IOx Application Hosting

This section contains the following topics:

Application Hosting

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

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

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.

Cisco devices support third-party off-the-shelf applications built using Linux tool chains. Users can run custom applications cross-compiled with the software development kit that Cisco provides.

IOx Overview

IOx is a Cisco-developed end-to-end application framework that provides application hosting capabilities for different application types on Cisco network platforms.

IOx architecture for the IR8100 is different compared to other Cisco platforms that use the hypervisor approach. In other platforms, IOx runs as a virtual machine. IOx runs as a process on the IR8100.

Cisco Application Hosting Overview

The IR8100 allows you to deploy applications using the application hosting CLI commands. You can also deploy applications using the Local Manager and Fog Director.

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 via REST APIs.
- Provides a CLI endpoint.
- Provides an application hosting infrastructure referred to as Cisco Application Framework (CAF).
- Helps in the setup of platform-specific networking (packet-path) via VirtualPortGroup and management interfaces.

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

Applications 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 file.

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

IOXMAN

IOXMAN is a process that establishes a tracing infrastructure to provide logging or tracing services for guest applications, except Libvirt, that emulates serial devices. IOXMAN is based on the lifecycle of the guest application to enable and disable the tracing service, to send logging data to IOS syslog, to save tracing data to IOx tracelog, and to maintain IOx tracelog for each guest application.

Application Hosting on the IR8100 Industrial Integrated Services Router

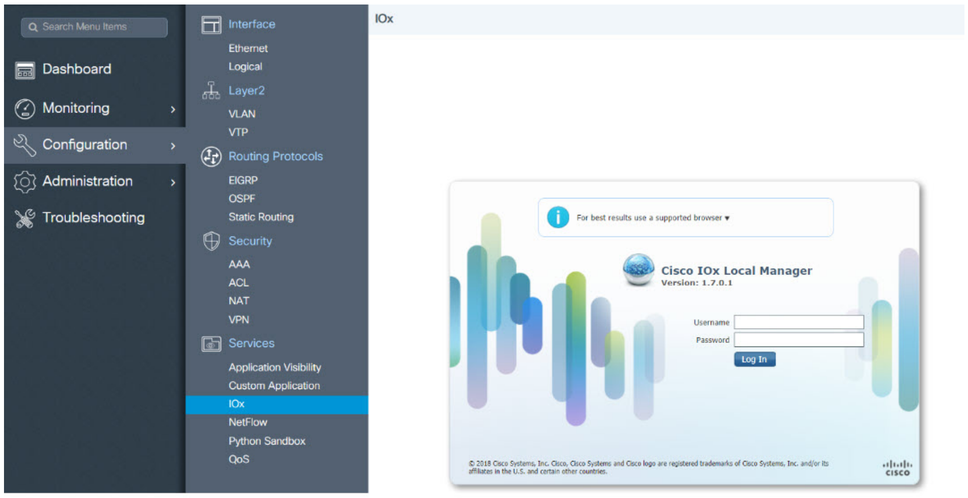
This section describes the application hosting characteristics specific to the IR8100.

Note The IR8100 CPU is not based on x86 architecture like other routers. Therefore, this requires the application to comply with the ARM 64-bits architecture.

Application hosting can be achieved using the application hosting CLI commands as well as using Local Manager and Fog Director. Application hosting using Local Manager is done through WebUI. To deploy the applications using Local Manager, enable WebUI and then log in to Local Manager.

Application Management is available using FND.

Figure 1: Local Manager



1. From WebUI, click on **Configuration** > **Services** > **IOx**

2. Log in using the username and password configured.

3. Follow the steps for the application lifecycle in the **Cisco IOx Local Manager Reference Guide** using this link: https://www.cisco.com/c/en/us/td/docs/routers/access/800/software/guides/iox/lm/reference-guide/1-7/b_iox_lm_ref_guide_1_7/b_iox_lm_ref_guide_1_7_chapter_011.html

The next section explains the deployment of an application using the application hosting CLI commands.

