Application Hosting

A hosted application is a software as a service (SaaS) solution, and it can be run remotely using commands. Application hosting gives administrators a platform for leveraging their own tools and utilities.

Note Application hosting supports only Docker applications.

This module describes the Application Hosting feature and how to enable it.

Prerequisites for Application Hosting

- Applications hosted by Catalyst 9000 Series Switches must be configured in the underlay Switch Virtual Interface (SVI). This applies to Cisco Software-Defined Access deployments as well.

Restrictions for Application Hosting

- Application hosting is not virtual routing and forwarding aware (VRF-aware).
- In releases prior to Cisco IOS XE Amsterdam 17.3.3, application hosting requires dedicated storage allocations, and is disabled on the bootflash.

  In Cisco IOS XE Amsterdam 17.3.3 and later releases, application hosting is enabled on the bootflash, however, only Cisco-signed applications are hosted.
- The front-panel Universal Serial Bus (USB) stick is not supported.

  Cisco Catalyst 9300 Series Switches support only back-panel Cisco-certified USB.

- Cisco Catalyst 9500-High Performance Series Switches and Cisco Catalyst 9600 Series Switches do not support front-panel USB for application hosting.
- Cisco Catalyst 9500 and 9500-High Performance Series Switches and Cisco Catalyst 9600 Series Switches do not support AppGigabitEthernet interfaces.
- Cisco Catalyst 9410R Switches do not support application-hosting in release prior to Cisco IOS XE Bengaluru 17.5.1.

  Configure the **enable** command on the AppGigabitEthernet interfaces to enable application hosting on Cisco Catalyst 9410R Switches.
- Cisco Catalyst 9200CX Series Switches do not support the Management interface, AppGigabitEthernet interface, or VirtualPortGroup interface. Applications or scripts running in the Guest Shell will not be able to communicate with the external network.

Information About Application Hosting

This section provides information about Application Hosting.

Need for Application Hosting

The move to virtual environments has given rise to the need to build applications that are reusable, portable, and scalable. Application hosting gives administrators a platform for leveraging their own tools and utilities. An application, hosted on a network device, can serve a variety of purposes. This ranges from automation, configuration management monitoring, and integration with existing tool chains.

Note In this document, *container* refers to Docker applications.

Cisco IOx Overview

Cisco IOx (IOs + linuX) is an end-to-end application framework that provides application-hosting capabilities for different application types on Cisco network platforms. The Cisco Guest Shell, a special container deployment, is one such application, that is useful in system deployment.

Cisco IOx facilitates the life cycle management of applications and data exchange by providing a set of services that helps developers to package prebuilt applications, and host them on a target device. IOx life cycle management includes distribution, deployment, hosting, starting, stopping (management), and monitoring of applications and data. IOx services also include application distribution and management tools that help users discover and deploy applications to the IOx framework.

Cisco IOx application hosting provides the following features:

- Hides network heterogeneity.
- Cisco IOx application programming interfaces (APIs) remotely manage the life cycle of applications hosted on a device.
- Centralized application life cycle management.

- Cloud-based developer experience.

Application Hosting Overview

The Cisco application-hosting framework is an IOx Python process that manages virtualized and container applications that run on devices.

Application hosting provides the following services:

- Launches designated applications in containers.
- Checks available resources (memory, CPU, and storage), and allocates and manages them.
- Provides support for console logging.
- Provides access to services through REST APIs.
- Provides a CLI endpoint.
- Provides an application-hosting infrastructure referred to as Cisco Application Framework (CAF).
- Helps setup platform-specific networking (packet-path) through management interfaces.

  Data ports are supported on platforms that have AppGigabitEthernet port functionality.

The application-hosting container that is referred to as the virtualization environment is provided to run a guest application on the host operating system. The Cisco IOS-XE virtualization services provide manageability and networking models for running a guest application. The virtualization infrastructure allows an administrator to define a logical interface that specifies the connectivity between the host and the guest. Cisco IOx maps the logical interface into a Virtual Network Interface Card (vNIC) that the guest application uses.

Applications that are to be deployed in the containers are packaged as TAR files. The configuration that is specific to these applications is also packaged as part of the TAR files.

The management interface on the device connects the application-hosting network to the Cisco IOS management interface. The Layer 3 interface of the guest application receives the Layer 2-bridged traffic from the Cisco IOS management interface. The management interface connects to the container interface through the management bridge. The IP address of the application must be on the same subnet as the management interface IP address.

Note On all Cisco Catalyst stack and stackwise virtual models (all software versions), Guest Shell and the AppGigabitEthernet interface operate only on the active switch in the stack. Therefore, the AppGigabitEthernet interface configuration must be applied to the AppGigabitEthernet interface on all the switches in the stack. If the configuration is not applied to all the switches, the AppGigabitEthernet interface on the switch will not work after a switchover.

Cisco Catalyst 9000 Series Switches support multiple applications when hosted on the SSD. The applications must meet the following criteria:

- Cisco-signed
- Meet the switching infrastructure requirements:

- Network configuration on AppGigabitEthernet ports does not create a conflict between the applications.
- Enough resources are available to run the applications.

Multiple applications cannot be deployed if an application consumes all the available App-hosting resources. For example, if one application consumes all the compute and run time resources, other applications are prevented from getting installed on the device.

Application Hosting on Front-Panel Trunk and VLAN Ports

Front-panel VLAN and trunk ports are supported for application hosting. Layer 2 traffic is delivered through these ports to software components that run outside of the Cisco IOS daemon.

For application hosting, you can configure the front-panel port as either a trunk interface or a VLAN-specific interface. When using as a trunk interface, the front-panel port is extended to work as a Layer 2 trunk port, and all the traffic received by the port is available to the application. When using the port as a VLAN interface, the application is connected to a specific VLAN network.

Note When using a back-panel USB or an M2 SATA drive for application hosting, the storage medium should be formatted as an *ext4* file system.

Application Hosting on Cisco Catalyst 9300 Series Switches

This section describes application-hosting on Cisco Catalyst 9300 Series Switches.

For application hosting, Cisco Catalyst 9300 Series Switches support the management interface and front-panel ports.

The USB 3.0 SSD is enabled on Cisco Catalyst 9300 Series Switches. The USB 3.0 SSD provides an extra 120 GB storage for application hosting. For more information, see the "Configuring USB 3.0 SSD" chapter in the *Interfaces and Hardware Configuration Guide*.

The following two types of networking applications are supported:

- Control plane: Applications that access the management interface.
- Data plane: Applications that access the front-panel ports.

Front-Panel App Hosting on Cisco Catalyst 9300X Series Switches

Front-panel application hosting is enabled on Cisco Catalyst 9300X Series Switches in Cisco IOS XE Bengaluru 17.6.1.

Applications can use dedicated front-panel ports for hosting. Use the **app-vnic AppGigabitEthernet port** command to specify the port to be used for application hosting. Both the front-panel ports can be attached to the same Layer 2 application.

These switches support application hosting in both access mode and trunk mode. Application hosting can be enabled on both modes simultaneously.

Note Any configuration done under the **app-vnic** command can be rejected during activation.

Table 1: Sample Configuration Scenarios for App Hosting in Access and Trunk Modes

Scenario	Supported/Unsupported
Single application with two front-panel ports in access mode.	Supported. No overlapping VLANs.
Single application with two front-panel ports in trunk mode.	Supported. No overlapping VLANs.
Single application with two front-panel ports in trunk and access modes.	Supported. No overlapping VLANs.
Single application with two front-panel ports in trunk mode with the default app-gateway configured.	Supported. The same application with two interfaces is configured in different subnets; but the default gateway is connected to one VLAN, which has external connectivity.
Single application with two front-panel ports in trunk and access modes with an overlapping VLAN.	Not a valid configuration. VLAN overlapping with both ports.
Single application in access mode and two front-panel ports configured on the same VLAN.	Not a valid configuration.
Single application in trunk mode and two front-panel ports configured on an overlapping VLAN range.	Not a valid configuration. The traffic is not isolated, and the VLAN range is overlapping.
Single application in trunk mode and two front-panel ports configured on an overlapping VLAN range.	Not a valid configuration. This configuration will be rejected during activation. Both the front-panel ports are in trunk mode, so any VLAN can be used. However, the same VLAN is configured for both the ports, and as a result, the VLAN overlaps with both the ports. **Note** The same scenario applies for access mode.
Single application in trunk and access modes, and front-panel ports with an overlapping VLAN.	Not a valid configuration. The same VLAN is configured in trunk mode and access mode. Because of the configuration, the VLAN overlaps with both ports.

