

# Understand App Hosting on Catalyst 9000 Switches

## Contents

---

### [Introduction](#)

### [Prerequisites](#)

[Requirements](#)

[Components Used](#)

### [Background Information](#)

### [Restrictions for Application Hosting Framework](#)

### [App Hosting Architecture](#)

[Supported Catalyst Models for App Hosting](#)

[Hardware Resources for Applications](#)

[Application High Availability](#)

[Application Hosting Framework](#)

### [Container Networking](#)

[Network for App Hosting](#)

[Lifecycle for and Application](#)

### [Docker Runtime Options](#)

### [List of Unsupported Docker Runtime Options](#)

### [Configuration](#)

### [Troubleshoot](#)

[Modify the Docker Container](#)

### [Related Information](#)

---

## Introduction

This document describes how to implement and troubleshoot the App Hosting on Catalyst 9000 Series Platforms.

## Prerequisites

### Requirements

Cisco recommends that you have knowledge of these topics:

- LAN switching fundamentals.
- Familiarity with Cisco IOS® XE and licensing.
- Understanding of switching architectures.
- Familiarity with basic Linux commands.

### Components Used

The information in this document is based on these software and hardware versions:

- Catalyst 9300
- Catalyst 9400
- Catalyst 9500
- Cisco IOS® XE & 16.12.X or 17.X software
- See the *Supported Catalyst Models for App Hosting* for more information.

---

 **Note:** Consult the appropriate configuration guide for the commands that are used in order to enable these features on other Cisco platforms.

---

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

## Background Information

Applications are essential in enterprise networks for addressing a variety of business-critical use cases, including administrative tools like protocol analyzers and security solutions such as intrusion detection services. Traditionally, these applications operated on external physical or cloud-based virtual servers, but modern demands have highlighted the limitations of such approaches.

The Cisco Catalyst 9000 series switches have revolutionized enterprise networking by integrating advanced capabilities designed for the modern digital era. Powered by Cisco IOS® XE operating system and an x86 CPU, these switches are more than just traditional network devices; they serve as platforms for application hosting, enabling businesses to consolidate infrastructure and unlock new possibilities for edge computing. This allows applications like security agents, IoT sensors, and traffic monitoring tools to run directly on the switch, eliminating the need for external compute hardware.

## Restrictions for Application Hosting Framework

To enable Application Hosting Framework on Catalyst 9000, these are the requirements:

- The Switch must be running release version 16.12. Docker App is supported only on release 16.12, as this version supports the native Docker engine.
- Before Application Hosting can be enabled on Catalyst 9000, a Cisco certified USB3.0 Flash Drive must be installed in the device back-panel USB3.0 port. App hosting only works on the back-panel USB3.0.
- Cisco Smart Licensing is required for Catalyst 9000 platforms. Cisco DNA-Advantage licensing is required to enable app hosting.
- Only Docker containers are supported. The Docker apps can be installed on Cisco supported SSD storage.
- Internal flash is not supported for third-party applications, only Cisco signed applications like ThousandEyes Agents.
- Third party containers must reside in one of the external Solid State Disk storage options, provided by the Catalyst 9000 switches. SSD storage is supported on Catalyst 9300 Series switches. SATA storage is supported on Catalyst 9400 Series and Catalyst 9500 Series High Performance models and Catalyst 9600 Series Switches. SSD storage or SATA storage cannot be used in the front panel USB port of the switch.
- Cisco does not provide any pre-packaged third party unsupported apps. Customers have to package it themselves. Certain third party vendors with whom Cisco has collaborated with can share a link specific to Cisco installation.

- Front panel USB ports for application hosting are only supported on Catalyst 9300LM models.

