of IP addresses that additional bridge networks can use by defining the `bridge-network-range` option when you run `vic-machine create`. For more detailed information about how to set bridge network ranges, see the section on the `bridge-network-range` option.
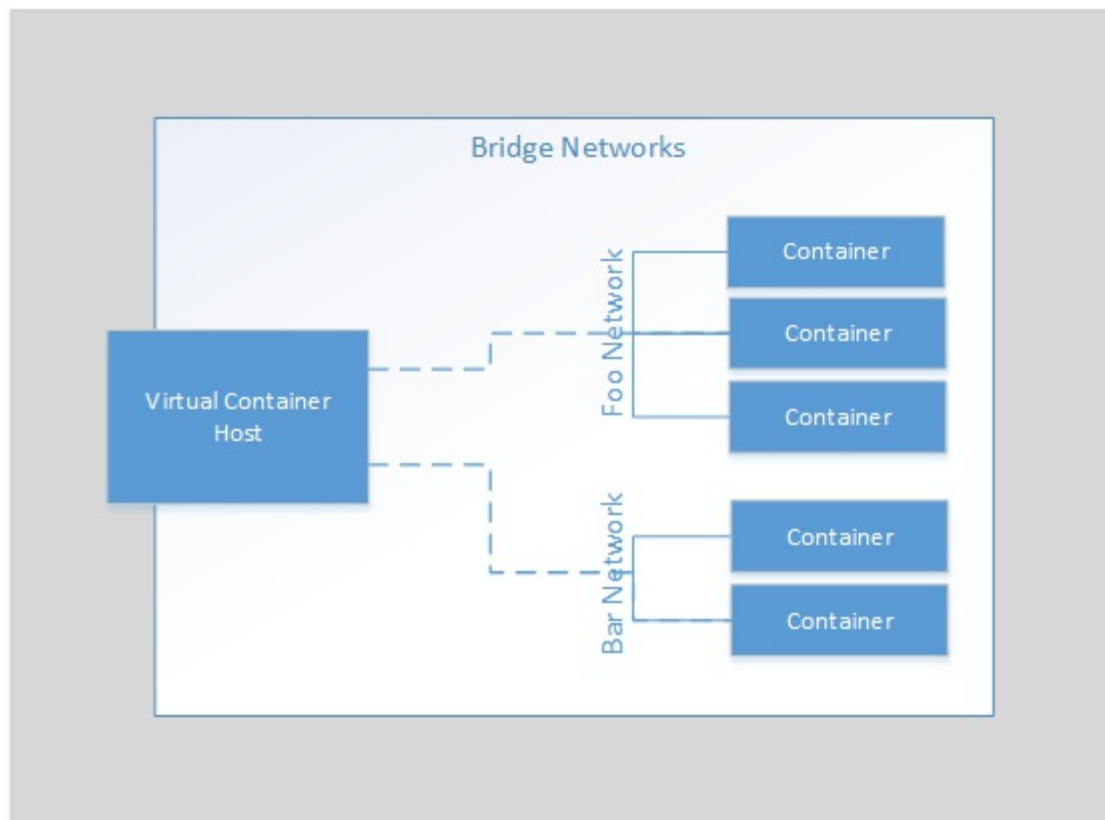
External Network

The network that container VMs use to connect to the internet. Containers can use this external network to publish network services. After defining the external network, you can deploy containers directly on the external interface.

You define the external network by setting the `external-network` option when you run `vic-machine create` . For more detailed information about management networks, see the section on the `external-network` option in Virtual Container Host Deployment Options.
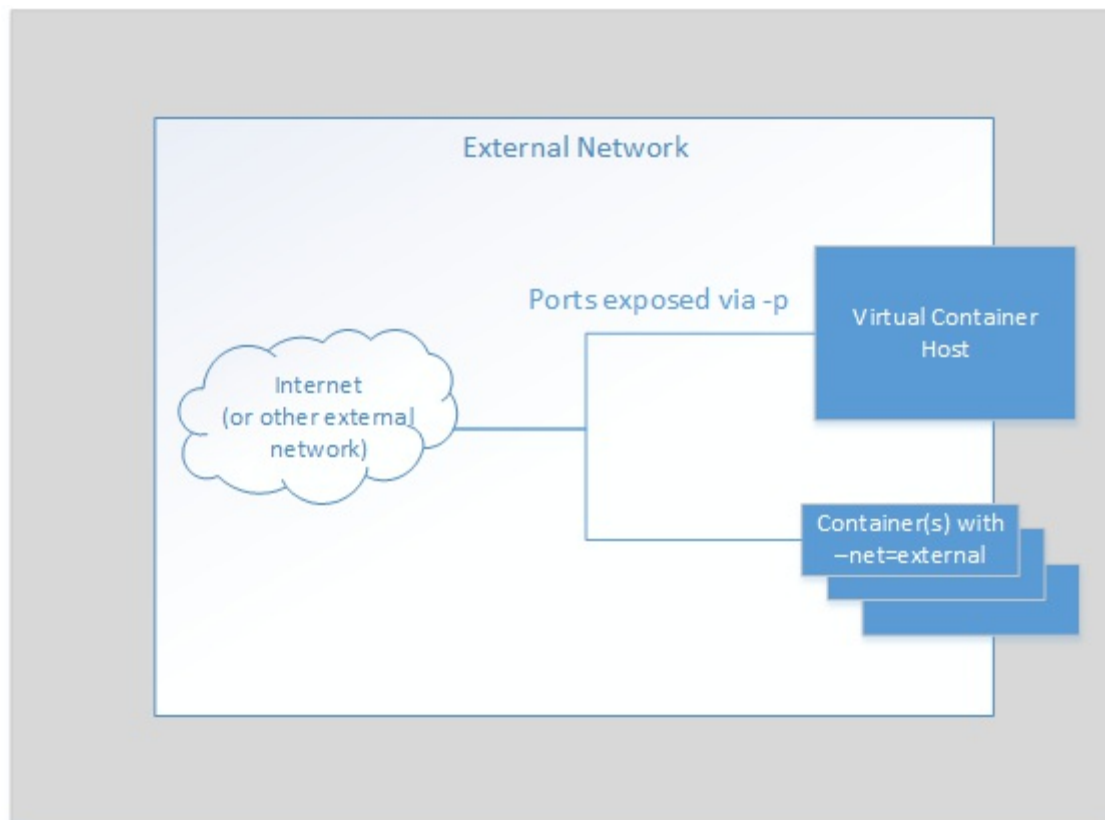
vSphere Management Network

The network for communication between the virtual container host and vCenter Server and ESXi hosts. This network also serves as a tether within container VMs for communication with the virtual container host.

You define the management network by setting the `management-network` option when you run `vic-machine create` . For more detailed information about management networks, see the section on the `management-network` option in Virtual Container Host Deployment Options.

Docker Management Endpoint Network

Connects virtual container hosts to Docker clients and isolates the Docker endpoints from the more public external network.

You define the Docker management endpoint network by setting the `client-network` option when you run `vic-machine` `create`. For more detailed information about Docker management endpoint networks, see the section on the `client-network` option in Virtual Container Host Deployment Options.
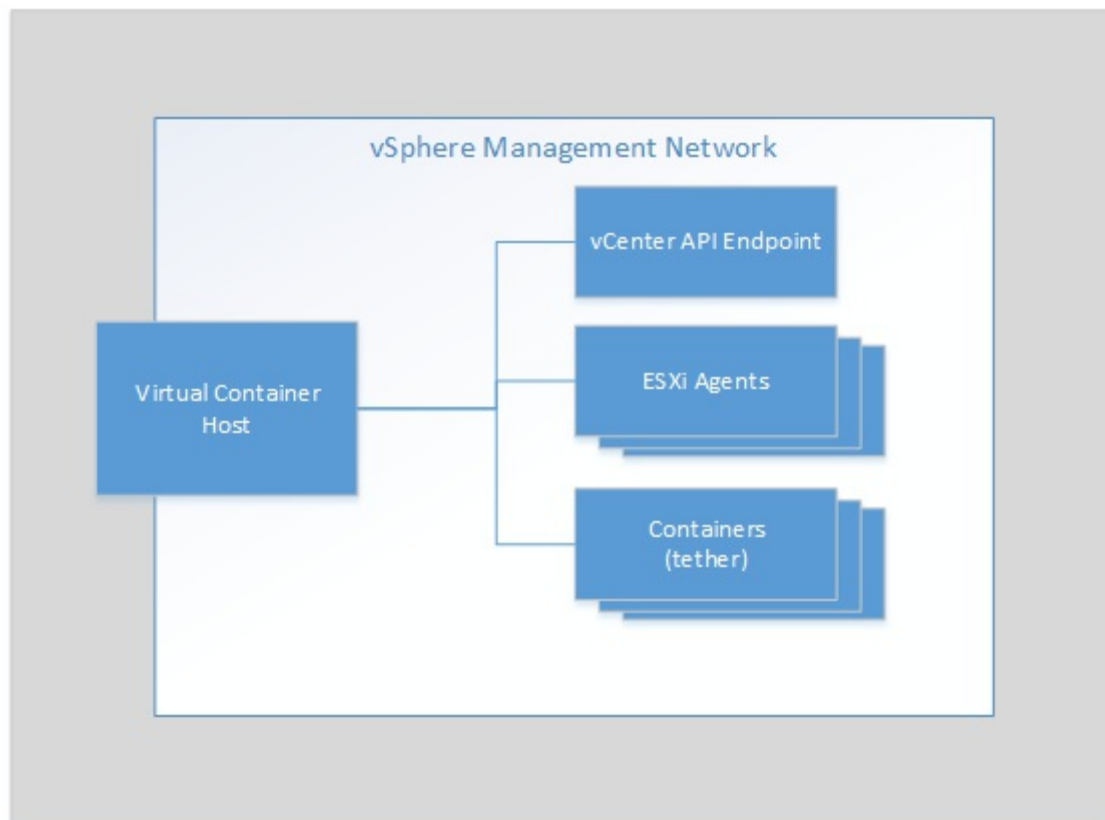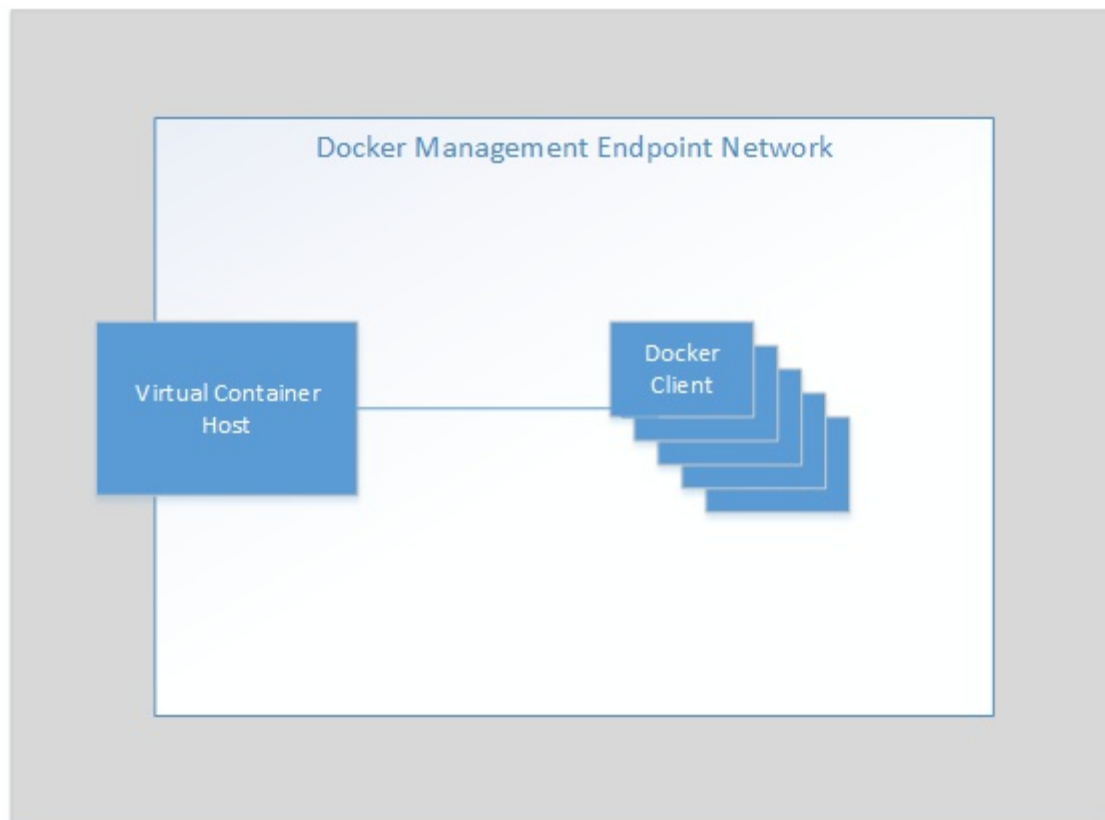
Container Networks