Scenario	Supported/Unsupported
Multiple application in trunk mode.	Not a valid configuration. The traffic is not isolated.
Two applications, one in trunk mode and the other in access mode.	Not a valid configuration. Overlapping VLAN.

High Availability on Cisco Catalyst 9300X Series Switches

With mixed mode stacking available on Cisco Catalyst 9300X Series Switches, the active and standby devices use the 1+1 redundancy for application hosting. Mixed mode support is when different model variants and different network modules are used in a stack.

When Cisco Catalyst 9300X Series Switches and Cisco Catalyst 9300 Series Switches are stacked, one of the two front-panel ports on Cisco Catalyst 9300X Series Switches are dynamically disabled. Only the AppGigabitEthernet 1/0/1 interface is displayed as enabled.

This section describes some of the high availability scenarios:

Stack Mode	Apps	Ports Used	Behavior
Cisco Catalyst 9300X Series Switch active + Cisco Catalyst 9300X Series Switch standby	2	1	Supported
Cisco Catalyst 9300X Series Switch active + Cisco Catalyst 9300X Series Switch standby	1	1	Supported
Cisco Catalyst 9300X Series Switch active + Cisco Catalyst 9300 Series Switch standby	1	1 Only if port 1 is used.	Supported. This configuration is supported, if port 1 is configured by using the **app-vnic Appgigabitethernet port 1 trunk** or the **app-vnic AppgigabitEthernet trunk** commands. If a port number is not specified, the default port 1 is used, when a switchover happens.

Stack Mode	Apps	Ports Used	Behavior
Cisco Catalyst 9300X Series Switch active + Cisco Catalyst 9300 Series Switch standby	1	2	Not supported. In this scenario, when a switchover happens, the new active will not have two front-panel ports, and the app configuration fails. After the switchover, the application is not restarted on Cisco Catalyst 9300 Series Switch, because only one front-panel port is configured, and this configuration fails. The application must be reconfigured using the available front-panel port.
Cisco Catalyst 9300X Series Switch active + Cisco Catalyst 9300 Series Switch standby	2	2	Not supported. **Note** Two applications, for example, app1 and app2 are running, with each application using a different front-panel port, for example, port1 and port2 respectively. After a switchover, app1 on front-panel port1 starts on Cisco Catalyst 9300 Series Switch in a running state. However, app2 is not started on Cisco Catalyst 9300 Series Switch as there is no front-panel port2.
Catalyst 9300 Series Switch active + Catalyst 9300X Series Switch standby	1 or more	1	Supported. **Note** After a switchover, the application is restarted on Catalyst 9300X Series Switch using the front-panel port.

Application Hosting on Cisco Catalyst 9400 Series Switches

This section describes application-hosting on Cisco Catalyst 9400 Series Switches.

Cisco Catalyst 9400 Series Switches support the management interface and front-panel ports for application hosting. Applications can be hosted on C9400-SSD-240GB, C9400-SSD-480GB, and C9400-SSD-960GB solid state drives (SSDs).

These switches use the M2 SATA module for application hosting. For more information, see the "M2 SATA Module" chapter in the *Interfaces and Hardware Configuration Guide*.

On Cisco Catalyst 9400 Series Switches, applications can be hosted only on active supervisors. After a switchover, the AppGigbitEthernet interface on the newly active supervisor becomes active and can be used for application hosting.

Application Hosting on Cisco Catalyst 9410 Series Switches

In Cisco IOS XE Bengaluru 17.5.1, application hosting is supported on Cisco Catalyst 9410 Series Switches. To enable the AppGigabitEthernet interface for application hosting, configure the **enable** command in interface configuration mode.

Note The **enable** command is available only on Cisco Catalyst 9410 Series Switches.

When using slot 4 of the 48-port linecard for application hosting, the port must be in the default shutdown mode. If slot 4 of the 48-port linecard is active, application hosting is rejected. If the linecard port is disabled, slot 4 of the 48-port linecard is marked as *inactive*.

If slot 4 of the 48-port linecard is populated, the port 4/0/48 will not come up. If linecard 4 is empty or if it is a 24-port linecard, no ports are disabled.

To enable the port (4/0/48), disable application hosting by using the **no iox** command. No system messages are displayed on the console when the port is enabled or disabled.

During an In-Service Software Upgrade (ISSU), the linecard port is not automatically disabled, because the AppGigabitEthernet interface has to be enabled. Before a software downgrade, the AppGigabitEthernet interface must be disabled to disable the front-panel port.

Online Insertion and Removal

Table 2: Online Insertion and Removal (OIR) Scenarios

OIR Scenario	Action
The linecard on slot 4 is empty, and the AppGigabitEthernet interface is enabled.	No port is disabled.
The linecard on slot 4 is a 48-port linecard, and the AppGigabitEthernet interface is enabled.	Port 48 on slot 4 is disabled. After the port is disabled, no configuration is applied to the port. Port 48 is marked as inactive.
The linecard on slot 4 is a 24-port linecard.	No port on slot 4 is disabled.
The linecard on slot 4 is a 48-port linecard that is replaced by a 24-port linecard, and the AppGigabitEthernet interface is enabled.	No port on slot 4 is disabled.
The linecard on slot 4 is a 24-port linecard that is replaced by a 48-port linecard, and the AppGigabitEthernet interface is enabled.	Port 48 on slot 4 is disabled.

OIR Scenario	Action
During OIR, the standby Supervisor becomes the new active, and the front-panel port on the new active is used for app hosting.	No state change will happen to port 48 on slot 4. The standby Supervisor OIR has no effect on the active Supervisor front-panel port.

Cisco StackWise Virtual

This section describes the scenarios when uplink ports in a dual Supervisor are used as StackWise Virtual links:

- When application hosting is enabled, and port 48 on linecard 4 is not up, it is disabled on both the active and standby chassis.
- If the link is up on either the active or standby chassis on port 48 linecard 4, then the **enable** command is rejected.
- If port 48 on linecard 4 is used as a dual-active detection (DAD) link, remove the DAD link, and configure it on another port.
- If port 48 on linecard 4 is used as a StackWise Virtual link, and the front-panel port must be enabled, remove the StackWise Virtual link on port 48 and use another port as the StackWise Virtual link. Port 48 on linecard 4 cannot be used as a StackWise Virtual or DAD link.

Application Hosting on Cisco Catalyst 9500 Series Switches

Cisco Catalyst 9500-High Performance Series Switches support only M2 SATA modules, SSD-240G, SSD-480G, and SSD-960 (C9k-F1-SSD-240GB). Front-panel USB is not supported.

For more information, see the "M2 SATA Module" of the *Interface and Hardware Components Configuration Guide, Cisco IOS XE Amsterdam 17.2.x (Catalyst 9500 Switches)*.

In Cisco IOS XE Cupertino 17.7.1, Cisco Catalyst 9500X Series Switches support application hosting on AppGigabitEthernet interfaces. Application Hosting is supported on the M2 SATA modules: SSD-240G, SSD-480G, and SSD-960 (C9k-F1-SSD-240GB).

Application Hosting on Cisco Catalyst 9600 Series Switches

Cisco Catalyst 9600 Series Switches support only M2 SATA modules for application hosting; front-panel USB is not supported. The following M2 SATA modules are supported: SSD-240G, SSD-480G, and SSD-960 (C9k-F2-SSD-240GB)

For more information, see the "M2 SATA Module" of the *Interface and Hardware Components Configuration Guide, Cisco IOS XE Amsterdam 17.2.x (Catalyst 9600 Switches)*.

