Programmability Configuration Guide

First Published: 2025-08-08

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

Preface

This preface describes the conventions of this document and information on how to obtain other documentation. It also provides information on what's new in Cisco product documentation.

Document Conventions

This document uses the following conventions:

Convention	Description
^ or Ctrl	Both the ^ symbol and Ctrl represent the Control (Ctrl) key on a keyboard. For example, the key combination ^**D** or **Ctrl-D** means that you hold down the Control key while you press the D key. (Keys are indicated in capital letters but are not case sensitive.)
bold font	Commands and keywords and user-entered text appear in **bold** font.
Italic font	Document titles, new or emphasized terms, and arguments for which you supply values are in *italic* font.
`Courier` font	Terminal sessions and information the system displays appear in `courier` font.
`Bold Courier` font	**`Bold Courier`** font indicates text that the user must enter.
[x]	Elements in square brackets are optional.
...	An ellipsis (three consecutive nonbolded periods without spaces) after a syntax element indicates that the element can be repeated.
|	A vertical line, called a pipe, indicates a choice within a set of keywords or arguments.
[x | y]	Optional alternative keywords are grouped in brackets and separated by vertical bars.

Convention	Description
{x | y}	Required alternative keywords are grouped in braces and separated by vertical bars.
[x {y | z}]	Nested set of square brackets or braces indicate optional or required choices within optional or required elements. Braces and a vertical bar within square brackets indicate a required choice within an optional element.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
< >	Nonprinting characters such as passwords are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

Reader Alert Conventions

This document may use the following conventions for reader alerts:

Note Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the manual.

Tip Means *the following information will help you solve a problem.*

Caution Means *reader be careful*. In this situation, you might do something that could result in equipment damage or loss of data.

Timesaver Means *the described action saves time.* You can save time by performing the action described in the paragraph.

Take note of the following general safety warnings:

Warning IMPORTANT SAFETY INSTRUCTIONS

Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents. Read the installation instructions before using, installing, or connecting the system to the power source. Use the statement number at the beginning of each warning statement to locate its translation in the translated safety warnings for this device.

SAVE THESE INSTRUCTIONS

Related Documentation

Note Before installing or upgrading the device, refer to the device release notes.

Obtaining Documentation and Submitting a Service Request

For information on obtaining documentation, submitting a service request, and gathering additional information, see the monthly *What's New in Cisco Product Documentation*, which also lists all new and revised Cisco technical documentation, at:

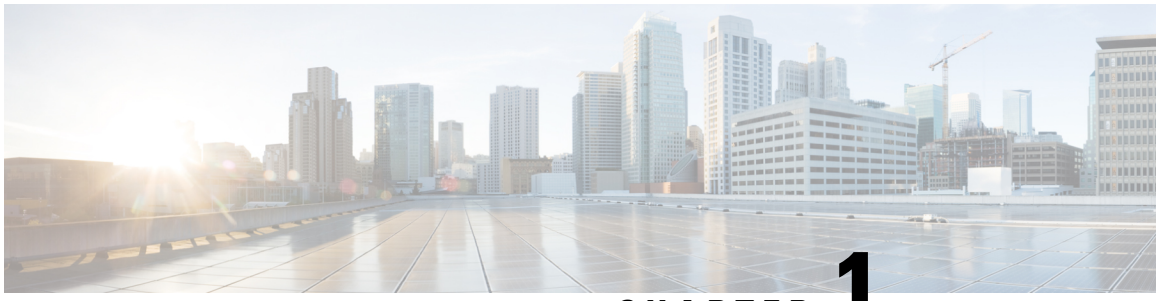
http://www.cisco.com/c/en/us/td/docs/general/whatsnew/whatsnew.html

Subscribe to the *What's New in Cisco Product Documentation* as a Really Simple Syndication (RSS) feed and set content to be delivered directly to your desktop using a reader application. The RSS feeds are a free service and Cisco currently supports RSS version 2.0.

CONTENTS

PART I

Provisioning



CHAPTER 1

Provisioning

Feature Information for Zero-Touch Provisioning

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 1: Feature Information for Zero-Touch Provisioning

Feature Name	Release	Feature Information
Zero-Touch Provisioning	Cisco IOS XE 17.18.1	To address network provisioning challenges, Cisco introduces a zero-touch provisioning model. This feature was implemented on the Cisco C9350 Series Switches.
Zero-Touch Provisioning: HTTP Download	Cisco IOS XE 17.18.1	Zero-Touch Provisioning supports HTTP and TFTP file download. This feature was implemented on the Cisco C9350 Series Switches.
DHCPv6 Support for Zero-Touch Provisioning	Cisco IOS XE 17.18.1	This feature was implemented on the Cisco C9350 Series Switches.

Feature Name	Release	Feature Information
Side-Effect Synchronization of the Configuration Database	Cisco IOS XE 17.18.1	During configuration changes in the DMI, a partial synchronization of the changes that are triggered when a command or RPC is configured occurs. This is called the side-effect synchronization, and it reduces the synchronization time and NETCONF downtime. This feature was implemented on the Cisco C9350 Series Switches and Cisco C9610 Series Switches.
Zero-Touch Provisioning Through YANG Models	Cisco IOS XE 17.18.1	ZTP is enabled through YANG models when NETCONF is enabled. This feature is supported on all platforms that support NETCONF-YANG.
Zero-Touch Provisioning Support on Data Port	Cisco IOS XE 17.18.1	ZTP is supported on data port for both IPv4 and IPv6. This feature was implemented on the Cisco C9350 Series Switches.
Secure Zero-Touch Provisioning	Cisco IOS XE 17.18.1	Secure ZTP provisions a device securely while booting in factory default state. This feature was implemented on the Cisco C9350 Series Switches.

Restrictions for Zero-Touch Provisioning

- Zero-touch provisioning is supported only on Cisco Catalyst 9350 Series Switches.

Information About Zero-Touch Provisioning

This section provides information about the DHCP server configuration, DHCPv6 support, Secure ZTP, bootstrapping information, and so on.

Zero-Touch Provisioning Overview

Zero-touch provisioning (ZTP) provides open bootstrap interfaces to automate network device provisioning in heterogeneous network environments.

When a device that supports ZTP starts up, and does not find the startup configuration (during initial installation), the device enters the zero-touch provisioning mode. The device searches for a DHCP server, bootstraps itself with its interface IP address, gateway, and Domain Name System (DNS) server IP address,

and enables Guest Shell. The device then obtains the IP address or URL of an HTTP or a TFTP server, and downloads the Python script from an HTTP or a TFTP server to configure the device.

ZTP is supported through data ports on all supported platforms.

Guest Shell provides the environment for the Python script to run. Guest Shell runs the downloaded Python script and applies an initial configuration to the device.

After initial provisioning is complete, Guest Shell remains enabled. For more information, see the Guest Shell chapter.

Note If ZTP fails, the device falls back on AutoInstall to load the configuration file. For more information, see Using AutoInstall and Setup.

DHCP Server Configuration for Zero-Touch Provisioning

In ZTP, a DHCP server must be running on the same network as the new device that is being provisioned. ZTP is supported on both management ports and in-band ports.

When the new device is switched on, it retrieves the IP address information of the HTTP or TFTP server where the Python script resides, and the folder path of the Python script from the DHCP server. For more information on Python Scripts, see the *Python API* chapter.

The DHCP server responds to DHCP discovery events with the following options:

- Option 150: (Optional) Contains a list of IP addresses that point to the HTTP or TFTP server in the management network that hosts the Python scripts to be run.
- Option 67: Contains the Python script file path in the HTTP/TFTP server.

After receiving these DHCP options, the device connects to the HTTP or TFTP server, and downloads the Python script. At this point, because the device does not have a route to reach the HTTP or TFTP server, it uses the default route provided by the DHCP server.

DHCP Client Behavior

From Cisco IOS XE 17.18.1 release onwards, the device product identification (PID), Cisco Network Plug and Play (PnP), the serial number, and the MAC address is set in the DHCP client options.

These client options can be used as variables at the DHCP server. The DHCP client options can be used to identify the device and send a unique Python file through option 67 or boot filename.

Table 2: DHCP Client Options

Device Information	DHCPv4	DHCPv6
Product ID	124	16/toggles
Serial number	61/toggles	16/toggles
MAC address	61/toggles	1
Cisco Network PnP	60	15

DHCPv6 Support

In Cisco IOS XE 17.18.1, DHCP Version 6 support is added to the Zero-Touch Provisioning feature. DHCPv6 is enabled by default, and will work on any device that boots without a startup configuration.

Note DHCPv6 is only supported on Catalyst 9350 Series Switches.

DHCPv6 is supported by both TFTP and HTTP download of Python scripts. If this download fails, the device reverts to the initial or factory settings (without any configuration). For both DHCPv4 and DHCPv6 to work, the correct HTTP or TFTP file path must be available in the DHCP configuration.

There can be scenarios where the same interface can have both IPv4 and IPv6 addresses, or two different interfaces in a network—one can receive IPv4 traffic and the other, IPv6 traffic. We recommend that you use either the DHCPv4 or DHCPv6 option in your deployment.

The following is a sample DHCPv4: {/etc/dhcp/dhcpd.conf:}

```
host <hostname> {
  hardware ethernet xx:xx:xx:xx:xx:xx;
  option dhcp-client-identifier "xxxxxxxxxxxxx";
  option host-name "<hostname>".
  option log-servers x.x.x.x;
  fixed-address x.x.x.x;
  if option vendor-class-identifier = "..." {
    option vendor-class-identifier "...";
    if exists user-class and option user-class = "iPXE" {
      filename "http://x.x.x.x/…/<image>";
    } else {
      filename "http://x.x.x.x/…/<script-name>";
    }
  }
}
```

The following is a sample ISC DHCPv6 server configuration:

```
 option dhcp6.bootfile-url  "http://[2001:DB8::21]/sample_day0_script.py";
```

Secure ZTP

As per RFC 8572, Secure ZTP is a technique to securely provision a device, while it is booting in a factory default state. The provisioning updates the boot image, commits an initial configuration, and runs customer-specific scripts.

The existing ZTP can download a configuration script from an HTTP or a TFTP server, and run it on the device, but it does not securely provision a device.

To securely provision a device, the following requirements must be met:

- The management system must validate that it is provisioning a valid device.
- The device must validate that it is being deployed in the correct network.
- The device must validate that the provisioning data has not been tampered with.
- Provisioning must use a secure transport protocol for data communication.

The Secure ZTP feature is enabled by default, and it cannot be disabled. The feature coexists with the classic ZTP. Based on the response that comes from the DHCP server, either ZTP or Secure ZTP is enabled. Option 67 triggers ZTP, and Option 143 or 136 triggers Secure ZTP.

To use Secure ZTP, users must get an ownership voucher from a Cisco Manufacturer Authorized Signing Authority (MASA) server.

Note Secure ZTP supports only Python scripts.

Secure ZTP Transport Protocol

Secure ZTP uses HTTPS as the transport protocol when communicating with the RESTCONF bootstrapping servers.

When establishing a connection, the device provides its Cisco Secure Unique Device Identifier (SUDI) certificate during the TLS handshake. The certificate serves as an authentication token. No other authentication is supported. The SUDI certificate validates the device on the server side.

The bootstrapping server provides its own certificate during the TLS handshake to the device. If the device has the server-trust anchor corresponding to this server, the device performs an X.509 certificate validation of the server certificate against the server-trust anchor. (The device can get the server-trust anchor from the redirect server.)

If the device does have a server-trust anchor, but the X.509 certificate is not validated successfully, the device must discard the server-trust anchor, accept the connection, and mark the server as not trusted.

If the device does not have a server-trust anchor, it accepts the server certificate, but marks it as not trusted.

According to the RESTCONF protocol, a device must first initiate the root-resource discovery, before it can proceed with RPC requests. The following is a sample root-discovery request:

```
GET /.well-known/host-meta HTTP/1.1
Host: example.com
Accept: application/xrd+xml
```

The following is a sample root-discovery response:

```
HTTP/1.1 200 OK
Content-Type: application/xrd+xml
Content-Length: nnn
<XRD xmlns='http://docs.oasis-open.org/ns/xri/xrd-1.0'>
    <Link rel='restconf' href='/restconf_root'/>
</XRD>
```

The value in the HREF attribute is used in the follow-up requests to the RESTCONF server, for example, when

```
href='/restconf_root'
```

is received, the header for the get-bootstrapping-data RPC is

```
POST /restconf_root/operations/ietf-sztp-bootstrap-server:
get-bootstrapping-data HTTP/1.1
```

The device receives a plain XML file, and the fields in the file are Base64 encoded. The device runs Base64 decoding before extracting the information and attempting to recognize its structure. The decoded structure format can either be JSON or XML.

DHCP Options for Secure ZTP

DHCP redirect Options 136 (DHCPv6) and 143 (DHCPv4) are supported for Secure ZTP. These options are used to provision a client with one or more Uniform Resource Identifiers (URIs) for bootstrap servers.

A device in the Zero-Touch Provisioning mode carries out a DHCP discovery to find a DHCP server, and to receive the IP address and other information. When the DHCP client receives Options 136 and 143, the Secure ZTP procedures are started, and the Secure ZTP handling functions are invoked. If the Secure ZTP handling functions fail, the handling procedure is repeated indefinitely, until it is either successful, or is interrupted by the user.

Bootstrapping Data

Bootstrapping data refers to a collection of data that a device obtains during the bootstrapping process.

Bootstrapping data (both redirect and onboarding information) can be signed or unsigned. An ownership voucher, owner certificate, and conveyed information must be present in signed bootstrapping data.

Signed data is trusted. If the signed data is redirect information, trust anchors for the bootstrap servers that are being redirected to are saved for future TLS handshakes. Signed onboarding information is trusted and is applied to the device.

When a bootstrap server provides signed data, the data is considered trusted. However; this does not make the server trusted. If the server is considered untrusted before receiving the data, it remains untrusted, and the server does not receive progress reports.

Unsigned data do not need to present the ownership voucher and the owner certificate artifacts; infact, these are discarded, if present. Unsigned data is accepted only if it is redirect data or is received from a trusted server. Receiving unsigned data from an untrusted onboarding server is considered an error, and the device continues to the next server on its list.

Bootstrapping Servers

A bootstrap server is a RESTCONF server that uses the HTTPS data transport protocol. A bootstrap server can be a redirect server or an onboarding server.

A redirect server sends a list of bootstrapping servers to a device. The device can attempt bootstrapping from any server in the list. Each server entry has a server address, a server port, and a trust anchor (X.509 certificate) that verifies the server's certificate presented during the TLS handshake. The trust anchor is accepted only if the redirect information is signed, or if the current redirect server is trusted. Otherwise, all the trust anchors received from the redirect server are discarded.

An onboarding server is a server that sends the actual bootstrapping data. The bootstrapping data contains the following types of information:

- Required updated image that the device must run: The required version, URLs for downloading the image, and information about how to validate the downloaded image.
- Preconfiguration script: A Python script that must be run before the initial device configuration is applied.

- Configuration: The initial device configuration. This configuration can be in the Cisco IOS CLI format of a NETCONF edit-config format.
- Postconfiguration script: A Python script that must be run after the initial device configuration is applied.

A bootstrapping server can be trusted or untrusted. A trusted server can send bootstrapping data that is not necessarily signed. A trusted server can be demoted into an untrusted server if the server certificate it sends during the TLS handshake fails to validate the server-trust anchor on the device.

Image Update Support

When the bootstrapping data requires a specific OS version, the device checks the version of the software that it is currently running. In case of a mismatch, even if the running version is newer, the device uses the URL provided in the bootstrapping data, and downloads the specified image. No modifications are done to the URL, and no additional authentication information is added by the device. If several URLs are specified in the bootstrapping data, the device will loop over the list of URLs until the first successful attempt or until the list is exhausted.

After an image is downloaded, if the *image-verification* data is included in the bootstrapping information, the device uses the specified algorithm to calculate the hash of the image and compares the result with the hash string included in the bootstrapping data. If multiple algorithms or hashes are provided in the bootstrapping data, the device picks the first-supported algorithm in the list to calculate and compare the hash.

After the image is verified, the image is installed, and the device reboots. State information is not stored before the reboot. When the device boots up, it enters the Secure ZTP process again and repeats all the steps till the image updating. Because the device is running the correct image, it does not need an image update, and the device will proceed with applying the on-boarding information.

Owner Certificate

The owner certificate is an X.509 certificate that binds an owner's identity to a public key, which a device can use to validate a signature in the conveyed information.

After verifying the ownership voucher, the device uses the pinned domain certificate extracted from the ownership voucher to validate the owner certificate.

The data field in the bootstrapping data that represents the owner certificate is Base64 encoded. The device then performs a Base64 decoding. The output of this decoding is a degenerate CMS structure, is DER-encoded, and is of the content type signedData. (Degenerate structure is a format that is commonly used to distribute X.509 certificates. These structures do not contain any signatures, but can have attached intermediate certificates.) The owner certificate is an x509 certificate, and the owner certificate artifact is a degenerate CMS structure that carries the x509 certificate.

Ownership Voucher

Assigning ownership is important to bootstrapping mechanisms. The ownership voucher is defined in Section 5.3 of RFC 8366. The primary purpose of a voucher is to securely assign a pledge to an owner. The voucher provides the pledge with details of the entity, which the pledge should consider as its owner.

The ownership voucher has a Cryptographic Message Syntax (CMS) structure, is Distinguished Encoding Rules-encoded (DER-encoded), and encloses YANG-modeled data.

This is a sample ownership voucher:

```
yang-data voucher-artifact:
+---- voucher
+---- created-on                    yang:date-and-time
+---- expires-on?                   yang:date-and-time
+---- assertion                     enumeration
+---- serial-number                 string
+---- idevid-issuer?                binary
+---- pinned-domain-cert            binary
+---- domain-cert-revocation-checks? boolean
+---- nonce?                        binary
+---- last-renewal-date?            yang:date-and-time
```

The CMS structure is signed using a private key that corresponds to the device's trust anchor. The device carries the public key (the trust anchor), which is the only thing it needs to verify the signature on the voucher. The private key is used by the Cisco MASA server when creating the voucher. The private key is securely stored in the Cisco Software Image Management (SWIM) server.

The device uses its trust anchor to extract the actual ownership voucher from the CMS structure. The signature on the ownership voucher is verified by using the public key enclosed in the device's trust anchor. This verification may require additional intermediate X.509 certificates, in which case, these must be attached to the ownership voucher.

After the signature on the ownership voucher is verified, the YANG-modeled data fields are extracted. If the *created-on* field is present, the device verifies whether the voucher was created in the past. If the *expires-on* field is present, the device verifies whether the voucher is expired or not. If these fields are not present, the voucher is rejected.

The device then verifies the required *serial-number* data field. The serial number of the device is learned by the server from the device's SUDI certificate sent to the server during the TLS handshake. The serial number helps the server to send the right configuration data to the right device.

When all the checks are successful, the device extracts the data field *pinned-domain-cert*, which is the main payload of the ownership voucher. The pinned domain certificate must be Base64 encoded. The device performs a Base64 decoding before handling the certificate. The decoded certificate is an X.509 certificate in DER format.

If an error occurs during validation, the ownership voucher is not verified, and the bootstrapping data is considered invalid.

Conveyed Information

The conveyed information artifact encodes the essential bootstrapping data for a device. This artifact is used to encode the redirect and onboarding information types.

The final step in validating the signed information is the verification of the signed-CMS structure that contains the conveyed information. The data field for the conveyed information is Base64 encoded. The device first performs Base64 decoding.

The device then verifies the signature using the certificates from the owner certificate's CMS structure and the pinned domain certificate as the trust anchor.

Progress Reporting

The device sends progress reports to the bootstrapping server. RFC 8572 provides more information on progress reporting.

When a bootstrapping server is trusted, it receives appropriate POST-request messages with the *ietf-sztp-bootstrap-server:report-progress* RPC.

There are two types of progress report messages—mandatory and optional. The mandatory reports indicate the start or termination of the bootstrapping process and include the following types of reports:

- bootstrap-initiated
- bootstrap-error
- bootstrap-complete
- boot-image-installed-rebooting
- config-error
- parsing-error
- pre-script-error
- post-script-error

The rest of progress report messages are optional. Optional progress report messages are only sent to the server if the bootstrapping data has set the *reporting-level* parameter to *verbose*.

