

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.

- Guidelines and Limitations for Application Hosting, on page 1
- Information About Application Hosting, on page 2
- How to Configure Application Hosting, on page 3
- Copying Application Data, on page 12
- Deleting Application Data, on page 12
- Verifying the Application-Hosting Configuration, on page 12
- Configuration Examples for Application Hosting, on page 16
- Additional References, on page 17
- Feature Information for Application Hosting, on page 17

Guidelines and Limitations for Application Hosting

This sections lists the guidelines and limitations for the application hosting feature:

- The application hosting feature does not support IPv6 configuration in Cisco NX-OS Release 10.3(1)F.
- The application hosting feature is supported on Cisco Nexus 9300 series FX, FX2, FX3, GX, and GX2 platforms and Cisco Nexus 9500 series modular switches with FX and GX line cards.
- Only 1 interface per container is supported.
- Application hosting does not support the configuration replace feature.
- Beginning with Cisco NX-OS Release 10.3(3)F, the application hosting feature is supported on Cisco Nexus 9808 platform switches.
- Beginning with Cisco NX-OS Release 10.3(3)F, the application hosting feature is supported on Cisco Nexus 9504 and 9508 platform switches with -R and -R2 line cards. This feature is also supported on Cisco N3K-C36180YC-R, N3K-C3636C-R, and N3K-C36480LD-R2 switches.

- When performing a non-disruptive upgrade on Cisco Nexus 9300-FX2, 9300-GX, and 9300-GX2 platform switches from a release which does not support app-hosting to Cisco NX-OS Release 10.3(3)F, post upgrade, enabling feature app-hosting requires a system reload for TCAM entry to take effect. Cisco Nexus 9300-FX switches do not require a system reload if SPAN is enabled.
- Testing connectivity by pinging into the app container from NX-OS will not work because ping from NX-OS uses the netstack process. If bidirectional reachability needs to be verified, use bash ping: **conf t** ; **feature bash** ; **run bash** ; **ping <app-ip-address>**.
- After configuring **system vrf-member-change retain-l3-config** command, when we change vrf membership on interface such that it falls in the same VRF as app-hosting bridge, and HSRP VIP matches subnet of app-hosting bridge IP, then VIP wont be removed from **show running-config** or DME and no faults will be raised. Rather, an HSRP syslog (severity 3) will be thrown notifying the user of misconfiguration.

Information About Application Hosting

This section provides information about Application Hosting.

Need for Application Hosting

The move to virtual environments has brought 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 such as automation, configuration management monitoring, and integration with the existing tool chains.



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

Application Hosting Overview

The Cisco application-hosting framework is an NX-OS 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 access to services through REST APIs.
- Provides a CLI endpoint.
- Provides an application-hosting infrastructure referred to as Cisco Application Framework (CAF).
- Helps set up platform-specific networking (packet-path) through a special application bridge interface.

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 NX-OS application hosting feature provides

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 NX-OS 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 or .tar.gz files. The configuration that is specific to these applications is also packaged as part of the .tar or .tar.gz files.

How to Configure Application Hosting

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

Enabling Application Hosting Feature

Perform this task to enable the Cisco application hosting feature, which enables the user interface command and API interfaces to manage, administer, monitor, and troubleshoot the applications on the host system, and to perform a variety of related activities.

SUMMARY STEPS

1. **configure terminal**
2. **feature app-hosting**
3. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**feature app-hosting** **Example:** `switch(config)# feature app-hosting`	Enables the Cisco application hosting feature.
Step 3	**end** **Example:** `switch(config)# end`	Exits global configuration mode and returns to privileged EXEC configuration mode.

Configuring Application Hosting Bridge Connections

Layer 3 connectivity to the application containers requires its own endpoint IPv4 addresses. In NX-OS, a virtual bridge mechanism called app-hosting bridge hosts the application containers inside the Cisco Nexus switch.

The bridge acts as a gateway to the application containers and helps route the traffic to the attached VRF routing context. The bridge forwards the subnet traffic of the application over the switch interface per the VRF context.

The hosting of the application containers with network connections across the switch interface requires a dedicated endpoint IP subnet with a minimum of 2 assignable addresses. One IP address is for the application container guest interface and the other IP address is for the application container gateway.

Internally, the application container guest interface is a Virtual Network Interface Card (vNIC), off the app-hosting virtual bridge.