Autotransfer and Auto-Install of Apps from Internal Flash to SSD

When IOx is enabled, it chooses the best available media, and starts the IOx service using that media. IOx also selects the media to run the applications at startup.

When IOx is restarted and a different media is selected, all applications (only Docker applications are supported.) must be migrated to the new media, and the containers must be restored to the same state as before the change. All persistent data and volumes attached to an application must also be migrated.

During a restart, IOx selects the media in the following order of precedence:

1. Harddisk
2. Flash

   Flash only supports Guest Shell; no other applications are allowed.

Use Cases

This section describes a couple of use cases during autotransfer and auto-install of applications.

Table 3: Use Cases for the AutoTransfer and Auto-Install of Applications

Use Case	Result
SSD is plugged in while IOx is running on flash.	If the SSD is plugged in while IOx is already running, there is no impact to the running applications or to IOx. IOx is migrated to the SSD only when IOx is restarted by disabling and then enabling IOx through the CLI, or due to a system restart.
System reboots, while IOx data is being copied to the new media.	While IOx data is getting migrated from one media to another, and the system reboots, the migration process will continue, when the system restarts. The data from the old media is deleted only when the copy operation is complete.

Native Docker Container: Application Auto-Restart

The Application Auto-Restart feature helps applications deployed on platforms to retain the last configured operational state in the event of a system switchover or restart. The underlying hosting framework is also retained during switchovers. This feature is enabled by default, and cannot be disabled by users.

The persistent data of applications is not synchronized; only secure data storage and persistent data that is known to Cisco Application Framework (CAF) is synchronized.

IOx media present on the active and standby devices must be in-sync to restart IOx in the same state upon a switchover or system restart.

Cisco Catalyst 9300 Series Switches only support Solid State Drive (SSD) for application hosting. When a new SSD is inserted, it needs to be brought up to the same sync state as the others. The standby device must have an SSD that is compatible with IOx for application auto-restart synchronization to work.

The output of the **show iox-service** command displays the status of the synchronization.

The Application Auto-Restart feature is supported only on Cisco Catalyst 9300 Series Switches.

Application Auto-Restart Scenarios

This section describes various application auto-restart scenarios:

Table 4: Application Auto-Restart Scenarios

Scenario	Single Media in the Active Device	Media in the Active and Standby Devices
System bootup	Starts IOx and the application at system bootup. The USB SSD is visible immediately because it is a local device. No synchronization happens at this time.	Starts IOx and the application on system bootup. Does a bulk synchronization of the existing information to the standby device.
Switchover	Media is not found on the new active device. IOx starts on the system flash with no previously installed applications and with minimum capabilities.	Starts IOx and the application in the previous state on the new active device after the system switchover (SSO). Does a bulk synchronization of the information to the new standby device after it boots up.
Bootup or switchover: USB SSD is present on a member device.	No synchronization of the SSD present in member devices. The member SSD is not used to host IOx and applications.	No synchronization of the SSD present in member devices. The member SSD is not used to host IOx and applications.
Device removal: Local USB SSD is removed from the active device.	When the local USB SSD is removed, IOx takes care of the graceful exit. User-triggered IOx restart is required once SSD is plugged back in the active device.	IOx takes care of the graceful exit. Since IOx operates only on the local disk, the standby SSD is not used to start IOx. User-triggered IOx restart is required once SSD is plugged back in the active device.
Device removal: USB SSD is removed from the standby device.	NA	IOx synchronization operation fails. IOx is no longer SSO ready.
Device removal: Remote USB SSD is removed from a remote member device.	IOx does not use any member SSD, and hence, there is no impact.	IOx does not use any member SSD, and hence, there is no impact.
Device going down: The active device on which IOx is running goes down.	Media is not found on the new active device. IOx starts up on the system flash with no previously installed applications and with minimum capabilities.	Starts IOx and applications in the state before the SSO on the new active device. Does a bulk synchronization of the information to the new standby device once it boots up.
Designated active-standby device change (stack environment 1:1)	The change is reflected after the reboot. IOx starts from the new active device after the reboot.	The change is reflected after the reboot. IOx starts from the new active device after the reboot.

Application Auto-Restart on Cisco Catalyst 9300 Series Switches

This section describes how application auto-restart works on Cisco Catalyst 9300 Series Switches in a multimember stack:

On Cisco Catalyst 9300 Series Switches, application auto-restart is supported in 1+1 switch redundancy or StackWise Virtual modes that assign the active and standby roles to specific devices in the stack.

Application auto-restart is not supported when the switch stack in is N+1 mode. If the device is in N+1 mode, the following log message is displayed on the console:

```
Feb 5 20:29:17.022: %IOX-3-IOX_RESTARTABITLITY: Switch 1 R0/0: run_ioxn_caf:Stack is in N+1
 mode,
disabling sync for IOx restartability
```

IOx uses a Cisco-certified USB3.0 flash drive in the back-panel USB port as storage for application hosting. This media may not be present in all the stack members.

Data is synced using the rsync utility from the active to the standby device.

Supported Network Types

This section lists the types of networks supported on Cisco Catalyst Switches.

Table 5: Supported Network Types

Network Type	Supported Platform and Release
Management port	• Catalyst 9300 Series Switches and C9300L in Cisco IOS XE Gibraltar 16.12.1 • Catalyst 9400 Series Switches in Cisco IOS XE Amsterdam 17.1.1 • Catalyst 9500 Series Switches and Catalyst 9500-High Performance Series Switches in Cisco IOS XE Amsterdam 17.2.1 • Catalyst 9600 Series Switches in Cisco IOS XE Amsterdam 17.2.1

Network Type	Supported Platform and Release
Front-panel port (trunk and VLAN)	• Catalyst 9300 Series Switches and C9300L in Cisco IOS XE Gibraltar 16.12.1 • Catalyst 9400 Series Switches in Cisco IOS XE Amsterdam 17.1.1 • Catalyst 9500- High Performance Series Switches in Cisco IOS XE Amsterdam 17.5.1 • Catalyst 9600 Series Switches in Cisco IOS XE Amsterdam 17.5.1 • Catalyst 9300X Series Switches in Cisco IOS XE Bengaluru 17.6.1 **Note** Catalyst 9300X Series Switches support multiple AppGigabitEthernet ports.
Cisco IOS Network Address Translation (NAT)	• Catalyst 9300 Series Switches and C9300L in Cisco IOS XE Gibraltar 16.12.1 • Catalyst 9400 Series Switches in Cisco IOS XE Amsterdam 17.1.1 On both these platforms, NAT is supported through the hardware data-port features applied on the front-panel data ports and on the AppGigabitEthernet port.
Cisco IOx NAT	Not supported

Virtual Network Interface Card

To manage the life cycle of an application container, the Layer 3 routing model that supports one container per internal logical interface is used. This means that a virtual Ethernet pair is created for each application, and one interface of this pair, called the Virtual Network Interface Card (vNIC) is part of the application container.

NIC is the standard Ethernet interface inside the container that connects to the platform data plane for the sending and receiving of packets. Cisco IOx is responsible for assigning the IP address and unique MAC address for each vNIC in the container.

The vNICs inside a container are considered as standard Ethernet interfaces.

ERSPAN Support on the AppGigabitEthernet Port

Encapsulated Remote Switch Port Analyzer (ERSPAN) support on an AppGigabitEthernet port enables the mirroring of data traffic from a device to an application that runs on the AppGigabitEthernet port by using IOx.

Note The Cisco IOx process must be running before the IOx virtual application can be hosted on a Cisco device.

How to Configure Application Hosting

The following sections provide information about the various tasks that comprise the configuration of application hosting.