This is a sample report-progress request from the device to the server:

```
POST /restconf/operations/ietf-sztp-bootstrap-server:report-progress/HTTP/1.1
HOST: example.com
Content-Type: application/yang.data+xml

<input xmlns="urn:ietf:params:xml:ns:yang:ietf-sztp-bootstrap-server">
<progress-type>bootstrap-error</progress-type>
<message>Failed to decode data</message>
</input>
```

This is a sample *No content* response from the server:

```
HTTP/1.1 204 No Content
Date: Sat, 31 May 2021 17:02:40 GMT
Server: example-server
```

Sample Zero-Touch Provisioning Configurations

Sample DHCP Server Configuration on a Management Port Using TFTP Copy

The following is a sample DHCP server configuration using TFTP copy when the DHCP server is connected through the management port on a device:

```
Device> enable
Device# configure terminal
Device(config)# ip dhcp excluded-address 10.1.1.1
Device(config)# ip dhcp excluded-address vrf Mgmt-vrf 10.1.1.1 10.1.1.10
Device(config)# ip dhcp pool pnp_device_pool
Device(config-dhcp)# vrf Mgmt-vrf
```

```
Device(config-dhcp)# network 10.1.1.0 255.255.255.0
Device(config-dhcp)# default-router 10.1.1.1
Device(config-dhcp)# option 150 ip 203.0.113.254
Device(config-dhcp)# option 67 ascii /sample_python_dir/python_script.py
Device(config-dhcp)# exit
Device(config)# interface gigabitethernet 1/0/2
Device(config-if)# no ip dhcp client request tftp-server-address
Device(config-if)# end
```

Sample DHCP Server Configuration on a Management Port Using HTTP Copy

The following is a sample DHCP server configuration using HTTP copy when the DHCP server is connected through the management port on a device:

```
Device> enable
Device# configure terminal
Device(config)# ip dhcp pool pnp_device_pool
Device(config-dhcp)# vrf Mgmt-vrf
Device(config-dhcp)# network 10.1.1.0 255.255.255.0
Device(config-dhcp)# default-router 10.1.1.1
Device(config-dhcp)# option 67 ascii http://198.51.100.1:8000/sample_python_2.py
Device(config-dhcp)# end
```

Sample DHCP Server Configuration on an In-Band Port Using TFTP Copy

The following is a sample DHCP server configuration using TFTP copy when the DHCP server is connected through the in-band port on a device:

```
Device> enable
Device# configure terminal
Device(config)# ip dhcp excluded-address 10.1.1.1
Device(config)# ip dhcp pool pnp_device_pool
Device(config-dhcp)# network 10.1.1.0 255.255.255.0
Device(config-dhcp)# default-router 10.1.1.1
Device(config-dhcp)# option 150 ip 203.0.113.254
Device(config-dhcp)# option 67 ascii /sample_python_dir/python_script.py
Device(config-dhcp)# exit
Device(config)# interface gigabitethernet 1/0/2
Device(config-if)# no ip dhcp client request tftp-server-address
Device(config-if)# end
```

Sample DHCP Server Configuration on an In-Band Port Using HTTP Copy

The following is a sample DHCP server configuration using HTTP copy when the DHCP server is connected through the in-band port on a device:

```
Device> enable
Device# configure terminal
Device(config)# ip dhcp excluded-address 10.1.1.1
Device(config)# ip dhcp pool pnp_device_pool
```

```
Device(config-dhcp)# network 10.1.1.0 255.255.255.0
Device(config-dhcp)# default-router 10.1.1.1
Device(config-dhcp)# option 67 ascii http://192.0.2.1:8000/sample_python_2.py
Device(config-dhcp)# end
```

Sample DHCP Server Configuration on a Linux Ubuntu Device

The following sample DHCP server configuration shows that the server is either connected to the management port or the in-band port in a device, and that a Python script is copied from a TFTP server.

```
root@ubuntu-server:/etc/dhcp# more dhcpd.conf
subnet 10.1.1.0 netmask 255.255.255.0 {
range 10.1.1.2  10.1.1.255;
      host 3850 {
          fixed-address                     10.1.1.246 ;
          hardware ethernet                 CC:D8:C1:85:6F:00;
          option bootfile-name !<opt 67>     " /python_dir/python_script.py";
         option tftp-server-name !<opt 150>  "203.0.113.254";
      }
  }
```

The following sample DHCP configuration shows that a Python script is copied from an HTTP server to the device:

```
Day0_with_mgmt_port_http
------------------------
subnet 192.168.1.0 netmask 255.255.255.0 {
 range 192.168.1.2 192.168.1.255;
      host C2-3850 {
          fixed-address                  192.168.1.246 ;
          hardware ethernet              CC:D8:C1:85:6F:00;
          option bootfile-name           "http://192.168.1.46/sample_python_2.py";
      }
  }
```

After the DHCP server starts running, you must start the management network-connected device. The rest of the configuration is automatic.

Sample DHCPv6 Server Configuration on a Management Port Using TFTP Copy

The following is a sample configuration using TFTP copy when the DHCPv6 server is connected through the management port on a device:

```
Device> enable
Device# configure terminal
Device(config)# ipv6 dhcp pool ztp
Device(config-dhcpv6)# address prefix 2001:DB8::1/64
Device(config-dhcpv6)# domain-name cisco.com
Device(config-dhcpv6)# bootfile-url tftp://[2001:db8::46]/sample_day0_script.py
Device(config-dhcpv6)# exit
Device(config)# interface vlan 20
```

```
Device(config-if)# ipv6 dhcp server ztp
Device(config-if)# end
```

Sample Python Provisioning Script

The following is a sample Python script that can be used from either an HTTP server or a TFTP server:

```
print "\n\n *** Sample ZTP Day0 Python Script *** \n\n"

# Importing cli module
import cli


print "\n\n *** Executing show platform  *** \n\n"
cli_command = "show platform"
cli.executep(cli_command)

print "\n\n *** Executing show version *** \n\n"
cli_command = "show version"
cli.executep(cli_command)

print "\n\n *** Configuring a Loopback Interface *** \n\n"
cli.configurep(["interface loop 100", "ip address 10.10.10.10 255.255.255.255", "end"])


print "\n\n *** Executing show ip interface brief  *** \n\n"
cli_command = "sh ip int brief"
cli.executep(cli_command)

print "\n\n *** ZTP Day0 Python Script Execution Complete *** \n\n"
```

Boot Log for Cisco 4000 Series Integrated Services Routers

The following sample Zero-Touch Provisioning boot log displays that Guest Shell is successfully enabled, the Python script is downloaded to the Guest Shell, and the Guest Shell is running the downloaded Python script and configuring the device for day zero.

```
% failed to initialize nvram
! <This message indicates that the startup configuration
is absent on the device. This is the first indication that the Day Zero work flow is
going to start.>

This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply
third-party authority to import, export, distribute or use encryption.
Importers, exporters, distributors and users are responsible for
compliance with U.S. and local country laws. By using this product you
agree to comply with applicable laws and regulations. If you are unable
to comply with U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html
```

```
If you require further assistance please contact us by sending email to
export@cisco.com.

cisco ISR4451-X/K9 (2RU) processor with 7941237K/6147K bytes of memory.
Processor board ID FJC1950D091
4 Gigabit Ethernet interfaces
32768K bytes of non-volatile configuration memory.
16777216K bytes of physical memory.
7341807K bytes of flash memory at bootflash:.
0K bytes of WebUI ODM Files at webui:.

%INIT: waited 0 seconds for NVRAM to be available


         --- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: %
!!<DO NOT TOUCH. This is Zero-Touch Provisioning>>
Generating 2048 bit RSA keys, keys will be non-exportable...
[OK] (elapsed time was 1 seconds)
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
Guestshell enabled successfully


 *** Sample ZTP Day0 Python Script ***


 *** Configuring a Loopback Interface ***


Line 1 SUCCESS: interface loop 100
Line 2 SUCCESS: ip address 10.10.10.10 255.255.255.255
Line 3 SUCCESS: end


 *** Executing show ip interface brief  ***


Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0/0   unassigned      YES unset  down                  down
GigabitEthernet0/0/1   unassigned      YES unset  down                  down
GigabitEthernet0/0/2   unassigned      YES unset  down                  down
GigabitEthernet0/0/3   192.168.1.246   YES DHCP   up                    up
GigabitEthernet0       192.168.1.246   YES DHCP   up                    up
Loopback100            10.10.10.10     YES TFTP   up                    up


 *** ZTP Day0 Python Script Execution Complete ***



Press RETURN to get started!
```

The day zero provisioning is complete, and the Cisco IOS prompt is accessible.

Boot Log for Cisco Catalyst 9000 Series Switches

The following sections display sample Zero-Touch Provisioning boot logs. These logs shows that Guest Shell is successfully enabled, the Python script is downloaded to the Guest Shell, and the Guest Shell executes the downloaded Python script and configures the device for Day Zero.

=

```
% Checking backup nvram
% No config present. Using default config

FIPS: Flash Key Check : Begin
FIPS: Flash Key Check : End, Not Found, FIPS Mode Not Enabled

! <This message indicates that the startup configuration
is absent on the device. This is the first indication that the Day Zero work flow is
going to start.>
```

Cisco IOS XE Everest 16.6.x to Cisco IOS XE Fuji 16.8.x

The following example shows the sample boot logs before the .py script is run:

```
Press RETURN to get started!

The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable


 *** Sample ZTP Day0 Python Script ***

...

 *** ZTP Day0 Python Script Execution Complete ***
```

The following example shows how to configure the device for Day Zero provisioning:

```
Initializing Hardware...

System Bootstrap, Version 17.2.1r[FC1], RELEASE SOFTWARE (P)
Compiled Thu 02/20/2020 23:47:51.50 by rel

Current ROMMON image : Primary
Last reset cause     : SoftwareReload
C9300-48UXM platform with 8388608 Kbytes of main memory

Preparing to autoboot. [Press Ctrl-C to interrupt]  0
boot: attempting to boot from [flash:cat9k_iosxe.16.06.05.SPA.bin]
boot: reading file cat9k_iosxe.16.06.05.SPA.bin
##################################################################################################
```

```
Both links down, not waiting for other switches
Switch number is 1

              Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

           cisco Systems, Inc.
           170 West Tasman Drive
           San Jose, California 95134-1706



Cisco IOS Software [Everest], Catalyst L3 Switch Software (CAT9K_IOSXE),
Version 16.6.5, RELEASE SOFTWARE (fc3)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2018 by Cisco Systems, Inc.
Compiled Mon 10-Dec-18 12:52 by mcpre



Cisco IOS-XE software, Copyright (c) 2005-2018 by cisco Systems, Inc.
All rights reserved.  Certain components of Cisco IOS-XE software are
licensed under the GNU General Public License ("GPL") Version 2.0.  The
software code licensed under GPL Version 2.0 is free software that comes
with ABSOLUTELY NO WARRANTY.  You can redistribute and/or modify such
GPL code under the terms of GPL Version 2.0.  For more details, see the
documentation or "License Notice" file accompanying the IOS-XE software,
or the applicable URL provided on the flyer accompanying the IOS-XE
software.


% Checking backup nvram
% No config present. Using default config


FIPS: Flash Key Check : Begin
FIPS: Flash Key Check : End, Not Found, FIPS Mode Not Enabled

This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply
third-party authority to import, export, distribute or use encryption.
Importers, exporters, distributors and users are responsible for
compliance with U.S. and local country laws. By using this product you
agree to comply with applicable laws and regulations. If you are unable
to comply with U.S. and local laws, return this product immediately.

A summary of U.S. laws governing Cisco cryptographic products may be found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html

If you require further assistance please contact us by sending email to
export@cisco.com.

cisco C9300-48UXM (X86) processor with 1392780K/6147K bytes of memory.
Processor board ID FCW2144L045
2048K bytes of non-volatile configuration memory.
```

```
8388608K bytes of physical memory.
1638400K bytes of Crash Files at crashinfo:.
11264000K bytes of Flash at flash:.
0K bytes of WebUI ODM Files at webui:.

Base Ethernet MAC Address          : ec:1d:8b:0a:68:00
Motherboard Assembly Number        : 73-17959-06
Motherboard Serial Number          : FOC21418FPQ
Model Revision Number              : B0
Motherboard Revision Number        : A0
Model Number                       : C9300-48UXM
System Serial Number               : FCW2144L045


%INIT: waited 0 seconds for NVRAM to be available

SETUP: new interface Vlan1 placed in "shutdown" state


Press RETURN to get started!


*Sep  4 20:35:07.330: %SMART_LIC-6-AGENT_READY: Smart Agent for Licensing is initialized
*Sep  4 20:35:07.493: %IOSXE_RP_NV-3-NV_ACCESS_FAIL: Initial read of NVRAM contents failed
*Sep  4 20:35:07.551: %IOSXE_RP_NV-3-BACKUP_NV_ACCESS_FAIL: Initial read of backup NVRAM
contents failed
*Sep  4 20:35:10.932: dev_pluggable_optics_selftest attribute table internally inconsistent
 @ 0x1D4

*Sep  4 20:35:13.406: %CRYPTO-4-AUDITWARN: Encryption audit check could not be performed
*Sep  4 20:35:13.480: %SPANTREE-5-EXTENDED_SYSID: Extended SysId enabled for type vlan
*Sep  4 20:35:13.715: %LINK-3-UPDOWN: Interface Lsmpi18/3, changed state to up
*Sep  4 20:35:13.724: %LINK-3-UPDOWN: Interface EOBC18/1, changed state to up
*Sep  4 20:35:13.724: %LINEPROTO-5-UPDOWN: Line protocol on Interface LI-Null0, changed
state to up
*Sep  4 20:35:13.724: %LINK-3-UPDOWN: Interface GigabitEthernet0/0, changed state to down
*Sep  4 20:35:13.725: %LINK-3-UPDOWN: Interface LIIN18/2, changed state to up
*Sep  4 20:35:13.749: WCM-PKI-SHIM: buffer allocation failed for SUDI support check
*Sep  4 20:35:13.749: PKI/SSL unable to send Sudi support to WCM
*Sep  4 20:35:14.622: %IOSXE_MGMTVRF-6-CREATE_SUCCESS_INFO: Management vrf Mgmt-vrf created
 with ID 1,
    ipv4 table-id 0x1, ipv6 table-id 0x1E000001
*Sep  4 20:34:42.022: %STACKMGR-6-STACK_LINK_CHANGE: Switch 1 R0/0: stack_mgr:  Stack port
 1 on Switch 1 is nocable
*Sep  4 20:34:42.022: %STACKMGR-6-STACK_LINK_CHANGE: Switch 1 R0/0: stack_mgr:  Stack port
 2 on Switch 1 is down
*Sep  4 20:34:42.022: %STACKMGR-6-STACK_LINK_CHANGE: Switch 1 R0/0: stack_mgr:  Stack port
 2 on Switch 1 is nocable
*Sep  4 20:34:42.022: %STACKMGR-6-SWITCH_ADDED: Switch 1 R0/0: stack_mgr:  Switch 1 has
been added to the stack.
*Sep  4 20:34:42.022: %STACKMGR-6-SWITCH_ADDED: Switch 1 R0/0: stack_mgr:  Switch 1 has
been added to the stack.
*Sep  4 20:34:42.022: %STACKMGR-6-SWITCH_ADDED: Switch 1 R0/0: stack_mgr:  Switch 1 has
been added to the stack.
*Sep  4 20:34:42.022: %STACKMGR-6-ACTIVE_ELECTED: Switch 1 R0/0: stack_mgr:  Switch 1 has
been elected ACTIVE.
*Sep  4 20:35:14.728: %LINEPROTO-5-UPDOWN: Line protocol on Interface Lsmpi18/3, changed
state to up
*Sep  4 20:35:14.728: %LINEPROTO-5-UPDOWN: Line protocol on Interface EOBC18/1, changed
state to up
*Sep  4 20:35:15.506: %HMANRP-6-HMAN_IOS_CHANNEL_INFO: HMAN-IOS channel event for switch
1: EMP_RELAY: Channel UP!
*Sep  4 20:35:15.510: %LINEPROTO-5-UPDOWN: Line protocol on Interface Vlan1, changed state
 to down
```

```
*Sep  4 20:35:34.501: %LINK-5-CHANGED: Interface Vlan1, changed state to administratively
down
*Sep  4 20:35:34.717: %SYS-5-RESTART: System restarted --
Cisco IOS Software [Everest], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 16.6.5,
RELEASE SOFTWARE (fc3)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2018 by Cisco Systems, Inc.
Compiled Mon 10-Dec-18 12:52 by mcpre
*Sep  4 20:35:34.796: %LINK-3-UPDOWN: Interface GigabitEthernet0/0, changed state to up
*Sep  4 20:35:35.266: %SYS-6-BOOTTIME: Time taken to reboot after reload =  283 seconds
*Sep  4 20:35:35.796: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet0/0,
changed state to up
*Sep  4 20:35:36.607: %LINK-3-UPDOWN: Interface GigabitEthernet1/1/1, changed state to down
*Sep  4 20:35:36.607: %LINK-3-UPDOWN: Interface GigabitEthernet1/1/2, changed state to down
*Sep  4 20:35:36.607: %LINK-3-UPDOWN: Interface GigabitEthernet1/1/3, changed state to down
*Sep  4 20:35:36.608: %LINK-3-UPDOWN: Interface GigabitEthernet1/1/4, changed state to down
*Sep  4 20:35:36.608: %LINK-3-UPDOWN: Interface TenGigabitEthernet1/1/1, changed state to
down
*Sep  4 20:35:36.608: %LINK-3-UPDOWN: Interface TenGigabitEthernet1/1/2, changed state to
down
*Sep  4 20:35:36.608: %LINK-3-UPDOWN: Interface TenGigabitEthernet1/1/3, changed state to
down
*Sep  4 20:35:36.608: %LINK-3-UPDOWN: Interface TenGigabitEthernet1/1/4, changed state to
down
*Sep  4 20:35:36.608: %LINK-3-UPDOWN: Interface TenGigabitEthernet1/1/5, changed state to
down
*Sep  4 20:35:36.609: %LINK-3-UPDOWN: Interface TenGigabitEthernet1/1/6, changed state to
down
*Sep  4 20:35:36.609: %LINK-3-UPDOWN: Interface TenGigabitEthernet1/1/7, changed state to
down
*Sep  4 20:35:36.609: %LINK-3-UPDOWN: Interface TenGigabitEthernet1/1/8, changed state to
down
*Sep  4 20:35:36.609: %LINK-3-UPDOWN: Interface FortyGigabitEthernet1/1/1, changed state
to down
*Sep  4 20:35:36.609: %LINK-3-UPDOWN: Interface FortyGigabitEthernet1/1/2, changed state
to down
*Sep  4 20:35:37.607: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet1/1/1,
 changed state to down
*Sep  4 20:35:37.608: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet1/1/2,
 changed state to down
*Sep  4 20:35:37.608: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet1/1/3,
 changed state to down
*Sep  4 20:35:37.609: %LINEPROTO-5-UPDOWN: Line protocol on Interface GigabitEthernet1/1/4,
 changed state to down
*Sep  4 20:35:37.609: %LINEPROTO-5-UPDOWN: Line protocol on Interface TenGigabitEthernet1/1/1,
 changed state to down
*Sep  4 20:35:37.609: %LINEPROTO-5-UPDOWN: Line protocol on Interface TenGigabitEthernet1/1/2,
 changed state to down
*Sep  4 20:35:37.609: %LINEPROTO-5-UPDOWN: Line protocol on Interface TenGigabitEthernet1/1/3,
 changed state to down
*Sep  4 20:35:37.609: %LINEPROTO-5-UPDOWN: Line protocol on Interface TenGigabitEthernet1/1/4,
 changed state to down
*Sep  4 20:35:37.609: %LINEPROTO-5-UPDOWN: Line protocol on Interface TenGigabitEthernet1/1/5,
 changed state to down
*Sep  4 20:35:37.609: %LINEPROTO-5-UPDOWN: Line protocol on Interface TenGigabitEthernet1/1/6,
 changed state to down
*Sep  4 20:35:43.511: AUTOINSTALL: Obtain tftp server address (opt 150) 159.14.27.2
*Sep  4 20:35:43.511: PNPA: Setting autoinstall complete to true for 159.14.27.2
*Sep  4 20:35:57.673: %PLATFORM_PM-6-FRULINK_INSERTED: 8x10G uplink module inserted in the
 switch 1 slot 1
*Sep  4 20:36:19.562: [IOX DEBUG] Guestshell start API is being invoked

*Sep  4 20:36:19.562: [IOX DEBUG] provided idb is mgmt interface
```

```
*Sep  4 20:36:19.562: [IOX DEBUG] Setting up guestshell to use mgmt-intf

*Sep  4 20:36:19.562: [IOX DEBUG] Setting up chasfs for iox related activity

*Sep  4 20:36:19.562: [IOX DEBUG] Setting up for iox pre-clean activity if needed

*Sep  4 20:36:19.562: [IOX DEBUG] Waiting for iox pre-clean setup to take affect

*Sep  4 20:36:19.562: [IOX DEBUG] Waited for 1 sec(s) for iox pre-clean setup to take affect

*Sep  4 20:36:19.562: [IOX DEBUG] Auto-configuring iox feature

*Sep  4 20:36:19.563: [IOX DEBUG] Waiting for CAF and ioxman to be up, in that order

*Sep  4 20:36:20.076: %UICFGEXP-6-SERVER_NOTIFIED_START: Switch 1 R0/0: psd:  Server iox
has been notified to start
*Sep  4 20:36:23.564: [IOX DEBUG] Waiting for another 5 secs

*Sep  4 20:36:28.564: [IOX DEBUG] Waiting for another 5 secs
The process for the command is not responding or is otherwise unavailable

*Sep  4 20:36:33.564: [IOX DEBUG] Waiting for another 5 secs
The process for the command is not responding or is otherwise unavailable

*Sep  4 20:36:34.564: [IOX DEBUG] Waited for 16 sec(s) for CAF and ioxman to come up

*Sep  4 20:36:34.564: [IOX DEBUG] Validating if CAF and ioxman are running

*Sep  4 20:36:34.564: [IOX DEBUG] CAF and ioxman are up and running

*Sep  4 20:36:34.564: [IOX DEBUG] Building the simple mgmt-intf enable command string

*Sep  4 20:36:34.564: [IOX DEBUG] Enable command is: request platform software iox-manager

    app-hosting guestshell enable

*Sep  4 20:36:34.564: [IOX DEBUG] Issuing guestshell enable command and waiting for it to
be up
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable

*Sep  4 20:36:38.578: [IOX DEBUG] Waiting for another 5 secs
The process for the command is not responding or is otherwise unavailable

*Sep  4 20:36:39.416: %LINK-3-UPDOWN: Interface TenGigabitEthernet1/0/48, changed state to
 up
*Sep  4 20:36:40.416: %LINEPROTO-5-UPDOWN: Line protocol on Interface
TenGigabitEthernet1/0/48,
    changed state to upThe process for the command is not responding or is otherwise
unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable

*Sep  4 20:36:43.586: [IOX DEBUG] Waiting for another 5 secs
Guestshell enabled successfully

*Sep  4 20:37:45.321: [IOX DEBUG] Checking for guestshell mount path

*Sep  4 20:37:45.321: [IOX DEBUG] Validating if guestshell is ready for use

*Sep  4 20:37:45.321: [IOX DEBUG] Guestshell enabled successfully
```

```
 *** Sample ZTP Day0 Python Script ***



 *** Executing show platform  ***


Switch  Ports    Model               Serial No.   MAC address     Hw Ver.       Sw Ver.
------  -----   ---------            -----------  --------------  -------       --------
 1       62     C9300-48UXM          FCW2144L045  ec1d.8b0a.6800  V01           16.6.5

Switch/Stack Mac Address : ec1d.8b0a.6800 - Local Mac Address
Mac persistency wait time: Indefinite
                                     Current
Switch#   Role        Priority      State
-----------------------------------------
*1       Active          1          Ready


 *** Executing show version ***


Cisco IOS XE Software, Version 16.06.05
Cisco IOS Software [Everest], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 16.6.5,
RELEASE SOFTWARE (fc3)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2018 by Cisco Systems, Inc.
Compiled Mon 10-Dec-18 12:52 by mcpre
Cisco IOS-XE software, Copyright (c) 2005-2018 by cisco Systems, Inc.
All rights reserved.  Certain components of Cisco IOS-XE software are
licensed under the GNU General Public License ("GPL") Version 2.0.  The
software code licensed under GPL Version 2.0 is free software that comes
with ABSOLUTELY NO WARRANTY.  You can redistribute and/or modify such
GPL code under the terms of GPL Version 2.0.  For more details, see the
documentation or "License Notice" file accompanying the IOS-XE software,
or the applicable URL provided on the flyer accompanying the IOS-XE
software.
ROM: IOS-XE ROMMON
BOOTLDR: System Bootstrap, Version 17.2.1r[FC1], RELEASE SOFTWARE (P)
Switch uptime is 2 minutes
Uptime for this control processor is 4 minutes
System returned to ROM by Reload Command
System image file is "flash:cat9k_iosxe.16.06.05.SPA.bin"
Last reload reason: Reload Command
This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply
third-party authority to import, export, distribute or use encryption.
Importers, exporters, distributors and users are responsible for
compliance with U.S. and local country laws. By using this product you
agree to comply with applicable laws and regulations. If you are unable
to comply with U.S. and local laws, return this product immediately.
A summary of U.S. laws governing Cisco cryptographic products may be found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html
If you require further assistance please contact us by sending email to
export@cisco.com.
Technology Package License Information:
------------------------------------------------------------------
Technology-package                  Technology-package
Current             Type            Next reboot
------------------------------------------------------------------
network-advantage   Permanent       network-advantage
```

```
cisco C9300-48UXM (X86) processor with 1392780K/6147K bytes of memory.
Processor board ID FCW2144L045
36 Ethernet interfaces
1 Virtual Ethernet interface
4 Gigabit Ethernet interfaces
20 Ten Gigabit Ethernet interfaces
2 Forty Gigabit Ethernet interfaces
2048K bytes of non-volatile configuration memory.
8388608K bytes of physical memory.
1638400K bytes of Crash Files at crashinfo:.
11264000K bytes of Flash at flash:.
0K bytes of WebUI ODM Files at webui:.
Base Ethernet MAC Address          : ec:1d:8b:0a:68:00
Motherboard Assembly Number        : 73-17959-06
Motherboard Serial Number          : FOC21418FPQ
Model Revision Number              : B0
Motherboard Revision Number        : A0
Model Number                       : C9300-48UXM
System Serial Number               : FCW2144L045
Switch Ports Model              SW Version        SW Image              Mode
------ ----- -----              ----------        ----------            ----
*    1 62    C9300-48UXM        16.6.5            CAT9K_IOSXE           BUNDLE
Configuration register is 0x102


 *** Configuring a Loopback Interface ***


Line 1 SUCCESS: interface loop 100
Line 2 SUCCESS: ip address 10.10.10.10 255.255.255.255
Line 3 SUCCESS: end


 *** Executing show ip interface brief  ***


Interface              IP-Address      OK? Method Status                Protocol
Vlan1                  unassigned      YES unset  administratively down down
GigabitEthernet0/0     10.127.128.3    YES DHCP   up                    up
Tw1/0/1                unassigned      YES unset  down                  down
Tw1/0/2                unassigned      YES unset  down                  down
Tw1/0/3                unassigned      YES unset  down                  down
Tw1/0/4                unassigned      YES unset  down                  down
Tw1/0/5                unassigned      YES unset  down                  down
Tw1/0/6                unassigned      YES unset  down                  down
Tw1/0/7                unassigned      YES unset  down                  down
Tw1/0/8                unassigned      YES unset  down                  down
Tw1/0/9                unassigned      YES unset  down                  down
Tw1/0/10               unassigned      YES unset  down                  down
Tw1/0/11               unassigned      YES unset  down                  down
Tw1/0/12               unassigned      YES unset  down                  down
Tw1/0/13               unassigned      YES unset  down                  down
Tw1/0/14               unassigned      YES unset  down                  down
Tw1/0/15               unassigned      YES unset  down                  down
Tw1/0/16               unassigned      YES unset  down                  down
Tw1/0/17               unassigned      YES unset  down                  down
Tw1/0/18               unassigned      YES unset  down                  down
Tw1/0/19               unassigned      YES unset  down                  down
Tw1/0/20               unassigned      YES unset  down                  down
Tw1/0/21               unassigned      YES unset  down                  down
Tw1/0/22               unassigned      YES unset  down                  down
Tw1/0/23               unassigned      YES unset  down                  down
Tw1/0/24               unassigned      YES unset  down                  down
Tw1/0/25               unassigned      YES unset  down                  down
```

```
Tw1/0/26               unassigned      YES unset  down                  down
Tw1/0/27               unassigned      YES unset  down                  down
Tw1/0/28               unassigned      YES unset  down                  down
Tw1/0/29               unassigned      YES unset  down                  down
Tw1/0/30               unassigned      YES unset  down                  down
Tw1/0/31               unassigned      YES unset  down                  down
Tw1/0/32               unassigned      YES unset  down                  down
Tw1/0/33               unassigned      YES unset  down                  down
Tw1/0/34               unassigned      YES unset  down                  down
Tw1/0/35               unassigned      YES unset  down                  down
Tw1/0/36               unassigned      YES unset  down                  down
Te1/0/37               unassigned      YES unset  down                  down
Te1/0/38               unassigned      YES unset  down                  down
Te1/0/39               unassigned      YES unset  down                  down
Te1/0/40               unassigned      YES unset  down                  down
Te1/0/41               unassigned      YES unset  down                  down
Te1/0/42               unassigned      YES unset  down                  down
Te1/0/43               unassigned      YES unset  down                  down
Te1/0/44               unassigned      YES unset  down                  down
Te1/0/45               unassigned      YES unset  down                  down
Te1/0/46               unassigned      YES unset  down                  down
Te1/0/47               unassigned      YES unset  down                  down
Te1/0/48               unassigned      YES unset  up                    up
GigabitEthernet1/1/1   unassigned      YES unset  down                  down
GigabitEthernet1/1/2   unassigned      YES unset  down                  down
GigabitEthernet1/1/3   unassigned      YES unset  down                  down
GigabitEthernet1/1/4   unassigned      YES unset  down                  down
Te1/1/1                unassigned      YES unset  down                  down
Te1/1/2                unassigned      YES unset  down                  down
Te1/1/3                unassigned      YES unset  down                  down
Te1/1/4                unassigned      YES unset  down                  down
Te1/1/5                unassigned      YES unset  down                  down
Te1/1/6                unassigned      YES unset  down                  down
Te1/1/7                unassigned      YES unset  down                  down
Te1/1/8                unassigned      YES unset  down                  down
Fo1/1/1                unassigned      YES unset  down                  down
Fo1/1/2                unassigned      YES unset  down                  down
Loopback100            10.10.10.10     YES TFTP   up                    up

 *** Configuring username, password, SSH  ***


Line 1 SUCCESS: username cisco privilege 15 password cisco
Line 2 SUCCESS: ip domain name domain
Line 3 SUCCESS: line vty 0 15
Line 4 SUCCESS: login local
Line 5 SUCCESS: transport input all
Line 6 SUCCESS: end


 *** ZTP Day0 Python Script Execution Complete ***
```