SUMMARY STEPS

1. **configure terminal**
2. **app-hosting bridge** *bridge-index*
3. **ip address** *ip-address/mask*
4. **vrf member** *name*
5. **exit**
6. **app-hosting appid** *name*
7. **app-vnic gateway bridge** *bridge-index* **guest-interface** *guest-interface-number*
8. **guest-ipaddress** *ip-address/mask*
9. **exit**
10. **app-default-gateway** *ip-address* **guest-interface** *guest-interface*
11. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 2	**app-hosting bridge** *bridge-index* **Example:** `switch(config)# app-hosting bridge 1`	Configures the app-hosting bridge and enters the app-hosting bridge configuration mode. • <1-8> Bridge index
Step 3	**ip address** *ip-address/mask* **Example:** `switch(config-app-hosting-bridge)# ip address 172.25.44.1/30`	Configures the app bridge IPv4 address which acts as the gateway to the application container. **Note**

	Command or Action	Purpose
		Beginning with Cisco NX-OS Release 10.3(2)F, the subnet will be rejected if the IP is in use by either an interface or a virtual IP.
Step 4	**vrf member** *name* **Example:** `switch(config-app-hosting-bridge)# vrf member overlay-VRF`	Sets the VRF context. If not configured, it belongs to VRF *default*. **Note** Configuring of management VRF is not supported. Configuring management VRF fails the bridge configuration.
Step 5	**exit** **Example:** `switch(config-app-hosting-bridge)# exit`	Exits app bridge configuration mode and returns to global configuration mode.
Step 6	**app-hosting appid** *name* **Example:** `switch(config)# app-hosting appid te_app`	Configures an application and enters the application-hosting configuration mode.
Step 7	**app-vnic gateway bridge** *bridge-index* **guest-interface** *guest-interface-number* **Example:** `switch(config-app-hosting)# app-vnic gateway bridge 1 guest-interface 0`	Configures the guest VNIC interface for an application and enters the application-hosting vnic interface mode. **Note** Beginning with Cisco NX-OS Release 10.3(3)F, configuring more than 1 VNIC is not supported.
Step 8	**guest-ipaddress** *ip-address/mask* **Example:** `switch(config-app-hosting-app-vnic)# guest-ipaddress 172.25.44.2/30`	Configures one of the available IPv4 addresses from the *bridge 1* subnet.
Step 9	**exit** **Example:** `switch(config-app-hosting-app-vnic)# exit`	Exits app vnic interface configuration mode and returns to app-hosting configuration mode.
Step 10	**app-default-gateway** *ip-address* **guest-interface** *guest-interface* **Example:** `switch(config-app-hosting-appid)# app-default-gateway 172.25.44.1 guest-interface 0`	Configures the available IPv4 address from the *bridge 1* subnet. Configures the gateway address that is configured in Step 3.
Step 11	**end** **Example:** `switch(config-app-hosting)# end`	Exits the application-hosting configuration mode and returns to the privileged EXEC mode.

Lifecycle of an Application

The following EXEC commands describe the lifecycle of an application.

Note If you make any configuration changes after you install an application, the application in the running state does not reflect these changes. To make changes after starting the application, stop and deactivate the application before making any changes, and then activate and start the application again.

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

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `switch# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**app-hosting install appid** *application-name* **package** *package-path* **Example:** `switch# app-hosting install appid te_app package bootflash:my_te_app.tar`	Installs an application from the specified location. • You can install an application from a local storage location, that is, bootflash.
Step 3	**app-hosting activate appid** *application-name* **Example:** `switch# app-hosting activate appid te_app`	Activates the application. • This command validates all the application resource requests. If all the resources are available, the command activates the application, otherwise the activation fails.
Step 4	**app-hosting start appid** *application-name* **Example:** `switch# app-hosting start appid te_app`	Starts the application. • Activates the application start-up scripts.

	Command or Action	Purpose
Step 5	**app-hosting stop appid** *application-name* **Example:** `switch# app-hosting stop appid te_app`	(Optional) Stops the application.
Step 6	**app-hosting deactivate appid***application-name* **Example:** `switch# app-hosting deactivate appid te_app`	(Optional) Deactivates all the resources that are allocated for the application.
Step 7	**app-hosting uninstall appid** *application-name* **Example:** `switch# app-hosting uninstall appid te_app`	(Optional) Uninstalls the application. • Uninstalls all the packaging and images stored. Also removes all the changes and updates to the application.