Enabling Cisco IOx

Perform this task to enable access to Cisco IOx, which provides a CLI-based user interface that you can use to manage, administer, monitor, and troubleshoot the apps on the host system, and to perform a variety of related activities.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **iox**
4. **username** *name* **privilege** *level* **password** {**0** | **7** | *user-password*}*encrypted-password*
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**iox** **Example:** `Device(config)# iox`	Enables Cisco IOx.
Step 4	**username** *name* **privilege** *level* **password** {**0** | **7** | *user-password*}*encrypted-password* **Example:** `Device(config)# username cisco privilege 15 password 0 ciscoI`	Establishes a username-based authentication system and privilege level for the user. • The username privilege level must be configured as 15.

	Command or Action	Purpose
Step 5	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC configuration mode.

Configuring Application Hosting on Front-Panel VLAN Ports

Note This task is applicable to Cisco IOS XE Amsterdam 17.1.1 and later releases.

In application-hosting trunk-configuration mode, all the allowed AppGigabitEthernet VLAN ports are connected to a container. Native and VLAN-tagged frames are transmitted and received by the container guest interface. Only one container guest interface can be mapped to the AppGigabitEthernet trunk port.

Concurrent configuration of both *trunk* and *vlan-access* ports are supported.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *AppGigabitEthernet number*
4. **switchport trunk allowed vlan** *vlan-ID*
5. **switchport mode trunk**
6. **exit**
7. **app-hosting appid** *name*
8. **app-vnic AppGigabitEthernet trunk**
9. **vlan** *vlan-ID* **guest-interface** *guest-interface-number*
10. **guest-ipaddress** *ip-address* **netmask** *netmask*
11. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *AppGigabitEthernet number* **Example:** `Device(config)# interface AppGigabitEthernet 1/0/1`	Configures the AppGigabitEthernet and enters interface configuration mode. • For stackable switches, the *number* argument is *switch-number/0/1*.

	Command or Action	Purpose
Step 4	**switchport trunk allowed vlan** *vlan-ID* **Example:** `Device(config-if)# switchport trunk allowed vlan 10-12,20`	Configures the list of VLANs allowed on the trunk.
Step 5	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Sets the interface into permanent trunking mode and negotiates to convert the neighboring link into a trunk link.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 7	**app-hosting appid** *name* **Example:** `Device(config)# app-hosting appid iox_app`	Configures an application and enters application-hosting configuration mode.
Step 8	**app-vnic AppGigabitEthernet trunk** **Example:** `Device(config-app-hosting)# app-vnic AppGigabitEthernet trunk`	Configures a trunk port as the front-panel port for an application, and enters application-hosting trunk-configuration mode.
Step 9	**vlan** *vlan-ID* **guest-interface** *guest-interface-number* **Example:** `Device(config-config-app-hosting-trunk)# vlan 10 guest-interface 2`	Configures a VLAN guest interface and enters application-hosting VLAN-access IP configuration mode. • Multiple VLAN-to-guest interface mapping is supported.
Step 10	**guest-ipaddress** *ip-address* **netmask** *netmask* **Example:** `Device(config-config-app-hosting-vlan-access-ip)# guest-ipaddress 192.168.0.2 netmask 255.255.255.0`	(Optional) Configures a static IP address.
Step 11	**end** **Example:** `Device(config-config-app-hosting-vlan-access-ip)# end`	Exits application-hosting VLAN-access IP configuration mode and returns to privileged EXEC mode.

Configuring Application Hosting on Front-Panel Trunk Ports

In application-hosting trunk-configuration mode, all the allowed AppGigabitEthernet VLAN ports are connected to a container. Native and VLAN-tagged frames are transmitted and received by the container guest interface. Only one container guest interface can be mapped to the AppGigabitEthernet trunk port.

In Cisco IOS XE Gibraltar 16.2.1, you can configure an app-ID in either application-hosting trunk configuration mode or application-hosting VLAN-access configuration mode; but not in both modes.

In Cisco IOS XE Amsterdam 17.1.1 and later releases, concurrent configuration of both *trunk* and *vlan-access* ports is supported.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface** *AppGigabitEthernet number*
4. **switchport trunk allowed vlan** *vlan-ID*
5. **switchport mode trunk**
6. **exit**
7. **app-hosting appid** *name*
8. **app-vnic AppGigabitEthernet trunk**
9. **guest-interface** *guest-interface-number*
10. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface** *AppGigabitEthernet number* **Example:** `Device(config)# interface AppGigabitEthernet 1/0/1`	Configures the AppGigabitEthernet and enters interface configuration mode. • For stackable switches, the *number* argument is *switch-number/0/1*.
Step 4	**switchport trunk allowed vlan** *vlan-ID* **Example:** `Device(config-if)# switchport trunk allowed vlan 10-12,20`	Configures the list of VLANs allowed on the trunk.
Step 5	**switchport mode trunk** **Example:** `Device(config-if)# switchport mode trunk`	Sets the interface into permanent trunking mode and negotiates to convert the neighboring link into a trunk link.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 7	**app-hosting appid** *name* **Example:** `Device(config)# app-hosting appid iox_app`	Configures an application and enters application-hosting configuration mode.
Step 8	**app-vnic AppGigabitEthernet trunk** **Example:** `Device(config-app-hosting)# app-vnic AppGigabitEthernet trunk`	Configures a trunk port as the front-panel port for an application, and enters application-hosting trunk-configuration mode.
Step 9	**guest-interface** *guest-interface-number* **Example:** `Device(config-config-app-hosting-trunk)# guest-interface 2`	Configures an application's interface that is connected to the AppGigabitEthernet interface trunk.
Step 10	**end** **Example:** `Deviceconfig-config-app-hosting-trunk)# end`	Exits application-hosting trunk-configuration mode and returns to privileged EXEC mode.

Starting an Application in Configuration Mode

The **start** command in application-hosting configuration mode is equivalent to the **app-hosting activate appid** and **app-hosting start appid** commands.

The **no start** command in application-hosting configuration mode is equivalent to the **app-hosting stop appid** and **app-hosting deactivate appid** commands.

Note If the **start** command is configured before an application is installed, and then the **install** command is configured, Cisco IOx automatically performs internal **activate** and **start** actions. This allows the application to be automatically started by configuring the **install** command.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **app-hosting appid** *application-name*
4. **start**
5. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:**	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
	`Device> enable`	
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**app-hosting appid** *application-name* **Example:** `Device(config)# app-hosting appid iox_app`	Configures an application and enters application-hosting configuration mode.
Step 4	**start** **Example:** `Device(config-app-hosting)# start`	(Optional) Starts and runs an application. • Use the **no start** command to stop the application.
Step 5	**end** **Example:** `Device(config-app-hosting)# end`	Exits application-hosting configuration mode and returns to privileged EXEC mode.

Lifecycle of an Application

The following EXEC commands take you through an application's lifecycle.

Note If any configuration changes are made after an application is installed, the application in the running state will not reflect these changes. The application must be explicitly stopped and deactivated, and then activated and started again for the configuration changes to take effect.

SUMMARY STEPS

1. **enable**
2. **app-hosting install appid** *application-name* **package** *package-path*
3. **app-hosting activate appid** *application-name*
4. **app-hosting start appid** *application-name*
5. **app-hosting stop appid** *application-name*
6. **app-hosting deactivate appid** *application-name*
7. **app-hosting uninstall appid** *application-name*

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.

	Command or Action	Purpose
Step 2	**app-hosting install appid** *application-name* **package** *package-path* **Example:** `Device# app-hosting install appid iox_app package usbflash1:my_iox_app.tar`	Installs an application from the specified location. • An application can be installed from a local storage location such as, flash, bootflash, usbflash0, usbflash1, and harddisk.
Step 3	**app-hosting activate appid** *application-name* **Example:** `Device# app-hosting activate appid iox_app`	Activates the application. • This command validates all the application resource requests, and if all the resources are available, the application is activated; if not, the activation fails.
Step 4	**app-hosting start appid** *application-name* **Example:** `Device# app-hosting start appid iox_app`	Starts the application. • Application start-up scripts are activated.
Step 5	**app-hosting stop appid** *application-name* **Example:** `Device# app-hosting stop appid iox_app`	(Optional) Stops the application.
Step 6	**app-hosting deactivate appid** *application-name* **Example:** `Device# app-hosting deactivate appid iox_app`	(Optional) Deactivates all the resources allocated for the application.
Step 7	**app-hosting uninstall appid** *application-name* **Example:** `Device# app-hosting uninstall appid iox_app`	(Optional) Uninstalls the application. • Uninstalls all the packaging and images stored. All the changes and updates to the application are also removed.