Cisco IOS XE Fuji 16.9.x to Cisco IOS XE Gibraltar 16.11.x

The following example shows the sample boot logs before the .py script is run:

```
--- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: The process for the
command is not
responding or is otherwise unavailable
```

```
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
guestshell installed successfully
Current state is: DEPLOYED
guestshell activated successfully
Current state is: ACTIVATED
guestshell started successfully
Current state is: RUNNING
Guestshell enabled successfully
```

The following example shows how to configure the device for Day Zero provisioning:

```
Both links down, not waiting for other switches
Switch number is 1

              Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

           Cisco Systems, Inc.
           170 West Tasman Drive
           San Jose, California 95134-1706



Cisco IOS Software [Fuji], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 16.9.4, RELEASE
 SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2019 by Cisco Systems, Inc.
Compiled Thu 22-Aug-19 18:14 by mcpre



PLEASE READ THE FOLLOWING TERMS CAREFULLY. INSTALLING THE LICENSE OR
LICENSE KEY PROVIDED FOR ANY CISCO SOFTWARE PRODUCT, PRODUCT FEATURE,
AND/OR SUBSEQUENTLY PROVIDED SOFTWARE FEATURES (COLLECTIVELY, THE
"SOFTWARE"), AND/OR USING SUCH SOFTWARE CONSTITUTES YOUR FULL
ACCEPTANCE OF THE FOLLOWING TERMS. YOU MUST NOT PROCEED FURTHER IF YOU
ARE NOT WILLING TO BE BOUND BY ALL THE TERMS SET FORTH HEREIN.

Your use of the Software is subject to the Cisco End User License Agreement
(EULA) and any relevant supplemental terms (SEULA) found at
http://www.cisco.com/c/en/us/about/legal/cloud-and-software/software-terms.html.

You hereby acknowledge and agree that certain Software and/or features are
licensed for a particular term, that the license to such Software and/or
features is valid only for the applicable term and that such Software and/or
features may be shut down or otherwise terminated by Cisco after expiration
of the applicable license term (e.g., 90-day trial period). Cisco reserves
the right to terminate any such Software feature electronically or by any
other means available. While Cisco may provide alerts, it is your sole
responsibility to monitor your usage of any such term Software feature to
ensure that your systems and networks are prepared for a shutdown of the
Software feature.
```

```
% Checking backup nvram
% No config present. Using default config

FIPS: Flash Key Check : Key Not Found, FIPS Mode Not Enabled
cisco C9300-48UXM (X86) processor with 1419044K/6147K bytes of memory.
Processor board ID FCW2144L045
2048K bytes of non-volatile configuration memory.
8388608K bytes of physical memory.
1638400K bytes of Crash Files at crashinfo:.
11264000K bytes of Flash at flash:.
0K bytes of WebUI ODM Files at webui:.

Base Ethernet MAC Address          : ec:1d:8b:0a:68:00
Motherboard Assembly Number        : 73-17959-06
Motherboard Serial Number          : FOC21418FPQ
Model Revision Number              : B0
Motherboard Revision Number        : A0
Model Number                       : C9300-48UXM
System Serial Number               : FCW2144L045


%INIT: waited 0 seconds for NVRAM to be available


         --- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: The process for the
command is not
    responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
```

```
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
The process for the command is not responding or is otherwise unavailable
guestshell installed successfully
Current state is: DEPLOYED
guestshell activated successfully
Current state is: ACTIVATED
guestshell started successfully
Current state is: RUNNING
Guestshell enabled successfully


HTTP server statistics:
Accepted connections total: 0

 *** Sample ZTP Day0 Python Script ***



 *** Executing show platform  ***


Switch  Ports    Model                Serial No.   MAC address     Hw Ver.       Sw Ver.
------  -----   ---------             -----------  --------------  -------       --------
 1       64     C9300-48UXM           FCW2144L045  ec1d.8b0a.6800  V01           16.9.4

Switch/Stack Mac Address : ec1d.8b0a.6800 - Local Mac Address
Mac persistency wait time: Indefinite
                                   Current
Switch#   Role        Priority      State
-------------------------------------------
*1       Active          1          Ready


 *** Executing show version ***


Cisco IOS XE Software, Version 16.09.04
Cisco IOS Software [Fuji], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 16.9.4, RELEASE
 SOFTWARE (fc2)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2019 by Cisco Systems, Inc.
Compiled Thu 22-Aug-19 18:14 by mcpre
Cisco IOS-XE software, Copyright (c) 2005-2019 by cisco Systems, Inc.
```

```
All rights reserved.  Certain components of Cisco IOS-XE software are
licensed under the GNU General Public License ("GPL") Version 2.0.  The
software code licensed under GPL Version 2.0 is free software that comes
with ABSOLUTELY NO WARRANTY.  You can redistribute and/or modify such
GPL code under the terms of GPL Version 2.0.  For more details, see the
documentation or "License Notice" file accompanying the IOS-XE software,
or the applicable URL provided on the flyer accompanying the IOS-XE
software.
ROM: IOS-XE ROMMON
BOOTLDR: System Bootstrap, Version 17.2.1r[FC1], RELEASE SOFTWARE (P)
Switch uptime is 4 minutes
Uptime for this control processor is 5 minutes
System returned to ROM by Reload Command
System image file is "flash:cat9k_iosxe.16.09.04.SPA.bin"
Last reload reason: Reload Command
This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply
third-party authority to import, export, distribute or use encryption.
Importers, exporters, distributors and users are responsible for
compliance with U.S. and local country laws. By using this product you
agree to comply with applicable laws and regulations. If you are unable
to comply with U.S. and local laws, return this product immediately.
A summary of U.S. laws governing Cisco cryptographic products may be found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html
If you require further assistance please contact us by sending email to
export@cisco.com.
Technology Package License Information:
------------------------------------------------------------------------------
Technology-package                                          Technology-package
Current                       Type                             Next reboot
------------------------------------------------------------------------------
network-advantage       Smart License                    network-advantage
None                    Subscription Smart License       None
Smart Licensing Status: UNREGISTERED/EVAL EXPIRED
cisco C9300-48UXM (X86) processor with 1419044K/6147K bytes of memory.
Processor board ID FCW2144L045
36 Ethernet interfaces
1 Virtual Ethernet interface
4 Gigabit Ethernet interfaces
20 Ten Gigabit Ethernet interfaces
2 TwentyFive Gigabit Ethernet interfaces
2 Forty Gigabit Ethernet interfaces
2048K bytes of non-volatile configuration memory.
8388608K bytes of physical memory.
1638400K bytes of Crash Files at crashinfo:.
11264000K bytes of Flash at flash:.
0K bytes of WebUI ODM Files at webui:.
Base Ethernet MAC Address          : ec:1d:8b:0a:68:00
Motherboard Assembly Number        : 73-17959-06
Motherboard Serial Number          : FOC21418FPQ
Model Revision Number              : B0
Motherboard Revision Number        : A0
Model Number                       : C9300-48UXM
System Serial Number               : FCW2144L045
Switch Ports Model              SW Version        SW Image              Mode
------ ----- -----              ----------        ----------            ----
*    1 64    C9300-48UXM        16.9.4            CAT9K_IOSXE           BUNDLE
Configuration register is 0x102


 *** Configuring a Loopback Interface ***
```

```
Line 1 SUCCESS: interface loop 100
Line 2 SUCCESS: ip address 10.10.10.10 255.255.255.255
Line 3 SUCCESS: end


 *** Executing show ip interface brief  ***


Any interface listed with OK? value "NO" does not have a valid configuration
Interface              IP-Address      OK? Method Status                Protocol
Vlan1                  unassigned      NO  unset  up                    up
GigabitEthernet0/0     10.127.128.5    YES DHCP   up                    up
Tw1/0/1                unassigned      YES unset  down                  down
Tw1/0/2                unassigned      YES unset  down                  down
Tw1/0/3                unassigned      YES unset  down                  down
Tw1/0/4                unassigned      YES unset  down                  down
Tw1/0/5                unassigned      YES unset  down                  down
Tw1/0/6                unassigned      YES unset  down                  down
Tw1/0/7                unassigned      YES unset  down                  down
Tw1/0/8                unassigned      YES unset  down                  down
Tw1/0/9                unassigned      YES unset  down                  down
Tw1/0/10               unassigned      YES unset  down                  down
Tw1/0/11               unassigned      YES unset  down                  down
Tw1/0/12               unassigned      YES unset  down                  down
Tw1/0/13               unassigned      YES unset  down                  down
Tw1/0/14               unassigned      YES unset  down                  down
Tw1/0/15               unassigned      YES unset  down                  down
Tw1/0/16               unassigned      YES unset  down                  down
Tw1/0/17               unassigned      YES unset  down                  down
Tw1/0/18               unassigned      YES unset  down                  down
Tw1/0/19               unassigned      YES unset  down                  down
Tw1/0/20               unassigned      YES unset  down                  down
Tw1/0/21               unassigned      YES unset  down                  down
Tw1/0/22               unassigned      YES unset  down                  down
Tw1/0/23               unassigned      YES unset  down                  down
Tw1/0/24               unassigned      YES unset  down                  down
Tw1/0/25               unassigned      YES unset  down                  down
Tw1/0/26               unassigned      YES unset  down                  down
Tw1/0/27               unassigned      YES unset  down                  down
Tw1/0/28               unassigned      YES unset  down                  down
Tw1/0/29               unassigned      YES unset  down                  down
Tw1/0/30               unassigned      YES unset  down                  down
Tw1/0/31               unassigned      YES unset  down                  down
Tw1/0/32               unassigned      YES unset  down                  down
Tw1/0/33               unassigned      YES unset  down                  down
Tw1/0/34               unassigned      YES unset  down                  down
Tw1/0/35               unassigned      YES unset  down                  down
Tw1/0/36               unassigned      YES unset  down                  down
Te1/0/37               unassigned      YES unset  down                  down
Te1/0/38               unassigned      YES unset  down                  down
Te1/0/39               unassigned      YES unset  down                  down
Te1/0/40               unassigned      YES unset  down                  down
Te1/0/41               unassigned      YES unset  down                  down
Te1/0/42               unassigned      YES unset  down                  down
Te1/0/43               unassigned      YES unset  down                  down
Te1/0/44               unassigned      YES unset  down                  down
Te1/0/45               unassigned      YES unset  down                  down
Te1/0/46               unassigned      YES unset  down                  down
Te1/0/47               unassigned      YES unset  down                  down
Te1/0/48               unassigned      YES unset  up                    up
GigabitEthernet1/1/1   unassigned      YES unset  down                  down
GigabitEthernet1/1/2   unassigned      YES unset  down                  down
GigabitEthernet1/1/3   unassigned      YES unset  down                  down
GigabitEthernet1/1/4   unassigned      YES unset  down                  down
```

```
Te1/1/1                unassigned      YES unset  down                  down
Te1/1/2                unassigned      YES unset  down                  down
Te1/1/3                unassigned      YES unset  down                  down
Te1/1/4                unassigned      YES unset  down                  down
Te1/1/5                unassigned      YES unset  down                  down
Te1/1/6                unassigned      YES unset  down                  down
Te1/1/7                unassigned      YES unset  down                  down
Te1/1/8                unassigned      YES unset  down                  down
Fo1/1/1                unassigned      YES unset  down                  down
Fo1/1/2                unassigned      YES unset  down                  down
TwentyFiveGigE1/1/1    unassigned      YES unset  down                  down
TwentyFiveGigE1/1/2    unassigned      YES unset  down                  down
Loopback100            10.10.10.10     YES TFTP   up                    up


 *** Configuring username, password, SSH  ***


Line 1 SUCCESS: username cisco privilege 15 password cisco
**CLI Line # 1:  WARNING: Command has been added to the configuration using a type 0 password.

    However, type 0 passwords will soon be deprecated. Migrate to a supported password type
Line 2 SUCCESS: ip domain name domain
Line 3 SUCCESS: line vty 0 15
Line 4 SUCCESS: login local
Line 5 SUCCESS: transport input all
Line 6 SUCCESS: end


 *** ZTP Day0 Python Script Execution Complete ***




Press RETURN to get started!
```