VirtualPortGroup

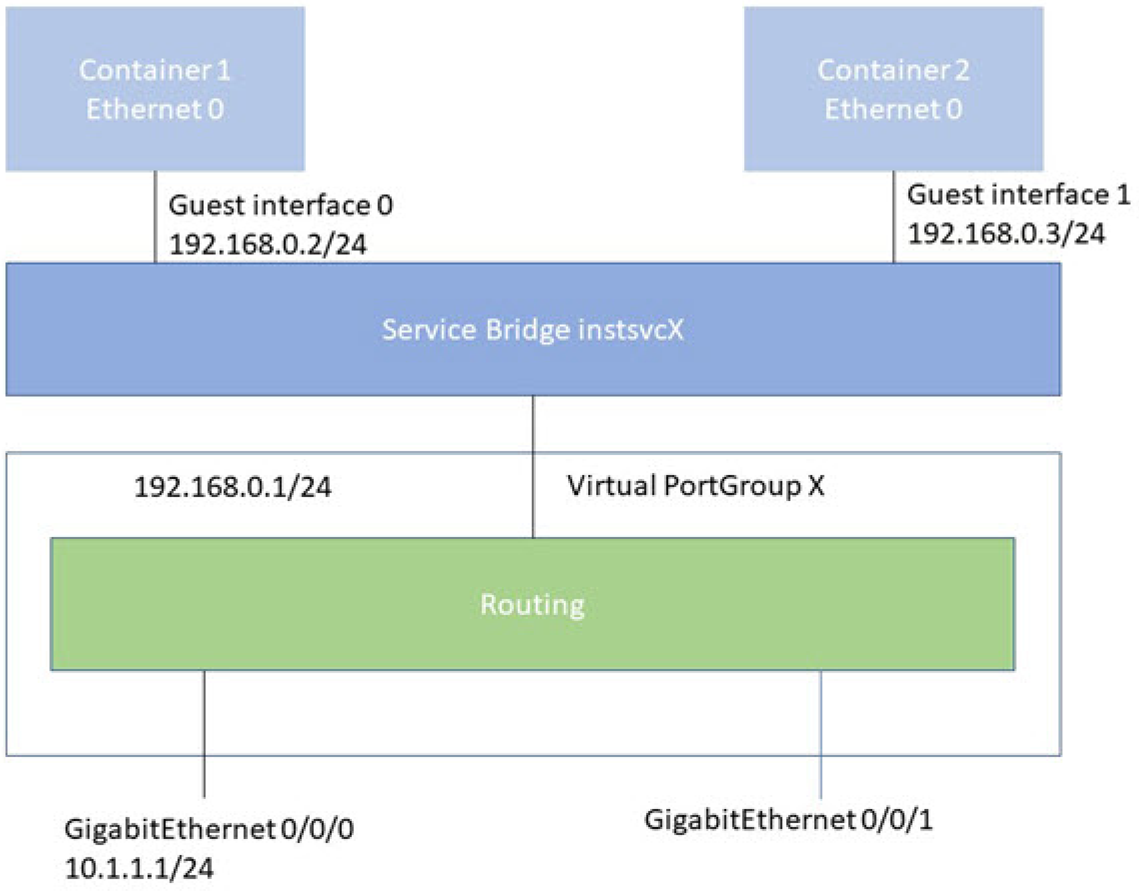
The VirtualPortGroup is a software construct on Cisco IOS that maps to a Linux bridge IP address. As such, the VirtualPortGroup represents the switch virtual interface (SVI) of the Linux container. Each bridge can contain multiple interfaces; each mapping to a different container. Each container can also have multiple interfaces.

VirtualPortGroup interfaces are configured by using the interface virtualportgroup command. Once these interfaces are created, IP address and other resources are allocated.

The VirtualPortGroup interface connects the application hosting network to the IOS routing domain. The Layer 3 interface of the application receives routed traffic from IOS. The VirtualPortGroup interface connects through the SVC Bridge to the container/application interface.

The following graphic helps to understand the relationship between the VirtualPortGroup and other interfaces, as it is different than the IR8x9 routers.