Configuring Docker Run Time Options

You can add a maximum of 30 lines of run time options. The system generates a concatenated string from line 1 though line 30. A string can have more than one Docker run time option.

When a run time option is changed, stop, deactivate, activate, and start the application for the new run time options to take effect.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **app-hosting appid** *application-name*
4. **app-resource docker**
5. **run-opts** *options*
6. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**app-hosting appid** *application-name* **Example:** `Device(config)# app-hosting appid iox_app`	Configures an application and enters application-hosting configuration mode.
Step 4	**app-resource docker** **Example:** `Device(config-app-hosting)# app-resource docker`	Enters application-hosting docker-configuration mode to specify application resource updates.
Step 5	**run-opts** *options* **Example:** `Device(config-app-hosting-docker)# run-opts 1 "-v $(APP_DATA):/data"`	Specifies the Docker run time options.
Step 6	**end** **Example:** `Device(config-app-hosting-docker)# end`	Exits application-hosting docker-configuration mode and returns to privileged EXEC mode.

Configuring a Static IP Address in a Container

When configuring a static IP address in a container, the following guidelines apply:

- Only the last configured default gateway configuration is used.
- Only the last configured name server configuration is used.

You can configure the IP address of a container through Cisco IOS CLIs.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **app-hosting appid** *name*
4. **name-server#** *ip-address*
5. **app-vnic management guest-interface** *interface-number*
6. **guest-ipaddress** *ip-address* **netmask** *netmask*
7. **exit**

8. **app-default-gateway** *ip-address* **guest-interface** *network-interface*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**app-hosting appid** *name* **Example:** `Device(config)# app-hosting appid iox_app`	Configures an application and enters application-hosting configuration mode.
Step 4	**name-server#** *ip-address* **Example:** `Device(config-app-hosting)# name-server0 10.2.2.2`	Configures the Domain Name System (DNS) server.
Step 5	**app-vnic management guest-interface** *interface-number* **Example:** `Device(config-app-hosting)# app-vnic management guest-interface 0`	Configures the management gateway of the virtual network interface and guest interface, and enters application-hosting management-gateway configuration mode.
Step 6	**guest-ipaddress** *ip-address* **netmask** *netmask* **Example:** `Device(config-app-hosting-mgmt-gateway)# guest-ipaddress 172.19.0.24 netmask 255.255.255.0`	Configures the management guest interface details.
Step 7	**exit** **Example:** `Device(config-app-hosting-mgmt-gateway)# exit`	Exits application-hosting management-gateway configuration mode and returns to application-hosting configuration mode.
Step 8	**app-default-gateway** *ip-address* **guest-interface** *network-interface* **Example:** `Device(config-app-hosting)# app-default-gateway 172.19.0.23 guest-interface 0`	Configures the default management gateway.
Step 9	**end** **Example:** `Device(config-app-hosting)# end`	Exits application-hosting configuration mode and returns to privileged EXEC mode.

Configuring Application Hosting on the Management Port

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **interface gigabitethernet0/0**
4. **vrf forwarding** *vrf-name*
5. **ip address** *ip-address mask*
6. **exit**
7. **app-hosting appid** *name*
8. **app-vnic management guest-interface** *network-interface*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**interface gigabitethernet0/0** **Example:** `Device(config)# interface gigabitethernet0/0`	Configures an interface and enters interface configuration mode. • On Cisco Catalyst 9000 Series Switches, the management interface is GigabitEthernet0/0.
Step 4	**vrf forwarding** *vrf-name* **Example:** `Device(config-if)# vrf forwarding Mgmt-vrf`	Associates a Virtual Routing and Forwarding (VRF) instance or a virtual network with an interface or subinterface. • *Mgmt-vrf* is automatically set for the management interface on the Cisco Catalyst 9000 Series Switch.
Step 5	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 198.51.100.1 255.255.255.254`	Configures an IP address for the interface.
Step 6	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.

	Command or Action	Purpose
Step 7	**app-hosting appid** *name* **Example:** `Device(config)# app-hosting appid iox_app`	Configures an application and enters application-hosting configuration mode.
Step 8	**app-vnic management guest-interface** *network-interface* **Example:** `Device(config-app-hosting)# app-vnic management guest-interface 1`	Connects the guest interface to the management port, and enters application-hosting management-gateway configuration mode. • The **management** keyword specifies the Cisco IOS management GigabitEthernet0/0 interface that is connected to the container. • The **guest-interface** *network-interface* keyword-argument pair specifies the container's internal Ethernet interface number that is connected to the Cisco IOS management interface. The example provided here uses *guest-interface 1* for the container's Ethernet 1 interface.
Step 9	**end** **Example:** `Device(config-app-hosting-mgmt-gateway)# end`	Exits application-hosting management-gateway configuration mode and returns to privileged EXEC mode.

Manually Configuring the IP Address for an Application

You can set up the IP address of a container using the following methods:

- Log into the container, and configure the **ifconfig** Linux command.
  1. Log in to the application by using the following command:

     ```
     app-hosting connect appid APPID {session | console}
     ```

  2. Based on the application's Linux support, use the standard Linux interface configuration commands:

     ```
     - ifconfig dev IFADDR/subnet-mask-length
     ```

     Or

     ```
     - ip address {add|change|replace} IFADDR dev IFNAME [ LIFETIME ] [ CONFFLAG-LIST ]
     ```

- Enable the Dynamic Host Configuration Protocol (DHCP) in the container, and configure the DHCP server and relay agent in the Cisco IOS configuration.
  - Cisco IOx provides a DHCP client to run within the application container that is used for an application DHCP interface.

Overriding App Resource Configuration

For resource changes to take effect, you must first stop and deactivate an app using the **app-hosting stop** and **app-hosting deactivate** commands, and then restart the app using the **app-hosting activate** and **app-hosting start** commands.

If you are using the **start** command in application-hosting configuration mode, configure the **no start** and **start** commands.

You can use these commands to reset both resources and the app-hosting appid iox_app configuration.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **app-hosting appid** *name*
4. **app-resource profile** *name*
5. **cpu** *unit*
6. **memory** *memory*
7. **vcpu** *number*
8. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**app-hosting appid** *name* **Example:** `Device(config)# app-hosting appid iox_app`	Enables application hosting and enters application-hosting configuration mode.
Step 4	**app-resource profile** *name* **Example:** `Device(config-app-hosting)# app-resource profile custom`	Configures the custom application resource profile, and enters custom application resource profile configuration mode. • Only the custom profile name is supported.
Step 5	**cpu** *unit* **Example:** `Device(config-app-resource-profile-custom)# cpu 7400`	Changes the default CPU allocation for the application. • Resource values are application specific, and any adjustment to these values must ensure that the application can run reliably with the changes.

	Command or Action	Purpose
Step 6	**memory** *memory* **Example:** `Device(config-app-resource-profile-custom)# memory 2048`	Changes the default memory allocation.
Step 7	**vcpu** *number* **Example:** `Device(config-app-resource-profile-custom)# vcpu 2`	Changes the virtual CPU (vCPU) allocation for the application.
Step 8	**end** **Example:** `Device(config-app-resource-profile-custom)# end`	Exits custom application resource profile configuration mode and returns to privileged EXEC mode.

Configuring ERSPAN Support on the AppGigabitEthernet Port

Perform the following tasks to configure ERSPAN through an AppGigabitEthernet interface.

Note The IOx process must be running before the IOx virtual application can be hosted on a Cisco device.