Cisco IOS XE Gibraltar 16.12.x to Cisco IOS XE Amsterdam 17.1.x

The following example shows the sample boot logs before the .py script is run:

```
--- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: day0guestshell installed
 successfully
Current state is: DEPLOYED
day0guestshell activated successfully
Current state is: ACTIVATED
day0guestshell started successfully
Current state is: RUNNING
Guestshell enabled successfully


 *** Sample ZTP Day0 Python Script ***

...

 *** ZTP Day0 Python Script Execution Complete ***
```

```
Guestshell destroyed successfully
```

The following example shows how to configure the device for Day Zero provisioning:

```
Both links down, not waiting for other switches
Switch number is 1

              Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

           Cisco Systems, Inc.
           170 West Tasman Drive
           San Jose, California 95134-1706



Cisco IOS Software [Gibraltar], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 16.12.3a,

RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2020 by Cisco Systems, Inc.
Compiled Tue 28-Apr-20 09:37 by mcpre


This software version supports only Smart Licensing as the software licensing mechanism.


PLEASE READ THE FOLLOWING TERMS CAREFULLY. INSTALLING THE LICENSE OR
LICENSE KEY PROVIDED FOR ANY CISCO SOFTWARE PRODUCT, PRODUCT FEATURE,
AND/OR SUBSEQUENTLY PROVIDED SOFTWARE FEATURES (COLLECTIVELY, THE
"SOFTWARE"), AND/OR USING SUCH SOFTWARE CONSTITUTES YOUR FULL
ACCEPTANCE OF THE FOLLOWING TERMS. YOU MUST NOT PROCEED FURTHER IF YOU
ARE NOT WILLING TO BE BOUND BY ALL THE TERMS SET FORTH HEREIN.

Your use of the Software is subject to the Cisco End User License Agreement
(EULA) and any relevant supplemental terms (SEULA) found at
http://www.cisco.com/c/en/us/about/legal/cloud-and-software/software-terms.html.

You hereby acknowledge and agree that certain Software and/or features are
licensed for a particular term, that the license to such Software and/or
features is valid only for the applicable term and that such Software and/or
features may be shut down or otherwise terminated by Cisco after expiration
of the applicable license term (e.g., 90-day trial period). Cisco reserves
the right to terminate any such Software feature electronically or by any
other means available. While Cisco may provide alerts, it is your sole
responsibility to monitor your usage of any such term Software feature to
ensure that your systems and networks are prepared for a shutdown of the
Software feature.


% Checking backup nvram
% No config present. Using default config

FIPS: Flash Key Check : Key Not Found, FIPS Mode Not Enabled

All TCP AO KDF Tests Pass
cisco C9300-48UXM (X86) processor with 1343703K/6147K bytes of memory.
```

```
Processor board ID FCW2144L045
2048K bytes of non-volatile configuration memory.
8388608K bytes of physical memory.
1638400K bytes of Crash Files at crashinfo:.
11264000K bytes of Flash at flash:.
0K bytes of WebUI ODM Files at webui:.

Base Ethernet MAC Address          : ec:1d:8b:0a:68:00
Motherboard Assembly Number        : 73-17959-06
Motherboard Serial Number          : FOC21418FPQ
Model Revision Number              : B0
Motherboard Revision Number        : A0
Model Number                       : C9300-48UXM
System Serial Number               : FCW2144L045


         --- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]: day0guestshell installed
 successfully
Current state is: DEPLOYED
day0guestshell activated successfully
Current state is: ACTIVATED
day0guestshell started successfully
Current state is: RUNNING
Guestshell enabled successfully


HTTP server statistics:
Accepted connections total: 0

 *** Sample ZTP Day0 Python Script ***



 *** Executing show platform  ***


Switch  Ports    Model              Serial No.   MAC address     Hw Ver.       Sw Ver.
------  -----   ---------           -----------  --------------  -------       --------
 1       65     C9300-48UXM         FCW2144L045  ec1d.8b0a.6800  V01           16.12.3a

Switch/Stack Mac Address : ec1d.8b0a.6800 - Local Mac Address
Mac persistency wait time: Indefinite
                                    Current
Switch#   Role        Priority      State
-----------------------------------------
*1       Active          1          Ready


 *** Executing show version ***


Cisco IOS XE Software, Version 16.12.03a
Cisco IOS Software [Gibraltar], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 16.12.3a,

RELEASE SOFTWARE (fc1)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2020 by Cisco Systems, Inc.
Compiled Tue 28-Apr-20 09:37 by mcpre
Cisco IOS-XE software, Copyright (c) 2005-2020 by cisco Systems, Inc.
All rights reserved.  Certain components of Cisco IOS-XE software are
```

```
licensed under the GNU General Public License ("GPL") Version 2.0.  The
software code licensed under GPL Version 2.0 is free software that comes
with ABSOLUTELY NO WARRANTY.  You can redistribute and/or modify such
GPL code under the terms of GPL Version 2.0.  For more details, see the
documentation or "License Notice" file accompanying the IOS-XE software,
or the applicable URL provided on the flyer accompanying the IOS-XE
software.
ROM: IOS-XE ROMMON
BOOTLDR: System Bootstrap, Version 17.2.1r[FC1], RELEASE SOFTWARE (P)
Switch uptime is 4 minutes
Uptime for this control processor is 9 minutes
System returned to ROM by Reload Command
System image file is "flash:cat9k_iosxe.16.12.03a.SPA.bin"
Last reload reason: Reload Command
This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply
third-party authority to import, export, distribute or use encryption.
Importers, exporters, distributors and users are responsible for
compliance with U.S. and local country laws. By using this product you
agree to comply with applicable laws and regulations. If you are unable
to comply with U.S. and local laws, return this product immediately.
A summary of U.S. laws governing Cisco cryptographic products may be found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html
If you require further assistance please contact us by sending email to
export@cisco.com.
Technology Package License Information:
------------------------------------------------------------------------------
Technology-package                                          Technology-package
Current                       Type                             Next reboot
------------------------------------------------------------------------------
network-advantage       Smart License                    network-advantage
None                    Subscription Smart License       None
AIR License Level: AIR DNA Advantage
Next reload AIR license Level: AIR DNA Advantage
Smart Licensing Status: UNREGISTERED/EVAL EXPIRED
cisco C9300-48UXM (X86) processor with 1343703K/6147K bytes of memory.
Processor board ID FCW2144L045
1 Virtual Ethernet interface
4 Gigabit Ethernet interfaces
36 2.5 Gigabit Ethernet interfaces
20 Ten Gigabit Ethernet interfaces
2 TwentyFive Gigabit Ethernet interfaces
2 Forty Gigabit Ethernet interfaces
2048K bytes of non-volatile configuration memory.
8388608K bytes of physical memory.
1638400K bytes of Crash Files at crashinfo:.
11264000K bytes of Flash at flash:.
0K bytes of WebUI ODM Files at webui:.
Base Ethernet MAC Address          : ec:1d:8b:0a:68:00
Motherboard Assembly Number        : 73-17959-06
Motherboard Serial Number          : FOC21418FPQ
Model Revision Number              : B0
Motherboard Revision Number        : A0
Model Number                       : C9300-48UXM
System Serial Number               : FCW2144L045
Switch Ports Model              SW Version        SW Image              Mode
------ ----- -----              ----------        ----------            ----
*    1 65    C9300-48UXM        16.12.3a          CAT9K_IOSXE           BUNDLE
Configuration register is 0x102


 *** Configuring a Loopback Interface ***
```

```
Line 1 SUCCESS: interface loop 100
Line 2 SUCCESS: ip address 10.10.10.10 255.255.255.255
Line 3 SUCCESS: end


 *** Executing show ip interface brief  ***


Interface              IP-Address      OK? Method Status                Protocol
Vlan1                  unassigned      YES unset  up                    up
GigabitEthernet0/0     10.127.128.10   YES DHCP   up                    up
Tw1/0/1                unassigned      YES unset  down                  down
Tw1/0/2                unassigned      YES unset  down                  down
Tw1/0/3                unassigned      YES unset  down                  down
Tw1/0/4                unassigned      YES unset  down                  down
Tw1/0/5                unassigned      YES unset  down                  down
Tw1/0/6                unassigned      YES unset  down                  down
Tw1/0/7                unassigned      YES unset  down                  down
Tw1/0/8                unassigned      YES unset  down                  down
Tw1/0/9                unassigned      YES unset  down                  down
Tw1/0/10               unassigned      YES unset  down                  down
Tw1/0/11               unassigned      YES unset  down                  down
Tw1/0/12               unassigned      YES unset  down                  down
Tw1/0/13               unassigned      YES unset  down                  down
Tw1/0/14               unassigned      YES unset  down                  down
Tw1/0/15               unassigned      YES unset  down                  down
Tw1/0/16               unassigned      YES unset  down                  down
Tw1/0/17               unassigned      YES unset  down                  down
Tw1/0/18               unassigned      YES unset  down                  down
Tw1/0/19               unassigned      YES unset  down                  down
Tw1/0/20               unassigned      YES unset  down                  down
Tw1/0/21               unassigned      YES unset  down                  down
Tw1/0/22               unassigned      YES unset  down                  down
Tw1/0/23               unassigned      YES unset  down                  down
Tw1/0/24               unassigned      YES unset  down                  down
Tw1/0/25               unassigned      YES unset  down                  down
Tw1/0/26               unassigned      YES unset  down                  down
Tw1/0/27               unassigned      YES unset  down                  down
Tw1/0/28               unassigned      YES unset  down                  down
Tw1/0/29               unassigned      YES unset  down                  down
Tw1/0/30               unassigned      YES unset  down                  down
Tw1/0/31               unassigned      YES unset  down                  down
Tw1/0/32               unassigned      YES unset  down                  down
Tw1/0/33               unassigned      YES unset  down                  down
Tw1/0/34               unassigned      YES unset  down                  down
Tw1/0/35               unassigned      YES unset  down                  down
Tw1/0/36               unassigned      YES unset  down                  down
Te1/0/37               unassigned      YES unset  down                  down
Te1/0/38               unassigned      YES unset  down                  down
Te1/0/39               unassigned      YES unset  down                  down
Te1/0/40               unassigned      YES unset  down                  down
Te1/0/41               unassigned      YES unset  down                  down
Te1/0/42               unassigned      YES unset  down                  down
Te1/0/43               unassigned      YES unset  down                  down
Te1/0/44               unassigned      YES unset  down                  down
Te1/0/45               unassigned      YES unset  down                  down
Te1/0/46               unassigned      YES unset  down                  down
Te1/0/47               unassigned      YES unset  down                  down
Te1/0/48               unassigned      YES unset  up                    up
GigabitEthernet1/1/1   unassigned      YES unset  down                  down
GigabitEthernet1/1/2   unassigned      YES unset  down                  down
GigabitEthernet1/1/3   unassigned      YES unset  down                  down
GigabitEthernet1/1/4   unassigned      YES unset  down                  down
```

```
Te1/1/1                unassigned      YES unset  down                  down
Te1/1/2                unassigned      YES unset  down                  down
Te1/1/3                unassigned      YES unset  down                  down
Te1/1/4                unassigned      YES unset  down                  down
Te1/1/5                unassigned      YES unset  down                  down
Te1/1/6                unassigned      YES unset  down                  down
Te1/1/7                unassigned      YES unset  down                  down
Te1/1/8                unassigned      YES unset  down                  down
Fo1/1/1                unassigned      YES unset  down                  down
Fo1/1/2                unassigned      YES unset  down                  down
TwentyFiveGigE1/1/1    unassigned      YES unset  down                  down
TwentyFiveGigE1/1/2    unassigned      YES unset  down                  down
Ap1/0/1                unassigned      YES unset  up                    up
Loopback100            10.10.10.10     YES TFTP   up                    up


 *** Configuring username, password, SSH  ***


Line 1 SUCCESS: username cisco privilege 15 password cisco
**CLI Line # 1:  WARNING: Command has been added to the configuration using a type 0 password.

However, type 0 passwords will soon be deprecated. Migrate to a supported password type
Line 2 SUCCESS: ip domain name domain
Line 3 SUCCESS: line vty 0 15
Line 4 SUCCESS: login local
Line 5 SUCCESS: transport input all
Line 6 SUCCESS: end


 *** ZTP Day0 Python Script Execution Complete ***



Guestshell destroyed successfully



Press RETURN to get started!
```

Cisco IOS XE Amsterdam 17.2.x and Later Releases

This following example shows the sample boot logs before the .py script is run:

```
--- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]:
Acquired IPv4 address 10.127.128.8 on Interface GigabitEthernet0/0
Received following DHCPv4 options:
        bootfile        : test.py
        tftp-server-ip  : 159.14.27.2

OK to enter CLI now...

pnp-discovery can be monitored without entering enable mode

Entering enable mode will stop pnp-discovery

Attempting bootfile tftp://159.14.27.2/test.py
day0guestshell activated successfully
Current state is: ACTIVATED
day0guestshell started successfully
```

```
Current state is: RUNNING
Guestshell enabled successfully


 *** Sample ZTP Day0 Python Script ***

...

 *** ZTP Day0 Python Script Execution Complete ***

Guestshell destroyed successfully
```

The following example shows how to configure the device for Day Zero provisioning:

```
Both links down, not waiting for other switches
Switch number is 1

              Restricted Rights Legend

Use, duplication, or disclosure by the Government is
subject to restrictions as set forth in subparagraph
(c) of the Commercial Computer Software - Restricted
Rights clause at FAR sec. 52.227-19 and subparagraph
(c) (1) (ii) of the Rights in Technical Data and Computer
Software clause at DFARS sec. 252.227-7013.

           Cisco Systems, Inc.
           170 West Tasman Drive
           San Jose, California 95134-1706



Cisco IOS Software [Amsterdam], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 17.2.1,
 RELEASE SOFTWARE (fc4)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2020 by Cisco Systems, Inc.
Compiled Thu 26-Mar-20 03:29 by mcpre


This software version supports only Smart Licensing as the software licensing mechanism.


PLEASE READ THE FOLLOWING TERMS CAREFULLY. INSTALLING THE LICENSE OR
LICENSE KEY PROVIDED FOR ANY CISCO SOFTWARE PRODUCT, PRODUCT FEATURE,
AND/OR SUBSEQUENTLY PROVIDED SOFTWARE FEATURES (COLLECTIVELY, THE
"SOFTWARE"), AND/OR USING SUCH SOFTWARE CONSTITUTES YOUR FULL
ACCEPTANCE OF THE FOLLOWING TERMS. YOU MUST NOT PROCEED FURTHER IF YOU
ARE NOT WILLING TO BE BOUND BY ALL THE TERMS SET FORTH HEREIN.

Your use of the Software is subject to the Cisco End User License Agreement
(EULA) and any relevant supplemental terms (SEULA) found at
http://www.cisco.com/c/en/us/about/legal/cloud-and-software/software-terms.html.

You hereby acknowledge and agree that certain Software and/or features are
licensed for a particular term, that the license to such Software and/or
features is valid only for the applicable term and that such Software and/or
features may be shut down or otherwise terminated by Cisco after expiration
of the applicable license term (e.g., 90-day trial period). Cisco reserves
the right to terminate any such Software feature electronically or by any
other means available. While Cisco may provide alerts, it is your sole
responsibility to monitor your usage of any such term Software feature to
ensure that your systems and networks are prepared for a shutdown of the
```

```
Software feature.


% Checking backup nvram
% No config present. Using default config

FIPS: Flash Key Check : Key Not Found, FIPS Mode Not Enabled

All TCP AO KDF Tests Pass
cisco C9300-48UXM (X86) processor with 1338934K/6147K bytes of memory.
Processor board ID FCW2144L045
2048K bytes of non-volatile configuration memory.
8388608K bytes of physical memory.
1638400K bytes of Crash Files at crashinfo:.
11264000K bytes of Flash at flash:.

Base Ethernet MAC Address          : ec:1d:8b:0a:68:00
Motherboard Assembly Number        : 73-17959-06
Motherboard Serial Number          : FOC21418FPQ
Model Revision Number              : B0
Motherboard Revision Number        : A0
Model Number                       : C9300-48UXM
System Serial Number               : FCW2144L045
CLEI Code Number                   :


No startup-config, starting autoinstall/pnp/ztp...

Autoinstall will terminate if any input is detected on console

Autoinstall trying DHCPv4 on GigabitEthernet0/0

Autoinstall trying DHCPv6 on GigabitEthernet0/0


         --- System Configuration Dialog ---

Would you like to enter the initial configuration dialog? [yes/no]:
Acquired IPv4 address 10.127.128.8 on Interface GigabitEthernet0/0
Received following DHCPv4 options:
        bootfile        : test.py
        tftp-server-ip  : 159.14.27.2

OK to enter CLI now...

pnp-discovery can be monitored without entering enable mode

Entering enable mode will stop pnp-discovery

Attempting bootfile tftp://159.14.27.2/test.py
day0guestshell activated successfully
Current state is: ACTIVATED
day0guestshell started successfully
Current state is: RUNNING
Guestshell enabled successfully


 *** Sample ZTP Day0 Python Script ***



 *** Executing show platform  ***
```

```
Switch  Ports    Model               Serial No.   MAC address     Hw Ver.       Sw Ver.
------  -----   ---------            -----------  --------------  -------       --------
 1       65     C9300-48UXM          FCW2144L045  ec1d.8b0a.6800  V01           17.02.01

Switch/Stack Mac Address : ec1d.8b0a.6800 - Local Mac Address
Mac persistency wait time: Indefinite
                                    Current
Switch#   Role        Priority      State
-----------------------------------------
*1       Active          1          Ready


 *** Executing show version ***


Cisco IOS XE Software, Version 17.02.01
Cisco IOS Software [Amsterdam], Catalyst L3 Switch Software (CAT9K_IOSXE), Version 17.2.1,
 RELEASE SOFTWARE (fc4)
Technical Support: http://www.cisco.com/techsupport
Copyright (c) 1986-2020 by Cisco Systems, Inc.
Compiled Thu 26-Mar-20 03:29 by mcpre
Cisco IOS-XE software, Copyright (c) 2005-2020 by cisco Systems, Inc.
All rights reserved.  Certain components of Cisco IOS-XE software are
licensed under the GNU General Public License ("GPL") Version 2.0.  The
software code licensed under GPL Version 2.0 is free software that comes
with ABSOLUTELY NO WARRANTY.  You can redistribute and/or modify such
GPL code under the terms of GPL Version 2.0.  For more details, see the
documentation or "License Notice" file accompanying the IOS-XE software,
or the applicable URL provided on the flyer accompanying the IOS-XE
software.
ROM: IOS-XE ROMMON
BOOTLDR: System Bootstrap, Version 17.2.1r[FC1], RELEASE SOFTWARE (P)
Switch uptime is 2 minutes
Uptime for this control processor is 8 minutes
System returned to ROM by Reload Command
System image file is "flash:cat9k_iosxe.17.02.01.SPA.bin"
Last reload reason: Reload Command
This product contains cryptographic features and is subject to United
States and local country laws governing import, export, transfer and
use. Delivery of Cisco cryptographic products does not imply
third-party authority to import, export, distribute or use encryption.
Importers, exporters, distributors and users are responsible for
compliance with U.S. and local country laws. By using this product you
agree to comply with applicable laws and regulations. If you are unable
to comply with U.S. and local laws, return this product immediately.
A summary of U.S. laws governing Cisco cryptographic products may be found at:
http://www.cisco.com/wwl/export/crypto/tool/stqrg.html
If you require further assistance please contact us by sending email to
export@cisco.com.
Technology Package License Information:
------------------------------------------------------------------------------
Technology-package                                      Technology-package
Current                       Type                         Next reboot
------------------------------------------------------------------------------
network-advantage       Smart License                    network-advantage
None                    Subscription Smart License       None
AIR License Level: AIR DNA Advantage
Next reload AIR license Level: AIR DNA Advantage
Smart Licensing Status: UNREGISTERED/EVAL EXPIRED
cisco C9300-48UXM (X86) processor with 1338934K/6147K bytes of memory.
Processor board ID FCW2144L045
1 Virtual Ethernet interface
4 Gigabit Ethernet interfaces
```

```
36 2.5 Gigabit Ethernet interfaces
20 Ten Gigabit Ethernet interfaces
2 TwentyFive Gigabit Ethernet interfaces
2 Forty Gigabit Ethernet interfaces
2048K bytes of non-volatile configuration memory.
8388608K bytes of physical memory.
1638400K bytes of Crash Files at crashinfo:.
11264000K bytes of Flash at flash:.
Base Ethernet MAC Address          : ec:1d:8b:0a:68:00
Motherboard Assembly Number        : 73-17959-06
Motherboard Serial Number          : FOC21418FPQ
Model Revision Number              : B0
Motherboard Revision Number        : A0
Model Number                       : C9300-48UXM
System Serial Number               : FCW2144L045
CLEI Code Number                   :
Switch Ports Model              SW Version        SW Image              Mode
------ ----- -----              ----------        ----------            ----
*    1 65    C9300-48UXM        17.02.01          CAT9K_IOSXE           BUNDLE
Configuration register is 0x102


 *** Configuring a Loopback Interface ***


Line 1 SUCCESS: interface loop 100
Line 2 SUCCESS: ip address 10.10.10.10 255.255.255.255
Line 3 SUCCESS: end


 *** Executing show ip interface brief  ***


Interface              IP-Address      OK? Method Status                Protocol
Vlan1                  unassigned      YES unset  up                    up
GigabitEthernet0/0     10.127.128.8    YES DHCP   up                    up
Tw1/0/1                unassigned      YES unset  down                  down
Tw1/0/2                unassigned      YES unset  down                  down
Tw1/0/3                unassigned      YES unset  down                  down
Tw1/0/4                unassigned      YES unset  down                  down
Tw1/0/5                unassigned      YES unset  down                  down
Tw1/0/6                unassigned      YES unset  down                  down
Tw1/0/7                unassigned      YES unset  down                  down
Tw1/0/8                unassigned      YES unset  down                  down
Tw1/0/9                unassigned      YES unset  down                  down
Tw1/0/10               unassigned      YES unset  down                  down
Tw1/0/11               unassigned      YES unset  down                  down
Tw1/0/12               unassigned      YES unset  down                  down
Tw1/0/13               unassigned      YES unset  down                  down
Tw1/0/14               unassigned      YES unset  down                  down
Tw1/0/15               unassigned      YES unset  down                  down
Tw1/0/16               unassigned      YES unset  down                  down
Tw1/0/17               unassigned      YES unset  down                  down
Tw1/0/18               unassigned      YES unset  down                  down
Tw1/0/19               unassigned      YES unset  down                  down
Tw1/0/20               unassigned      YES unset  down                  down
Tw1/0/21               unassigned      YES unset  down                  down
Tw1/0/22               unassigned      YES unset  down                  down
Tw1/0/23               unassigned      YES unset  down                  down
Tw1/0/24               unassigned      YES unset  down                  down
Tw1/0/25               unassigned      YES unset  down                  down
Tw1/0/26               unassigned      YES unset  down                  down
Tw1/0/27               unassigned      YES unset  down                  down
Tw1/0/28               unassigned      YES unset  down                  down
```

```
Tw1/0/29               unassigned      YES unset  down                  down
Tw1/0/30               unassigned      YES unset  down                  down
Tw1/0/31               unassigned      YES unset  down                  down
Tw1/0/32               unassigned      YES unset  down                  down
Tw1/0/33               unassigned      YES unset  down                  down
Tw1/0/34               unassigned      YES unset  down                  down
Tw1/0/35               unassigned      YES unset  down                  down
Tw1/0/36               unassigned      YES unset  down                  down
Te1/0/37               unassigned      YES unset  down                  down
Te1/0/38               unassigned      YES unset  down                  down
Te1/0/39               unassigned      YES unset  down                  down
Te1/0/40               unassigned      YES unset  down                  down
Te1/0/41               unassigned      YES unset  down                  down
Te1/0/42               unassigned      YES unset  down                  down
Te1/0/43               unassigned      YES unset  down                  down
Te1/0/44               unassigned      YES unset  down                  down
Te1/0/45               unassigned      YES unset  down                  down
Te1/0/46               unassigned      YES unset  down                  down
Te1/0/47               unassigned      YES unset  down                  down
Te1/0/48               unassigned      YES unset  up                    up
GigabitEthernet1/1/1   unassigned      YES unset  down                  down
GigabitEthernet1/1/2   unassigned      YES unset  down                  down
GigabitEthernet1/1/3   unassigned      YES unset  down                  down
GigabitEthernet1/1/4   unassigned      YES unset  down                  down
Te1/1/1                unassigned      YES unset  down                  down
Te1/1/2                unassigned      YES unset  down                  down
Te1/1/3                unassigned      YES unset  down                  down
Te1/1/4                unassigned      YES unset  down                  down
Te1/1/5                unassigned      YES unset  down                  down
Te1/1/6                unassigned      YES unset  down                  down
Te1/1/7                unassigned      YES unset  down                  down
Te1/1/8                unassigned      YES unset  down                  down
Fo1/1/1                unassigned      YES unset  down                  down
Fo1/1/2                unassigned      YES unset  down                  down
TwentyFiveGigE1/1/1    unassigned      YES unset  down                  down
TwentyFiveGigE1/1/2    unassigned      YES unset  down                  down
Ap1/0/1                unassigned      YES unset  up                    up
Loopback100            10.10.10.10     YES TFTP   up                    up


 *** Configuring username, password, SSH  ***


Line 1 SUCCESS: username cisco privilege 15 password cisco
**CLI Line # 1:  WARNING: Command has been added to the configuration using a type 0 password.

However, type 0 passwords will soon be deprecated. Migrate to a supported password type
Line 2 SUCCESS: ip domain name domain
Line 3 SUCCESS: line vty 0 15
Line 4 SUCCESS: login local
Line 5 SUCCESS: transport input all
Line 6 SUCCESS: end


 *** ZTP Day0 Python Script Execution Complete ***



Guestshell destroyed successfully
Script execution success!



Press RETURN to get started!
```