Networks for container VMs to use for external communication when container developers run `docker run` or `docker create` with the `--net` option. For more detailed information about setting up container networks, see the sections on the `container-network-xxx` options in Virtual Container Host Deployment Options.

The Port Layer

You can configure networks that are tied into the vSphere infrastructure. Pre-configured networks available to a virtual container host are determined by the networks that you define when you configure the virtual container host.

The port layer augments the vSphere API with low level, platform-specific primitives to allow you to implement a simple container engine:

- Port Layer Execution: Handles container management, such as create, start, and stop.
- Port Layer Interaction: Handles interaction with a running container.
- Port Layer Networking: Handles specific vSphere network mappings into the Docker network namespace as well as mapping existing network entities such as database servers into the Docker container namespace with defined aliases.
- Port Layer Storage: Provides storage manipulation, including container image storage, layering with volume creation and manipulation. imagec, the docker registry client library, uses this component to translate registry images into a layered format that VMDK disk chains can use directly.

Tether Process

The tether process is a minimal agent in the container VM that starts and stops processes and provides monitoring statistics.

12

Contents of the vSphere Integrated Containers Engine Binaries

After you download and unpack a vSphere Integrated Containers Engine binary bundle, you obtain following files:

File	Description
`appliance.iso`	The ISO from which a virtual container host appliance boots.
`bootstrap.iso`	A Photon OS kernel from which container VMs boot.
`ui/`	A folder that contains the files and scripts for the deployment of the vSphere Web Client Plug-in for vSphere Integrated Containers Engine.
`vic-machine-darwin`	The Mac OS command line utility for the installation and management of virtual container hosts.
`vic-machine-linux`	The Linux command line utility for the installation and management of virtual container hosts.
`vic-machine-windows.exe`	The Windows command line utility for the installation and management of virtual container hosts.
`vic-ui-darwin`	The Mac OS executable for the deployment of the vSphere Web Client Plug-in for vSphere Integrated Containers Engine. **NOTE**: Do not run this executable directly.[1]
`vic-ui-linux`	The Linux executable for the deployment of the vSphere Web Client Plug-in for vSphere Integrated Containers Engine. **NOTE**: Do not run this executable directly.[1]
`vic-ui-windows.exe`	The Windows executable for the deployment of the vSphere Web Client Plug-in for vSphere Integrated Containers Engine. **NOTE**: Do not run this executable directly.[1]
`README`	Contains a link to the vSphere Integrated Containers Engine repository on GitHub.
`LICENSE`	The license file for vSphere Integrated Containers Engine

If you build the vSphere Integrated Containers Engine binaries manually, you find the ISO files and the `vic_machine` utility in the `<git_installation_dir>/vic/bin` folder.

[1] For information about how to install the vSphere Integrated Containers Engine client plug-in, see Installing the vSphere Web Client Plug-in for vSphere Integrated Containers Engine.

Deploy a Virtual Container Host

You install vSphere Integrated Containers Engine by deploying a vSphere Integrated Containers Engine virtual container host. You use the `vic-machine create` command to deploy a virtual container host.

The `vic-machine` utility can deploy a virtual container host in one of the following setups:

- vCenter Server with a cluster
- vCenter Server with one or more standalone ESXi hosts
- A standalone ESXi host

When you deploy a virtual container host, `vic-machine` registers the virtual container host as a vSphere extension. Authentication between the virtual container host and vSphere is handled via key pair authentication against the vSphere extension.

The virtual container host allows you to use an ESXi host or vCenter Server instance as the Docker endpoint for a Docker client. The containers that you pull or create by using your Docker client are stored and managed in the vSphere environment.

Prerequisites

- Verify that your vSphere infrastructure meets the requirements in Environment Prerequisites for vSphere Integrated Containers Engine Installation.
- In a vCenter Server environment, before you deploy a virtual container host, you must create a distributed virtual switch and a distributed port group for use as the bridge network for container VMs. For information about how to create a distributed virtual switch and port group, see *Network Requirements* in Environment Prerequisites for vSphere Integrated Containers Engine Installation.
- Obtain either a verified build, the latest daily build, or the source code of vSphere Integrated Containers Engine:
  - Download the most recent verified build of vSphere Integrated Containers Engine from https://github.com/vmware/vic/releases and unpack it. This version has been tested and approved, but it does not reflect the most up-to-date version of the code.
  - Download the latest daily build of vSphere Integrated Containers Engine from https://bintray.com/vmware/vic-repo/build/view#files and unpack it. This version reflects the version of the code as it was at the last daily build. It has not been tested or approved.
  - For the very latest version, for example to include changes that you have made since the last daily build, build the vSphere Integrated Containers Engine binaries from the source code.
- Add the folder that contains the vSphere Integrated Containers Engine binaries to the `PATH` environment variable on the machine on which you are running `vic-machine`.
- Familiarize yourself with the vSphere Integrated Containers Engine binaries, as described in Contents of the vSphere Integrated Containers Engine Binaries.
- Familiarize yourself with the options of the `create` command of the `vic-machine` utility described in Virtual Container Host Deployment Options.
- For examples of commands to deploy a virtual container host in various vSphere configurations, see Examples of Deploying a Virtual Container Host.

Procedure

1. Open a terminal on the system on which you downloaded and unpacked the vSphere Integrated Containers Engine binary bundle.
2. Navigate to the directory that contains the `vic-machine` utility:
   - If you downloaded the most recent verified build or the latest daily build, go to the location in which you unpacked the vSphere Integrated Containers Engine bundle.
   - If you built the vSphere Integrated Containers Engine binaries, go to `installation_dir/vic/bin`.

3. Run the `create` command of the `vic-machine` utility.

   The following examples include the fewest possible options for installation in a simple vCenter Server environment with a cluster.

   Deploy a virtual container host from a Mac OS system:

   ```
   $ vic-machine-darwin create
   --target vcenter_server_address
   --image-store datastore_name
   --user username
   --bridge-network network_name
   ```

   Deploy a virtual container host from a Linux OS system:

   ```
   $ vic-machine-linux create
   --target vcenter_server_address
   --image-store datastore_name
   --user username
   --bridge-network network_name
   ```

   Deploy a virtual container host from a Windows system:

   ```
   $ vic-machine-windows create
   --target vcenter_server_address
   --image-store datastore_name
   --user username
   --bridge-network network_name
   ```

   At the end of a successful installation, `vic-machine` displays a success message:

   ```
   Initialization of appliance successful
   vic-admin portal:
   https://vch_address:2378
   DOCKER_HOST=vch_address:2376
   Connect to docker:
   docker -H vch_address:2376 --tls info
   Installer completed successfully
   ```

4. (Optional) Copy the generated certificate and key files to the Docker client system.

   If you did not explicitly disable TLS certificate generation by using the `no-tls` option, and if your Docker client is not on the same system as the one that you used to run `vic-machine`, you must copy the `vch_name-cert.pem` and `vch_name-key.pem` files to the Docker client system.

What to Do Next

To test your virtual container host, see Verify the Deployment of a Virtual Container Host to vCenter Server or Verify the Deployment of a Virtual Container Host to an ESXi Host.

Virtual Container Host Deployment Options

The command line utility for vSphere Integrated Containers Engine, `vic-machine`, provides a `create` command that has options to allow you to customize the deployment of a virtual container host to correspond to your vSphere environment.

- Mandatory Options
- Networking Options
- Compute Resource Options
- Datastore Options
- Security Options
- vApp Deployment Options
- Advanced Options

Mandatory Options

The `create` command of the `vic-machine` utility requires you to provide information about where in your vSphere environment to deploy the virtual container host, the user account to use, the bridge network to use, and the location in which to store container image files. The options in this section are either always **mandatory** or mandatory in most circumstances.