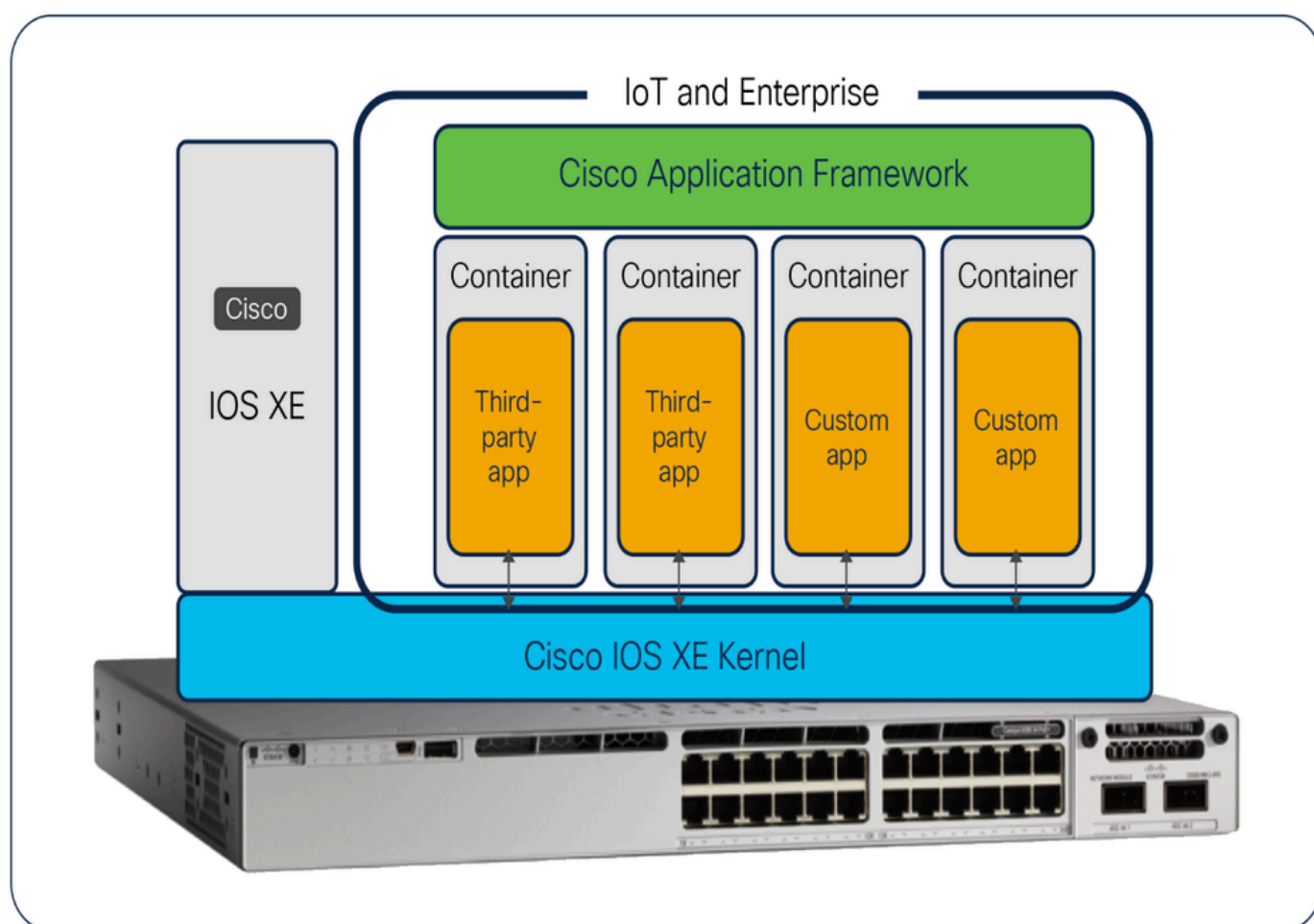
## App Hosting Architecture

To support application hosting capabilities on the Cisco Catalyst 9000 switches, the switch provides hardware resources where applications can reside and execute. Cisco IOS XE running on the Cisco Catalyst 9000 switches reserves dedicated memory and CPU resources for application hosting to provide a separate execution space for user applications without compromising the integrity and performance of the switch.

Moreover, applications must reside in one of the external Solid-State Drive (SSD) storage options (USB or M2 SATA), depending on the specific Cisco Catalyst 9000 platforms. Applications have no access to the internal device flash storage, which is reserved for Cisco IOS XE to protect its integrity.

For maximum flexibility and total isolation from the main operating system, the Cisco IOS XE kernel and Cisco Application Framework on the Cisco Catalyst 9000 switches support containerized application by leveraging control groups (Cgroups) and user namespace. Cgroups limit access to physical resources such as CPU and memory for applications. The Cisco Application Framework checks that there are sufficient resources to activate and install the application. If hardware resources are not available for the application, then it cannot activate the application, and relevant messages are given to the administrator.

**Figure 1.** Shows a visual representation of the Cisco Application Framework on the Cisco Catalyst 9000 platform:



Moreover, SSD storage offers best-in-class security by providing AES-256 hardware encryption on SSD storage and passcode authentication on both SSD storage and the switch.

The AES-256 encryption is completely done in hardware. When passcode authentication is used, the passcode has to set on both the SSD and the switch. When a SSD with passcode authentication pre-configured is inserted to the Catalyst 9000 switch that does not have the matching passcode configuration, then the authentication fails because the switch does not have the correct passcode configured. The passcode must match on both the SSD storage and on the switch for successful de-authentication as show on Figure 2.

If the passcode configured SSD storage is removed from the Catalyst 9000 switches and inserted into a non-Catalyst switch, then the contents are secured and not accessible. Any sensitive data is only accessible once unlocked in a Catalyst switch with the correct passcode.

**Figure 2.** Shows the passcode authentication on SSD storage:



For example, you can use these commands to enable or disable security and set up a password:

```
<#root>
```

```
Cat9k#
```

```
hw-module switch 1 usbflash1 security ?
```

```
disable  disable security on USB3.0
enable   Enable security on USB3.0
unlock   Unlock USB3.0
```

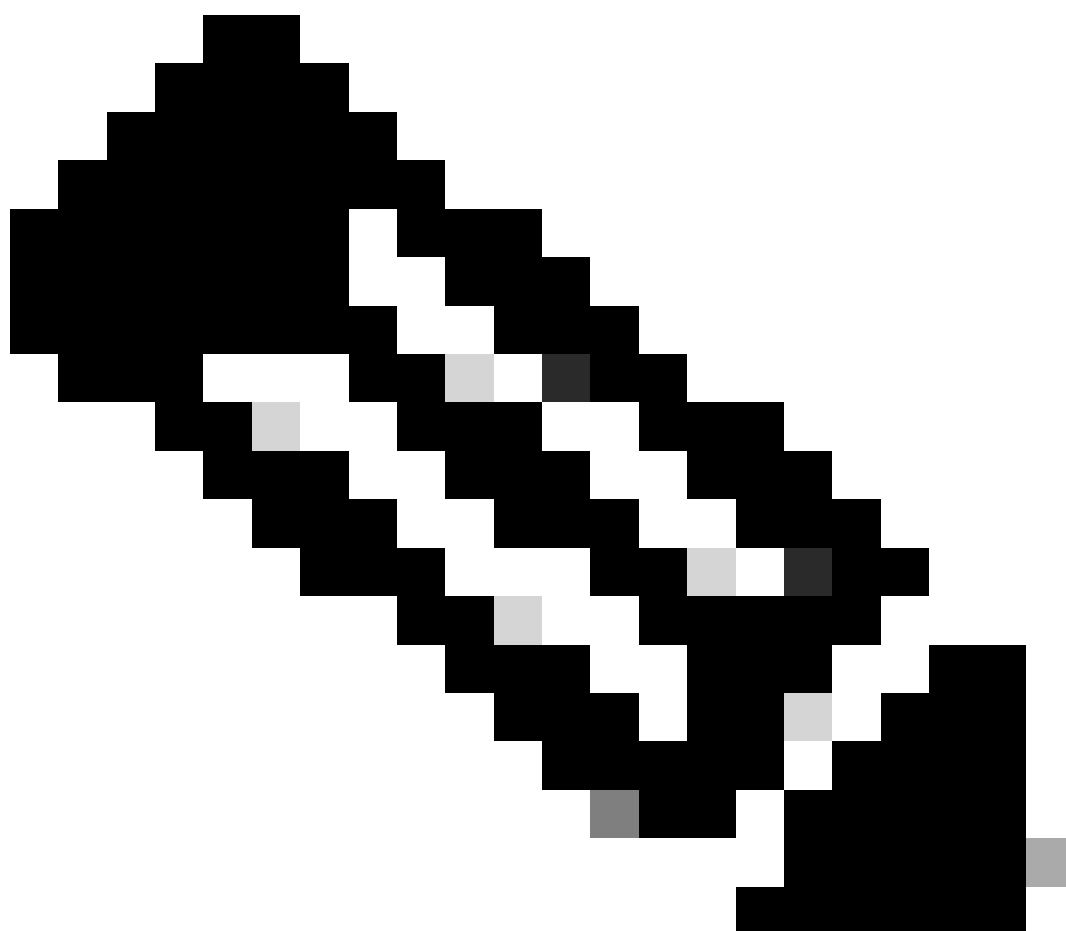
```
Cat9k(config)#hw-module switch 1 usbflash1-password <password>
```