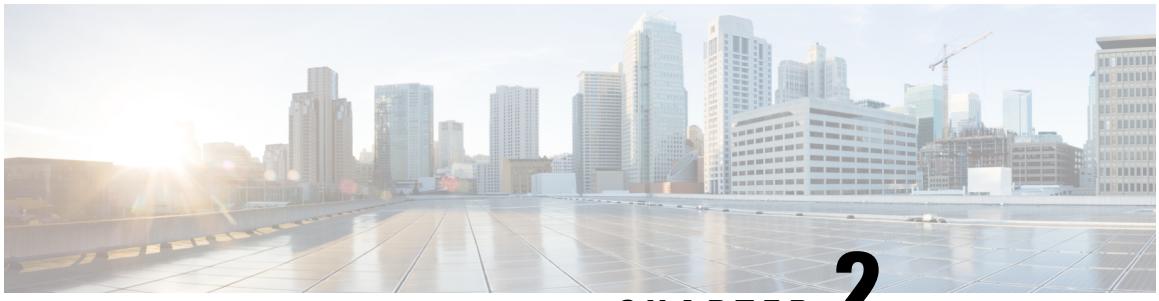
Additional References for Zero-Touch Provisioning

Standards and RFCs

Standard/RFC	Title
RFC 5652	*Cryptographic Message Syntax (CMS)*
RFC 8040	*RESTCONF Protocol*
RFC 8366	*A Voucher Artifact for Bootstrapping Protocols*
RFC 8572	*Secure Zero Touch Provisioning (SZTP)*

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.	https://www.cisco.com/c/en/us/support/index.html



CHAPTER 2

iPXE

iPXE is an enhanced version of the Pre-boot eXecution Environment (PXE), which is an open standard for network booting. This module describes the iPXE feature and how to configure it.

Feature Information for iPXE

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 3: Feature Information for iPXE

Feature Name	Release	Feature Information
iPXE	Cisco IOS XE 17.18.1	Network Bootloaders support booting from an IPv4/IPv6 device-based or network-based source. A network boot source must be detected automatically by using an iPXE-like solution. This feature was implemented on the following platforms: • Catalyst 9350 Series Switches • Catalyst 9610 Series Switches

Feature Name	Release	Feature Information
IPXE IPv6 Support	Cisco IOS XE 17.8.1	IPXE supports the IPv6 protocol. This feature was implemented on the following platforms: • Catalyst 9350 Series Switches

Information About iPXE

About iPXE

iPXE is an enhanced version of the Pre-boot eXecution Environment (PXE), which is an open standard for network booting.

iPXE netboot provides:

- IPv4 and IPv6 protocols
- FTP/HTTP/TFTP boot image download
- Embedded scripts into the image
- Stateless and stateful address auto-configuration (SLAAC) using Dynamic Host Configuration Protocol Version 4 (DHCPv4) and/or DHCPv6, boot URI, and parameters for DHCPv6 options depending on the IPv6 router advertisement.

Netboot Requirements

The following are the primary requirements for netbooting:

- DHCP server with proper configuration.
- Boot image available on the FTP/HTTP/TFTP server.
- Device configured to boot from a network-based source.

iPXE Overview

Network bootloaders support booting from a network-based source. The bootloaders boot an image located on an HTTP, FTP, or TFTP server. A network boot source is detected automatically by using an iPXE-like solution.

iPXE enables network boot for a device that is offline. The following are the three types of boot modes:

- iPXE Timeout—Boots through iPXE network boot. Configures a timeout in seconds for iPXE network boot by using the IPXE_TIMEOUT rommon variable. Use the **boot ipxe timeout** command to configure iPXE timeout. When the timeout expires, device boot is activated.
- iPXE Forever—Boots through iPXE network boot. The device sends DHCP requests forever, when the **boot ipxe forever** command is configured. This is an iPXE-only boot (which means that the bootloader

will not fall back to a device boot or a command prompt, because it will send DHCP requests forever until it receives a valid DHCP response.)

- Device—Boots using the local device BOOT line configured on it. When device boot is configured, the configured IPXE_TIMEOUT rommon variable is ignored. You can activate device boot as specified below:
  - If BOOTMODE=ipxe-forever, device boot is not activated without user intervention (this is possible only if ENABLE_BREAK=yes).
  - If BOOTMODE=ipxe-timeout, device boot is activated when the specified IPXE_TIMEOUT variable (in seconds) has elapsed.
  - If BOOTMODE=device, device boot is activated. This is the default active mode.
  - Device boot can also be activated through the CLI.

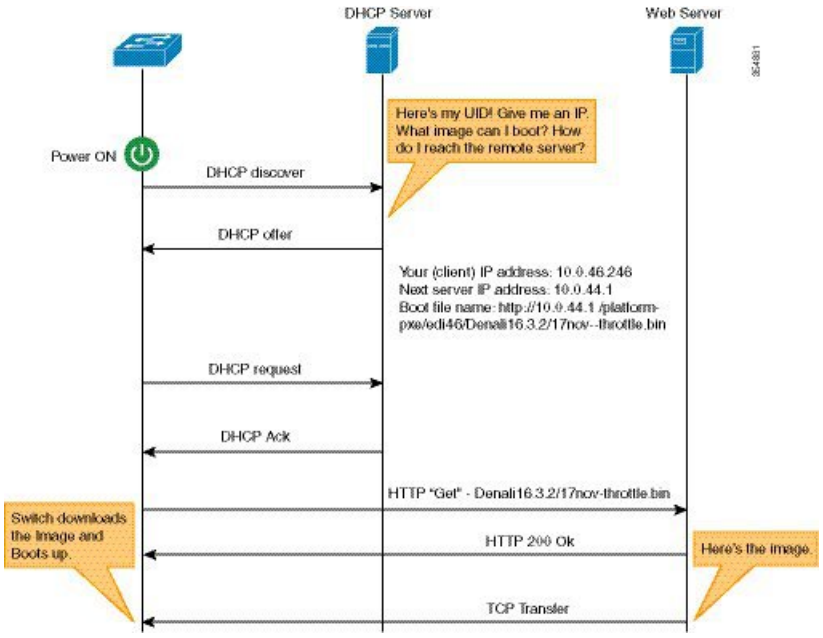
Note Device boot is the default boot mode.

Note Manual boot is another term used in this document. Manual boot is a flag that determines whether to do a rommon reload or not. When the device is in rommon mode, you have to manually issue the **boot** command.

If manual boot is set to YES, the rommon or device prompt is activated. If manual boot is set to NO, the autoboot variable is executed; this means that the value set in the BOOT variable is followed.

The following section describes how an iPXE bootloader works:

Figure 1: iPXE Bootloader Workflow



1. Bootloader sends a DHCP discover message, and when the server replies, the Bootloader sends a DHCP request.
2. The DHCP response includes the IP address and boot file name. The boot file name indicates that the boot image is to be retrieved from a TFTP server (tftp://server/filename), FTP server (ftp://userid:password@server/filename), or an HTTP server (http://server/filename).
3. Bootloader downloads and boots the image from the network source.
4. If no DHCP response is received, the bootloader keeps sending DHCP requests forever or for a specified period of time, based on the boot mode configured. When a timeout occurs, the bootloader reverts to a device-based boot. The device sends DHCP requests forever only if the configured boot mode is **ipxe-forever**. If the **ipxe-timeout** boot mode command is configured, DHCP requests are sent for the specified amount of time, and when the timeout expires, device boot mode is activated.

Note Because the current iPXE implementation works only via the management port (GigabitEthernet0/0), DHCP requests sent through the front panel ports are not supported.

When using a static network configuration to network boot, ROMMON uses the following environment variables (and all of them are required):

- BOOT—URLs separated by semicolon (;) to boot from.
- IP_ADDRESS—Statically assigned IP address of a device.
- DEFAULT_GATEWAY—Default gateway of the device.
- IP_SUBNET_MASK—IPv4 or IPv6 prefix information.

  IPv4—Subnet mask of the device in the format WWW.XXX.YYY.ZZZ eg. 255.255.255.0.

  IPv6—Subnet prefix length of the device in the format NNN eg. 64 or 112.

When manual boot is disabled, the bootloader determines whether to execute a device boot or a network boot based on the configured value of the rommon iPXE variable. Irrespective of whether manual boot is enabled or disabled, the bootloader uses the BOOTMODE variable to determine whether to do a device boot or a network boot. Manual boot means that the user has configured the **boot manual switch** command. When manual boot is disabled, and when the device reloads, the boot process starts automatically.

When iPXE is disabled, the contents of the existing BOOT variable are used to determine how to boot the device. The BOOT variable may contain a network-based uniform resource identifier (URI) (for example, http://, ftp://, tftp://), and a network boot is initiated; however DHCP is not used to get the network image path. The static network configuration is taken from the IP_ADDRESS, DEFAULT_GATEWAY, and IP_SUBNET_MASK variables. The BOOT variable may also contain a device filesystem-based path, in which case, a device filesystem-based boot is initiated.

The DHCP server used for booting can identify a device through the Product ID (PID) (available in DHCP Option 60), chassis serial number (available in DHCP option 61), or the MAC address of the device. The **show inventory** and **show switch** commands also display these values on the device.

The following is sample output from the **show inventory** command:

```
Device# show inventory

NAME:“c38xx Stack”, DESCR:“c38xx Stack”
```

```
PID:WS-3850-12X-48U-L, VID:V01 ,  SN: F0C1911V01A

NAME:“Switch 1”, DESCR:“WS-C3850-12X48U-L”
PID:WS-C3850-12X48U-L, VID:V01 , SN:F0C1911V01A

NAME:”Switch1 -Power Supply B”, DESCR:“Switch1 -Power Supply B”
PID:PWR-C1-1100WAC, VID:V01, SN:LIT1847146Q
```

The following is sample output from the **show switch** command:

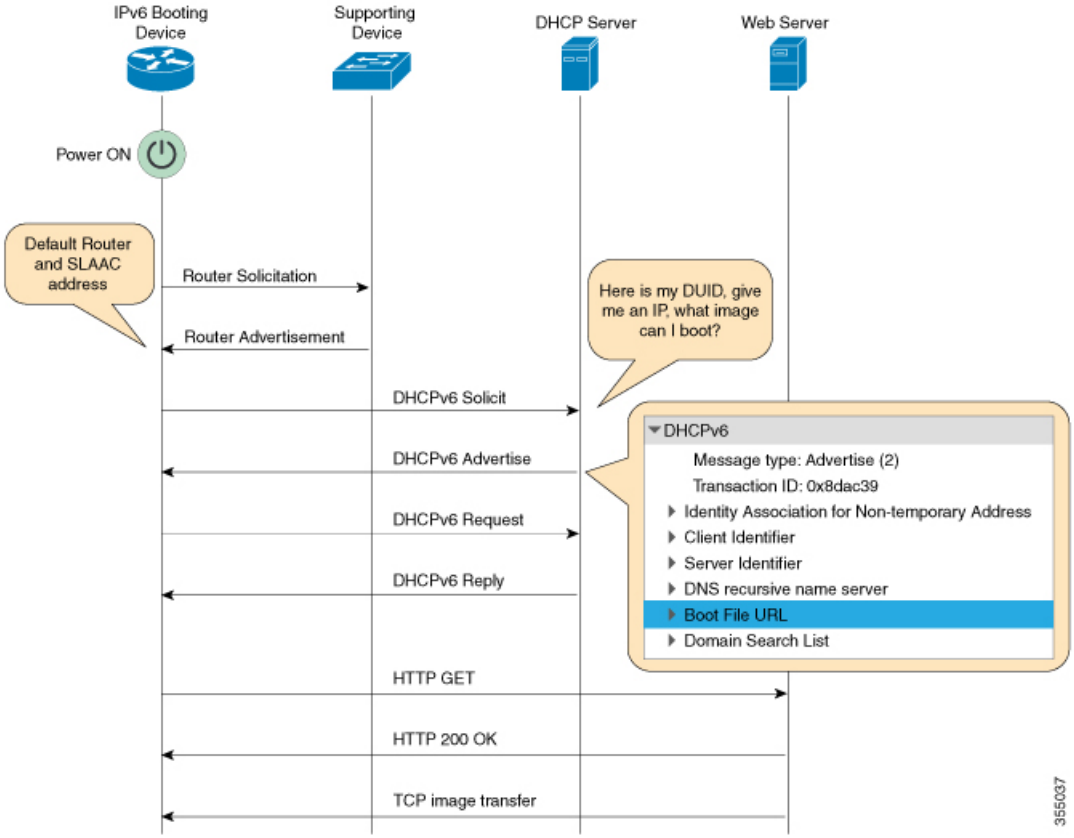
```
Device# show switch

Switch/Stack Mac Address : 046c.9d01.7d80 - Local Mac Address
Mac persistency wait time: Indefinite
                                             H/W   Current
Switch#   Role    Mac Address     Priority Version  State
-----------------------------------------------------------------------------------
 1       Member   046c.9d1e.1a00     1              Ready
 2       Standby  046c.9d01.7d80     1              Ready
*3       Active   f8b7.e24e.9a00     1      P2B     Ready
```

The following rommon variables should be configured for iPXE:

- BOOTMODE = ipxe-forever | ipxe-timeout | device
- IPXE_TIMEOUT = seconds

IPv6 iPXE Network Boot

This illustration displays how IPv6 iPXE network boot works on a Cisco device:



The four elements in the above illustration are described below:

- IPv6 Booting Device—The device that is booting through iPXE boot.
- Supporting Device—A Cisco device that is configured with an IPv6 address to generate Router Advertisement (RA) messages.

Note In this illustration, the IPv6 booting device, the supporting device, and the DHCP server are on the same subnet. However; if the supporting device and the DHCP server are on different subnets, then there must be a relay agent in the network.

- DHCP server—Any DHCP server.
- Web server—Any web server.

This section describes the IPv6 iPXE boot process:

1. The device sends a router solicitation Internet Control Message Protocol IPv6 (ICMPv6) type 133 packet to the IPv6 device on the local subnet.
2. The IPv6 device on the local subnet replies with a router advertisement (RA) message, ICMPv6 type 134 packet. The device that sent the router solicitation message, gets the default router and prefix information for Stateless Address AutoConfiguration (SLAAC) address completion from the RA packet.