Upgrading an Application

The following EXEC commands describe how to upgrade an application.

SUMMARY STEPS

1. **enable**
2. **switch# app-hosting upgrade appid** *application-name* **package** *package-path*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `switch# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**switch# app-hosting upgrade appid** *application-name* **package** *package-path* **Example:** `switch# app-hosting upgrade appid tea package bootflash:thousandeyes-enterprise-agent-4.1.0.cisco.tar`	Upgrades the existing application to a newer version. While doing so, this command stops, upgrades, and reverts the application to the pre-upgrade state. **Note** • If you upgrade an application when it is in a STOPPED state, after a successful upgrade the new app-hosting state changes to ACTIVATED. • You can upgrade an application from a local storage location, that is, bootflash.

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 options.

Note To change the run time option, the application must be in a deactivated state.

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

Procedure

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

	Command or Action	Purpose
	`switch(config-app-hosting-docker)# end`	

Configuring Application Hosting on the Management Interface

NX-OS allows application containers to share the network connections over the Cisco NX-OS management interface. You can internally set up a virtual NAT bridge and assign a private IP address to the guest vNIC interface. The guest interface private IP address gets automatically assigned by the Apphosting framework.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **app-hosting appid** *name*
4. **app-vnic management guest-interface** *network-interface*
5. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `switch# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 3	**app-hosting appid** *name* **Example:** `switch(config)# app-hosting appid te_app`	Configures an application and enters application-hosting configuration mode.
Step 4	**app-vnic management guest-interface** *network-interface* **Example:** `switch(config-app-hosting)# app-vnic management guest-interface 0`	Connects the guest interface to the management port, and enters application-hosting management-gateway configuration mode. • The **management** keyword specifies the Cisco NX-OS *interface mgmt0* that connects to the container through private IPNAT mode. • The **guest-interface** *network-interface* keyword-argument pair specifies the container's internal Ethernet interface number that connects to the Cisco NX-OS management interface *mgmt0*. The example here uses *guest-interface 0* for the container's Ethernet 0 interface.

	Command or Action	Purpose
Step 5	**end** **Example:** `switch(config-app-hosting-mgmt-gateway)# end`	Exits application-hosting management-gateway configuration mode and returns to privileged EXEC mode.

Overriding Application 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.

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

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `switch# enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 3	**app-hosting appid** *name* **Example:** `switch(config)# app-hosting appid te_app`	Enables application hosting and enters application-hosting configuration mode.
Step 4	**app-resource profile** *name* **Example:** `switch(config-app-hosting)# app-resource profile custom`	Configures the custom application resource profile, and enters the custom application resource profile configuration mode. • Only the custom profile name is supported.

	Command or Action	Purpose
Step 5	**cpu** *unit* **Example:** `switch(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.
Step 6	**memory** *memory* **Example:** `switch(config-app-resource-profile-custom)# memory 2048`	Changes the default memory allocation.
Step 7	**end** **Example:** `switch(config-app-resource-profile-custom)# end`	Exits the custom application resource profile configuration mode and returns to the privileged EXEC mode.

Advanced Application Hosting Features

By default, App-hosting feature allows only Cisco supported and signed application packages. To install non-Cisco signed application docker images, the sign verification functionality must be turned off. This is a global configuration and affects all the applications that are being installed. The **app-hosting signed-verification** [**disable** | **enable**] command disables the sign verification and helps install the non-Cisco Docker applications.

When the application hosting feature is configured, it reserves 2 GB of file space from the bootflash as application storage space. If more space is required for a particular application, then you can increase the partition size. Alternatively, you can decrease the space based on the space requirements of the applications using the **app-hosting bootflash backend storage limit size** global configuration command. The application restarts.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **app-hosting signed-verification [disable | enable]**
4. **app-hosting bootflash backend storage limit** *size*
5. **end**

DETAILED STEPS

Procedure

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

	Command or Action	Purpose
Step 2	**configure terminal** **Example:** `switch# configure terminal`	Enters global configuration mode.
Step 3	**app-hosting signed-verification [disable | enable]** **Example:** `switch(config)# app-hosting signed-verification disable`	Disables the package verification to allow non-Cisco applications. • Sign verification is enabled by default.
Step 4	**app-hosting bootflash backend storage limit** *size* **Example:** `switch(config)# app-hosting bootflash backend storage limit 600`	Configures the applications storage size required considering all the applications going to be installed. • 2048 MB would be used by default. • Size in MB; size must be less than the available free space in bootflash.
Step 5	**end** **Example:** `switch(config-app-resource-profile-custom)# end`	Exits the custom application resource profile configuration mode and returns to the privileged EXEC mode.