## Supported Catalyst Models for App Hosting

Application hosting capabilities are supported as follow:

Supported Platforms	Cisco IOS XE Release
Catalyst 9300/L	16.12.1 release or later
Catalyst 9300X	17.5.1 release or later
Catalyst 9300LM	17.9.1 release or later

Catalyst 9404 and 9407	17.1.1 release or later
Catalyst 9410	17.5.1 release or later
Catalyst 9400X	17.8.1 release or later
Catalyst 9500 High Performance and 9600 Series	17.5.1 release or later
Catalyst 9500X and 9600X Series	17.8.1 release or later



**Note:** The Cisco Catalyst 9200 Series and the Catalyst 9500 (UADP 2.0 based: C9500-40X, C9500-16X, C9500-24Q, C9500-12Q) models do not support application hosting.

## Hardware Resources for Applications

**Table 1.** Contains the Cisco Catalyst 9000 platform hardware resources for applications:

Cisco Models	Networking (AppGig Port)	Memory (RAM)	CPU	Storage
Catalyst 9300X	2x10G	Up to 8 GB	50% of total CPU (2 CPU Cores)	120/240 GB (USB 3.0 SSD)
Catalyst 9300/L/LM	1x1G	2 GB	25% of total CPU (1 CPU Core)	120/240 GB (USB 3.0 SSD)
Catalyst 9400X	2x10G	Up to 8 GB	25% of total CPU (1 CPU Core)	240 - 960 GB (SATA)
Catalyst 9400	1x1G	Up to 8 GB	25% of total CPU (1 CPU Core)	240 - 960 GB (SATA)
Catalyst 9500X	2x10G	Up to 8 GB	25% of total CPU (1 CPU Core)	240 - 960 GB (SATA)
Catalyst 9500 High performance	Management Port <sup>*</sup>	Up to 8 GB	25% of total CPU (1 CPU Core)	240 - 960 GB (SATA)
Catalyst 9600X	Management Port <sup>*</sup>	Up to 8 GB	25% of total CPU (1 CPU Core)	240 - 960 GB (SATA)
Catalyst 9600	Management Port <sup>*</sup>	Up to 8 GB	25% of total CPU (1 CPU Core)	240 - 960 GB (SATA)

The resources, CPU, memory and vCPU can be reserved with custom resource profile if the default options are not sufficient. For device resource limits refer to this information:

Platform	Memory (GB)	vCPUs	CPU Units	USB Back Storage (GB)	M2 SATA Storage (GB)
Catalyst 9300	2	2	7400	120	NA
Catalyst 9400	4-10	2	7400	NA	960
Catalyst 9500	8	2	7400	120	NA
Catalyst 9500 High-performance	8	2	7400	NA	960

- **vCPUs:** This indicates the maximum number of virtual CPUs that a single application can utilize concurrently.
- **CPU Units:** Represents the total CPU load resource allocated to application hosting. Each application specifies its guaranteed minimum CPU load required to ensure reliable operation.
- **USB Back Storage:** Refers to the back-panel recessed USB 3.0 slot. Application hosting is supported only on this back-panel USB 3.0 slot.
- **M.2 SATA Storage:** Refers to an internal solid-state drive (SSD) used for application data storage.
- **NA (Not Applicable):** Indicates that the feature or resource is not available for the specified configuration.

The example illustrates how to create a custom profile:

```
<#root>

Cat9k(config)#
app-hosting appid MYAPP

Cat9k(config-app-hosting)#
app-resource profile custom

Cat9k(config-app-resource-profile-custom)#
cpu 7400

Cat9k(config-app-resource-profile-custom)#
memory 2048

Cat9k(config-app-resource-profile-custom)#
vcpu 2
```

## Application High Availability

Catalyst 9000 switches support application auto-restart feature which can retain the last configured operational state of app in the event of system switchover or restart. This feature is enabled by default and same storage type required on both Active and Standby switches.

**Table 2.** Cisco Catalyst 9000 platform App auto-restart feature

Supported Platforms	IOS XE Release
Catalyst 9300/L StackWise 480/360 (1+1 mode only)	17.2.1
Catalyst 9300LM StackWise 360 (1+1 mode only)	17.9.1
Catalyst 9300X StackWise 1T (1+1 mode only)	17.6.1
Catalyst 9400 Dual Sup (Single Chassis and StackWise Virtual)	17.5.1
Catalyst 9400X Dual Sup (Single Chassis and StackWise Virtual)	17.9.1
Catalyst 9500 High Performance StackWise Virtual	17.5.1
Catalyst 9600 Dual Sup (Single Chassis and StackWise Virtual)	17.5.1
Catalyst 9600X Dual Sup (Single Chassis)	17.9.1

## Application Hosting Framework

Application Hosting framework is not enabled by default. The Cisco Application Framework (CAF), is built by Cisco to manage containerized applications running on any network device. CAF is also known as IOx. Cisco IOx enables the execution of IoT applications at the network edge (fog computing) while ensuring secure connectivity with Cisco IOS software. **It reserves 4GB of disk space for application hosting.** The partition and application data are cleared when IOx is disabled via the CLI or when the IOx infrastructure utilizes an SSD.

This is the Cisco IOS XE CLI Config to enable App hosting infrastructure:

```
Cat9k(config)#iox
```





**Note:** Starting with Cisco IOS XE 16.12.1, IoX packaging is no longer mandatory. Docker apps can be installed as is. This enables users to build and bring their own applications without additional packaging. Application developers can find more information about application hosting on the [Cisco DevNet site](#)