`target`

Short name: `-t`

The IPv4 address, fully qualified domain name (FQDN), or URL of the ESXi host or vCenter Server instance on which you are deploying a virtual container host.

To facilitate IP address changes in your infrastructure, provide an FQDN whenever possible, rather than an IP address.

- If the target ESXi host is not managed by vCenter Server, provide the address of the ESXi host.

```
--target esxi_host_address
```

- If the target ESXi host is managed by vCenter Server, or if you are deploying to a cluster, provide the address of vCenter Server.

```
--target vcenter_server_address
```

- You can include the user name and password in the target URL.

```
--target vcenter_or_esxi_username:password@vcenter_or_esxi_address
```

Wrap the user name or password in single quotes (Linux or Mac OS) or double quotes (Windows) if they include special characters.

```
'vcenter_or_esxi_usern@me':'p@ssword'@vcenter_or_esxi_address
```

If you do not include the user name in the target URL, you must specify the `user` option. If you do not specify the `password` option or include the password in the target URL, `vic-machine create` prompts you to enter the password.

- If you are deploying a virtual container host on a vCenter Server instance that includes more than one datacenter, include the datacenter name in the target URL. If you include an invalid datacenter name, `vic-machine create` fails and suggests the available datacenters that you can specify.

```
--target vcenter_server_address/datacenter_name
```

Wrap the datacenter name in single quotes (Linux or Mac OS) or double quotes (Windows) if it includes spaces.

```
--target vcenter_server_address/'datacenter name'
```

`user`

Short name: `-u`

The username for the ESXi host or vCenter Server instance on which you are deploying a virtual container host.

If you are deploying a virtual container host on vCenter Server, specify a username for an account that has the Administrator role on that vCenter Server instance.

```
--user esxi_or_vcenter_server_username
```

Wrap the user name in single quotes (Linux or Mac OS) or double quotes (Windows) if it includes special characters.

```
--user 'esxi_or_vcenter_server_usern@me'
```

You can also specify the username in the URL that you pass to `vic-machine create` in the `target` option, in which case the `user` option is not required.

`image-store`

Short name: `-i`

The datastore in which to store container image files. The `image-store` option is **mandatory** if there is more than one datastore in your vSphere environment. If there is only one datastore in your vSphere environment, the `image-store` option is not required.

When you deploy a virtual container host, `vic-machine` creates a folder named `VIC` on the target datastore, in which to store all of the container images that you pull into a virtual container host. The `vic-machine` utility also places the VM files for the virtual container host in the datastore that you designate as the image store, in a folder that has the same name as the virtual container host.

If you are deploying the virtual container host to a vCenter Server cluster, the datastore that you designate in the `image-store` option must be shared by at least two ESXi hosts in the cluster. Using non-shared datastores is possible, but limits the use of vSphere features such as DRS and High Availability.

You can designate the same datastore as the image store for multiple virtual container hosts. In this case, only one `VIC` folder is created in the datastore and the container image files are made available to all of the virtual container hosts that use that image store.

vSphere Integrated Containers Engine supports all alphanumeric characters, hyphens, and underscores in datastore paths and datastore names, but no other special characters.

```
--image-store datastore_name
```

You can specify a datastore folder to use as the image store in the format `datastore/path`. In this case, the virtual container host uses `path` as the image store instead of using the folder with the same name as the virtual container host. If the folder that you specify does not already exist, `vic-machine create` creates it.

```
--image-store datastore_name/path
```

Wrap the datastore name or path in single quotes (Linux or Mac OS) or double quotes (Windows) if they include spaces.

```
--image-store 'datastore name'/'datastore path'
```

If you specify an invalid datastore name, `vic-machine create` fails and suggests valid datastores.

NOTE: In the current builds the `container-store` option is not enabled. As a consequence, container VM files are also stored in the datastore that you designate as the image store.

`bridge-network`

Short name: `-b`

A distributed port group that container VMs use to communicate with each other.

The `bridge-network` option is **mandatory** if you are deploying a virtual container host to vCenter Server.

In a vCenter Server environment, before you run `vic-machine create`, you must create a distributed virtual switch and a distributed port group. You must add the target ESXi host or hosts to the distributed virtual switch, and assign a VLAN ID to the port group, to ensure that the bridge network is isolated. For information about how to create a distributed virtual switch and port group, see *Network Requirements* in Environment Prerequisites for vSphere Integrated Containers Engine Installation.

You pass the name of the distributed port group to the `bridge-network` option. Each virtual container host requires its own distributed port group.

IMPORTANT

- Do not assign the same `bridge-network` distributed port group to multiple virtual container hosts. Sharing a distributed port group between virtual container hosts might result in multiple container VMs being assigned the same IP address.
- Do not use the `bridge-network` distributed port group as the target for any of the other `vic-machine create` networking options.

If you specify an invalid network name, `vic-machine create` fails and suggests valid networks.

The `bridge-network` option is **optional** when you are deploying a virtual container host to an ESXi host with no vCenter Server. In this case, if you do not specify `bridge-network`, `vic-machine` creates a virtual switch and a port group that each have the same name as the virtual container host. You can optionally specify this option to assign

an existing port group for use as the bridge network for container VMs. You can also optionally specify this option to create a new virtual switch and port group that have a different name to the virtual container host.

```
--bridge-network distributed_port_group_name
```

Wrap the distributed port group name in single quotes (Linux or Mac OS) or double quotes (Windows) if it includes spaces.

```
--bridge-network 'distributed port group name'
```

Networking Options

The `vic-machine create` utility allows you to specify different networks for the different types of traffic between containers, the virtual container host, the external internet, and your vSphere environment.

`bridge-network`

See bridge-network in the section on mandatory options.

`bridge-network-range`

Short name: `--bnr`

The range of IP addresses that additional bridge networks can use when container application developers use `docker network create` to create new bridge networks. If you do not specify the `bridge-network-range` option, the IP range for bridge networks is 172.16.0.0/12.

When you specify the bridge network IP range, you specify the IP range as a CIDR.

```
--bridge-network-range 192.168.100.0/24
```

`external-network`

Short name: `--en`

The network for containers to use to connect to the Internet. Containers use the external network to pull container images, for example from https://hub.docker.com/, and to publish network services. If you define the external network, you can deploy containers directly on the external interface.

If not specified, containers use the default VM Network for external traffic. If you specify an invalid network name, `vic-machine create` fails and suggests valid networks.

```
--external-network network_name
```

Wrap the network name in single quotes (Linux or Mac OS) or double quotes (Windows) if it includes spaces.

```
--external-network 'network name'
```

`management-network`

Short name: `--mn`

The network that the virtual container host uses to communicate with vCenter Server and ESXi hosts. Container VMs use this network to communicate with the virtual container host.

If not specified, the virtual container host uses the external network for management traffic. If you specify an invalid network name, `vic-machine create` fails and suggests valid networks.

```
--management-network network_name
```

Wrap the network name in single quotes (Linux or Mac OS) or double quotes (Windows) if it includes spaces.

```
--management-network 'network name'
```

client-network

Short name: `--cln`

The network that the virtual container host uses to generate the Docker API. The Docker API only uses this network.

If not specified, the virtual container host uses the external network for client traffic. If you specify an invalid network name, `vic-machine create` fails and suggests valid networks.

```
--client-network network_name
```

Wrap the network name in single quotes (Linux or Mac OS) or double quotes (Windows) if it includes spaces.

```
--client-network 'network name'
```

container-network

Short name: `--cn`

A network for container VMs to use for external communication when container developers run `docker run` or `docker create` with the `--net` option.

To specify a container network, you provide the name of a distributed port group for the container VMs to use, and an optional descriptive name for the container network for use by Docker. If you do not specify a descriptive name, Docker uses the vSphere network name. If you specify an invalid network name, `vic-machine create` fails and suggests valid networks.

- The distributed port group must exist before you run `vic-machine create`.
- You cannot use the same distributed port group as you use for the bridge network.
- You can create the distributed port group on the same distributed virtual switch as the distributed port group that you use for the bridge network.
- The descriptive name appears under `Networks` when you run `docker info` on the deployed virtual container host.
- Container developers use the descriptive name in the `--net` option when they run `docker run` or `docker create`.

If you do not specify the `container-network` option, or if container developers run `docker run` or `docker create` without specifying `--net`, container VMs use the bridge network.

If the network that you specify in the `container-network` option does not support DHCP, you must also specify the `container-network-gateway` option.

22

```
--container-network distributed_port_group_name:container_network_name
```

Wrap the distributed port group name in single quotes (Linux or Mac OS) or double quotes (Windows) if it includes spaces. The descriptive name cannot include spaces.

```
--container-network 'distributed port group name':container_network_name
```

container-network-gateway

Short name: `--cng`

The gateway for the subnet of the container network. This option is required if the network that you specify in the `container-network` option does not support DHCP. Specify the gateway in the format `container_network:subnet`. If you specify this option, it is recommended that you also specify the `container-network-dns` option.

When you specify the container network gateway, you use the distributed port group that you specify in the `container-network` option.

```
--container-network-gateway distributed_port_group_name:gateway_ip_address/subnet_mask
```

Wrap the distributed port group name in single quotes (Linux or Mac OS) or double quotes (Windows) if it includes spaces.

```
--container-network-gateway 'distributed port group name':gateway_ip_address/subnet_mask
```

container-network-dns

Short name: `--cnd`

The address of the DNS server for the container network. This option is recommended if the network that you specify in the `container-network` option does not support DHCP.

When you specify the container network DNS server, you use the distributed port group that you specify in the `container-network` option. You can specify `container-network-dns` multiple times, to configure multiple DNS servers.

```
--container-network-dns distributed_port_group_name:8.8.8.8
```

Wrap the distributed port group name in single quotes (Linux or Mac OS) or double quotes (Windows) if it includes spaces.

```
--container-network-dns 'distributed port group name':8.8.8.8
```

container-network-ip-range

Short name: `--cnr`

The range of IP addresses that container VMs can use if the network that you specify in the `container-network` option does not support DHCP. If you do not specify this option, the IP range for container VMs is the entire subnet that you specify in `container-network-gateway`.