3. The device sends a DHCPv6 solicit message to the multicast group address of ff02::1:2 for all DHCP agents.

   The following sample displays the fields in a DHCPv6 solicit packet during iPXE boot:

```
DHCPv6
 Message type: Solicit (1)
 Transaction ID: 0x36f5f1
 Client Identifier
 Vendor Class
 Identity Association for Non-Temporary Address
 Option Request
 User Class
 Vendor-specific Information
```

   The DHCPv6 solicit message contains the following information:

   - DHCP Unique Identifier (DUID)—Identifies the client. iPXE supports DUID-EN. EN stands for Enterprise Number, and this DUID is based on the vendor-assigned unique identifier.
   - DHCP and DHCPv6 Options

4. If the DHCPv6 server is configured, it responds with a DHCPv6 advertise packet that contains the 128 Bit IPv6 address, the boot file Uniform Resource Identifier (URI), the Domain Name System (DNS) server and domain search list, and the client and server IDs. The client ID contains the DUID of the client (In this illustration, the IPv6 Booting Device), and the Server ID contains the DUID of the DHCPv6 server.
5. The client then sends a DHCPv6 request packet to the multicast group address ff02::1:2, requesting for advertised parameters.
6. The server responds with a unicast DHCPv6 reply to the Link Local (FE80::) IPv6 address of the client. The following sample displays the fields in a DHCPv6 reply packet:

```
DHCPv6
 Message type: Reply (7)
 Transaction ID: 0x790950
 Identity Association for Non-Temporary Address
 Client Identifier
 Server Identifier
 DNS recursive name server
 Boot File URL
 Domain Search List
```

7. The device then sends an HTTP GET request to the web server.
8. If the requested image is available at the specified path, the web server responds with an OK for the HTTP GET request.
9. The TCP image transfer copies the image, and the device boots up.

IPv6 Address Assignment in Rommon Mode

The DHCP client uses the following order-of-precedence to decide which IPv6 address to use in rommon mode:

1. DHCP Server-assigned address

2. Stateless Address Auto-Configuration (SLAAC) address
3. Link-local address
4. Site-local address

The device uses the DHCP server-assigned address to boot an image. If the DHCPv6 server fails to assign an address, the device tries to use the SLAAC address. If both the DHCP server-assigned address and the SLAAC address are not available, the device uses the link-local address. However, the remote FTP/HTTP/TFTP servers must be on the same local subnet as that of the device for the image copy to succeed.

If the first three addresses are not available, the device uses the automatically generated site-local address.

Supported ROMMON Variables

The following ROMMON variables are supported in Cisco IOS XE 17.8.1:

- BAUD: Changes the device console BAUD rate to one of the Cisco standard baud rate; such as 1200, 2400, 4800, 9600, 19200, 38400, 57600, and 115200). Any invalid value will be rejected. If the BAUD variable is not set, the default will be 9600. The corresponding CLI command is
- ENABLE_BREAK: Enables a rommon break. The default value is NO.
- MANUAL_BOOT: If manual boot is set to 1, the rommon or device prompt is activated. If manual boot is set to 0, the device is reloaded; but rommon mode is not activated.
- SWITCH_IGNORE_STARTUP_CFG: If the value is 1, it causes the device to ignore the startup configuration. If the value is not set, the value is treated as zero. This is a read-only variable, and can only be modified by IOS.

iPXE-Supported DHCP Options

iPXE boot supports the following DHCPv4 and DHCPv6 options in rommon mode.

Note Catalyst 9000 Series Switches support DHCP Option 60, Option 77, DHCPv6 Options 1, Option 15, and Option 16. DHCP Option 61 is only supported on Catalyst 9350 Series Switches.

- DHCP Option 60—Vendor Class Identifier. This option is populated with the value of the ROMMON environment variable MODEL_NUM.
- DHCP Option 61—Client Identifier. This option is populated with the value of theROMMON environment variable SYSTEM_SERIAL_NUM.

Note This option is not supported on Catalyst 9400 Series Switches.

- DHCP Option 77—User Class Option. This option is added to a DHCP Discover packet, and contains the value equal to the string *iPXE*. This option helps to isolate iPXE DHCP clients looking for an image to boot from a DHCP server.

The following is sample DHCPv4 configuration from the ISC DHCP Server that displays the use of Option 77. The *if* condition in this sample implies that if Option 77 exists, and is equal to the string *iPXE*, then advertise the Boot File URI for the image.

```
host Switch2 {
     fixed-address 192.168.1.20 ;
     hardware ethernet CC:D8:C1:85:6F:11 ;
           #user-class = length of string + ASCII code for iPXE
     if exists user-class and option user-class = 04:68:50:58:45 {
         filename "http://192.168.1.146/test-image.bin"
     }
}
```

- DHCPv6 Option 1—Client Identifier Option. This option is populated with the value of the ROMMON environment variable SYSTEM_SERIAL_NUM as specified in RFC 3315. The recommended format for the ROMMON environment variable is MAC_ADDR.
- DHCPv6 Option 15—User Class Option. This option is the IPv6 User Class option in a DHCPv6 solicit message, and is populated with the string, iPXE. The following sample shows Option 15 defined in the ISC DHCP server:

```
option dhcp6.user-class code 15 = string ;
```

The following is a sample DHCP Server configuration that uses the DHCPv6 Option 15:

```
#Client-specific parameters
host switch1 {
     #assigning a fixed IPv6 address
     fixed-address6 2001:DB8::CAFE ;
     #Client DUID in hexadecimal format contains: DUID-type"2" + "EN=9" + "Chassis
serial number"
     host-identifier option dhcp6.client-id       00:02:00:00:00:09:46:4F:43:31:38:33:
31:58:31:41:53;
    #User class 00:04:69:50:58:45 is len 4 + "iPXE"
    if option dhcp6.user-class = 00:04:69:50:58:45 {
       option dhcp6.bootfile-url
"http://[2001:DB8::461/platform-pxe/edi46/test-image.bin";
    }
}
```

- DHCPv6 Option 16—Vendor Class Option. Contains the device product ID (PID). The PID can be determined from the output of the **show inventory** command or from the MODEL_NUM rommon variable. Option 16 is not a default option in the ISC DHCP Server and can be defined as follows:

```
option dhcp6.vendor-class-data code 16 = string;
```

The following sample configuration illustrates the use of DHCPv6 Option 16:

```
# Source: dhcpd6ConfigPD

host host1-ipxe6-auto-host1 {
     fixed-address6 2001:DB8::1234;
     host-identifier option dhcp6.client-id 00:02:00:00:00:09:46:4F:
     43:31:38:33:31:58:31:41:53;
     if option dhcp6.vendor-class-data = 00:00:00:09:00:0E:57:53:2D:
     43:33:38:35:30:2D:32:34:50:2D:4D {
     option dhcp6.bootfile-url
```

```
"http://[2001:DB8::46]/platform-pxe/host1/17jan-polaris.bin";
```

The table below describes the significant fields shown in the display.

Table 4: Sample Output Field Descriptions

Field	Description
dhcp6.client-id	DHCP Unique Identifier (DUID) to identify the client.
dhcp6.user-class	DHCPv6 Option 15, the User Class option
dhcp6.vendor-class-data	DHCPv6 Option 16, the Vendor Class option that contains the switch Product ID (PID).
dhcp6.bootfile-url	DHCPv6 Option 6 to request for the Boot File URI

DHCPv6 Unique Identifiers

There are three types of DHCPv6 Identifiers (DUIDs) defined by RFC 3315; these are:

- DUID-LLT—DUID Link Layer address plus time, this is the link layer address of the network interface connected to the DHCP device plus the time stamp at which it is generated.
- DUID-EN—EN stands for Enterprise Number, this DUID is based on vendor-assigned unique ID.
- DUID-LL—DUID formed using the Link Layer address of any network interface that is permanently connected to the DHCP (client/server) device.

Cisco devices that support this feature use the DUID-EN (DUID Type 2) to identify the DHCP client (that is the device in the DHCPv6 Solicit packet). Catalyst 9000 Series Switches support not only DUID-EN, but also DUID-LL (DUID Type 3). DUID-EN is the preferred type; however, if switches are unable to create it, then DUID-LL is constructed and used.

How to Configure iPXE

Configuring iPXE

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. • **boot ipxe forever** [*switch number*]
   • **boot ipxe timeout** *seconds* [*switch number*]
4. **boot system** {**switch** *switch-number* | **all**} {**flash:** | **ftp:** | **http:** | **usbflash0** | **tftp:**}
5. **end**

DETAILED STEPS

Procedure

	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	• **boot ipxe forever** [*switch number*]
	• **boot ipxe timeout** *seconds* [*switch number*]
	Example:
	`Device(config)# boot ipxe forever switch 2`
	Example:
	`Device(config)# boot ipxe timeout 30 switch 2`
		Configures the BOOTMODE rommon variable.
		• The **forever** keyword configures the BOOTMODE rommon variable as IPXE-FOREVER.
		• The **timeout** keyword configures the BOOTMODE rommon variable as IPXE-TIMEOUT.

Step 4	**boot system** {**switch** *switch-number* | **all**} {**flash:** | **ftp:** | **http:** | **usbflash0** | **tftp:**}
	Example:
	`Device(config)# boot system switch 1 http://192.0.2.42/image-filename`
	or
	`Device(config)# boot system switch 1 http://[2001:db8::1]/image-filename`
		Boots an image from the specified location.
		• You can either use an IPv4 or an IPv6 address for the remote FTP/HTTP/TFTP servers.
		• You must enter the IPv6 address inside the square brackets (as per RFC 2732); if not the device will not boot.

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

Configuring Device Boot

You can either use the **no boot ipxe** or the **default boot ipxe** command to configure device boot.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. • **no boot ipxe**
   • **default boot ipxe**
4. **end**

DETAILED STEPS

Procedure

	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	• **no boot ipxe**
	• **default boot ipxe**
	Example:
	`Device(config)# no boot ipxe`
	Example:
	`Device(config)# default boot ipxe`
		Configures device boot. The default bot mode is device boot.
		Enables default configuration on the device.

Step 4	**end** **Example:** `Device(config)# end`	Exits global configuration mode and returns to privileged EXEC mode.

Configuring iPXE

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. • **boot ipxe forever** [*switch number*]
   • **boot ipxe timeout** *seconds* [*switch number*]
4. **boot system** {**switch** *switch-number* | **all**} {**flash:** | **ftp:** | **http:** | **usbflash0** | **tftp:**}
5. **end**

DETAILED STEPS

Procedure

	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	• **boot ipxe forever** [*switch number*]
	• **boot ipxe timeout** *seconds* [*switch number*]
	Example:
	`Device(config)# boot ipxe forever switch 2`
	Example:
	`Device(config)# boot ipxe timeout 30 switch 2`
		Configures the BOOTMODE rommon variable.
		• The **forever** keyword configures the BOOTMODE rommon variable as IPXE-FOREVER.
		• The **timeout** keyword configures the BOOTMODE rommon variable as IPXE-TIMEOUT.

Step 4	**boot system** {**switch** *switch-number* | **all**} {**flash:** | **ftp:** | **http:** | **usbflash0** | **tftp:**}
	Example:
	`Device(config)# boot system switch 1 http://192.0.2.42/image-filename`
	or
	`Device(config)# boot system switch 1 http://[2001:db8::1]/image-filename`
		Boots an image from the specified location.
		• You can either use an IPv4 or an IPv6 address for the remote FTP/HTTP/TFTP servers.
		• You must enter the IPv6 address inside the square brackets (as per RFC 2732); if not the device will not boot.

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

Configuration Examples for iPXE

Example: iPXE Configuration

The following example shows that iPXE is configured to send DHCP requests forever until the device boots with an image:

```
Device# configure terminal
Device(config)# boot ipxe forever switch 2
Device(config)# end
```

The following example shows how to configure the boot mode to ipxe-timeout. The configured timeout is 200 seconds. If an iPXE boot failure occurs after the configured timeout expires, the configured device boot is activated. In this example, the configured device boot is http://[2001:db8::1]/image-filename.

```
Device# configure terminal
Device(config)# boot ipxe timeout 200 switch 2
Device(config)# boot system http://[2001:db8::1]/image-filename
Device(config)# end
```

Sample iPXE Boot Logs

The following are sample boot logs from a device in rommon mode. Here, manual boot using the **ipxe-timeout** command is configured:

```
switch: boot

pxemode:(ipxe-timeout) 60s timeout
00267.887 ipxe_get_booturl: Get URL from DHCP; timeout 60s
00267.953 ipxe_get_booturl: trying DHCPv6 (#1) for 10s
IPv4:
           ip addr 192.168.1.246
           netmask 255.255.255.0
           gateway 192.168.1.46
IPv6:
link-local addr fe80::ced8:c1ff:fe85:6f00
site-local addr fec0::ced8:c1ff:fe85:6f00
      DHCP addr 2001:db8::cafe
    router addr fe80::f29e:63ff:fe42:4756
     SLAAC addr 2001:db8::ced8:c1ff:fe85:6f00 /64
Common:
        macaddr cc:d8:c1:85:6f:00
            dns 2001:db8::46
       bootfile
http://[2001:DB8::461/platform-pxe/edi46/17jan-dev.bin--13103--2017-Feb28--13-54-50
         domain cisco.com
00269.321 ipxe_get_booturl: got URL
(http://[2001:DB8::461/platform-pxe/edi46/17jan-dev.bin--13103--2017-Feb-28--13-54-50)
Reading full image into memory …….…….…….…….…….…….…….…….…….…….…….…….…….….....
Bundle Image
-----------------------------------------------
Kernel Address    : 0x5377a7e4
Kernel Size       : 0x365e3c/3563068
Initramfs Address : 0x53ae0620
Initramfs Size    : 0x13a76f0/20608752
Compression Format: mzip
```

Sample DHCPv6 Server Configuration for iPXE

The following is a sample DHCPv6 server configuration taken from an Internet Systems Consortium (ISC) DHCP Server for reference. The lines preceded by the character #, are comments that explain the configuration that follows.

```
Default-least-time 600;
max-lease-time-7200;
log-facility local7;

#Global configuration
#domain search list
option dhcp6.domain-search "cisco.com" ;
#User-defined options:new-name code new-code = definition ;
option dhcp6.user-class code 15 = string ;
option dhcp6.vendor-class-data code 16 = string;
```

```
subnet6 2001:db8::/64 {
 #subnet range for clients requiring an address
 range6 2001:db8:0000:0000::/64;

#DNS server options
option dhcp6.name-servers 2001:db8::46;

}
#Client-specific parameters
host switch1 {
    #assigning a fixed IPv6 address
    fixed-address6 2001:DB8::CAFE ;
    #Client DUID in hexadecimal that contains: DUID-type "2" + "EN=9" + "Chassis serial
number"
    host-identifier option dhcp6.client-id 00:02:00:00:00:09:46:4F:43:31:38:33:
31:58:31:41:53;
   option dhcp6.bootfile-url "http://[2001:DB8::461/platform-pxe/edi46/test-image.bin";
}
```

For more information on DHCP server commands, see the *ISC DHCP Server* website.

In this sample configuration, the dhcp6.client-id option identifies the switch, and it is followed by the Enterprise Client DUID. The client DUID can be broken down for understanding as 00:02 + 00:00:00:09 + chassis serial number in hexadecimal format, where 2 refers to the Enterprise Client DUID Type, 9 refers to the reserved code for Cisco's Enterprise DUID, followed by the ASCII code for the Chassis serial number in hexadecimal format. The chassis serial number for the switch in this sample is FOC1831X1AS.

The Boot File URI is advertised to the switch only using the specified DUID.

The DHCPv6 Vendor Class Option 16 can also be used to identify the switch on the DHCP Server. To define Option 16 as a user-defined option, configure the following:

```
option dhcp6.vendor-class-data code 16 = string;
```

The following is a sample DHCP server configuration that identifies the switch based on the DHCPv6 Vendor Class Option 16 that is formed by using the switch Product ID:

```
# Source: dhcp6ConfigPID

host edi-46-ipxe6-auto-edi46 {
    fixed-address6 2001:DB8::1234;
    host-identifier option dhcp6.client-id 00:02:00:00:00:09:
    46:4F:43:31:38:33:31:58:31:58:31:41:53;
    if option dhcp6.vendor-class-data = 00:00:00:09:00:0E:57:
    53:2D:43:33:38:35:30:2D:32:34:50:2D:4C {
       option dhcp6.bootfile-url "http://[2001:DB8::461/platform-pxe/edi46/17jan-dev.bin";
    }
}
```

In this sample configuration, the dhcp6.vendor-class-data option refers to the DHCPv6 Option 16. In the dhcp6.vendor-class-data, 00:00:00:09 is Cisco's Enterprise DUID, 0E is the length of the PID, and the rest is the PID in hexadecimal format. The PID can also be found from the output of the **show inventory** command or from the CFG_MODEL_NUM rommon variable. The PID used in this sample configuration is WS-C3850-24P-L.

DHCPv6 options and DUIDs in the server configuration must be specified in the hexadecimal format, as per the ISC DHCP server guidelines.

Additional References for iPXE

Related Topic	Document Title
Programmability commands	Programmability Command Reference, Cisco IOS XE Everest 16.6.1

Standards and RFCs

Standard/RFC	Title
RFC 3315	*Dynamic Host Configuration Protocol for IPv6 (DHCPv6)*
RFC 3986	*Uniform Resource Identifier (URI): Generic Syntax*

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

Troubleshooting Tips for iPXE

This section provides troubleshooting tips.

- When iPXE boot is enabled on power up, the device first attempts to send a DHCPv6 Solicit message, followed by a DHCPv4 Discover message. If boot mode is **ipxe-forever** the device keeps iterating between the two forever.
- If the boot-mode is iPXE timeout, the device first sends a DHCPv6 Solicit message, and then a DHCPv4 Discover message, and the device falls back to device boot after the timeout expires.
- To interrupt iPXE boot, send a serial break to the console.

  When using a UNIX telnet client, type CTRL-] and then send break. When you are using a different TELNET client, or you are directly attached to a serial port, sending a break may be triggered by a different keystroke or command.
- If the DHCP server responds with an image, but the DNS server cannot resolve the hostname, enable DNS debugs.

Note We recommend the use of ISC DHCP server. This feature has not been verified on IOS DHCP.

- To test the HTTP server connectivity, use HTTP copy to copy a small sample file from your HTTP server to your device. For example, at the rommon prompt, enter **copy http://192.168.1.1/test null:** (the flash is normally locked and you need to use the null device for testing) or **http://[2001:db8::99]/test**.
- When manual boot is enabled, and boot mode is ipxe-timeout, the device will not automatically boot on power up. Issue the **boot** command in rommon mode. To automate the boot process on power up, disable manual boot.
- Use the **net6-show** command to display the current IPv6 parameters, including IPv6 addresses and the default router in rommon mode

Note On Catalyst 9000 Series Switches, use the **net-show** show command.

- Use the **net-dhcp** or the **net6-dhcp** commands based on your configuration, The **net-dhcp** command is a test command for DHCPv4 and the **net6-dhcp** command is for DHCPv6.

Note On Catalyst 9000 Series Switches, use the **net-dhcp -6** command for DHCPv6.

- Use the **dig** command to resolve names.

Note On Catalyst 9000 Series Switches, use the **dns-lookup** commmand to resolve names.

- Enable HTTP debug logs to view the HTTP response code from the web server.
- If Stateless Address Auto-Configuration (SLAAC) addresses are not generated, there is no router that is providing IPv6 RA messages. iPXE boot for IPv6 can still work but only with link or site-local addresses.

Troubleshooting Tips for iPXE

This section provides troubleshooting tips.

- When iPXE boot is enabled on power up, the device first attempts to send a DHCPv6 Solicit message, followed by a DHCPv4 Discover message. If boot mode is **ipxe-forever** the device keeps iterating between the two forever.
- If the boot-mode is iPXE timeout, the device first sends a DHCPv6 Solicit message, and then a DHCPv4 Discover message, and the device falls back to device boot after the timeout expires.
- To interrupt iPXE boot, send a serial break to the console.

When using a UNIX telnet client, type CTRL-] and then send break. When you are using a different TELNET client, or you are directly attached to a serial port, sending a break may be triggered by a different keystroke or command.

- If the DHCP server responds with an image, but the DNS server cannot resolve the hostname, enable DNS debugs.

Note We recommend the use of ISC DHCP server. This feature has not been verified on IOS DHCP.

- To test the HTTP server connectivity, use HTTP copy to copy a small sample file from your HTTP server to your device. For example, at the rommon prompt, enter **copy http://192.168.1.1/test null:** (the flash is normally locked and you need to use the null device for testing) or **http://[2001:db8::99]/test**.
- When manual boot is enabled, and boot mode is ipxe-timeout, the device will not automatically boot on power up. Issue the **boot** command in rommon mode. To automate the boot process on power up, disable manual boot.
- Use the **net6-show** command to display the current IPv6 parameters, including IPv6 addresses and the default router in rommon mode

Note On Catalyst 9000 Series Switches, use the **net-show** show command.

- Use the **net-dhcp** or the **net6-dhcp** commands based on your configuration, The **net-dhcp** command is a test command for DHCPv4 and the **net6-dhcp** command is for DHCPv6.

Note On Catalyst 9000 Series Switches, use the **net-dhcp -6** command for DHCPv6.