---

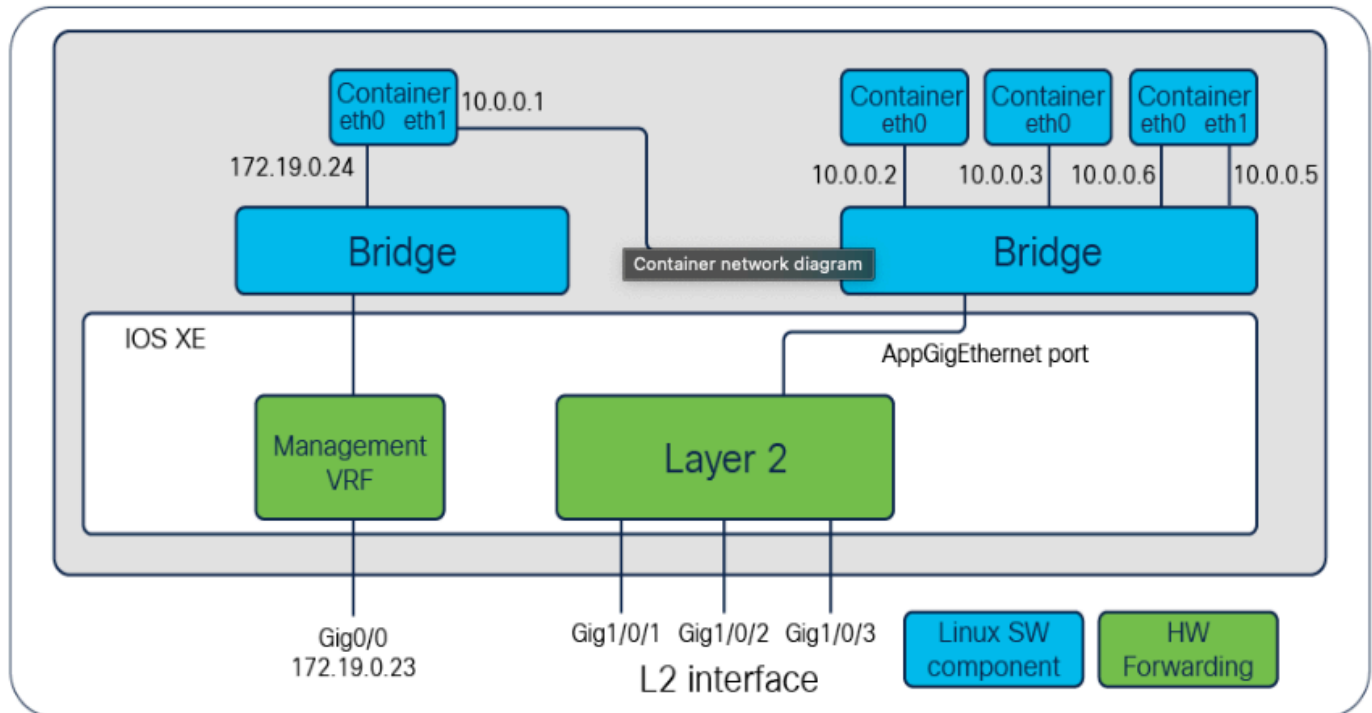
## Container Networking

The image illustrates the connectivity architecture for containers. It highlights all the possible network modes the Cat9k support for app-hosting. They include:

- Front Panel Data port switching using the dedicated, internal **AppGigabitEthernet** interface
- Management Interface (**GigabitEthernet 0/0**)

AppGigabitEthernet is an internal hardware data port which is hardware switched to the front panel data ports. Containers are connected using both the management interface and the front-panel data ports. Within the containers, virtual network interface cards (vNICs) appear as standard Ethernet interfaces, labeled as eth0, eth1, and so on. This design facilitates seamless integration and consistent network configuration across containerized environments.

**Figure 3.** Container network diagram:



An AppGigabitEthernet interface introduced on Cisco Catalyst 9300, Catalyst 9400, and Catalyst 9500X Series switches offers a dedicated application traffic feature. It is an internal hardware data port that is hardware-switched to the front panel data ports. The AppGigabitEthernet interface can be configured as a trunk or VLAN specific interface. For a trunk interface, it is extended to work as a Layer 2 trunk port, and all traffic received by the port is available to the application. For a VLAN interface, the application is connected to a specific VLAN network by specifying the VLAN ID number.

The AppGigabitEthernet; interface is only available on the Cisco Catalyst 9300 series, Catalyst 9400 series and Catalyst 9500X switches. Catalyst 9410 chassis with Supervisor 1 requires disabling Slot 4 port 48 (if applicable) to enable AppGigabitEthernet port. Catalyst 9500 High Performance, and 9600 series switches do not support AppGigabitEthernet interface. The connectivity for applications hosted on these models is achieved through management interface via loopback from any front panel ports.



**Note:** C9300X, C9400X-Sup-2/2XL and C9500X models have 2 x 10G of AppGigabitEthernet ports.

---

## Network for App Hosting

For trunk interface, all traffic received by port is available to App:

```
<#root>
```

```
Cat9k(config)#
```

```
interface AppGigabitEthernet 1/0/1
```

```
Cat9k(config)#
```

```
switchport mode trunk
```

```
Cat9k(config-if)#exit
```

```
Cat9k(config)#  
app-hosting appid MYAPP  
  
Cat9k(config-app-hosting)#  
app-vnic AppGigabitEthernet trunk  
  
Cat9k(config-config-app-hosting-trunk)#  
guest-interface  
  
<guest_interface_id>  
  
Cat9k(config-config-app-hosting-trunk-mode-guest)#  
end
```

For VLAN interface, Application is connected to specific VLAN:

```
<#root>  
  
Cat9k(config)#  
interface AppGigabitEthernet 1/0/1  
  
Cat9k(config)#  
switchport trunk allowed vlan  
  
<vlan_id>  
  
Cat9k(config-if)#  
exit  
  
Cat9k(config)#  
app-hosting appid MYAPP  
  
Cat9k(config-app-hosting)#  
app-vnic AppGigabitEthernet trunk  
  
Cat9k(config-config-app-hosting-trunk)#  
vlan  
  
<vlan_id>
```

```
guest-interface
```

```
<guest_interface_id>
```

```
Cat9k(config-config-app-hosting-trunk-mode-guest)#  
end
```

For management interface, Application is connected to Management port (**GigabitEthernet0/0**):

```
<#root>
```

```
Cat9k(config)#  
interface gigabitEthernet 0/0
```

```
Cat9k(config-if)#  
ip address
```

```
<ip_address> <subnet_mask>
```

```
Cat9k(config-if)#  
exit
```

```
Cat9k(config)#  
app-hosting appid MYAPP
```

```
Cat9k(config-app-hosting)#(config-app-hosting)#  
app-vnic management guest-interface
```

```
<guest_interface_id>
```

```
Cat9k((config-app-hosting-mgmt-gateway)#  
end
```

**IP Address Assignment to App Container:** IP addresses for container interfaces can be explicitly assigned through the switch CLI or obtained dynamically via DHCP.