When you specify the container network IP range, you use the distributed port group that you specify in the `container-network` option.

```
--container-network-ip-range distributed_port_group_name:192.168.100.2-192.168.100.254
```

You can also specify the IP range as a CIDR.

```
--container-network-ip-range distributed_port_group_name:192.168.100.0/24
```

Wrap the distributed port group name in single quotes (Linux or Mac OS) or double quotes (Windows) if it includes spaces.

```
--container-network-ip-range 'distributed port group name':192.168.100.0/24
```

Compute Resource Options

If the vCenter Server instance on which you are deploying a virtual container host only includes a single instance of a standalone host or cluster, `vic-machine create` automatically detects and uses those resources. If you are deploying to an ESXi host that has no resource pools, `vic-machine create` automatically uses the default resource pool. In these cases, you do not need to specify a compute resource when you run `vic-machine create`.

`compute-resource`

Short name: `-r`

The relative path to the host, cluster, or resource pool in which to deploy the virtual container host. You specify the `compute-resource` option in the following circumstances:

- A vCenter Server instance includes multiple instances of standalone hosts or clusters, or a mixture of standalone hosts and clusters.
- An ESXi host includes multiple resource pools.
- You want to deploy the virtual container host to a specific resource pool in your environment.

If you do not specify the `compute-resource` option and multiple possible resources exist, `vic-machine create` fails and suggests valid targets for `compute-resource` in the failure message.

- To deploy to a specific resource pool on an ESXi host, specify the name of the resource pool:

  ```
  --compute-resource  resource_pool_name
  ```

- To deploy to a vCenter Server instance that has more than one standalone host that are not part of a cluster, specify the IPv4 address or fully qualified domain name (FQDN) of the target host:

  ```
  --compute-resource host_address
  ```

- To deploy to a vCenter Server with more than one cluster, specify the name of the target cluster:

  ```
  --compute-resource cluster_name
  ```

- To deploy to a specific resource pool on a standalone host that is managed by vCenter Server, specify the IPv4 address or FQDN of the target host and name of the resource pool:

24

```
    --compute-resource host_name/resource_pool_name
```

- To deploy to a specific resource pool in a cluster, specify the names of the target cluster and the resource pool:

```
    --compute-resource cluster_name/resource_pool_name
```

- Wrap the resource names in single quotes (Linux or Mac OS) or double quotes (Windows) if they include spaces:

```
    --compute-resource 'cluster name'/'resource pool name'
```

Datastore Options

The `vic-machine` utility allows you to specify the datastores in which to store container image files, the files for the virtual container host appliance, container VM files, and container volumes.

- vSphere Integrated Containers Engine fully supports VMware Virtual SAN datastores.
- vSphere Integrated Containers Engine supports all alphanumeric characters, hyphens, and underscores in datastore paths and datastore names, but no other special characters.

`image-store`

See image-store in the section on mandatory options.

`container-store`

Short name: `--cs`

The datastore in which to store container VM files. When you run a container, container VM files are stored in folders at the top level of the designated datastore. If multiple virtual container hosts use the same container store, all of the container VM files appear at the top level of the container store. You cannot currently designate a specific datastore folder for the VM files of the containers that run in a particular virtual container host.

If you are deploying the virtual container host to a vCenter Server cluster, the datastore that you designate in the `container-store` option must be shared by at least two ESXi hosts in the cluster. Using non-shared datastores is possible, but limits the use of vSphere features such as DRS and High Availability.

You can designate the same datastore as the container store for multiple virtual container hosts.

If you do not specify the `container-store` option, vSphere Integrated Containers Engine stores container VM files in the image store.

If you specify an invalid datastore name, `vic-machine create` fails and suggests valid datastores.

```
  --container-store datastore_name
```

Wrap the datastore name in single quotes (Linux or Mac OS) or double quotes (Windows) if it includes spaces.

```
  --container-store 'datastore name'
```

NOTE: In the current builds the `container-store` option is not enabled. Container VM files are stored in the datastore that you designate as the image store.

25

`volume-store`

Short name: `--vs`

The datastore in which to create volumes when container developers use the `docker volume create` command. When you specify the `volume-store` option, you provide the name of the target datastore and a label for the volume store. You can optionally provide a path to a specific folder in the datastore in which to create the volume store. If you specify an invalid datastore name, `vic-machine create` fails and suggests valid datastores.

If you are deploying the virtual container host to a vCenter Server cluster, the datastore that you designate in the `volume-store` option must be shared by at least two ESXi hosts in the cluster. Using non-shared datastores is possible, but limits the use of vSphere features such as DRS and High Availability.

You can designate the same datastore as the volume store for multiple virtual container hosts.

The label that you specify is the volume store name that Docker uses. For example, the volume store label appears in the information for a virtual container host when container developers run `docker info`. Container developers also specify the volume store label in the `docker volume create --opt VolumeStore=volume_store_label` option when they create a volume.

- If you only require one volume store, you can set the volume store label to `default`. If you set the volume store label to `default`, container developers do not need to specify the `--opt VolumeStore=volume_store_label` option when they run `docker volume create`.

    NOTE: If container developers intend to create anonymous volumes by using `docker create -v`, you must create a volume store with a label of `default`.

    ```
    --volume-store datastore_name:default
    ```

- If you specify the target datastore and the volume store label, `vic-machine create` creates a folder named `volumes` under the `VIC` folder on the target datastore. Any volumes that container developers create will appear in the `volumes` folder.

    ```
    --volume-store datastore_name:volume_store_label
    ```

- If you specify the target datastore, a datastore path, and the volume store label, `vic-machine create` creates a folder named `VIC/volumes` in the location that you specify in the datastore path. If the folders that you specify in the path do not already exist on the datastore, `vic-machine create` creates the appropriate folder structure. Any volumes that container developers create will appear in the `path/VIC/volumes` folder.

    ```
    --volume-store datastore_name/datastore_path:volume_store_label
    ```

- Wrap the datastore name and path in single quotes (Linux or Mac OS) or double quotes (Windows) if they include spaces. The volume store label cannot include spaces.

    ```
    --volume-store 'datastore name'/'datastore path':volume_store_label
    ```

- You can specify the `volume-store` option multiple times, to create multiple volume stores for the virtual container host.

```
--volume-store datastore_name/path:volume_store_label_1
--volume-store datastore_name/path:volume_store_label_2
[...]
--volume-store datastore_name/path:volume_store_label_n
```

Security Options

By default, virtual container hosts automatically generate TLS certificates to authenticate connections with Docker clients. You can configure a virtual container host to use no certificate or a Certificate Authority (CA) certificate to authenticate connections with Docker clients.

`no-tls`

Short name: `-k`

If you do not set the `no-tls` option, `vic-machine` by default generates a TLS certificate and key for the virtual container host to use to authenticate with a Docker client. Set the `no-tls` option if you do not require certificate-based authentication between the virtual container host and the Docker client.

```
--no-tls
```

If you use the `no-tls` option, you connect Docker clients to the virtual container host via port 2375, instead of via port 2376.

`cert`

Short name: none

The path to an X.509 certificate for the Docker API to use to authenticate the virtual container host with a Docker client.

- This option is mandatory if your Docker environment uses TLS certificates that are signed by a CA. For information about how to set up a Docker client to use CA certificates, see https://docs.docker.com/engine/security/https/.
- Use this option in combination with the `key` option, that provides the path to the private key file for the CA certificate.

If you use the `cert` and `key` options, `vic-machine` does not automatically generate certificates. Omit this option if your Docker environment does not use certificates that are signed by a CA. Include the names of the certificate and key files in the paths.

```
--cert path_to_certificate_file/certificate_file_name.pem
--key path_to_key_file/key_file_name.pem
```

Wrap the folder names in the paths in single quotes (Linux or Mac OS) or double quotes (Windows) if they include spaces.

```
--cert 'path to certificate file'/certificate_file_name.pem
--key 'path to key file'/key_file_name.pem
```

27

key

Short name: none

The path to the private key file for use with a custom CA certificate. This option is mandatory if your Docker environment uses certificates that are signed by a CA. For information about how to set up a Docker client to use CA certificates, see https://docs.docker.com/engine/security/https/.

Use this option in combination with the `cert` option, that provides the path to an X.509 certificate file. Include the names of the certificate and key files in the paths.

```
--cert path_to_certificate_file/certificate_file_name.pem
--key path_to_key_file/key_file_name.pem
```

Wrap the folder names in the paths in single quotes (Linux or Mac OS) or double quotes (Windows) if they include spaces.

```
--cert 'path to certificate file'/certificate_file_name.pem
--key 'path to key file'/key_file_name.pem
```

vApp Deployment Options

The `vic-machine` utility provides options to customize the deployment of virtual container hosts.

name

Short name: `-n`

A name for the virtual container host appliance. If not specified, `vic-machine` sets the name of the virtual container host to `virtual-container-host`. If a virtual container host of the same name exists on the ESXi host or in the vCenter Server inventory, or if a folder of the same name exists in the target datastore, the deployment of the virtual container host fails.

```
--name vch_appliance_name
```

Wrap the appliance name in single quotes (Linux or Mac OS) or double quotes (Windows) if it includes spaces.

```
--name 'vch appliance name'
```

password

Short name: `-p`

The password for the user account on the vCenter Server on which you are deploying the virtual container host, or the password for the ESXi host if you are deploying directly to an ESXi host. If not specified, `vic-machine` prompts you to enter the password during deployment.

```
--password esxi_host_or_vcenter_server_password
```

Wrap the password in single quotation marks (') on Mac OS and Linux and in double quotation (") marks on Windows if it includes special characters.

```
--password 'esxi_host_or_vcenter_server_p@ssword'
```

force

Short name: `-f`

Forces `vic-machine create` to ignore warnings and non-fatal errors and continue with the deployment of a virtual container host. Errors such as an incorrect compute resource still cause the installation to fail.

```
--force
```

timeout

Short name: none

The timeout period for uploading the vSphere Integrated Containers Engine appliance and container images to the ESXi host, and for powering on the appliance. Specify a value in the format `XmYs` if the default timeout of 3m0s is insufficient.

```
--timeout 5m0s
```

appliance-iso

Short name: `--ai`

The ISO image from which the virtual container host appliance boots. Omit this option to boot the appliance from the default ISO that is included with `vic-machine`. Set this option to boot the appliance from a different ISO file, for example to reinstall an existing virtual container host or to update it to a newer version. Include the name of the ISO file in the path.

```
--appliance-iso path_to_ISO_file/ISO_file_name.iso
```