Figure 2: Virtual Port Group Mapping



vNIC

For the container life cycle management, 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 vNIC is part of the application container. The other interface, called vpgX is part of the host system.

NIC is the standard Ethernet interface inside the container that connects to the platform dataplane for the sending and receiving of packets. IOx is responsible for the gateway (VirtualPortGroup interface), IP address, and unique MAC address assignment for each vNIC in the container.

The vNIC inside the container/application are considered as standard Ethernet interfaces.

How to Configure Application Hosting

Enabling IOx

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

Note In the steps that follow, IP HTTP commands do not enable IOX, but allow the user to access the WebUI to connect the IOX Local Manager.

SUMMARY STEPS

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

DETAILED STEPS

Steps	Command	Purpose
1.	**enable** Example: `Device>enable`	Enables privileged EXEC mode. Enter your password if prompted.
2.	**configure terminal** Example: `Device#configure terminal`	Enters global configuration mode.
3.	**iox** Example: `Device(config)#iox`	Enables IOx
4.	**ip http server** Example: `Device(config)#ip http server`	Enables the HTTP server on your IP or IPv6 system.
5.	**ip http secure-server** Example: `Device(config)#ip http secure-server`	Enables a secure HTTP (HTTPS) server.

Steps	Command	Purpose
6.	**username** *name* **privilege** *level* **password** {**0** | **7** | *user-password* }*encrypted-password* Example: `Device(config)#username cisco privilege 15 password 0 cisco`	Establishes a username-based authentication system and privilege level for the user. The username privilege level must be configured as 15.

7.	**end**
	Example:
	`Device(config-if)#end`
		Exits interface configuration mode and returns to privileged EXEC mode.

Configuring a VirtualPortGroup to a Layer 3 Data Port

Multiple Layer 3 data ports can be routed to one or more VirtualPortGroups or containers. VirutalPortGroups and Layer 3 data ports must be on different subnets.

Enable the **ip routing** command to allow external routing on the Layer 3 data-port.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **ip routing**
4. **interface** *type number*
5. **no switchport**
6. **ip address** *ip-address mask*
7. **exit**
8. **interface** *type number*
9. **ip address** *ip-address mask*
10. **end**

DETAILED STEPS

Step	Command
1.	**enable** Example: `Device>enable`
2.	**configure terminal** Example: `Device#configure terminal`
3.	**ip routing** Example: `Device(config)#ip routing`
4.	**interface type number** Example: `Device(config)#interface gigabitethernet 0/0/0`
5.	**no switchport** Example: `Device(config-if)#no switchport`
6.	**ip address ip-address mask** Example: `Device(config-if)#ip address 10.1.1.1 255.255.255.0`
7.	**exit** Example: `Device(config-if)#exit`
8.	**interface type number** Example: `Device(config)#interface virtualportgroup 0`

Step	Command
9.	**ip address ip-address mask** Example: `Device(config-if)#ip address 192.168.0.1 255.255.255.0`

10.	**end**
	Example:
	`Device(config-if)#end`

11.	**configure terminal** Enter configuration commands, one per line. End with CNTL/Z. Example: `Device#configure terminal`
12.	**app-hosting appid app1** Example: `Device(config)#app-hosting appid app1`
13.	**app-vnic gateway0 virtualportgroup 0 guest-interface 0** Example: `Device(config-app-hosting)#app-vnic gateway0 virtualportgroup 0 guest-interface 0`
14.	**guest-ipaddress 192.168.0.2 netmask 255.255.255.0** Example: `Device(config-app-hosting-gateway0)#guest-ipaddress 192.168.0.2 netmask 255.255.255.0`
15.	**app-default-gateway 192.168.0.1 guest-interface 0** Example: `Device(config-app-hosting-gateway0)#app-default-gateway 192.168.0.1 guest-interface 0`

16.	**end**
	Example:
	`Device#end`

Installing and Uninstalling Apps

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

Step	Command
1.	**enable** Example: `Device>enable`
2.	**app-hosting install appid** *application-name* **package** *package-path* Example: `Device#app-hosting install appid lxc_app package flash:my_iox_app.tar`
3.	**app-hosting activate appid** *application-name* Example: `Device#app-hosting activate appid app1`