Configure a static IP address for the App through AppGigabitEthernet:

<#root>

Cat9k(config)#

app-hosting appid MYAPP

Cat9k(config-app-hosting)#

app-vnic AppGigabitEthernet trunk

Cat9k(config-config-app-hosting-trunk)#

vlan

<vlan\_id>

guest-interface

<guest\_interface\_id>

Cat9k(config-config-app-hosting-vlan-access-ip)#

guest-ipaddress

<ip\_address>

netmask

<subnet\_mask> <-- Container IP Address

Cat9k(config-config-app-hosting-vlan-access-ip)#

exit

Cat9k(config-config-app-hosting-trunk)#

exit

Cat9k(config-app-hosting)#

app-default-gateway

<default\_gateway>

guest-interface

<guest\_interface\_id>

```
Cat9k(config-app-hosting)#  
exit
```

Configure a static IP address for the App through **GigabitEthernet 0/0**:

```
<#root>
```

```
Cat9k(config)#  
app-hosting appid MYAPP
```

```
Cat9k(config-app-hosting)#  
app-vnic management guest-interface
```

```
<guest_interface_id>
```

```
Cat9k(config-app-hosting-mgmt-gateway)#  
guest-ipaddress
```

```
<ip_address>
```

```
netmask
```

```
<subnet_mask>
```

```
Cat9k(config-app-hosting-mgmt-gateway)#  
exit
```

```
Cat9k(config-app-hosting)#  
app-default-gateway
```

```
<default_gateway>
```

```
guest-interface
```

```
<guest_interface_id>
```

```
Cat9k(config-app-hosting)#  
exit
```

Configure a dynamic IP address for the App (**DHCP**):

```
<#root>
```

```
Cat9k(config)#
```

```
app-hosting appid MYAPP
```

```
Cat9k(config-app-hosting)#
```

```
app-vnic AppGigabitEthernet trunk
```

```
Cat9k(config-config-app-hosting-trunk)#
```

```
vlan
```

```
<vlan_id>
```

```
guest-interface
```

```
<guest_interface_id>
```

```
Cat9k(config-config-app-hosting-vlan-access-ip)#
```

```
end
```

```
Cat9k#
```

Or through Interface

```
GigabitEthernet 0/0
```

```
Cat9k(config)#
```

```
app-hosting appid MYAPP
```

```
Cat9k(config-app-hosting)#
```

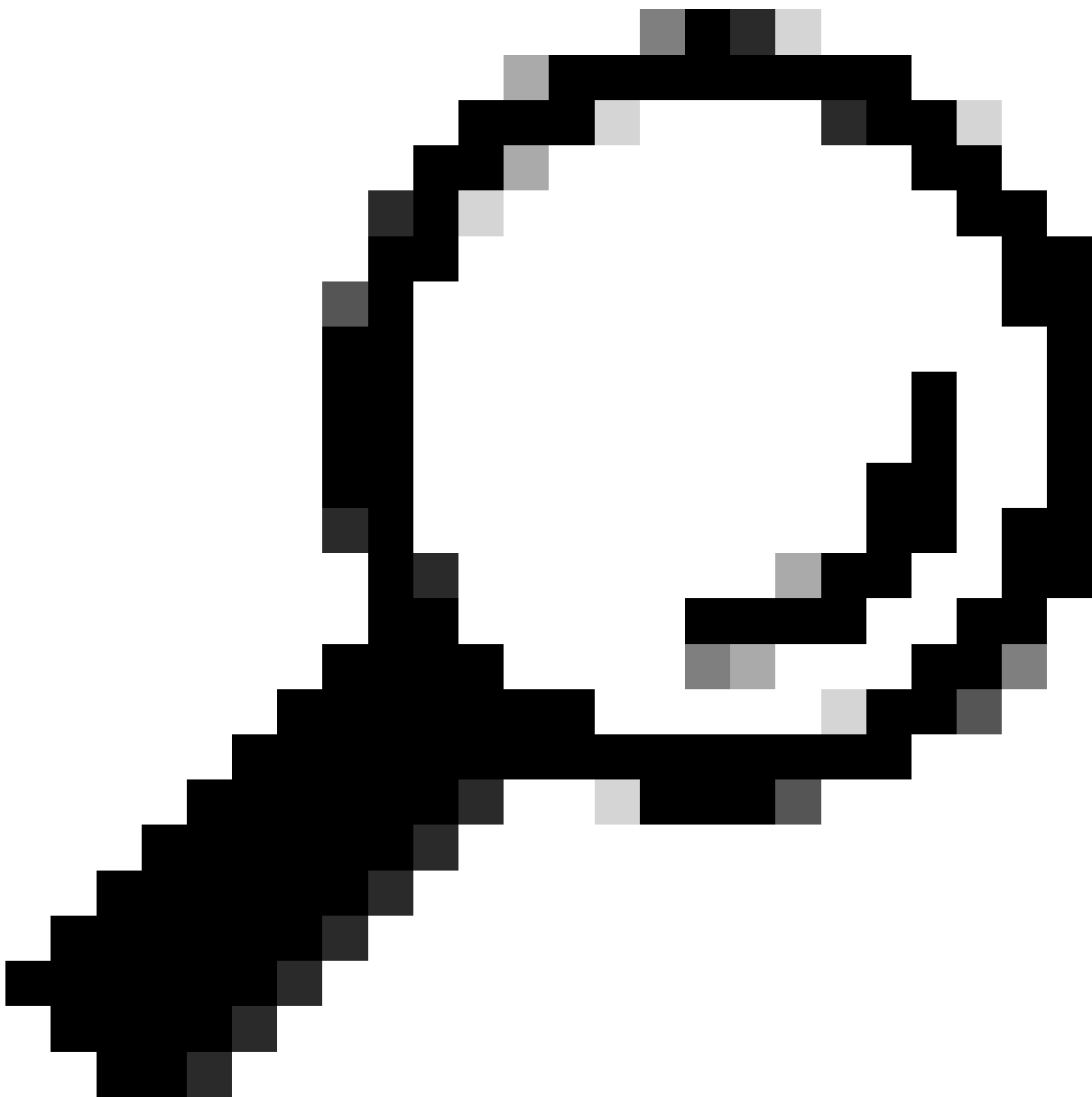
```
app-vnic management guest-interface
```

```
<guest_interface_id>
```

```
Cat9k(config-app-hosting-mgmt-gateway)#
```

```
end
```





**Tip:** Ensure networking is configured correctly before installing apps. **Stop**, **Deactivate**, and **Uninstall** if changes are required, then restart.