Wrap the folder names in the path in single quotes (Linux or Mac OS) or double quotes (Windows) if they include spaces.

```
--appliance-iso 'path to ISO file'/ISO_file_name.iso
```

bootstrap-iso

Short name: `--bi`

The ISO image from which container VMs boot. Omit this option to boot container VMs from the default Photon OS ISO that is included with `vic-machine`. Set this option to a different ISO file to boot container VMs with an operating system other than Photon OS. Include the name of the ISO file in the path.

```
--bootstrap-iso path_to_ISO_file/ISO_file_name.iso
```

Wrap the folder names in the path in single quotes (Linux or Mac OS) or double quotes (Windows) if they include spaces.

```
--bootstrap-iso 'path to ISO file'/ISO_file_name.iso
```

pool-memory-reservation

Short name: `--pmr`

Reserve a quantity of memory for use by the virtual container host vApp and container VMs. Specify the memory reservation value in MB. If not specified, `vic-machine create` sets the reservation to 1.

```
--pool-memory-reservation 1024
```

pool-memory-limit

Short name: `--pml`

Limit the amount of memory that is available for use by the virtual container host vApp and container VMs. Specify the memory limit value in MB. If not specified, `vic-machine create` sets the limit to 0 (unlimited).

```
--pool-memory-limit 1024
```

pool-memory-shares

Short name: `--pms`

Set memory shares on the virtual container host vApp. Specify the share value as a level or a number, for example `high`, `normal`, `low`, or `163840`. If not specified, `vic-machine create` sets the share to `normal`.

```
--pool-memory-shares low
```

pool-cpu-reservation

Short name: `--pcr`

Reserve a quantity of CPU capacity for use by the virtual container host vApp and container VMs. Specify the CPU reservation value in MHz. If not specified, `vic-machine create` sets the reservation to 1.

```
--pool-cpu-reservation 1024
```

pool-cpu-limit

Short name: `--pcl`

Limit the amount of CPU capacity that is available for use by the virtual container host vApp and container VMs. Specify the CPU limit value in MHz. If not specified, `vic-machine create` sets the reservation to 0 (unlimited).

```
--pool-cpu-limit 1024
```

`pool-cpu-shares`

Short name: `--pcs`

Set CPU shares on the virtual container host vApp. Specify the share value as a level or a number, for example `high`, `normal`, `low`, or `163840`. If not specified, `vic-machine create` sets the share to `normal`.

```
--pool-cpu-shares low
```

`debug`

Short name: `-v`

Provide verbose logging output, for troubleshooting purposes when running `vic-machine create`. If not specified, the `debug` value is set to 0 and verbose logging is disabled. Provide a value of 1 or greater to increase the verbosity of the logging. Note that setting debug to a value greater than 1 can affect the behavior of `vic-machine create`.

```
--debug 1
```

Advanced Options

The options in this section are exposed in the `vic-machine create` help if you run `vic-machine create --advanced-options`.

`appliance-cpu`

Short name: none

The number of virtual CPUs for the virtual container host VM. The default is 1. Set this option to increase the number of CPUs in the virtual container host VM, for example if the virtual container host will handle large volumes of containers, or containers that require a lot of processing power.

NOTE Use the `--pool-cpu` options instead of the `--appliance-cpu` option. The `--appliance-cpu` option is mainly intended for use by VMware Support.

```
--appliance-cpu number_of_CPUs
```

`appliance-memory`

Short name: none

The amount of memory for the virtual container host VM. The default is 2048MB. Set this option to increase the amount of memory in the virtual container host VM, for example if the virtual container host will handle large volumes of containers, or containers that consume a lot of memory.

NOTE Use the `--pool-memory` options instead of the `--appliance-memory` option. The `--appliance-memory` option is mainly intended for use by VMware Support.

```
--appliance-memory amount_of_memory
```

`use-rp`

Short name: none

Deploy the virtual container host to a resource pool rather than to a vApp. If you specify this option, `vic-machine` `create` creates a resource pool with the same name as the virtual container host.

```
--use-rp
```

Examples of Deploying a Virtual Container Host

This topic provides examples of the options of the `vic-machine create` command to use when deploying virtual container hosts in different vSphere configurations.

- Deploy a Virtual Container Host Directly on an ESXi Host with no Resource Pools and a Single Datastore
- Deploy a Virtual Container Host in a vCenter Server Cluster
- Deploy a Virtual Container Host to a Cluster and Specify External, Management, Client, and Container Networks
- Deploy a Virtual Container Host and Configure a Non-DHCP Container Network
- Deploy a Virtual Container Host and Specify an Image Store Folder and a Different Datastore for Container VM Files
- Deploy a Virtual Container Host and Specify One or More Volume Stores
- Deploy a Virtual Container Host on a Standalone Host in vCenter Server
- Deploy a Virtual Container Host in a Resource Pool on an ESXi Host
- Deploy a Virtual Container Host in a Resource Pool in a vCenter Server Cluster
- Deploy a Virtual Container Host for Use with a Docker Client that Requires CA Certificates
- Deploy a Virtual Container Host with Limits on Resource Use

For detailed descriptions of all of the `vic-machine create` options, see Virtual Container Host Deployment Options

Deploy a Virtual Container Host Directly on an ESXi Host with no Resource Pools and a Single Datastore

You can deploy a virtual container host directly on an ESXi host that is not managed by a vCenter Server instance. This example provides the minimum options required to deploy a virtual container host. The `vic-machine create` command prompts you for the password for the ESXi host and deploys a virtual container host with the default name `virtual-container-host`. If there is only one datastore on the host and there are no resource pools, you do not need to specify the `image-store` or `compute-resource` options. When deploying to an ESXi host, `vic-machine create` creates a standard virtual switch and a distributed port group, so you do not need to specify any network options if you do not have specific network requirements.

```
vic-machine-darwin-linux-windows create
--target esxi_host_address
--user root
```

Deploy a Virtual Container Host in a vCenter Server Cluster

If vCenter Server has more than one datacenter, you specify the datacenter in the `target` option.

If vCenter Server manages more than one cluster, you use the `compute-resource` option to specify the cluster on which to deploy the virtual container host.

When deploying a virtual container host to vCenter Server, you must use the `bridge-network` option to specify an existing distributed port group for container VMs to use to communicate with each other. For information about how to create a distributed virtual switch and port group, see *Network Requirements* in Environment Prerequisites for vSphere Integrated Containers Engine Installation.

This example deploys a virtual container host with the following configuration:

33

- Provides the vCenter Single Sign-On user and password in the `target` option. Note that the user name is wrapped in quotes, because it contains the `@` character. Use single quotes if you are using `vic-machine` on a Linux or Mac OS system and double quotes on a Windows system.
- Deploys a virtual container host named `vch1` to the cluster `cluster1` in datacenter `dc1`.
- Uses a distributed port group named `vic-bridge` for the bridge network.
- Designates `datastore1` as the datastore in which to store container images and the files for the virtual container host appliance.

```
vic-machine-darwin-linux-windows create
--target 'Administrator@vsphere.local':password@vcenter_server_address/dc1
--compute-resource cluster1
--image-store datastore1
--bridge-network vic-bridge
--name vch1
```

Deploy a Virtual Container Host to a Cluster and Specify External, Management, Client, and Container Networks

In addition to the mandatory bridge network, if your vCenter Server environment includes multiple networks, you can direct different types of traffic to different networks.

- You can direct the traffic between the virtual container host, container VMs, and the internet to a specific network by specifying the `external-network` option. If you do not specify the `external-network` option, the virtual container host uses the default VM Network for external traffic.
- You can direct traffic between ESXi hosts, vCenter Server, and the virtual container host to a specific network by specifying the `management-network` option. If you do not specify the `management-network` option, the virtual container host uses the external network for management traffic.
- You can designate a specific network for use by the Docker API by specifying the `client-network` option. If you do not specify the `client-network` option, the Docker API uses the external network.
- You can designate a specific network for container VMs to use by specifying the `container-network` option. Containers use this network if the container developer runs `docker run` or `docker create` with the `--net` option when they run or create a container. This option requires a distributed port group that must exist before you run `vic-machine create`. You cannot use the same distributed port group that you use for the bridge network. You can provide a descriptive name for the network, for use by Docker. If you do not specify a descriptive name, Docker uses the vSphere network name. For example, the descriptive name appears as an available network in the output of `docker info`.

This example deploys a virtual container host with the following configuration:

- Specifies the user name, password, datacenter, cluster, image store, bridge network, and name for the virtual container host.
- Directs external, management, and Docker API traffic to network 1, network 2, and network 3 respectively. Note that the network names are wrapped in quotes, because they contain spaces. Use single quotes if you are using `vic-machine` on a Linux or Mac OS system and double quotes on a Windows system.
- Designates a distributed port group named `vic-containers` for use by container VMs that are run with the `--net` option.
- Gives the container network the name `vic-container-network`, for use by Docker.

```
vic-machine-darwin-linux-windows create
--target 'Administrator@vsphere.local':password@vcenter_server_address/dc1
--compute-resource cluster1
--image-store datastore1
--bridge-network vic-bridge
--external-network 'network 1'
--management-network 'network 2'
--client-network 'network 3'
--container-network vic-containers:vic-container-network
--name vch1
```

For more information about the networking options, see the Networking Options section in Virtual Container Host Deployment Options.

Deploy a Virtual Container Host and Configure a Non-DHCP Container Network

If the network that you designate as the container network in the `container-network` option does not support DHCP, you can configure the gateway, DNS server, and a range of IP addresses for container VMs to use.