Copying Application Data

To copy application data into an application's persistent data mount, use the **app-hosting data appid** <*appid*> **copy** command in the privileged EXEC mode.

```
app-hosting data appid tea copy bootflash:src dest
```

where,

`src` is the source file from bootflash, and `dest` is the destination file path.

Deleting Application Data

To delete application data from the application's persistent data mount, use the **app-hosting data appid** <*appid*> **delete** command in the privileged EXEC mode.

```
app-hosting data appid tea delete file
```

where,

`file` is the file to be deleted from the application's persistent data mount.

Verifying the Application-Hosting Configuration

Use these **show** commands to verify the configuration. You can use these commands in any order.

SUMMARY STEPS

1. **show app-hosting infra**
2. **show app-hosting list**
3. **show app-hosting bridge**
4. **show app-hosting detail**
5. **show app-hosting resource**
6. **show app-hosting utilization appid <app-name>**
7. **show tech-support app-hosting**

DETAILED STEPS

Procedure

Step 1 **show app-hosting infra**

Displays a summary of the app-hosting infra.

Note
Move CAF to running state before performing further operation.

Example:

```
switch(config)# show app-hosting infra
App signature verification: disabled
Docker partition size: 0 MB
Inband packet rate limit: 0 PPS
Services
---------------------------------
CAF 1.16.0.0 : Running
HA : Running
App Manager : Running
Libvirtd 4.7.0 : Running
Dockerd 18.09.0-ce : Running
Linux kernel 5.10.126 : Running
```

Step 2 **show app-hosting list**

Displays the list of apps that are running.

Example:

```
switch(config)# show app-hosting list
App id               State
------------------------------------------------------
nginx_1              started
```

Step 3 **show app-hosting bridge**

Displays the list of app-hosting bridges.

Example:

```
switch(config)# show app-hosting bridge
Bridge ID   VRF       IP Address       IPv6 Address
------------------------------------------------------------------------------
1           blue       172.10.23.45/24  ::/0
```

Step 4 **show app-hosting detail**

Displays detailed information about app-hosting.

Example:

```
switch(config)# show app-hosting detail
App id : nginx_1
Owner : appmgr
State : started
Application
Type : docker
Name : nginx
Version : latest
Description :
Author :
Path : /bootflash/nginx.tar.gz
URL Path :
Activated profile name : default

Resource reservation
Memory : 64 MB
Disk : 10 MB
CPU : 200 units

Platform resource profiles
Profile Name CPU(unit) Memory(MB) Disk(MB)
------------------------------------------------------------------

Attached devices
Name               Type               Alias
---------------------------------------------
iox_trace          serial/trace       serial3
iox_syslog         serial/syslog      serial2
iox_console_aux    serial/aux         serial1
iox_console_shell serial/shell        serial0

Network interfaces
---------------------------------------
eth0:
MAC address : 5254.9999.0000
IPv4 address : 192.168.10.130
IPv6 address : fe80::5054:99ff:fe99:0/64
Network name : iox-nat_docker0
Tx Packets : 9
Tx Bytes : 726
Tx Errors : 0
Rx Packets : 0
Rx Bytes : 0
Rx Errors : 0

Docker
------
Run-time information
Command :
Entry-point : /docker-entrypoint.sh nginx -g 'daemon off;'
Run options in use : --publish=40080:80
Package run options :
Application health information
Status : 0
Last probe error :
Last probe output :
```

Step 5 **show app-hosting resource**

Displays information about the resources for app-hosting.

Example:

```
switch(config)# show app-hosting resource
CPU:
Total: 7400 units
Available: 7200 units
VCPU:
Application Hosting
Additional References
Application Hosting
46
Count: 1
Memory:
Total: 3840(MB)
Available: 3776(MB)
Storage space:
Total: 110745(MB)
Available: 93273(MB)
vice
```

Step 6 **show app-hosting utilization appid <app-name>**

Displays the utilization for an application.