Configuring an ERSPAN Source Session

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **monitor session** *span-session-number* **type erspan-source**
4. **source interface** *interface-type interface-id*
5. **no shutdown**
6. **ip address** *ip-address*
7. **origin ip address** *ip-address*
8. **erspan-id** *erspan-flow-id*
9. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**monitor session** *span-session-number* **type erspan-source** **Example:** `Device(config)# monitor session 2 type erspan-source`	Defines an ERSPAN source session using the session ID and the session type, and enters ERSPAN monitor source session configuration mode.
Step 4	**source interface** *interface-type interface-id* **Example:** `Device(config-mon-erspan-src)# source interface gigabitethernet 1/0/3`	Configures the source interface and the traffic direction to be monitored.
Step 5	**no shutdown** **Example:** `Device(config-mon-erspan-src)# no shutdown`	Enables the configured sessions on an interface.
Step 6	**ip address** *ip-address* **Example:** `Device(config-mon-erspan-src-dst)# ip address 10.1.1.5`	Configures the IP address that is used as the destination of the ERSPAN traffic.
Step 7	**origin ip address** *ip-address* **Example:** `Device(config-mon-erspan-src-dst)# origin ip address 10.1.1.2`	Configures the IP address used as the source of the ERSPAN traffic.
Step 8	**erspan-id** *erspan-flow-id* **Example:** `Device(config-mon-erspan-src-dst)# erspan-id 5`	Configures the ID used by the source and destination sessions to identify the ERSPAN traffic, which must also be entered in the ERSPAN destination session configuration.
Step 9	**end** **Example:** `Device(config-mon-erspan-src-dst)# end`	Exits ERSPAN monitor source session configuration mode and returns to privileged EXEC mode.

Configuring the AppGigabitEthernet Interface for ERSPAN

Note You can either use a Layer 2 port or a Layer 3 port for the ERSPAN traffic. Use the **no switchport mode** command to change the port from a Layer 2 interface to a Layer 3 interface.

Before you begin

1. Step 1 to Step 9 show how to configure a VLAN for traffic mirroring.

2. Step 10 to Step 14 show how to configure the AppGigabitEthernet interface to transport ERSPAN-mirrored data traffic to the IOx virtual application.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **vtp mode off**
4. **vlan** {*vlan-ID* | *vlan-range*}
5. **exit**
6. **interface vlan** *vlan-ID*
7. **ip address** *ip-address mask*
8. **no shutdown**
9. **exit**
10. **interface AppGigabitEthernet** *number*
11. (Optional) **no switchport mode**
12. (Optional) **ip address** *ip-address mask*
13. (Optional) **switchport mode trunk**
14. **end**

DETAILED STEPS

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. Enter your password, if prompted.
Step 2	**configure terminal** **Example:** `Device# configure terminal`	Enters global configuration mode.
Step 3	**vtp mode off** **Example:** `Device(config)#vtp mode off`	Sets the VTP-device mode to Off for VLANs.
Step 4	**vlan** {*vlan-ID* | *vlan-range*} **Example:** `Device(config)# vlan 2508`	Adds a VLAN and enters config-VLAN configuration mode.
Step 5	**exit** **Example:** `Device(config-vlan)# exit`	Exits config-vlan configuration mode and returns to global configuration mode.
Step 6	**interface vlan** *vlan-ID* **Example:** `Device(config)# interface vlan 2508`	Creates a dynamic Switch Virtual Interface (SVI) and enters interface configuration mode.

	Command or Action	Purpose
Step 7	**ip address** *ip-address mask* **Example:** `Device(config-if)# ip address 192.0.2.1 255.255.255.252`	Configures an IP address.
Step 8	**no shutdown** **Example:** `Device(config-if)# no shutdown`	Restarts a disabled interface.
Step 9	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 10	**interface AppGigabitEthernet** *number* **Example:** `Device(config)# interface AppGigabitEthernet 1/1`	Configures the AppGigabitEthernet and enters interface configuration mode. For stackable switches, the number argument is *switch-number/0/1*. **Note** You can use either a Layer 2 or a Layer 3 port.
Step 11	(Optional) **no switchport mode** **Example:** `Device(config-if)# no switchport mode`	Changes the port from a Layer 2 interface to a Layer 3 interface.
Step 12	(Optional) **ip address** *ip-address mask* **Example:** `Device(config-if)# 10.1.1.2 255.255.255.0`	Configures an IP address for a Layer 3 port.
Step 13	(Optional) **switchport mode trunk** **Example:** `Device(config-if)# switchport mode  trunk`	Sets the interface into permanent trunking mode and negotiates to convert the neighboring link into a trunk link for a Layer 2 port.
Step 14	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Verifying the Application-Hosting Configuration

Use these **show** commands to verify the configuration. These commands can be used in any order.

SUMMARY STEPS

1. **enable**

2. **show iox-service**
3. **show app-hosting detail**
4. **show app-hosting device**
5. **show app-hosting list**
6. **show interfaces trunk**
7. **show controller ethernet-controller AppGigabitEthernet** *interface-number*

DETAILED STEPS

Step 1 enable

Enables privileged EXEC mode.

- Enter your password if prompted.

Example:

```
Device> enable
```

Step 2 show iox-service

Displays the status of all the Cisco IOx services.

Example:

```
Device# show iox-service

IOx Infrastructure Summary:
---------------------------
IOx service (CAF)          : Not Running
IOx service (HA)           : Not Running
IOx service (IOxman)       : Not Running
IOx service (Sec storage)  : Not Running
Libvirtd                   : Running
Dockerd                    : Not Running
Application DB Sync Info   : Not available
```

Step 3 show app-hosting detail

Displays detailed information about the application.

Example:

```
Device# show app-hosting detail

State                      : Running
Author                     : Cisco Systems, Inc
Application
 Type                      : vm
 App id                    : Wireshark
 Name                      : Wireshark
 Version                   : 3.4
 Activated Profile Name    : custom
 Description               : Ubuntu based Wireshark
Resource Reservation
 Memory                    : 1900 MB
 Disk                      : 10 MB
 CPU                       : 4000 units
 VCPU                      : 2
Attached devices
```

```
Type          Name         Alias
------------------------------------------------------------------------
Serial/shell
Serial/aux
Serial/Syslog              serial2
Serial/Trace               serial3
Network Interfaces
------------------------------------------------------------------
eth0:
 MAC address              : 52:54:dd:80:bd:59
 IPv4 address
eth1:
 MAC address              : 52:54:dd:c7:7c:aa
 IPv4 address
```

Step 4 **show app-hosting device**

Displays information about the USB device.

Example:

```
Device# show app-hosting device

USB port Device name Available
1 Front_USB_1 true


app-hosting appid testvm
app-vnic management guest-interface 0
app-device usb-port 1
```

Step 5 **show app-hosting list**

Displays the list of applications and their status.

Example:

```
Device# show app-hosting list

App id              State
---------------------------------------
Wireshark           Running
```

Step 6 **show interfaces trunk**

Displays trunk interface information.

Example:

```
Device# show interfaces trunk

Port Mode Encapsulation Status Native vlan
Gi3/0/1 on 802.1q trunking 1
Ap3/0/1 on 802.1q trunking 1

Port Vlans allowed on trunk
Gi3/0/1 1-4094
Ap3/0/1 1-4094

Port Vlans allowed and active in management domain
Gi3/0/1 1,8,10,100
Ap3/0/1 1,8,10,100
```

```
Port Vlans in spanning tree forwarding state and not pruned
Gi3/0/1 1,8,10,100
Ap3/0/1 1,8,10,100

Device# show runnning-config interface AppGigabitEthernet 3/0/1

Building configuration...

Current configuration : 64 bytes
!
interface AppGigabitEthernet3/0/1
switchport mode trunk
end
```

Step 7 **show controller ethernet-controller AppGigabitEthernet** *interface-number*

Displays the send and receive statistics for the AppGigabitEthernet interface that is read from the hardware.