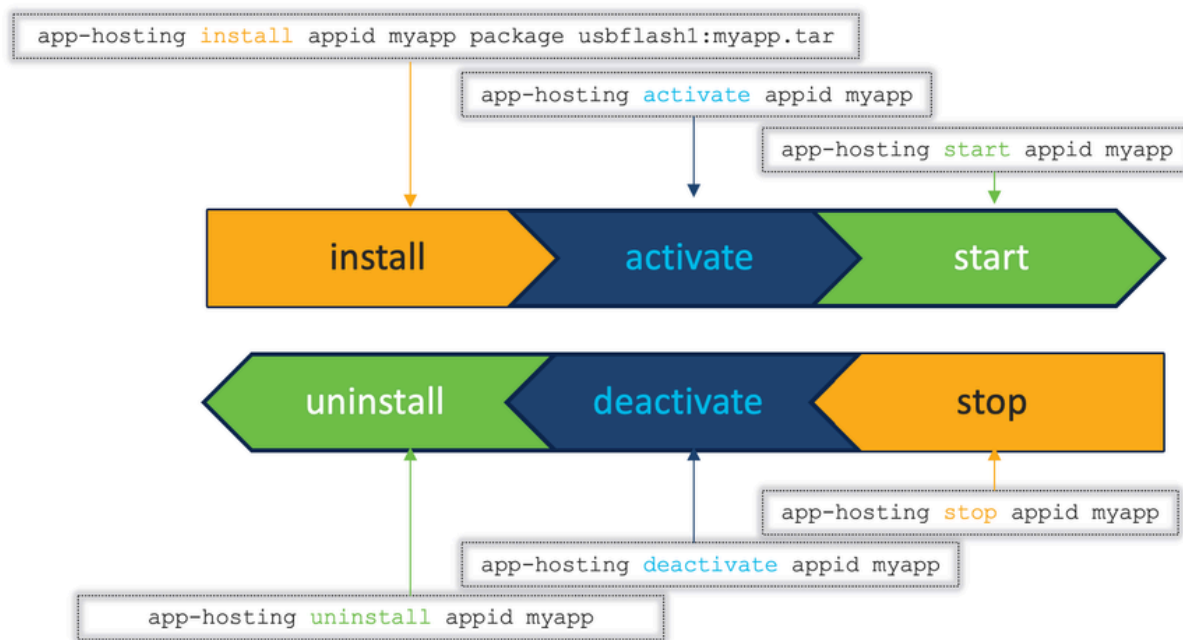
---

## Lifecycle for and Application

The application lifecycle on Cisco Catalyst 9000 switches consists of three stages, as illustrated in Figure 4:

- **Install:** During this stage, the application is installed on the device. However, the resources required by the application are not yet allocated.
- **Activate:** Hardware resources necessary for the application are committed to ensure its functionality.
- **Start:** The application transitions into an operational state, becoming fully active and functional.

**Figure 4.** Application lifecycle management:



## Docker Runtime Options

If the Container Application is required to have run time options, which are passed as command line options, like controller IP, Data directory and others, then, those options can be configured using **app-resource docker** command.

The system supports multiple lines of run-option string configuration. Here are the key considerations for the user:

- The user can enter/copy-paste up to a maximum of 30 lines of docker run options.
- The system generates a concatenated string from the strings in lines 1 through 30 in that order.
- Each string in each line of run-option can have a maximum of 235 characters.
- Each string can have more than one complete docker run-option as long as they are all contained in the 235 characters.
- Please note that no run-option string in any line can be split to the next line.
- There is no need to terminate the string in each line with a space. The system auto-generates a space for each line when it concatenates all the existing run-option strings into a single string.
- If the user makes any changes to the run option, the user needs to stop, deactivate, activate and start the application again for the new run options to take effect.

This is an example of a Docker option and its possible equivalent configuration on a Cat9k switch:

```
docker run -v $(APP_DATA):/data --entrypoint startup.sh
```

Configuration on a Catalyst 9000 switch:

```
<#root>
```

```
Cat9k(config)#
```

```
app-hosting appid
```

MYAPP

```
Cat9k(config-app-hosting)#
```

```
app-resource docker
```

```
Cat9k(config-app-hosting-docker)#
```

```
run-opts 1 "-v $(APP_DATA):/data"
```

```
Cat9k(config-app-hosting-docker)#
```

```
run-opts 2 "--entrypoint startup.sh"
```

For external persistent data storage, use the command: **run-opts 1 "-v /vol/usb1/iox\_host\_data\_share:/(APP\_DATA)"**

<#root>

```
Cat9k(config)#
```

```
app-hosting appid MYAPP
```

```
Cat9k(config-app-hosting)#
```

```
app-resource docker
```

```
Cat9k(config-app-hosting-docker)#
```

```
run-opts 1 "-v /vol/usb1/iox_host_data_share:/(APP_DATA)"
```

To remove a run option line from the configuration, execute a "no" command similar to this example:

<#root>

```
Cat9k(config)#
```

```
app-hosting appid MYAPP
```

```
Cat9k(config-app-hosting)#
```

```
app-resource docker
```

```
Cat9k(config-app-hosting-docker)#
```

```
no run-opts 1 "-v /vol/usb1/iox_host_data_share:/(APP_DATA)"
```

To delete all the run-options in the configuration of an application, you can execute the **no configuration** command:

<#root>

```
Cat9k(config)#  
app-hosting appid MYAPP  
  
Cat9k(config-app-hosting)#  
  
no  
  
app-resource docker
```

## List of Unsupported Docker Runtime Options

The Docker command-line options listed are not supported in the Application Hosting Framework due to security reasons or because the options are not applicable to the Linux platform.