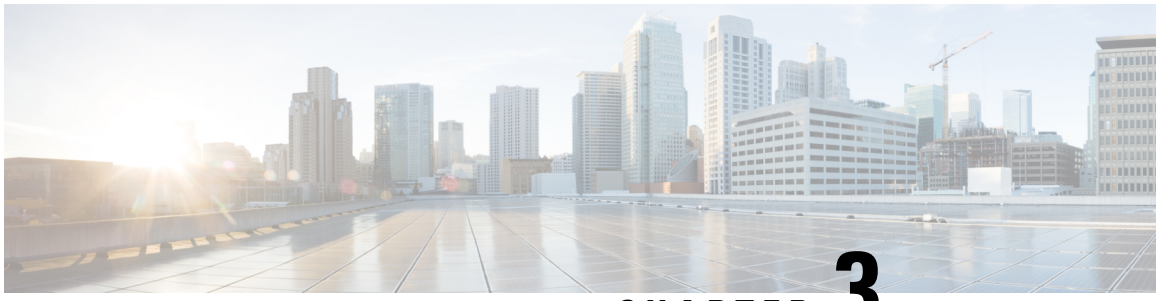
- Use the **dig** command to resolve names.

Note On Catalyst 9000 Series Switches, use the **dns-lookup** commmand to resolve names.

- Enable HTTP debug logs to view the HTTP response code from the web server.
- If Stateless Address Auto-Configuration (SLAAC) addresses are not generated, there is no router that is providing IPv6 RA messages. iPXE boot for IPv6 can still work but only with link or site-local addresses.

PART II

Shells and Scripting



CHAPTER 3

Guest Shell

Guestshell is a virtualized Linux-based environment, designed to run custom Linux applications, including Python for automated control and management of Cisco devices. It also includes the automated provisioning (Day zero) of systems. This container shell provides a secure environment, decoupled from the host device, in which users can install scripts or software packages and run them.

This module describes Guest Shell and how to enable it.

Feature Information for Guest Shell

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 5: Feature Information for Guest Shell

Feature Name	Release	Feature Information
Guest Shell	Cisco IOS XE 17.18.1	Guest Shell is a secure container that is an embedded Linux environment that allows customers to develop and run Linux and custom Python applications for automated control and management of Cisco switches. It also includes the automated provisioning of systems. This container shell provides a secure environment, decoupled from the host device, in which users can install scripts or software packages and run them. In Cisco IOS XE 17.18.1, this feature was implemented on the following platforms • Cisco Catalyst 9350 Series Switches
NETCONF Access from Guest Shell	Cisco IOS XE 17.18.1	NETCONF can be accessed from within the Guest Shell, so that users can run Python scripts and invoke Cisco-custom package CLIs using the NETCONF protocol. In 17.18.1, this feature was implemented on the following platforms: • Cisco Catalyst 9350 Series Switches • Cisco Catalyst 9610 Series Switches
Python 3 Support in Guest Shell	Cisco IOS XE 17.18.1	Python Version 3.6 is supported in Guest Shell. Python Version 3.6 is available on all supported platforms.

Restrictions for Guest Shell

- Guest Shell is not supported on Cisco Catalyst 9200L SKUs.
- NETCONF sessions cannot be established on the standby Route Processor (RP).

- Python scripts fail when running commands like the **show tech-support wireless** command, when the scale is set to 2000Aps, and clients are set to 10000.

  The output of commands like **show tech-support wireless** is huge and can cause memory exhaustion inside the Guest Shell. When using commands with huge output, redirect the output to a file. The IOS CLI can write the output to a file in the */bootflash/guest-share* directory, and it can be accessed from the Guest Shell.

- 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.

- In Cisco IOS XE 17.16.1 and later releases, Catalyst 9200CX Series Switches will use VirtualPortGroup interface with Network Address Translation (NAT) to communicate with the external network. This communication is only supported for day zero ZTP running in Guest Shell.

Information About the Guest Shell

Guest Shell Overview

The Guest Shell is a virtualized Linux-based environment, designed to run custom Linux applications, including Python, for automated control and management of Cisco devices. Using the Guest Shell, you can also install, update, and operate third-party Linux applications. The Guest Shell is bundled with the system image and can be installed using the **guestshell enable** Cisco IOS command.

The Guest Shell environment is intended for tools, Linux utilities, and manageability rather than networking.

Guest Shell shares the kernel with the host (Cisco switches and routers) system. Users can access the Linux shell of Guest Shell and update scripts and software packages in the container root filesystem. However, users within the Guest Shell cannot modify the host file system and processes.

Guest Shell container is managed using IOx. IOx is Cisco's Application Hosting Infrastructure for Cisco IOS XE devices. IOx enables hosting of applications and services developed by Cisco, partners, and third-party developers in network edge devices, seamlessly across diverse and disparate hardware platforms.

Guest Shell Software Requirements

The Guest Shell container allows users to run their scripts and apps on the system. The Guest Shell container on Intel x86 platforms will be a Linux container (LXC) with a CentOS 8.0 minimal rootfs. You can install other Python libraries such as, Python Version 3.0 during runtime using the Yum utility in CentOS 8.0. You can also install or update python packages using PIP.

Table 6: Guest Shell Software Requirements

	Guest Shell (LXC Container)
Operating System	Cisco IOS XE
Platform	All supported Cisco IOS XE platforms

	Guest Shell (LXC Container)
Guest Shell Environment	• CentOS 7 supported in Cisco IOS XE Amsterdam 17.2.1 and previous releases. • CentOS 8 supported in Cisco IOS XE Amsterdam 17.3.1 and later releases. **Note** CentOS supports only Python 3.6.
Python 2.7	Supported till Cisco IOS XE Amsterdam 17.3.1
Python 3.6	Supported in Cisco IOS XE Amsterdam 17.1.1 and later releases. In Cisco IOS XE Amsterdam 17.1.1 and Cisco IOS XE Amsterdam 17.2.1, Python V2 is the default. However, in Cisco IOS XE Amsterdam 17.3.1 and later releases, Python V3 is the default. **Note** Cisco Catalyst 9200 Series Switches support Python version 3 in Cisco IOS XE Amsterdam 17.3.1 and later releases.
Pre-installed Custom Python Libraries	• Cisco Embedded Event Manager • Cisco IOS XE CLIs • NCCLIENT library for the NETCONF API
Supported Rootfs	SSH, Yum install, and Python PIP install
GNU C Compiler	Not supported
RPM Install	Supported
Architecture	x86 and ARM

Guest Shell Security

Cisco provides security to ensure that users or apps in the Guest Shell do not compromise the host system. Guest Shell is isolated from the host kernel, and it runs as an unprivileged container.

Hardware Requirements for the Guest Shell

This section provides information about the hardware requirements for supported platforms which have variable memory configurations.

Table 7: Guest Shell Resource Requirements

Platforms	Minimum Memory
Cisco 1000 Series Integrated Services Routers	4 GB

Platforms	Minimum Memory
Cisco Catalyst 8000V Edge Software	4 GB
Cisco Cloud Services Router 1000V Series	4 GB
Cisco ISR 4000 Series Integrated Services Routers	8 GB DRAM (In Cisco IOS XE Fuji 16.8.1 and previous releases.) 4GB DRAM (In Cisco IOS XE Fuji 16.8.1 and later releases.)

All other platforms are shipped with sufficient resources to support Guest Shell.

Note Virtual-service installed applications and the Guest Shell container cannot co-exist.

Guest Shell Storage Requirements

Cisco Catalyst 9300 Series Switches and Cisco Catalyst 9500 Series Switches require 1100 MB free hard disk space for Guest Shell to install successfully.

On Cisco 4000 Series Integrated Services Routers, the Guest Shell is installed on the Network Interface Module (NIM)-Solid State Drive (SSD) (hard disk), if available. If the hard disk drive is available, there is no option to select bootflash to install Guest Shell. Cisco 4000 Series Integrated Services Routers require 1100 MB free hard disk (NIM-SSD) space for Guest Shell to install successfully.

For Cisco 4000 Series Integrated Services Routers and Cisco ASR 1000 Series Aggregation Services Routers (when an optional hard disk has been added to that router) you can only do resource resizing if you have installed the Guest Shell on the hard disk and inserted the hard disk into the router.

Note A Guest Shell installed via bootflash does not allow you to do resource resizing using application hosting configuration commands.

During Guest Shell installation, if enough hard disk space is not available, an error message is displayed.

The following is a sample error message on an Cisco ISR 4000 Series Integrated Services Router

```
% Error:guestshell_setup.sh returned error:255, message:
Not enough storage for installing guestshell. Need 1100 MB free space.
```

Bootflash or hard disk space can be used to store additional data by Guest Shell. On Cisco 4000 Series Integrated Services Routers, Guest Shell has 800 MB of storage space available. Because Guest Shell accesses the bootflash, it can use the entire space available.

Table 8: Resources Available to Guest Shell and Guest Shell Lite

Resource	Default	Minimum/Maximum
CPU	1% **Note** 1% is not standard; 800 CPU units/ total system CPU units.	1/100%
Memory	256 MB 512 MB (Cisco Catalyst 8000V Edge Software and Cisco Cloud Services Router 1000V Series)	256/256 MB 512/512 MB (Cisco Catalyst 8000V Edge Software and Cisco Cloud Services Router 1000V Series)

Enabling and Running the Guest Shell

The **guestshell enable** command installs Guest Shell. This command is also used to reactivate Guest Shell, if it is disabled.

When Guest Shell is enabled and the system is reloaded, Guest Shell remains enabled.

Note IOx must be configured before the **guestshell enable** command is used.

The **guestshell run bash** command opens the Guest Shell bash prompt. Guest Shell must already be enabled for this command to work.

Note If the following message is displayed on the console, it means that IOx is not enabled; check the output of the **show iox-service** command to view the status of IOx.

```
The process for the command is not responding or is otherwise unavailable
```

For more information on how to enable Guest Shell, see the "Configuring the AppGigabitEthernet Interface for Guest Shell" and "Enabling Guest Shell on the Management Interface" sections.

Disabling and Destroying the Guest Shell

The **guestshell disable** command shuts down and disables Guest Shell. When Guest Shell is disabled and the system is reloaded, Guest Shell remains disabled.

The **guestshell destroy** command removes the rootfs from the flash filesystem. All files, data, installed Linux applications and custom Python tools and utilities are deleted, and are not recoverable.

Accessing Guest Shell on a Device

Network administrators can use Cisco IOS commands to manage files and utilities in the Guest Shell.

During the Guest Shell installation, SSH access is setup with a key-based authentication. The access to the Guest Shell is restricted to the user with the highest privilege (15) in Cisco IOS. This user is granted access

into the Linux container as the *guestshell* Linux user, who is a sudoer, and can perform all root operations. Commands executed through the Guest Shell are executed with the same privilege that a user has when logged into the Cisco IOS terminal.

At the Guest Shell prompt, you can execute standard Linux commands.

Accessing Guest Shell Through the Management Port

By default, Guest Shell allows applications to access the management network. Users cannot change the management VRF networking configurations from inside the Guest Shell.

Note For platforms without a management port, a VirtualPortGroup can be associated with Guest Shell in the Cisco IOS configuration. For more information, see the *Sample VirtualPortGroup Configuration* section.

Cisco Catalyst 9200 Series Switches, Cisco Catalyst 9300 Series Switches, and Cisco Catalyst 9400 Series Switches support the AppGigabitEthernet interface and management interface (mgmt-if) to access Guest Shell.

Cisco Catalyst 9500 and 9500 High-Performance Series Switches and Cisco Catalyst 9600 Series Switches do not support AppGigabitEthernet interfaces.

Note Cisco Catalyst 9200L SKUs do not support Guest Shell.

Day Zero Guest Shell Provisioning Using Front-Panel Port or Fiber Uplink

On Day Zero, when the device has no management connectivity, and the only connectivity is either through the front-panel port or fibre uplink port, Guest Shell is internally configured to use the available port. The AppGigabitEthernet interface connects Guest Shell to the server.

When Guest Shell is connected to the server, the device downloads the configuration script, and configures the device. This configuration also includes downloading, setting, and starting of the virtual machine (VM). After the day zero configuration is complete, based on your configuration the system may reboot. Ensure that the system boots with only the user-specific configuration.

Guest Shell Connectivity Using the USB Port

The device uses a serial adapter to connect to multiple other devices. This serial adapter is connected through the USB port that is present on the front panel of the device.

The VM controls the serial adapter, and if there are any changes to the connected devices that are attached to the USB interface while VM is running, the VM is notified.

Stacking with Guest Shell

Guest Shell supports 1+1 high availability. 1+1 high availability is when one device is designated as the active, and the other is designated as the standby. N+1 high availability is not supported.

When Guest Shell is installed, a *guest-share* directory is automatically created in the flash file system. This directory is synchronized across stack members. Any files stored in the *guest-share* folder will be maintained when the active device goes down and the standby takes over. To preserve up to 50 MB of data during high

availability switchover, ensure that data is placed in this directory. If the size of the *guest-share* folder is more than 50 MB, it will not be synched to stack members.

During a high availability switchover, the new active device creates its own Guest Shell installation and restores Guest Shell to the synchronized state; the old file system is not maintained. Guest Shell state is internally synchronized across all stack members.

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.

IOx Tracing and Logging Overview

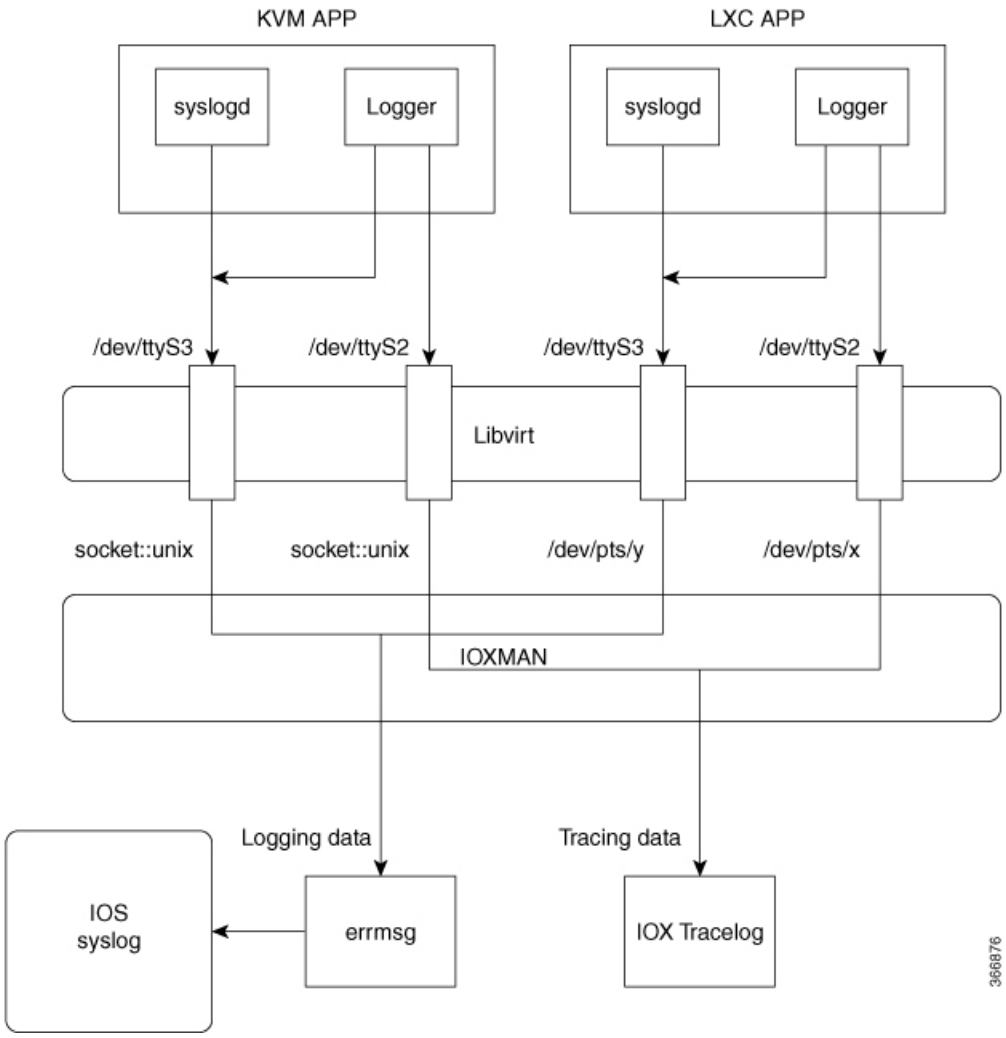
IOx tracing and logging feature allows guest application to run separately on the host device that can help reporting the logging and tracing of the data to the host. The tracing data is saved into IOx tracelog, and the logging data is saved into the Cisco IOS syslog on the host device.

You can redirect the tracing data to the appropriate storage device on the host device which can help in debugging of guest application.

IOXMAN Structure

Each guest application, a system LXC or a KVM instance is configured with its own syslogd and logfiles stored within a visible file system and are not accessible to the host device. To support logging data to the Cisco IOS syslog and tracing data to IOx tracelog on the host, two serial devices, **/dev/ttyS2** and **/dev/ttyS3**, are designated on the guest application for delivering data to the host as shown in the following figure.

Figure 2: IOXMAN Structure



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

NETCONF Access from Guest Shell

NETCONF-YANG can be accessed from within the Guest Shell, so that users can run Python scripts and invoke Cisco-custom package CLIs using the NETCONF protocol.

The Guest Shell application will establish an SSH connection without a passwordless SSH connection to the localhost and NETCONF port, by using *guestshell* as the username. This username does not correspond to any actual users configured on the device. Even if the device does have a *guestshell* user configured, there is no connection to this passwordless access. Only users with PRIV15 privilege level can access NETCONF from within the Guest Shell.

Authentication and authorization is not bypassed; instead, authentication and authorization happens while granting access to Guest Shell. Only users with the maximum privilege are granted this access.

Users can access the NETCONF service from Guest Shell without opening any external ports. Before connecting to the NETCONF-YANG server on the device, you must run the initializing commands in Guest Shell. These commands are:

```
iosp_client -f netconf_enable guestshell <port-number> and
iosp_client -f netconf_enable_passwordless guestshell <username>
```

The **iosp_client -f netconf_enable guestshell** *port-number* command configures the **netconf-yang ssh local-vrf guestshell** command, and blocks connections until NETCONF-YANG is up and running.

The **iosp_client -f netconf_enable_passwordless guestshell <username>** command creates the SSH keys required for Guest Shell access.

To remove the NETCONF-YANG access from Guest Shell, use the following commands:

```
iosp_client -f netconf_disable guestshell and
iosp_client -f netconf_disable_passwordless guestshell  <username>
```

The **iosp_client -f netconf_disable guestshell** command disables access to NETCONF from within the Guest Shell; however, the NETCONF-YANG configuration will still exist. To shut down NETCONF-YANG, use the **no netconf-yang** command.

The **iosp_client -f netconf_disable_passwordless guestshell** *username* command removes the SSH keys for the specified user. The user will not be able to access NETCONF without a password; however, the user would still be able to connect by using a password.

The *netconf_enable_guestshell* python API runs a combination of the iosp_client functions, *iosp_client -f netconf_enable guestshell 830* and *iosp_client -f netconf_enable_passwordless guestshell guestshell*. This API hides the *unfamiliar-to-user iosp_client* function. When this function is called, it does not return a response until all commands are completed. Unless the function returns an error, you can be sure that NETCONF is running, and the passwordless setup is complete; and you can start creating connections.

Logging and Tracing System Flow

The following sections describes how the IOx logging and tracing works:

LXC Logging

1. Guest OS enables **/dev/ttyS2** on the guest application.
2. Guest application writes data to **/dev/ttyS2**.
3. Libvirt emulates **/dev/ttyS2** to **/dev/pts/x** on the host.
4. IOXMAN gets the emulated serial device, **/dev/pts/x** from the XML file.
5. IOXMAN listens and reads available data from **/dev/pts/x**, sets the severity for the message, filters, parses and queues the message.
6. Start timer to send the message to **/dev/log** device on the host using **errmsg**.
7. Data is saved to the Cisco IOS syslog.

KVM Logging

1. Guest OS enables **/dev/ttyS2** on the guest application.
2. Guest application writes data to **/dev/ttyS2**.
3. Libvirt emulates **/dev/ttyS2** to **/dev/pts/x** on the host.
4. IOXMAN gets the emulated TCP path from the XML file.
5. IOXMAN opens an UNIX socket, and connects to the remote socket.
6. IOXMAN reads available data from the socket, sets the severity for the message, filters, parses, and queues the message.
7. Starts the timer to send the message to **/dev/log** device on the host using **errmsg**.
8. Data is saved to the Cisco IOS syslog.