Example:

```
Device# show controller ethernet-controller AppGigabitEthernet 1/0/1

Transmit                  AppGigabitEthernet1/0/1   Receive
            0 Total bytes                           0 Total bytes
            0 Unicast frames                        0 Unicast frames
            0 Unicast bytes                         0 Unicast bytes
            0 Multicast frames                      0 Multicast frames
            0 Multicast bytes                       0 Multicast bytes
            0 Broadcast frames                      0 Broadcast frames
            0 Broadcast bytes                       0 Broadcast bytes
            0 System FCS error frames               0 IpgViolation frames
            0 MacUnderrun frames                    0 MacOverrun frames
            0 Pause frames                          0 Pause frames
            0 Cos 0 Pause frames                    0 Cos 0 Pause frames
            0 Cos 1 Pause frames                    0 Cos 1 Pause frames
            0 Cos 2 Pause frames                    0 Cos 2 Pause frames
            0 Cos 3 Pause frames                    0 Cos 3 Pause frames
            0 Cos 4 Pause frames                    0 Cos 4 Pause frames
            0 Cos 5 Pause frames                    0 Cos 5 Pause frames
            0 Cos 6 Pause frames                    0 Cos 6 Pause frames
            0 Cos 7 Pause frames                    0 Cos 7 Pause frames
            0 Oam frames                            0 OamProcessed frames
            0 Oam frames                            0 OamDropped frames
            0 Minimum size frames                   0 Minimum size frames
            0 65 to 127 byte frames                 0 65 to 127 byte frames
            0 128 to 255 byte frames                0 128 to 255 byte frames
            0 256 to 511 byte frames                0 256 to 511 byte frames
            0 512 to 1023 byte frames               0 512 to 1023 byte frames
            0 1024 to 1518 byte frames              0 1024 to 1518 byte frames
            0 1519 to 2047 byte frames              0 1519 to 2047 byte frames
            0 2048 to 4095 byte frames              0 2048 to 4095 byte frames
            0 4096 to 8191 byte frames              0 4096 to 8191 byte frames
            0 8192 to 16383 byte frames             0 8192 to 16383 byte frames
            0 16384 to 32767 byte frame             0 16384 to 32767 byte frame
            0 > 32768 byte frames                   0 > 32768 byte frames
            0 Late collision frames                 0 SymbolErr frames
            0 Excess Defer frames                   0 Collision fragments
            0 Good (1 coll) frames                  0 ValidUnderSize frames
            0 Good (>1 coll) frames                 0 InvalidOverSize frames
            0 Deferred frames                       0 ValidOverSize frames
            0 Gold frames dropped                   0 FcsErr frames
            0 Gold frames truncated
            0 Gold frames successful
            0 1 collision frames
```

```
0 2 collision frames
0 3 collision frames
0 4 collision frames
0 5 collision frames
0 6 collision frames
0 7 collision frames
0 8 collision frames
0 9 collision frames
0 10 collision frames
0 11 collision frames
0 12 collision frames
0 13 collision frames
0 14 collision frames
0 15 collision frames
0 Excess collision frame
```

Configuration Examples for Application Hosting

The following are the various examples pertaining to the configuration of the Application Hosting feature.

Example: Enabling Cisco IOx

This example shows how to enable Cisco IOx.

```
Device> enable
Device# configure terminal
Device(config)# iox
Device(config)# username cisco privilege 15 password 0 ciscoI
Device(config)# end
```

Example: Configuring Application Hosting on Front-Panel VLAN Ports

Note This section is applicable to Cisco IOS XE Amsterdam 17.1.1 and later releases.

This example shows how to configure application hosting on front-panel VLAN ports.

```
Device# configure terminal
Device(config)# interface AppGigabitEthernet 1/0/1
Device(config-if)# switchport trunk allowed vlan 10-12,20
Device(config-if)# switchport mode trunk
Device(config-if)# exit
Device(config)# app-hosting appid iox_app
Device(config-app-hosting)# app-vnic AppGigabitEthernet trunk
Device(config-config-app-hosting-trunk)# vlan 10 guest-interface 2
Device(config-config-app-hosting-vlan-access-ip)# guest-ipaddress 192.168.0.1
netmask 255.255.255.0
```

```
Device(config-config-app-hosting-vlan access-ip)# end
```

Example: Configuring Application Hosting on Front-Panel Trunk Ports

This example shows how to configure application hosting on front-panel trunk ports.

```
Device# configure terminal
Device(config)# interface AppGigabitEthernet 3/0/1
Device(config-if)# switchport trunk allowed vlan 10-12,20
Device(config-if)# switchport mode trunk
Device(config-if)# exit
Device(config)# app-hosting appid iox_app
Device(config-app-hosting)# app-vnic AppGigabitEthernet trunk
Device(config-config-app-hosting-trunk)# guest-interface 2
Device(config-config-app-hosting-trunk)# end
```

Example: Installing an Application from disk0:

The following example shows how to install an application from disk0:

```
Device> enable
Device# app-hosting install appid iperf3 package disk0:iperf3.tar

Installing package 'disk0:iperf3.tar' for 'iperf3'. Use 'show app-hosting list' for progress.


Device# show app-hosting list
App id                                   State
---------------------------------------------------------
iperf3                                   DEPLOYED

Switch#app-hosting activate appid iperf3
iperf3 activated successfully
Current state is: ACTIVATED
Switch#
Switch#show app-hosting list
App id                                   State
---------------------------------------------------------
iperf3                                   ACTIVATED

Switch#app-hosting start appid iperf3
iperf3 started successfully
Current state is: RUNNING
Switch#show app-hosting list
App id                                   State
---------------------------------------------------------
iperf3                                   RUNNING

Device#
```

Example: Starting an Application

This example shows how to start an application.

```
Device> enable
Device# configure terminal
Device(config)# app-hosting appid iox_app
Device(config-app-hosting)# start
Device(config-app-hosting)# end
```

Example: Lifecycle for an Application

This example shows how to install and uninstall an application:

```
Device> enable
Device# app-hosting install appid iox_app package usbflash1:my_iox_app.tar.tar
Device# app-hosting activate appid iox_app
Device# app-hosting start appid iox_app
Device# app-hosting stop appid iox_app
Device# app-hosting deactivate appid iox_app
Device# app-hosting uninstall appid iox_app
```

Example: Configuring Docker Run Time Options

This example shows how to configure Docker run time options.

```
Device> enable
Device# configure terminal
Device(config)# app-hosting appid iox_app
Device(config-app-hosting)# app-resource docker
Device(config-app-hosting-docker)# run-opts 1 "-v $(APP_DATA):/data"
Device(config-app-hosting-docker)# run-opts 3 "--entrypoint '/bin/sleep 1000000'"
Device(config-app-hosting-docker)# end
```

Example: Configuring a Static IP Address in a Container

This example shows how to configure a static IP address in a container.

```
Device> enable
Device# configure terminal
Device(config)# app-hosting appid iox_app
Device(config-app-hosting)# name-server0 10.2.2.2
Device(config-app-hosting)# app-vnic management guest-interface 0
Device(config-app-hosting-mgmt-gateway)# guest-ipaddress 172.19.0.24 netmask 255.255.255.0
Device(config-app-hosting-mgmt-gateway)# exit
Device(config-app-hosting)# app-default-gateway 172.19.0.23 guest-interface 0
Device(config-app-hosting)# end
```

Example: Configuring Application Hosting on the Management Port

This example shows how to manually configure the IP address for an application.

```
Device# configure terminal
Device(config)# interface gigabitethernet 0/0
Device(config-if)# vrf forwarding Mgmt-vrf
Device(config-if)# ip address 198.51.100.1 255.255.255.254
Device(config-if)# exit
Device(config)# app-hosting appid iox_app
Device(config-app-hosting)# app-vnic management guest-interface 1
Device(config-app-hosting-mgmt-gateway)# end
```

Example: Overriding App Resource Configuration

This example shows how to override an app resource configuration.

```
Device# configure terminal
Device(config)# app-hosting appid iox_app
Device(config-app-hosting)# app-resource profile custom
Device(config-app-resource-profile-custom)# cpu 7400
Device(config-app-resource-profile-custom)# memory 2048
Device(config-app-resource-profile-custom)# vcpu 2
Device(config-app-resource-profile-custom)# end
```

Example: Configuring ERSPAN Support on an AppGigabitEthernet Port

These examples show how to configure ERSPAN on an AppGigabitEthernet port.