This example deploys a virtual container host with the following configuration:

- Specifies the user name, password, datacenter, cluster, image store, bridge network, and name for the virtual container host.
- Uses the default VM Network for the external, managment, and client networks.
- Designates a distributed port group named `vic-containers` for use by container VMs that are run with the `--net` option.
- Gives the container network the name `vic-container-network`, for use by Docker.
- Specifies the gateway, two DNS servers, and a range of IP addresses on the container network for container VMs to use.

```
vic-machine-darwin-linux-windows create
--target 'Administrator@vsphere.local':password@vcenter_server_address/dc1
--compute-resource cluster1
--image-store datastore1
--bridge-network vic-bridge
--container-network vic-containers:vic-container-network
--container-network-gateway vic-containers:gateway_ip_address/255.255.255.0
--container-network-dns vic-containers:dns1_ip_address
--container-network-dns vic-containers:dns2_ip_address
--container-network-ip-range vic-containers:192.168.100.0/24
--name vch1
```

For more information about the container network options, see the container network section in Virtual Container Host Deployment Options.

Deploy a Virtual Container Host and Specify an Image Store Folder and a Different Datastore for Container VM Files

If your vSphere environment includes multiple datastores, you can specify different datastores for the container image files and the container VM files. If you specify the `container-store` option, the VM files for container VMs are placed in folders at the top level of the designated datastore. The virtual container host files and the container image files are placed in the datastore that you specify in the `image-store` option. If you do not specify the `container-store` option, the virtual container host uses the `image-store` datastore for container VM files.

You specify a specific folder in which to store your container images by providing a path in the `image-store` option.

NOTE: In the current builds of vSphere Integrated Containers Engine, the `container-store` option is not enabled. Container VM files are stored in the datastore that you designate as the image store.

This example deploys a virtual container host with the following configuration:

- Specifies the user name, password, datacenter, cluster, bridge network, and name for the virtual container host.
- Designates the folder `vic images` on `datastore 1` as the location in which to store container image files. If the `vic images` folder does not already exist, `vic-machine create` creates it.
- Designates `datastore 2` as the datastore in which to store container VM files.
- Note that the datastore and folder names are wrapped in quotes, because they contain spaces. Use single quotes if you are using `vic-machine` on a Linux or Mac OS system and double quotes on a Windows system.

```
vic-machine-darwin-linux-windows create
--target 'Administrator@vsphere.local':password@vcenter_server_address/dc1
--compute-resource cluster1
--bridge-network vic-bridge
--image-store 'datastore 1/vic images'
--container-store 'datastore 2'
--name vch1
```

For more information about the datastore options, see the Datastore Options section in Virtual Container Host Deployment Options.

Deploy a Virtual Container Host and Specify One or More Volume Stores

If container application developers will use the `docker volume create` command to create containers that use volumes, you must create volume stores when you deploy virtual container hosts. You specify volume stores in the `volume-store` option. You can specify `volume-store` multiple times to create multiple volume stores.

When you create a volume store, you specify the name of the datastore to use and an optional path to a folder on that datastore. You also specify a descriptive name for that volume store for use by Docker.

This example deploys a virtual container host with the following configuration:

- Specifies the user name, password, datacenter, cluster, bridge network, and name for the virtual container host.
- Specifies the `volumes` folder on `datastore 1` as the default volume store. Creating a volume store named `default` allows container application developers to create anonymous volumes by using `docker create -v`.
- Specifies a second volume store named `volume_store_2` in the `volumes` folder on `datastore 2`.
- Note that the datastore names are wrapped in quotes, because they contain spaces. Use single quotes if you are using `vic-machine` on a Linux or Mac OS system and double quotes on a Windows system.

```
vic-machine-darwin-linux-windows create
--target 'Administrator@vsphere.local':password@vcenter_server_address/dc1
--compute-resource cluster1
--bridge-network vic-bridge
--image-store 'datastore 1'
--volume-store 'datastore 1'/volumes:default
--volume-store 'datastore 2'/volumes:volume_store_2
--name vch1
```

For more information about volume stores, see the volume-store section in Virtual Container Host Deployment Options.

Deploy a Virtual Container Host on a Standalone Host in vCenter Server

If vCenter Server manages multiple standalone ESXi hosts that are not part of a cluster, you use the `compute-resource` option to specify the address of the ESXi host to which to deploy the virtual container host.

This example deploys a virtual container host with the following configuration:

- Specifies the user name, password, image store, bridge network, and name for the virtual container host.
- Deploys the virtual container host on the ESXi host with the FQDN `esxihost1.organization.company.com` in the datacenter `dc1`. You can also specify an IP address.

```
vic-machine-darwin-linux-windows create
--target 'Administrator@vsphere.local':password@vcenter_server_address/dc1
--image-store datastore1
--bridge-network vic-bridge
--compute-resource esxihost1.organization.company.com
--name vch1
```

Deploy a Virtual Container Host in a Resource Pool on an ESXi Host

To deploy a virtual container host in a specific resource pool on an ESXi host that is not managed by vCenter Server, you specify the resource pool name in the `compute-resource` option.

This example deploys a virtual container host with the following configuration:

- Specifies the user name and password and a name for the virtual container host.
- Designates `rp 1` as the resource pool in which to place the virtual container host. Note that the resource pool name is wrapped in quotes, because it contains a space. Use single quotes if you are using `vic-machine` on a Linux or Mac OS system and double quotes on a Windows system.

```
vic-machine-darwin-linux-windows create
--target root:password@esxi_host_address
--compute-resource 'rp 1'
--name vch1
```

Deploy a Virtual Container Host in a Resource Pool in a vCenter Server Cluster

To deploy a virtual container host in a resource pool in a vCenter Server cluster, you specify the names of the cluster and resource pool in the `compute-resource` option.

This example deploys a virtual container host with the following configuration:

- Specifies the user name, password, datacenter, image store, bridge network, and name for the virtual container host.
- Designates `rp 1` in cluster `cluster 1` as the resource pool in which to place the virtual container host. Note that the resource pool and cluster names are wrapped in quotes, because they contain spaces. Use single quotes if you are using `vic-machine` on a Linux or Mac OS system and double quotes on a Windows system.

```
vic-machine-darwin-linux-windows create
--target 'Administrator@vsphere.local':password@vcenter_server_address/dc1
--compute-resource 'cluster 1'/'rp 1'
--image-store datastore1
--bridge-network vic-bridge
--name vch1
```

Deploy a Virtual Container Host for Use with a Docker Client that Requires CA Certificates

If your development environment uses custom Certificate Authority certificates to authenticate connections between Docker clients and virtual container hosts, use the `cert` and `key` options to provide the paths to an X.509 certificate and its key when you deploy a virtual container host. The paths to the certificate and key files must be relative to the location from which you are running `vic-machine create`.

This example deploys a virtual container host with the following configuration:

- Specifies the user name, password, image store, cluster, bridge network, and name for the virtual container host.
- Provides the paths relative to the current location of the `*.pem` files for the custom CA certificate and key files.

```
vic-machine-darwin-linux-windows create
--target 'Administrator@vsphere.local':password@vcenter_server_address/dc1
--compute-resource cluster1
--image-store datastore1
--bridge-network vic-bridge
--cert ../some/relative/path/certificate_file.pem
--key ../some/relative/path/key_file.pem
--name vch1
```

For more information about using CA certificates, see the Security Options section in Virtual Container Host Deployment Options.

Deploy a Virtual Container Host with Limits on Resource Use

To limit the amount of system resources that the container VMs in a virtual container host can use, you can set resource limits on the virtual container host vApp.

This example deploys a virtual container host with the following configuration:

- Specifies the user name, password, image store, cluster, bridge network, and name for the virtual container host.
- Sets resource limits on the virtual container host by imposing memory and CPU reservations, limits, and shares.

```
vic-machine-darwin-linux-windows create
--target 'Administrator@vsphere.local':password@vcenter_server_address/dc1
--compute-resource cluster1
--image-store datastore1
--bridge-network vic-bridge
--pool-memory-reservation 1024
--pool-memory-limit 1024
--pool-memory-shares low
--pool-cpu-reservation 1024
--pool-cpu-limit 1024
--pool-cpu-shares low
--name vch1
```

For more information about setting resource use limitations on virtual container hosts, see the vApp Deployment Options section in Virtual Container Host Deployment Options.

Verify the Deployment of a Virtual Container Host to vCenter Server

After you have deployed a virtual container host to vCenter Server, you can verify the deployment by connecting a Docker client to the virtual container host and running Docker operations. You can check the results in the vSphere Web Client.

Prerequisites

- You used `vic-machine` `create` to deploy a virtual container host to vCenter Server. For information about running `vic-machine`, see Deploy a Virtual Container Host and Virtual Container Host Deployment Options.
- You have installed a Docker client.
- If you did not explicitly disable certificate generation when you deployed the virtual container host, and if you are not running the Docker client in the same location as the one on which you ran `vic-machine`, copy the `vch_name-cert.pem` and `vch_name-key.pem` files to the system on which you you are running the Docker client. Copy the certificate and key files either to the folder in which you run Docker, or to the `~/.docker` folder. Alternatively, specify the following options when you run Docker commands:

  ```
  docker -H vch_address:2376 --tlscacert --tlscert=path --tlskey=path
  ```

- Connect a vSphere Web Client to the vCenter Server instance on which you deployed the virtual container host.

Procedure

1. In the vSphere Web Client, go to **Hosts and Clusters** and select the cluster or host on which you deployed the virtual container host.

   You should see a vApp with the name that you set for the virtual container host. The vApp contains the vSphere Integrated Containers Engine endpoint VM.

2. In your Docker client, run the `docker info` command to confirm that you can connect to the virtual container host.

   - With TLS authentication:

     ```
     $ docker -H vch_address:2376 --tls info
     ```

   - Without TLS authentication:

     ```
     $ docker -H vch_address:2375 info
     ```

   You should see confirmation that the Storage Driver is `vSphere Integrated Containers Backend Engine`.

3. Pull a Docker container image into the virtual container host, for example, the `BusyBox` container.

   - With TLS authentication:

     ```
     $ docker -H vch_address:2376 --tls pull busybox:latest
     ```

   - Without TLS authentication:

     ```
     $ docker -H vch_address:2375 pull busybox:latest
     ```

4. In the vSphere Web Client, go to **Storage**, select the datastore that you designated as the image store, click **Manage** > **Files**, and navigate to the **VIC** folder.

If you specified a specific datastore folder as the image store, the **VIC** folder appears inside that folder.

You should see a folder for every container image that you have pulled into the virtual container host. The folders contain the container image files.