Step	Command
4.	**app-hosting start appid** *application-name* Example: `Device#`**`app-hosting start appid app1`**
5.	**app-hosting stop appid** *application-name* Example: `Device#`**`app-hosting stop appid app1`**
6.	**app-hosting deactivate appid** *application-name* Example: `Device#`**`app-hosting deactivate appid app1`**
7.	**app-hosting uninstall appid** *application-name* Example: `Device#`**`app-hosting uninstall appid app1`**

Overriding the App Resource Configuration

Resource changes will take effect only after the app-hosting activate command is configured.

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

Step	Command
1.	**enable** Example: `Device>`**`enable`**
2.	**configure terminal** Example: `Device#`**`configure terminal`**
3.	**app-hosting appid** *name* Example: `Device(config)#`**`app-hosting appid app1`**
4.	**app-resource profile** *name* Example: `Device(config-app-hosting)#`**`app-resource profile custom`**
5.	**cpu** *unit* Example: `Device(config-app-resource-profile-custom)#` **`cpu 800`**
6.	**memory** *memory* Example: `Device(config-app-resource-profile-custom)#` **`memory 512`**
7.	**vcpu** *number* Example: `Device(config-app-resource-profile-custom)#` **`vcpu 2`**

Step	Command

8.	**end**
	Example:
	`Device(config-app-resource-profile-custom)# end`

Verifying the Application Hosting Configuration

SUMMARY STEPS

1. **enable**
2. **show iox-service**
3. **show app-hosting detail**
4. **show app-hosting list**

DETAILED STEPS

1. **enable**

Enables privileged EXEC mode. Enter your password if prompted.

Example:

```
Device>enable
```

2. show iox-service

Displays the status of all IOx services

Example:

```
Device# show iox-service
IOx Infrastructure Summary:
---------------------------
IOx service (CAF) : Running
IOx service (HA) : Not Supported
IOx service (IOxman) : Running
IOx service (Sec storage) : Running
Libvirtd 5.5.0 : Running
Dockerd 18.03.0 : Running
Device#
```

3. show app-hosting detail

Displays detailed information about the application.

Example:

```
Device#show app-hosting detail
App id : iperf
Owner : iox
State : RUNNING
Application
Type : lxc
```

```
Name : nt08-stress
Version : 0.1
Description : Stress Testing Application
Path : bootflash:sparrow_lxc.tar
URL Path :
Activated profile name : custom

Resource reservation
Memory : 64 MB
Disk : 2 MB
CPU : 500 units
CPU-percent : 31 %
VCPU : 1

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

Network interfaces
---------------------------------------
eth0:
MAC address : 52:54:dd:8e:55:19
IPv4 address : 192.168.11.2
IPv6 address : ::
Network name : VPG1
```

4. show app-hosting list

Displays the list of applications and their status.

Example:

```
Device#show app-hosting list
App id                           State
------------------------------------------------------
app1                             RUNNING
```

Configuration Examples for Application Hosting

See the following examples:

Example: Enabling IOx

```
Device> enable
Device# configure terminal
Device(config)# iox
Device(config)# ip http server
Device(config)# ip http secure-server
Device(config)# username cisco privilege 15 password 0 cisco
Device(config)# end
```

Example: Configuring a VirtualPortGroup to a Layer 3 Data Port

```
Device> enable
Device# configure terminal
```

```
Device(config)# ip routing
Device(config)# interface gigabitethernet 0/0/0
Device(config-if)# no switchport
Device(config-if)# ip address 10.1.1.1 255.255.255.0
Device(config-if)# exit
Device(config)# interface virtualportgroup 0
Device(config-if)# ip address 192.168.0.1 255.255.255.0
Device(config-if)# end
```