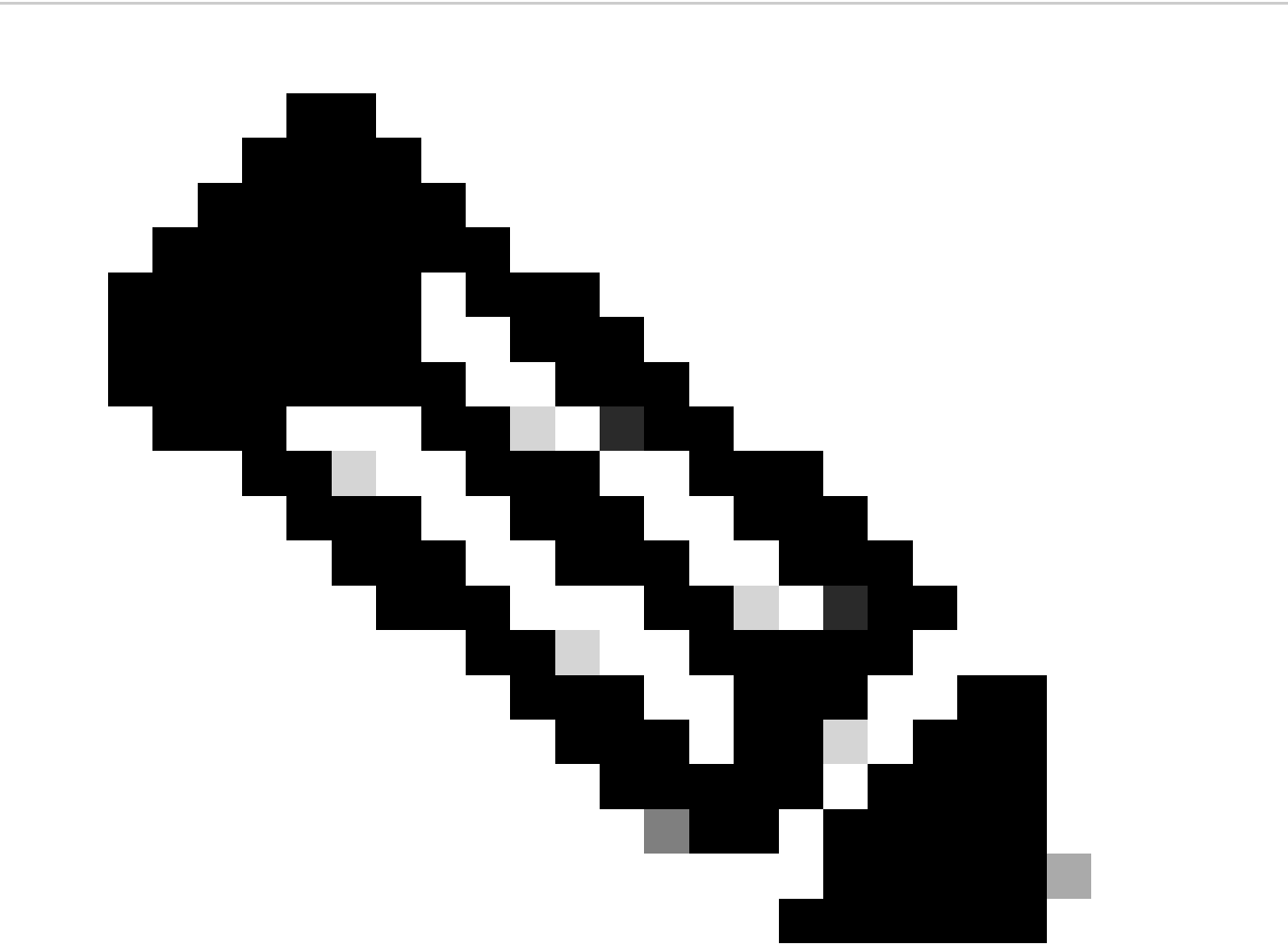
Docker Option	Description
--attach	Attach to STDIN, STDOUT, or STDERR.
--blkio-weight-device	Specify block IO weight (relative device weight).
--cgroup-parent	Optional parent cgroup for the container.
--cidfile	Write the container ID to a specified file.
--cpu-count	Specify the CPU count (Windows only).
--cpu-percent	Specify the CPU percentage (Windows only).
--cpus	Specify the number of CPUs (API 1.25+).
--device-cgroup-rule	Add a rule to the cgroup allowed devices list.
--device-read-bps	Limit the read rate (bytes per second) from a device.
--device-read-iops	Limit the read rate (I/O per second) from a device.
--device-write-bps	Limit the write rate (bytes per second) to a device.
--device-write-iops	Limit the write rate (I/O per second) to a device.

--disable-content-trust	Skip image verification.
--env-file	Load environment variables from a file.
--interactive (-i)	Keep STDIN open even if not attached.
--io-maxbandwidth	Maximum IO bandwidth limit for the system drive (Windows only).
--io-maxiops	Maximum IOps limit for the system drive (Windows only).
--ip	Specify an IPv4 address (example: 192.168.100.100).
--ip6	Specify an IPv6 address (example: 2001:db8::44).
--isolation	Specify container isolation technology.
--link	Add a link to another container.
--name	Assign a name to the container.
--oomkilldisable	Disable the OOM killer for the container.
--pid	Specify the PID namespace to use.
--platform	Specify the platform (experimental; API 1.32+).
--privileged	Grant extended privileges to the container.
--runtime	Specify the runtime to use for the container.
--storage-opt	Define storage driver options for the container.
--sysctl	Specify sysctl options.
--tty (-t)	Allocate a pseudo-TTY.

--usersns	Specify the user namespace to use.
--uts	Specify the UTS namespace to use.
--volume-driver	Specify an optional volume driver for the container.

## Configuration

The subsequent sections explain how to configure and deploy an APP on Catalyst 9000.



**Note:** Explore the catalog of partner solutions that work seamlessly on Catalyst 9000 series switches [Application Marketplace](#)

**Download** the Docker image and copy to your Cisco switch using SCP, FTP, TFTP, or USB storage:

<#root>

```
Cat9k#
```

```
dir usbflash1:/
```

```
Directory of usbflash1:/
```

```
17      -rw-          5843233 Jan 23 2025 20:50:01 +00:00  
MYAPP.tar
```

**Configure** the network parameters (For this example the AppGigabitEthernet interface is used, the **192.168.1.10/24**, under VLAN 10 and use Google resolver):

```
<#root>
```

```
Cat9k(config)#
```

```
interface AppGigabitEthernet 1/0/1
```

```
Cat9k(config-if)#
```

```
switchport trunk allowed vlan 10
```

```
Cat9k(config-if)#
```

```
exit
```

```
Cat9k(config)#
```

```
app-hosting appid MYAPP
```

```
Cat9k(config-app-hosting)#
```

```
app-vnic
```

```
AppGigabitEthernet
```

```
trunk
```

```
Cat9k(config-config-app-hosting-trunk)#
```

```
vlan 10 guest-interface 0
```

```
Cat9k(config-config-app-hosting-vlan-access-ip)#
```

```
guest-ipaddress 192.168.1.10 netmask 255.255.255.0
```

```
Cat9k(config-config-app-hosting-vlan-access-ip)#
```

```
exit
```

```
Cat9k(config-config-app-hosting-trunk)#
```

```
exit
```

```
Cat9k(config-app-hosting)#
```

```
app-default-gateway 192.168.1.1 guest-interface 0
```

```
Cat9k(config-app-hosting)#
```

```
name-server 8.8.8.8
```

```
Cat9k(config-app-hosting)#
```

```
exit
```