5. In your Docker client, run the Docker container that you pulled into the virtual container host.

   ○ With TLS authentication:

   ```
   $ docker -H vch_address:2376 --tls run busybox --name test
   ```

   ○ Without TLS authentication:

   ```
   $ docker -H vch_address:2375 run busybox --name test
   ```

6. In the vSphere Web Client, go to **Hosts and Clusters** and expand the virtual container host vApp

   You should see a VM for every container that you run, including a VM named `test` .

7. In the vSphere Web Client, go to **Storage** and select the datastore that you designated as the container store.

   At the top-level of the datastore, you should see a folder for every container that you run. The folders contain the container VM files.

Verify the Deployment of a Virtual Container Host to an ESXi Host

After you have deployed a virtual container host to a standalone ESXi host, you can verify the deployment by connecting a Docker client to the virtual container host and running Docker operations. You can check the results in the vSphere Client.

Prerequisites

- You used `vic-machine` `create` to deploy a virtual container host to a standalone ESXi host. For information about running `vic-machine`, see Deploy a Virtual Container Host and Virtual Container Host Deployment Options.
- You have installed a Docker client.
- If you did not explicitly disable certificate generation when you deployed the virtual container host, and if you are not running the Docker client in the same location as the one on which you ran `vic-machine`, copy the `vch_name-cert.pem` and `vch_name-key.pem` files to the system on which you you are running the Docker client. Copy the certificate and key files either to the folder in which you run Docker, or to the `~/.docker` folder. Alternatively, specify the following options when you run Docker commands:

  ```
  docker -H vch_address:2376 --tlscacert --tlscert=path --tlskey=path
  ```

- Connect a vSphere Client to the ESXi host on which you deployed the virtual container host.

Procedure

1. In the vSphere Client, go to **Inventory** and select the host on which you deployed the virtual container host.

   You should see a resource pool with the name that you set for the virtual container host. The resource pool contains the vSphere Integrated Containers Engine endpoint VM.

2. In your Docker client, run the `docker info` command to confirm that you can connect to the virtual container host.

   - With TLS authentication:

     ```
     $ docker -H vch_address:2376 --tls info
     ```

   - Without TLS authentication:

     ```
     $ docker -H vch_address:2375 info
     ```

   You should see confirmation that the Storage Driver is `vSphere Integrated Containers Backend Engine`.

3. Pull a Docker container image into the virtual container host, for example, the `BusyBox` container.

   - With TLS authentication:

     ```
     $ docker -H vch_address:2376 --tls pull busybox:latest
     ```

   - Without TLS authentication:

     ```
     $ docker -H vch_address:2375 pull busybox:latest
     ```

4. In the vSphere Client, click the **Summary** tab for the ESXi host, right-click the datastore that you designated as the image store, select **Browse Datastore**, and navigate to the **VIC** folder.

   If you specified a specific datastore folder as the image store, the **VIC** folder appears inside that folder.

   You should see a folder for every container image that you have pulled into the virtual container host. The folders contain the container image files.

5. In your Docker client, run the Docker container that you pulled into the virtual container host.

   - With TLS authentication:

     ```
     $ docker -H vch_address:2376 --tls run busybox --name test
     ```

   - Without TLS authentication:

     ```
     $ docker -H vch_address:2375 run busybox --name test
     ```

6. In the vSphere Client, go to **Inventory** and expand the virtual container host resource pool.

   You should see a running VM for every container that you run, including a VM named `test`.

7. In the vSphere Client, click the **Summary** tab for the ESXi host, right-click the datastore that you designated as the container store, and select **Browse Datastore**.

   At the top-level of the datastore, you should see a folder for every container that you run. The folders contain the container VM files.

Installing the vSphere Web Client Plug-In for vSphere Integrated Containers Engine

You can install a plug-in that adds information about virtual container hosts and container VMs in the vSphere Web Client.

You can install the plug-in for vSphere Integrated Containers Engine either on a vCenter Server instance that runs on Windows, or on a vCenter Server Appliance.

Information about virtual container hosts and container VMs appears in the **Summary** tabs for those VMs.

- Install the vSphere Integrated Containers Engine Plug-In on vCenter Server For Windows by Using a Web Server
- Install the vSphere Integrated Containers Engine Plug-In on vCenter Server For Windows, by Using SFTP
- Install the vSphere Integrated Containers Engine Plug-In on vCenter Server For Windows, without SFTP or a Web Server
- Install the vSphere Integrated Containers Engine Plug-In on a vCenter Server Appliance by Using a Web Server
- Install the vSphere Integrated Containers Engine Plug-In on a vCenter Server Appliance Without Access to a Web Server

Install the vSphere Integrated Containers Engine Plug-In on vCenter Server For Windows by Using a Web Server

If your vCenter Server instance runs on Windows, you can use a Web server to host the vSphere Web Client plug-in for vSphere Integrated Containers Engine.

Prerequisites

- You deployed at least one virtual container host to a vCenter Server instance that runs on Windows.
- You are running a Web server that your vCenter Server instance can access.
- You must use a Windows system to run the script to install the plug-in on a vCenter Server that runs on Windows. If you used a Linux or Mac OS system to deploy the virtual container host, download and unpack the vSphere Integrated Containers Engine package on a Windows system. For example, download the package to the system on which vCenter Server is running.

Procedure

1. On the Windows system on which you have downloaded and unpacked vSphere Integrated Containers Engine, navigate to the folder that contains the `vic-machine` utility and open the `ui` folder.
2. Upload the plug-in bundle to your Web server.

   ```
   vic_unpack_dir\vic\ui\vsphere-client-serenity\com.vmware.vicui.Vicui-0.0.1.zip
   ```

3. On the `vic-machine` system, open the `vic_unpack_dir\vic\ui\vCenterForWindows\configs` file in a text editor.
4. Enter the IPv4 address or FQDN of the vCenter Server instance on which to install the plug-in.

   ```
   SET target_vcenter_ip=vcenter_server_address
   ```

5. Enter the path to the folder on your Web server that contains the `com.vmware.vicui.Vicui-0.0.1.zip` file.

   ```
   VIC_UI_HOST_URL="vicui_zip_location"
   ```

6. (Optional) If you used an HTTPS address in `vic_ui_host_url`, provide the SHA-1 thumbprint of the Web server.

   ```
   SET vic_ui_host_thumbprint=thumbprint
   ```

7. Save and close the `configs` file.
8. Open a command prompt, navigate to `vic_unpack_dir\vic\ui\vCenterForWindows`, and run the installer.

   ```
   install.bat
   ```

9. Enter the user name and password for the vCenter Server administrator account.
10. When installation finishes, if you are logged into the vSphere Web Client, log out then log back in again.

Install the vSphere Integrated Containers Engine Plug-In on vCenter Server For Windows by Using SFTP

If your vCenter Server instance runs on a Windows system that has access to an SFTP service, you can use SFTP to install the vSphere Web Client plug-in for vSphere Integrated Containers Engine.

Prerequisites

- You deployed at least one virtual container host to a vCenter Server instance that runs on Windows.
- Your vCenter Server instance is running on a Windows system that has an SFTP service.
- You must use a Windows system to run the script to install the plug-in on a vCenter Server that runs on Windows. If you used a Linux or Mac OS system to deploy the virtual container host, download and unpack the vSphere Integrated Containers Engine package on a Windows system. For example, download the package to the system on which vCenter Server is running.

Procedure

1. On the Windows system on which you have downloaded and unpacked vSphere Integrated Containers Engine, navigate to the folder that contains the `vic-machine` utility and open the `ui` folder.
2. Open the `vic_unpack_dir\vic\ui\vCenterForWindows\configs` file in a text editor.
3. Enter the IPv4 address or FQDN of the vCenter Server instance on which to install the plug-in.

   ```
   SET target_vcenter_ip=vcenter_server_address
   ```

4. Make sure that `sftp_supported` is set to 1.

   ```
   SET sftp_supported=1
   ```

5. Provide the SFTP user name and password.

   ```
   SET sftp_username=username
   SET sftp_password=password
   ```

6. Provide the location of the `vsphere-client-serenity` folder on the vCenter Server Windows system relative to the root folder of the SFTP connection.

   For example, if the root folder of the SFTP connection is `vcenter_server_install_dir\vCenterServer\cfg`, set the relative path as follows:

   ```
   target_vc_packages_path=/vsphere-client/vc-packages/vsphere-client-serenity/
   ```

7. Save and close the `configs` file.
8. Open a command prompt, navigate to `vic_unpack_dir\vic\ui\vCenterForWindows`, and run the installer.

   ```
   install.bat
   ```

9. Enter the user name and password for the vCenter Server administrator account.
10. When installation finishes, if you are logged into the vSphere Web Client, log out then log back in again.

Install the vSphere Integrated Containers Engine Plug-In on vCenter Server For Windows, without SFTP or a Web Server

You can install the vSphere Web Client plug-in for vSphere Integrated Containers Engine on a vCenter Server instance for Windows that has neither access to an SFTP service nor to a Web Server.

Prerequisites

- You deployed at least one virtual container host to a vCenter Server instance that runs on Windows.
- You must use a Windows system to run the script to install the plug-in on a vCenter Server that runs on Windows. If you used a Linux or Mac OS system to deploy the virtual container host, download and unpack the vSphere Integrated Containers Engine package on a Windows system. For example, download the package to the system on which vCenter Server is running.

Procedure

1. On the Windows system on which you have downloaded and unpacked vSphere Integrated Containers Engine, navigate to the folder that contains the `vic-machine` utility and open the `ui` folder.
2. Copy the `com.vmware.vicui.Vicui-0.0.1` folder into the folder on the vCenter Server system that contains the vSphere Web Client packages.

   - Source location on `vic-machine` system:

     ```
     vic_unpack_dir\vic\ui\vsphere-client-serenity
     ```

   - Destination location on vCenter Server Windows system:

     ```
     instl_dir\vCenterServer\cfg\vsphere-client\vc-packages\vsphere-client-serenity
     ```

     `instl_dir` is the location in which vCenter Server is installed. If the `vc-packages\vsphere-client-serenity` folders do not exist under the `vsphere-client` folder, create them manually.

3. On the `vic-machine` system, open the `vic_unpack_dir\vic\ui\vCenterForWindows\configs` file in a text editor.
4. Enter the IPv4 address or FQDN of the vCenter Server instance on which to install the plug-in.