Example: Installing and Uninstalling Apps

```
Device> enable
Device# app-hosting install appid app1 package flash:my_iox_app.tar
Device# app-hosting activate appid app1
Device# app-hosting start appid app1
Device# app-hosting stop appid app1
Device# app-hosting deactivate appid app1
Device# app-hosting uninstall appid app1
```

Example: Overriding the App Resource Configuration

```
Device# configure terminal
Device(config)# app-hosting appid app1
Device(config-app-hosting)# app-resource profile custom
Device(config-app-resource-profile-custom)# cpu 800
Device(config-app-resource-profile-custom)# memory 512
Device(config-app-resource-profile-custom)# vcpu 2
Device(config-app-resource-profile-custom)# end
```

Native docker support

Native Docker Support enables users to deploy the docker applications on the IR1800. The application lifecycle process is similar to the procedure in the Installing and Uninstalling Apps section. For docker applications, entry point configuration is required as part of the application configuration. Please refer to the following example for the entry point configuration.

```
Router#conf t
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#app-hosting appid app3
Router(config-app-hosting)#app-vnic gateway0 virtualportgroup 0 guest-interface 0
Router(config-app-hosting-gateway0)#guest-ipaddress 192.168.0.7 netmask 255.255.255.0
Router(config-app-hosting-gateway0)#app-default-gateway 192.168.0.1 guest-interface 0
Router(config-app-hosting)#app-resource docker
Router(config-app-hosting-docker)#run-opts 1 "--entrypoint '/bin/sleep 10000'"
Router(config-app-hosting-docker)#end
Router#
```

The output for docker applications is shown in the following example:

```
Router#show app-hosting detail
App id : app1
Owner : iox
State : RUNNING
Application
Type : docker
Name : aarch64/busybox
Version : latest
```

```
Description :
Path : bootflash:busybox.tar
Activated profile name : custom
Resource reservation
Memory : 431 MB
Disk : 10 MB
CPU : 577 units
VCPU : 1
Attached devices
Type Name Alias
---------------------------------------------
serial/shell iox_console_shell serial0
serial/aux iox_console_aux serial1
serial/syslog iox_syslog serial2
serial/trace iox_trace serial3
Network interfaces
---------------------------------------
eth0:
MAC address : 52:54:dd:e9:ab:7a
IPv4 address : 192.168.0.7
Network name : VPG0
Docker
------
Run-time information
Command :
Entry-point : /bin/sleep 10000
Run options in use : --entrypoint '/bin/sleep 10000'
Application health information
Status : 0
Last probe error :
Last probe output :
Router#
```

Signed Application Support

Cisco Signed applications are now supported on the IR1800. In order to install a signed application, signed verification has to be enabled on the device. Signed verification can be enabled by following the following instructions.

```
Router#configure terminal
Enter configuration commands, one per line. End with CNTL/Z.
Router(config)#
Router(config)#app-hosting signed-verification
Router(config)#
Router(config)#exit
```

After enabling the signed verification, follow the instructions in the Installing and Uninstalling Apps section under IOx Application Hosting in order to install the application.

Cisco Cyber Vision and Edge Intelligence

Cisco Cyber Vision Center (CVC) gives more visibility into Industrial IoT networks across Industrial Control Systems (ICS) with real-time monitoring of control and data networks. On IoT IOS-XE platforms beginning with release 17.4, integration of CVC is supported by deploying IOX Cyber Vision sensor. With this sensor deployed on IoT Routers, the platform can forward the traffic from IOX applications to Cyber Vision Center for real-time monitoring and we can forward any captured PCAP files to Vision center from IOX application. The minimum Cybervision release is 3.1.1 to work with the IR8100. For more information about CVC, see

Deployment of Cyber Vision Center (CVC) on IOS-XE platform and Release Notes for Cisco Cyber Vision Release 3.1.1.

Cisco Edge Intelligence allows for simplified data extraction from IoT sensors, transformation, governance and delivery to applications that need this data. The release for the IR8100 is version 1.0.6, and is called:

ei_1.0.6_ir1101.K9.tar

Complete information about Cisco Edge Intelligence is found at:

https://developer.cisco.com/edge-intelligence/.