**Enable** the IOx framework on the switch and wait until all the services are running:

```
<#root>
```

```
Cat9k(config)#
```

```
iox
```

```
Cat9k#
```

```
show iox-service
```

```
IOx Infrastructure Summary:
```

```
-----  
IOx service (CAF)           : Running  
IOx service (HA)            : Running  
IOx service (IOxman)        : Running  
IOx service (Sec storage)    : Running  
Libvirtd 5.5.0               : Running  
Dockerd v19.03.13-ce        : Running  
Application DB Sync Info    : Available  
Sync Status                  : Disable
```

**Set up** the required Docker run options for the specific App:

```
<#root>
```

```
Cat9k(config)#
```

```
app-hosting appid MYAPP
```

```
Cat9k(config-app-hosting)#
```

```
app-resource docker
```

```
Cat9k(config-app-hosting-docker)#
```



```
run-opts 1 "<docker_opts_1>"
```

```
Cat9k(config-app-hosting-docker)#
```

```
run-opts 2 "<docker_opts_2>"
```

**Install** the application from the SSD and verify its deployment:

```
<#root>
```

```
Cat9k#
```

```
app-hosting install appid MYAPP package usbflash1:MYAPP.tar
```

Installing package 'usbflash1:MYAPP.tar' for 'MYAPP'. Use 'show app-hosting list' for progress.

```
Cat9k#
```

```
show app-hosting list
```

App id	State
-----	
MYAPP	
DEPLOYED	

**Activate** the application and verify its state:

```
<#root>
```

```
Cat9k#
```

```
app-hosting activate appid MYAPP
```

```
MYAPP activated successfully
```

```
Current state is: ACTIVATED
```

```
Cat9k#
```

```
show app-hosting list
```

App id	State
-----	
MYAPP	
ACTIVATED	

**Start** the application and verify its running state:

<#root>

Cat9k#

```
app-hosting start appid MYAPP
```

MYAPP started successfully

Current state is: RUNNING

Cat9k#

```
show app-hosting list
```

App id	State
-----	
MYAPP	
RUNNING	

**Save** your configuration changes to ensure that they persist across reboots:

<#root>

Cat9k#

```
app-hosting start appid MYAPP
```

MYAPP started successfully

Current state is: RUNNING

Cat9k#

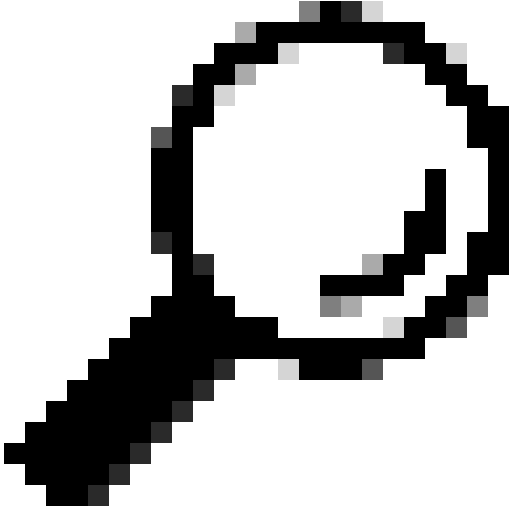
```
show app-hosting list
```

App id	State
-----	
MYAPP	
RUNNING	

## Troubleshoot

This table lists the various commands that can be used to troubleshoot App Hosting:

Command	Purpose
show iox-service	Displays the interface error counters

show app-hosting list	Displays the list of the appliance(s) installed
show app-hosting detail appid MYAPP	Displays detailed information about appliance
show app-hosting resource	Displays the available resources
show app-hosting utilization appid MYAPP	Displays utilization information about appliance
app-hosting move appid MYAPP log to bootflash:	Use this command to capture App specific tracelogs if they exist.
app-hosting move system techsupport to ?  bootflash:            Destination path crashinfo-1:        Destination path crashinfo:           Destination path flash-1:             Destination path flash:                Destination path webui:                Destination path	Use this command to move the system techsupport to an alternate directory.
show app-hosting infra	Use this command to check if signature verification is enabled.
	 <p><b>Tip:</b> The application signature verification can be disabled only when the application hosting is using USB/SSD as media.</p>
app-hosting verification ? disable      App verification disable	Use this command to enable/disable signature verification (form CLI privilege mode).

enable      App verification enable	
app-hosting connect appid MYAPP session	Use this command to access the app console and verify the status of processes within the container (from CLI privilege mode).

**Modify the Docker Container**

If Docker requires modification, adhere to these procedures:

- **Stop** the application:

```
<#root>
Cat9k#
app-hosting stop appid MYAPP

MYAPP stopped successfully

Current state is:
STOPPED
```

- **Deactivate** the application:

```
<#root>
Cat9k#
app-hosting deactivate appid MYAPP

MYAPP deactivated successfully

Current state is:
DEPLOYED
```

- **Modify** the Docker options:

```
<#root>
Cat9k#
app-hosting start appid MYAPP

Cat9k(config)#
```

```
app-hosting appid MYAPP
```

```
Cat9k(config-app-hosting)#
```

```
app-resource docker
```

```
Cat9k(config-app-hosting-docker)#
```

```
prepend-pkg-opts
```

```
Cat9k(config-app-hosting-docker)#
```

```
<run-opts command>
```

```
Cat9k(config-app-hosting-docker)#
```

```
exit
```

```
Cat9k(config-app-hosting)#
```

```
exit
```

```
Cat9k(config)#
```

```
exit
```

- **Reactivate** the application:

```
<#root>
```

```
Cat9k#
```

```
app-hosting activate appid MYAPP
```

```
MYAPP activated successfully
```

```
Current state is:
```

```
STOPPED
```

- **Start** the application:

```
<#root>
```

```
Cat9k#
```

```
app-hosting start appid MYAPP
```

```
MYAPP started successfully
```

Current state is:

**RUNNING**

## **Related Information**

- [Application Hosting on the Cisco Catalyst 9000 Series Switches White paper](#)
- [Application Marketplace](#)
- [Application Hosting in the Enterprise](#)