Example: Configuring an ERSPAN Source Session

```
Device> enable
Device# configure terminal
Device(config)# monitor session 2 type erspan-source
Device(config-mon-erspan-src)# source interface gigabitethernet 1/0/3
Device(config-mon-erspan-src)# no shutdown
Device(config-mon-erspan-src-dst)# ip address 10.1.1.5
Device(config-mon-erspan-src-dst)# origin ip address 10.1.1.2
Device(config-mon-erspan-src-dst)# erspan-id 5
Device(config-mon-erspan-src-dst)# end
```

Examples: Configuring ERSPAN Through an AppGigabitEthernet Interface

This example shows how to configure ERSPAN through an AppGigabitEthernet interface:

Note Layer 3 port used for ERSPAN traffic:

```
Device> enable
Device# configure terminal
Device(config)# vtp mode off
Device(config)# vlan 2508
Device(config-vlan)# exit
Device(config)# interface vlan 2508
Device(config-if)# ip address 192.0.2.1 255.255.255.252
Device(config-if)# no shutdown
Device(config-if)# exit
Device(config)# interface AppGigabitEthernet 1/1
Device(config-if)# no switchport mode
Device(config-if)# ip address 10.1.1.2 255.255.255.0
Device(config-if)# end
```

This example shows a Layer 2 port used for ERSPAN traffic:

```
Device> enable
Device# configure terminal
Device(config)# vtp mode off
Device(config)# vlan 2508
Device(config-vlan)# exit
Device(config)# interface vlan 2508
Device(config-if)# ip address 192.0.2.1 255.255.255.252
Device(config-if)# no shutdown
Device(config-if)# exit
Device(config)# interface AppGigabitEthernet 1/1
Device(config-if)# switchport mode trunk
Device(config-if)# end
```

Additional References

Related Documents

Related Topic	Document Title
Programmability commands	*Programmability Command Reference*
DevNet	https://developer.cisco.com/docs/app-hosting/
M2 SATA on Cisco Catalyst 9400 Series Switches	*M2 SATA Module*
M2 SATA on Cisco Catalyst 9500-High Performance Series Switches	*M2 SATA Module*
M2 SATA on Cisco Catalyst 9600 Series Switches	*M2 SATA Module*
USB3.0 SSD on Cisco Catalyst 9300 Series Switches	*Configuring USB 3.0 SSD*
USB3.0 SSD on Cisco Catalyst 9500 Series Switches	*Configuring USB 3.0 SSD*

Technical Assistance

Description	Link
The Cisco Support website provides extensive online resources, including documentation and tools for troubleshooting and resolving technical issues with Cisco products and technologies. To receive security and technical information about your products, you can subscribe to various services, such as the Product Alert Tool (accessed from Field Notices), the Cisco Technical Services Newsletter, and Really Simple Syndication (RSS) Feeds. Access to most tools on the Cisco Support website requires a Cisco.com user ID and password.	http://www.cisco.com/support

Feature Information for Application Hosting

The following table provides release information about the feature or features described in this module. This table lists only the software release that introduced support for a given feature in a given software release train. Unless noted otherwise, subsequent releases of that software release train also support that feature.

Use Cisco Feature Navigator to find information about platform support and Cisco software image support. To access Cisco Feature Navigator, go to www.cisco.com/go/cfn. An account on Cisco.com is not required.

Table 6: Feature Information for Application Hosting

Feature Name	Release	Feature Information
Application Hosting	Cisco IOS XE Gibraltar 16.12.1 Cisco IOS XE Amsterdam 17.1.1 Cisco IOS XE Amsterdam 17.2.1 Cisco IOS XE Bengaluru 17.5.1 Cisco IOS XE Cupertino 17.7.1	A hosted application is a software as a service (SaaS) solution, and users can execute and operate this solution entirely from the cloud. This module describes the Application Hosting feature and how to enable it. • In Cisco IOS XE Gibraltar 16.12.1, this feature was implemented on Cisco Catalyst 9300 Series Switches. • In Cisco IOS XE Amsterdam 17.1.1, this feature was implemented on Cisco Catalyst 9400 Series Switches. • In Cisco IOS XE Amsterdam 17.2.1, this feature was implemented on Cisco Catalyst 9500-High Performance Series Switches, and Cisco Catalyst 9600 Series Switches. • In Cisco IOS XE Bengaluru 17.5.1, this feature was implemented on Cisco Catalyst 9410 Series Switches. • In Cisco IOS XE Cupertino 17.7.1, this feature was implemented on Cisco Catalyst 9500X Series Switches.
Application Hosting: Autotransfer and Auto-Install of Apps from Internal Flash to SSD	Cisco IOS XE Bengaluru 17.6.1	When IOx is restarted and a different media is selected, all applications must be migrated to the new media, and containers must be restored to the same state as before the change. In Cisco IOS XE Bengaluru 17.6.1, this feature was introduced on the following platforms: • Cisco Catalyst 9300 and 9300L Series Switches • Cisco Catalyst 9400 Series Switches

Feature Name	Release	Feature Information
Application Hosting: Front-Panel Network Port Access	Cisco IOS XE Gibraltar 16.12.1 Cisco IOS XE Amsterdam 17.1.1	Introduces datapath connectivity between the Application Hosting container and the front-panel network ports. Also enables ZTP functionality on the front-panel network. • In Cisco IOS XE Gibraltar 16.12.1, this feature was implemented on Cisco Catalyst 9300 Series Switches. • In Cisco IOS XE Amsterdam 17.1.1, this feature was implemented on Cisco Catalyst 9400 Series Switches.
Application Hosting: Front-Panel USB Port Access	Cisco IOS XE Gibraltar 16.12.1 Cisco IOS XE Amsterdam 17.1.1	Introduces datapath connectivity between the Application Hosting container and the front-panel USB port. • In Cisco IOS XE Gibraltar 16.12.1, this feature was implemented on Cisco Catalyst 9300 Series Switches. • In Cisco IOS XE Amsterdam 17.1.1, this feature was implemented on Cisco Catalyst 9400 Series Switches.
ERSPAN Support on the AppGigabitEthernet Port	Cisco IOS XE Dublin 17.10.1	ERSPAN support on the AppGigabitEthernet port enables the mirroring of data traffic from the device to an application that runs on the AppGigabitEthernet port by using Cisco IOx.
Multicast Routing on the AppGigabitEthernet Port	Cisco IOS XE Dublin 17.11.1	Multicast traffic forwarding is supported on the AppGigabitEthernet interface. Applications can select thenetworks that allow multicast traffic. In Cisco IOS XE Dublin 17.11.1, this feature was introduced on the following platforms: • Cisco Catalyst 9300, 9300L, and 9300X Series Switches • Cisco Catalyst 9400 and 9400X Series Switches • Cisco Catalyst 9500-High Performance Series Switches • Cisco Catalyst 9600 and 9600X Series Switches

Feature Name	Release	Feature Information
Native Docker Container: Application Auto-Restart	Cisco IOS XE Amsterdam 17.2.1 Cisco IOS XE Bengaluru 17.5.1	The Application Auto-Restart feature helps applications deployed on platforms to retain the last configured operational state in the event of a system switchover or restart. This feature is enabled by default, and cannot be disabled by users. • In Cisco IOS XE Amsterdam 17.2.1, this feature was implemented on Cisco Catalyst 9300 Series Switches. • In Cisco IOS XE Bengaluru 17.5.1, this feature was implemented on Cisco Catalyst 9410 Series Switches.