Example:

```
switch(config)# show app-hosting utilization appid nginx_1
Application: nginx_1
CPU Utilization:
CPU Allocation: 200 units
CPU Used: 0 %
Memory Utilization:
Memory Allocation: 64 MB
Memory Used: 7000 KB
Disk Utilization:
Disk Allocation: 10 MB
Disk Used: 0 MB
```

Step 7 **show tech-support app-hosting**

Displays all the app-hosting logs and the dependent component logs that are relevant.

This show-tech command collects the details for the following show commands:

Example:

```
show system internal app-hosting
show system internal app-hosting event-history debug
show system internal app-hosting event-history error
show system internal app-hosting event-history msgs
show app-hosting list
show app-hosting detail
show app-hosting utilization
show app-hosting infra
show app-hosting resource
show app-hosting bridge
show routing appmgr vrf all
show routing ipv6 appmgr vrf all
```

Configuration Examples for Application Hosting

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

Example: Enabling AppHosting Feature

This example shows how to enable Cisco Apphosting feature.

```
switch# configure terminal
switch(config)# feature app-hosting
switch(config)# end
```

Example: Configuring Application Hosting Bridge Connections

This example shows how to configure application hosting bridge connections.

```
switch(config)# configure terminal
switch(config)# app-hosting bridge 1
switch(config-app-hosting-bridge)# ip address 172.25.44.1/30
switch(config-app-hosting-bridge)# vrf member overlay-VRF
switch(config-app-hosting-bridge)# exit
switch(config)# app-hosting appid te_app
switch(config- app-hosting)# app-vnic bridge 1 guest-interface 0
switch(config-app-hosting-app-vnic)# guest-ipaddress 172.25.44.2/30
switch(config-app-hosting-app-vnic)# exit
switch(config-app-hosting-appid)# app-default-gateway 172.25.44.1 guest-interface 0
switch(config-app-hosting)# end
```

Example: Configuring Docker Run Time Options

The following example shows how to configure docker run time options.

```
switch> enable
switch# configure terminal
switch(config)# app-hosting appid te_app
switch(config-app-hosting)# app-resource docker
switch(config-app-hosting-docker)# run-opts 1 "-v $(APP_DATA):/data"
switch(config-app-hosting-docker)# end
```

Example: Configuring Application Hosting on the Management Interface

This example shows how to configure application hosting on the management interface.

```
switch> enable
switch# configure terminal
switch(config)# app-hosting appid te_app
switch(config-app-hosting)# app-vnic management guest-interface 0
switch(config-app-hosting)# end
```

Example: Overriding App Resource Configuration

This example shows how to override an app resource configuration.

```
switch> enable
switch# configure terminal
switch(config)# app-hosting appid te_app
switch(config-app-hosting)# app-resource profile custom
switch(config-app-resource-profile-custom)# cpu 7400
switch(config-app-resource-profile-custom)# memory 2048
switch(config-app-resource-profile-custom)# end
```

Additional References

Related Documents

Related Topic	Document Title
Configuring Apphosting	Cisco Nexus 3000 and 9000 Series NX-API REST SDK User Guide and API Reference

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. 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, you do not need to have an account with Cisco.com.

Table 1: Feature Information for Application Hosting

Feature Name	Release	Feature Information
Support for Cisco Application Hosting Framework (CAF) on Cisco Nexus 9808 platform switches	Cisco NX-OS Release 10.3(3)F	The application hosting feature is now supported on Cisco Nexus 9808 platform switches.
Support for Cisco Application Hosting Framework (CAF) on Cisco Nexus 9504 and 9508 platform switches with -R and -R2 line cards	Cisco NX-OS Release 10.3(3)F	The application hosting feature is now supported on Cisco Nexus 9504 and 9508 platform switches with -R and -R2 line cards. This feature is also supported on Cisco N3K-C36180YC-R, N3K-C3636C-R, and N3K-C36480LD-R2 switches
Cisco Application Hosting Framework (CAF)	Cisco NX-OS Release 10.3(1)F	A hosted application is a software as a service (SaaS) solution, and you can execute and operate this solution entirely from the cloud. This module describes the Cisco application hosting feature and how to enable it. The application hosting feature is supported on Cisco Nexus 9300 series FX, FX2, GX, and GX2 platforms and Cisco Nexus 9500 series modular switches with FX and GX line cards.