   ```
   SET target_vcenter_ip=vcenter_server_address
   ```

5. Deactivate SFTP by changing the value of `sftp_supported` to 0.

   ```
   SET sftp_supported=0
   ```

6. Save and close the `configs` file.
7. Open a command prompt, navigate to `vic_unpack_dir\vic\ui\vCenterForWindows`, and run the installer.

   ```
   install.bat
   ```

8. Enter the user name and password for the vCenter Server administrator account.
9. When installation finishes, if you are logged into the vSphere Web Client, log out then log back in again.

Install the vSphere Integrated Containers Engine Plug-In on a vCenter Server Appliance by Using a Web Server

If you are running the vCenter Server Appliance, you can use a Web server to host the vSphere Web Client plug-in for vSphere Integrated Containers Engine.

Prerequisites

- You deployed at least one virtual container host to a vCenter Server Appliance instance.
- You are running a Web server that the vCenter Server Appliance can access.

Procedure

1. On the system on which you run `vic-machine`, navigate to the folder that contains the `vic-machine` utility and open the `ui` folder.
2. Upload the plug-in bundle to your Web server.

   ```
   vic_unpack_dir/vic/ui/vsphere-client-serenity/com.vmware.vicui.Vicui-0.0.1.zip
   ```

3. Open the `vic_unpack_dir/vic/ui/VCSA/configs` file in a text editor.
4. Enter the IPv4 address or FQDN of the vCenter Server instance on which to install the plug-in.

   ```
   VCENTER_IP="vcenter_server_address"
   ```

5. Enter the path to the folder on your Web server that contains the `com.vmware.vicui.Vicui-0.0.1.zip` file.

   ```
   VIC_UI_HOST_URL="vicui_zip_location"
   ```

6. (Optional) If you used an HTTPS address in `vic_ui_host_url`, provide the SHA-1 thumbprint of the Web server.

   ```
   VIC_UI_HOST_THUMBPRINT="thumbprint"
   ```

7. Save and close the `configs` file.
8. (Optional) If you run `vic-machine` on a Windows system, open the `vic_unpack_dir/vic/ui/VCSA/install.sh` file in a text editor and point `PLUGIN_MANAGER_BIN` to the Windows UI executable.

   - Before:
     ```
     if [[ $(echo $OS | grep -i "darwin") ]] ; then
     PLUGIN_MANAGER_BIN="../../vic-ui-darwin"
     else
     PLUGIN_MANAGER_BIN="../../vic-ui-linux"
     ```

   - After:
     ```
     if [[ $(echo $OS | grep -i "darwin") ]] ; then
     PLUGIN_MANAGER_BIN="../../vic-ui-darwin"
     else
     PLUGIN_MANAGER_BIN="../../vic-ui-windows"
     ```

9. Open a command prompt, navigate to `vic_unpack_dir/vic/ui/VCSA` , and run the installer.

```
./install.sh
```

- Make sure that `install.sh` is executable by running `chmod` before you run it.
- On Windows systems, run `install.sh` in a UNIX shell that supports SSH and SCP, for example Cygwyn or Git Bash. Do not use Windows 10 native Bash.

10. Enter the user name and password for the vCenter Server administrator account.
11. Answer the question about the version of vCenter Server that you are using.
    - Answer `y` if you are using vCenter Server 5.5.
    - Answer `n` if you are using vCenter Server 6.0.
12. When installation finishes, if you are logged into the vSphere Web Client, log out then log back in again.

Install the vSphere Integrated Containers Engine Plug-In on a vCenter Server Appliance Without Access to a Web Server

If you are running the vCenter Server Appliance and you do not have access to a Web server, you can manually install the vSphere Web Client plug-in for vSphere Integrated Containers Engine.

Prerequisites

You deployed at least one virtual container host to a vCenter Server Appliance instance.

Procedure

1. On the system on which you run `vic-machine`, open the `vic_unpack_dir/vic/ui/VCSA/configs` file in a text editor.
2. Enter the IPv4 address or FQDN of the vCenter Server instance on which to install the plug-in.

   ```
   VCENTER_IP="vcenter_server_address"
   ```

3. Save and close the `configs` file.
4. (Optional) If you run `vic-machine` on a Windows system, open the `vic_unpack_dir/vic/ui/VCSA/install.sh` file in a text editor and point `PLUGIN_MANAGER_BIN` to the Windows UI executable.

   - Before:

     ```
     if [[ $(echo $OS | grep -i "darwin") ]] ; then
     PLUGIN_MANAGER_BIN="../../vic-ui-darwin"
     else
     PLUGIN_MANAGER_BIN="../../vic-ui-linux"
     ```

   - After:

     ```
     if [[ $(echo $OS | grep -i "darwin") ]] ; then
     PLUGIN_MANAGER_BIN="../../vic-ui-darwin"
     else
     PLUGIN_MANAGER_BIN="../../vic-ui-windows"
     ```

5. Open a command prompt, navigate to `vic_unpack_dir/vic/ui/VCSA`, and run the installer.

   ```
   ./install.sh
   ```

   - Make sure that `install.sh` is executable by running `chmod` before you run it.
   - On Windows systems, run `install.sh` in a UNIX shell that supports SSH and SCP, for example Cygwyn or Git Bash. Do not use Windows 10 native Bash.
6. Enter the user name and password for the vCenter Server administrator account.

7. Answer the question about the version of vCenter Server that you are using.
   - Answer `y` if you are using vCenter Server 5.5.
   - Answer `n` if you are using vCenter Server 6.0.
8. Enter the root password for the vCenter Server Appliance.

   The installer requires the root password of the vCenter Server Appliance three times:

- Once to check whether the Bash shell is enabled on the vCenter Server Appliance. If the Bash shell is not enabled, the installation fails and the installer provides remedial instructions.
- Once to copy the files to the appliance over SSH.
- Once to set the correct ownership on the files and folders.

9. When installation finishes, if you are logged into the vSphere Web Client, log out then log back in again.

Troubleshooting vSphere Integrated Containers Engine Installation

This information provides solutions for common problems that you might encounter when deploying virtual container hosts.

- Installation Fails with Resource Pool Creation Error
- VCH Deployment Fails with Unknown or Non-Specified Argument Error or Incorrect User Name Error
- VCH Deployment Fails with Firewall Validation Error

VCH Deployment Fails with Resource Pool Creation Error

When you use `vic-machine create` to deploy a virtual container host directly on an ESXi host, the installation fails with a resource pool creation error.

Problem

Deployment on an ESXi host fails during the validation of the configuration that you provided:

```
Creating resource pool failed with ServerFaultCode:
Access to resource settings on the host is restricted to the server
that is managing it: vcenter_server_address.
Exiting ...
```

Cause

You set the `target` option to the address of an ESXi host that is managed by a vCenter Server instance.

Solution

- Set the `target` option to the address of the vCenter Server instance that manages the ESXi host.
- Disassociate the ESXi host from the vCenter Server instance.
- Set the `target` option to a different ESXi host.

VCH Deployment Fails with Unknown or Non-Specified Argument Error or Incorrect User Name Error

When you use the command line installer to deploy a virtual container host, the deployment fails with an error about unknown CLI arguments, unspecified mandatory options, or an invalid user name and password.

Problem

Deployment fails during the validation of the configuration that you provided, even if you did specify the options cited as missing or incorrect. For example:

```
Image datastore path must be specified; use format datastore/path
```

```
Unknown argument: argument
vic-machine failed: invalid CLI arguments
```

```
vic-machine failed: Failed to log in to vcenter_server_or_esxi_host_address:
ServerFaultCode: Cannot complete login due to an incorrect user name or password
```

Cause

String values that you provided for certain options contain spaces, or the user name and password contain special characters.

Solution

Wrap any arguments that contain spaces or special characters in single quotation marks (') on Mac OS and Linux and in double quotation (") marks on Windows.

Option arguments that might require quotation marks include the following:

- User names and passwords in `target`, or in `user` and `password`
- Datacenter names in `target`
- Virtual container host names in `name`
- Datastore names and paths in `image-store`, `container-store`, and `volume-store`
- Network and distributed port group names in all networking options.
- Cluster and resource pool names in `compute-resource`
- Folder names in the paths for `cert`, `key`, `appliance-iso`, and `bootstrap-iso`

For information about when to use quotation marks for different options, see the descriptions of those options in Virtual Container Host Deployment Options.

VCH Deployment Fails with Firewall Validation Error

When you use `vic-machine create` to deploy a virtual container host, deployment fails because firewall port 2377 is not open on the target ESXi host or hosts.

Problem

Deployment fails with a firewall error during the validation phase:

```
Firewall must permit 2377/tcp outbound to use VIC.
```

Cause

ESXi hosts communicate with the virtual container hosts through port 2377 via Serial Over LAN. For deployment of a virtual container host to succeed, port 2377 must be open for outgoing connections on all ESXi hosts before you run `vic-machine create`. Opening port 2377 for outgoing connections on ESXi hosts opens port 2377 for inbound connections on the virtual container hosts.

Solution

Set a firewall ruleset on the ESXi host or hosts. In test environments, you can disable the firewall on the hosts.

Set a Firewall Ruleset Manually

In production environments, if you are deploying to a standalone ESXi host, set a firewall ruleset on that ESXi host. If you are deploying to a cluster, set the firewall ruleset on all of the ESXi hosts in the cluster.

IMPORTANT: Firewall rulesets that you set manually are not persistent. If you reboot the ESXi hosts, any firewall rules that you set are lost. You must recreate firewall rules each time you reboot a host.

1. Use SSH to log in to each ESXi host as `root` user.
2. Follow the instructions in VMware KB 2008226 to add the following rule after the last rule in the file `/etc/vmware/firewall/service.xml` .

```
<service id='id_number'>
<id>vicoutgoing</id>
<rule id='0000'>
 <direction>outbound</direction>
 <protocol>tcp</protocol>
 <port type='dst'>2377</port>
</rule>
<enabled>true</enabled>
<required>true</required>
</service>
```

In this example, *id_number* is the number of the preceding ruleset in `service.xml` , incremented by 1.

Disable the Firewall

In test environments, you can disable the firewalls on the ESXi hosts instead of opening port 2377.

1. Use SSH to log in to each ESXi host as `root` user.
2. Run the following command:

```
$ esxcli network firewall set --enabled false
```