LXC Tracing

1. Guest OS enables **/dev/ttyS3** on the guest application.
2. Configures **syslogd** to copy message to **/dev/ttyS3**.
3. Guest application writes data to **/dev/ttyS3**.
4. Libvirt emulates **/dev/ttyS3** to **/dev/pts/y** on the host.
5. IOXMAN gets the emulated serial device, **/dev/pts/y** from the XML file.
6. IOXMAN listens and reads available data from **/dev/pts/y**, filters, parses, and saves the message to IOx tracelog.
7. If IOx tracelog is full, IOXMAN rotates the tracelog file to **/bootflash/tracelogs**.

KVM Tracing

1. Guest OS enables **/dev/ttyS3** on the guest application.
2. Configures syslog to copy the message to **/dev/ttyS3**.
3. Guest application writes data to **/dev/ttyS3**.
4. Libvirt emulates **/dev/ttyS3** to TCP path on the host.
5. IOXMAN gets the emulated TCP path from the XML file.
6. IOXMAN opens an UNIX socket, and connects to the remote socket.
7. IOXMAN reads the available data from the socket, sets the severity level for the message, filters, parses, and saves the message to IOx tracelog.
8. If IOx tracelog is full, IOXMAN rotates the tracelog file to **/bootflash/tracelogs**.

Logging and Tracing of Messages

The following sections explains the logging and tracing of messages in to the Cisco IOS syslog.

Logging Messages in Cisco IOS Syslog

For any logging messages received from a guest application, IOXMAN sets the severity of the message to NOTICE by default, before sending it to the Cisco IOS syslog. When a message is received by IOSd, it is displayed on the console and saved on the syslog in the following message format:

***Apr 7 00:48:21.911: %IM-5-IOX_INST_NOTICE:ioxman: IOX SERVICE guestshell LOG: Guestshell test**

To comply with the Cisco IOS syslog, the IOXMAN does support severity levels for logging messages. To report logging messages with severity, a guest application must append a header to the front of the message.

```
[a123b234,version,severity]

a123b234 is magic number.
Version:        severity support version.  Current version is 1.
Severity:       CRIT is 2
                ERR is 3
                WARN is 4
                NOTICE is 5
                INFO is 6
                DEBUG is 7
```

The following is an example of a message log:

```
echo "[a123b234,1,2]Guestshell failed" > /dev/ttyS2
```

Perform the following steps to report logging data from a guest application to the Cisco IOS syslog:

1. If you are using C programming, use **write()** to send logging data to the host.

   ```
   #define SYSLOG_TEST       "syslog test"
        int fd;
        fd = open("/dev/ttyS2", O_WRONLY);
        write(fd, SYSLOG_TEST, strlen(SYSLOG_TEST));
        close(fd);
   ```

2. If you are using a Shell console, use **echo** to send logging data to the host.

   ```
   echo "syslog test" > /dev/ttyS2
   ```

Tracing Message to IOx Tracelog

Perform the following steps to report tracing messages from a guest application to IOx tracelog:

1. If you are using C programming, use **write()** to send tracing message to the host.

   ```
   #define SYSLOG_TEST       "tracelog test"
        int fd;
        fd = open("/dev/ttyS3", O_WRONLY);
        write(fd, SYSLOG_TEST, strlen(SYSLOG_TEST));
        close(fd);
   ```

2. If you are using C programming, use **syslog()** to send tracing message to the host.

   ```
   #define SYSLOG_TEST       "tracelog test"

   syslog(LOG_INFO, "%s\n", SYSLOG_TEST);
   ```

3. If you are using a Shell console, use **echo** to send tracing data to the host.

```
echo "tracelog test" > /dev/ttyS3
   or
logger "tracelog test"
```

How to Enable the Guest Shell

Managing IOx

Before you begin

IOx takes upto two minutes to start. CAF, IOXman, and Libvirtd services must be running to enable Guest Shell successfully.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **iox**
4. **exit**
5. **show iox-service**
6. **show app-hosting list**

DETAILED STEPS

Procedure

	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`	Configures IOx services.
Step 4	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.

	Command or Action	Purpose
Step 5	**show iox-service** **Example:** `Device# show iox-service`	Displays the status of the IOx service
Step 6	**show app-hosting list** **Example:** `Device# show app-hosting list`	Displays the list of app-hosting services enabled on the device.

Example

The following is sample output from the **show iox-service** command:

```
Device# show iox-service

IOx Infrastructure Summary:
---------------------------
IOx service (CAF) 1.10.0.0 : Running
IOx service (HA)           : Running
IOx service (IOxman)       : Running
IOx service (Sec storage) : Not Running
Libvirtd   1.3.4           : Running
Dockerd    18.03.0         : Running
Application DB Sync Info  : Available
Sync Status              : Disabled
```

The following is sample output from the **show app-hosting list** command:

```
Device# show app-hosting list

App id                           State
---------------------------------------------------------
guestshell                       RUNNING
```

Managing the Guest Shell

Note VirtualPortGroups are supported only on routing platforms.

Before you begin

IOx must be configured, and running for Guest Shell access to work. If IOx is not configured, the following message is displayed on the device console.

```
iox feature is not enabled.
```

Removing IOx removes access to the Guest Shell, but the rootfs remains unaffected.

An application or management interface must also be configured to enable and operate Guest Shell. See "Configuring the AppGigabitEthernet Interface for Guest Shell" and "Enabling Guest Shell on the Management Interface" sections for more information on enabling an interface for Guest Shell.

SUMMARY STEPS

1. **enable**
2. **guestshell enable**
3. **guestshell run** *linux-executable*
4. **guestshell run bash**
5. **guestshell disable**
6. **guestshell destroy**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enables privileged EXEC mode. • Enter your password if prompted.
Step 2	**guestshell enable** **Example:** `Device# guestshell enable`	Enables the Guest Shell service. **Note** • The **guestshell enable** command uses the management virtual routing and forwarding (VRF) instance for networking. • When using VirtualPortGroups (VPGs) for front panel networking, the VPG must be configured first. • The guest IP address and the gateway IP address must be in the same subnet.

Step 3	**guestshell run** *linux-executable*
	Example:
	`Device# guestshell run python`
	or
	`Device# guestshell run python3`
		Executes or runs a Linux program in the Guest Shell.
		Note
		In Cisco IOS XE Amsterdam 17.3.1 and later releases, only Python version 3 is supported.

Step 4	**guestshell run bash** **Example:** `Device# guestshell run bash`	Starts a Bash shell to access the Guest Shell.
Step 5	**guestshell disable** **Example:**	Disables the Guest Shell service.

	Command or Action	Purpose
	`Device# guestshell disable`	
Step 6	**guestshell destroy** **Example:** `Device# guestshell destroy`	Deactivates and uninstalls the Guest Shell service.

Managing the Guest Shell Using Application Hosting

Note This section is applicable to Cisco routing platforms. VirtualPortGroups are not supported on Cisco Catalyst Switching platforms.

IOx must be configured, and running for Guest Shell access to work. If IOx is not configured, the following message is displayed on the device console.

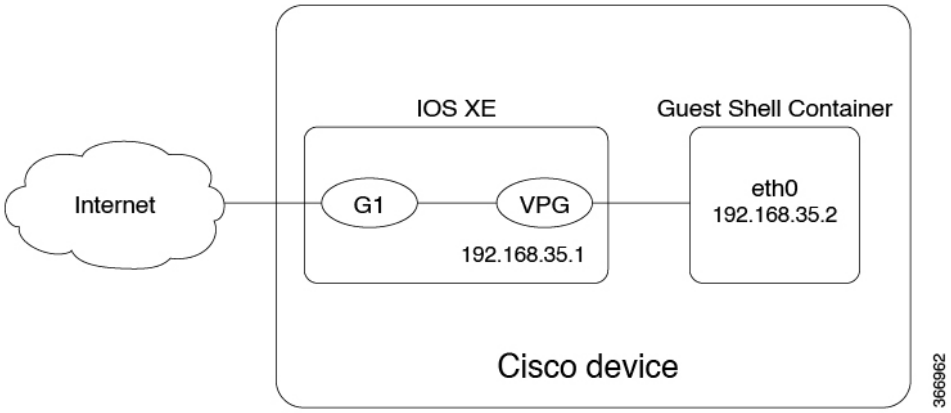
```
iox feature is not enabled.
```

Removing IOx removes access to the Guest Shell, but the rootfs remains unaffected.

Note Use this procedure (Managing the Guest Shell Using Application Hosting) to enable the Guest Shell in Cisco IOS XE Fuji 16.7.1 and later releases. For Cisco IOS XE Everest 16.6.x and previous releases, use the procedure in Managing the Guest Shell, on page 72.

```
Device(config)# interface GigabitEthernet1
Device(config-if)# ip address dhcp
Device(config-if)# ip nat outside
Device(config-if)# exit

Device(config-if)# interface VirtualPortGroup0
Device(config-if)# ip address 192.168.35.1 255.255.255.0
Device(config-if)# ip nat inside
Device(config-if)# exit

Device(config)# ip nat inside source list GS_NAT_ACL interface GigabitEthernet1 overload
Device(config)# ip access-list standard GS_NAT_ACL
Device(config)# permit 192.168.0.0 0.0.255.255

Device(config)# app-hosting appid guestshell
Device(config-app-hosting)# app-vnic gateway1 virtualportgroup 0 guest-interface 0
Device(config-app-hosting-gateway)# guest-ipaddress 192.168.35.2 netmask 255.255.255.0
Device(config-app-hosting-gateway)# exit
Device(config-app-hosting)# app-default-gateway 192.168.35.1 guest-interface 0
Device(config-app-hosting)# end

Device# guestshell enable
Device# guestshell run python
```

Figure 3: Managing the Guest Shell using Application Hosting



For front panel networking, you must configure the GigabitEthernet and VirtualPortGroup interfaces as shown above. The Guest Shell uses a Virtualportgroup as the source interface to connect to the outside network through NAT.

The following commands are used to configure inside NAT. They allow the Guest Shell to reach the internet; for example, to obtain Linux software updates:

```
ip nat inside source list
ip access-list standard
permit
```

The **guestshell run** command in the example above, runs a python executable. You can also use the **guestshell run** command to run other Linux executables; for example, see the example **guestshell run bash** command, which starts a Bash shell or the **guestshell disable** command which shuts down and disables the Guest Shell. If the system is later reloaded, the Guest Shell remains disabled.

Configuring the AppGigabitEthernet Interface for Guest Shell

Note The following task is applicable only to Catalyst switches that have the AppGigabitEthernet interface. All other Catalyst switches use the management port.

SUMMARY STEPS

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

11. **exit**
12. **app-default-gateway** *ip-address* **guest-interface** *network-interface*
13. **nameserver#** *ip-address*
14. **end**
15. **guestshell enable**

DETAILED STEPS

Procedure

	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** *interface-number* **Example:** `Device(config)# interface AppGigabitEthernet 1/0/1`	Configures the AppGigabitEthernet interface and enters interface configuration mode.
Step 4	**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 5	**exit** **Example:** `Device(config-if)# exit`	Exits interface configuration mode and returns to global configuration mode.
Step 6	**app-hosting appid** *name* **Example:** `Device(config)# app-hosting appid guestshell`	Configures an application and enters application-hosting configuration mode.
Step 7	**app-vnic AppGigabitEthernet trunk** **Example:** `Device(config-app-hosting)# app-vnic AppGigabitEthernet trunk`	Configures a trunk port as the front-panel port for application hosting, and enters application-hosting trunk configuration mode.
Step 8	**vlan** *vlan-ID* **guest-interface** *guest-interface-number* **Example:** `Device(config-config-app-hosting-trunk)# vlan 4094 guest-interface 0`	Configures a VLAN guest interface and enters application-hosting VLAN-access IP configuration mode.

	Command or Action	Purpose
Step 9	**guest-ipaddress** *ip-address* **netmask** *netmask* **Example:** `Device(config-config-app-hosting-vlan-access-ip)# guest-ipaddress 192.168.2.2 netmask 255.255.255.0`	(Optional) Configures a static IP address.
Step 10	**exit** **Example:** `Device(config-config-app-hosting-vlan-access-ip)# exit`	Exits application-hosting VLAN-access IP configuration mode and returns to application-hosting trunk configuration mode
Step 11	**exit** **Example:** `Device(config-config-app-hosting-trunk)# exit`	Exits application-hosting trunk configuration mode and returns to application-hosting configuration mode.
Step 12	**app-default-gateway** *ip-address* **guest-interface** *network-interface* **Example:** `Device(config-app-hosting)# app-default-gateway 192.168.2.1 guest-interface 0`	Configures the default management gateway.
Step 13	**nameserver#** *ip-address* **Example:** `Device(config-app-hosting)# name-server0 172.16.0.1`	Configures the Domain Name System (DNS) server.
Step 14	**end** **Example:** `Device(config-app-hosting)# end`	Exits application-hosting configuration mode and returns to privileged EXEC mode.
Step 15	**guestshell enable** **Example:** `Device# guestshell enable`	Enables the Guest Shell service.

Enabling Guest Shell on the Management Interface

Note This task is applicable to Cisco Catalyst 9200 Series Switches, Cisco Catalyst 9300 Series Switches, Cisco Catalyst 9400 Series Switches, Cisco Catalyst 9500 Series Switches, and Cisco Catalyst 9600 Series Switches.

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **app-hosting appid** *name*

4. **app-vnic management guest-interface** *interface-number*
5. **end**
6. **show app-hosting list**
7. **guestshell enable**

DETAILED STEPS

Procedure

	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 guestshell`	Configures an application and enters application-hosting configuration mode.
Step 4	**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 5	**end** **Example:** `Device(config-app-hosting-mgmt-gateway)# end`	Exits application-hosting management-gateway configuration mode and returns to privileged EXEC mode.
Step 6	**show app-hosting list** **Example:** `Device# show app-hosting list`	Displays the current status of the installed applications. **Note** Guest Shell is displayed in the list of applications, only if it is installed.
Step 7	**guestshell enable** **Example:** `Device# guestshell enable`	Enables the Guest Shell service.

Enabling and Disabling NETCONF Access from Guest Shell

Before you begin

Initialize the following commands from within the Guest Shell to initialize the NETCONF-YANG access:

SUMMARY STEPS

1. **iosp_client -f netconf_enable guestshell** *port-number*
2. **iosp_client -f netconf_enable_passwordless guestshell** *username*
3. **iosp_client -f netconf_disable guestshell**
4. **iosp_client -f netconf_disable_passwordless guestshell** *username*

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**iosp_client -f netconf_enable guestshell** *port-number* **Example:** `Guest Shell: iosp_client -f netconf_enable guestshell 3`	Configures the **netconf-yang ssh local-vrf guestshell** command, and blocks connections until NETCONF-YANG is up and running.
Step 2	**iosp_client -f netconf_enable_passwordless guestshell** *username* **Example:** `Guest Shell: iosp_client -f netconf_enable guestshell guestshell`	Creates the SSH keys required for Guest Shell access.
Step 3	**iosp_client -f netconf_disable guestshell** **Example:** `GuestShell: iosp_client -f netconf_disable guestshell`	Removes access to NETCONF from within the Guest Shell. • NETCONF-YANG configuration will still exist. To shut down NETCONF-YANG use the **no netconf-yang** command.
Step 4	**iosp_client -f netconf_disable_passwordless guestshell** *username* **Example:** `Guest Shell: iosp_client -f netconf_disable_passwordless guestshell guestshell`	Removes the access keys for the specified user. • NETCONF access is still enabled for the user; however the user will have to use a password to connect to NETCONF.

Example

Accessing the Python Interpreter

Python can be used interactively or Python scripts can be run in the Guest Shell. Use the **guestshell run python** command to launch the Python interpreter in Guest Shell and open the Python terminal.

Note In releases prior to Cisco IOS XE Amsterdam 17.3.1, Python V2 is the default. Python V3 is supported in Cisco IOS XE Amsterdam 17.1.1, and Cisco IOS XE Amsterdam 17.2.1. In Cisco IOS XE Amsterdam 17.3.1 and later releases, Python V3 is the default.

In Releases Prior to Cisco IOS XE Amsterdam 17.3.1

The **guestshell run** command is the Cisco IOS equivalent of running Linux executables, and when running a Python script from Cisco IOS, specify the absolute path. The following example shows how to specify the absolute path for the command:

```
Guestshell run python /flash/guest-share/sample_script.py parameter1 parameter2
```

The following example shows how to enable Python on a Cisco Catalyst 3650 Series Switch or a Cisco Catalyst 3850 Series Switch:

```
Device# guestshell run python

Python 2.7.11 (default, March 16 2017, 16:50:55)
[GCC 4.7.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>>>
```

The following example shows how to enable Python on a Cisco ISR 4000 Series Integrated Services Router:

```
Device# guestshell run python

Python 2.7.5 (default, Jun 17 2014, 18:11:42)
[GCC 4.8.2 20140120 (Red Hat 4.8.2-16)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>>>
```

In Cisco IOS XE Amsterdam 17.3.1 and Later Releases

The following example shows how to enable Python on Cisco Catalyst 9000 Series Switches:

```
Device# guestshell run python3

Python 3.6.8 (default, Nov 21 2019, 22:10:21)
[GCC 8.3.1 20190507 (Red Hat 8.3.1-4)] on linux
Type "help", "copyright", "credits" or "license" for more information.>>>>>
```

Configuration Examples for the Guest Shell

Example: Managing the Guest Shell

In Cisco IOS XE Amsterdam 17.1.x to Cisco IOS XE Amsterdam 17.2.x

The following example shows how to enable Guest Shell. In Cisco IOS XE Amsterdam 17.1.x and Cisco IOS XE Amsterdam 17.2.x, both Python V2.7 and Python V3.6 are supported. However, Python V2.7 is the default in these releases.

```
Device> enable
Device# guestshell enable
```

```
Management Interface will be selected if configured
Please wait for completion
Guestshell enabled successfully

Device# guestshell run python
or
Device# guestshell run python3

Python 2.7.5 (default, Jun 17 2014, 18:11:42)
[GCC 4.8.2 20140120 (Red Hat 4.8.2-16)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>>>


Device# guestshell run bash

[guestshell@guestshell ~]$

Device# guestshell disable

Guestshell disabled successfully

Device# guestshell destroy

Guestshell destroyed successfully
```

In Cisco IOS XE Amsterdam 17.3.1 and Later Releases

The following example shows how to enable Guest Shell. In Cisco IOS XE Amsterdam 17.3.1 and later releases, only Python V3.6 is supported.

```
Device> enable
Device# guestshell enable

Management Interface will be selected if configured
Please wait for completion
Guestshell enabled successfully

Device# guestshell run python3

Python 3.6.8 (default, Nov 21 2019, 22:10:21)
[GCC 8.3.1 20190507 (Red Hat 8.3.1-4)] on linux
Type "help", "copyright", "credits" or "license" for more information.>>>>>

>>>>>

Device# guestshell run bash

[guestshell@guestshell ~]$

Device# guestshell disable

Guestshell disabled successfully

Device# guestshell destroy

Guestshell destroyed successfully
```

Sample VirtualPortGroup Configuration

Note VirtualPortGroups are supported only on Cisco routing platforms.

When using the VirtualPortGroup interface for Guest Shell networking, the VirtualPortGroup interface must have a static IP address configured. The front port interface must be connected to the Internet and Network Address Translation (NAT) must be configured between the VirtualPortGroup and the front panel port.

The following is a sample VirtualPortGroup configuration:

```
Device> enable
Device# configure terminal
Device(config)# interface VirtualPortGroup 0
Device(config-if)# ip address 192.168.35.1 255.255.255.0
Device(config-if)# ip nat inside
Device(config-if)# no mop enabled
Device(config-if)# no mop sysid
Device(config-if)# exit
Device(config)# interface GigabitEthernet 0/0/3
Device(config-if)# ip address 10.0.12.19 255.255.0.0
Device(config-if)# ip nat outside
Device(config-if)# negotiation auto
Device(config-if)# exit
Device(config)# ip route 0.0.0.0 0.0.0.0 10.0.0.1
Device(config)# ip route 10.0.0.0 255.0.0.0 10.0.0.1
!Port forwarding to use ports for SSH and so on.
Device(config)# ip nat inside source static tcp 192.168.35.2 7023 10.0.12.19 7023 extendable
Device(config)# ip nat outside source list NAT_ACL interface GigabitEthernet 0/0/3 overload
Device(config)# ip access-list standard NAT_ACL
Device(config-std-nacl)# permit 192.168.0.0 0.0.255.255
Device(config-std-nacl)# exit

! App-hosting configuration
Device(config)# app-hosting appid guestshell
Device(config-app-hosting)# app-vnic gateway1 virtualportgroup 0 guest-interface 0
Device(config-app-hosting-gateway)# guest-ipaddress 192.168.35.2 netmask 255.255.255.0
Device(config-app-hosting-gateway)# exit
Device(config-app-hosting)# app-resource profile custom
Device(config-app-resource-profile-custom)# cpu 1500
Device(config-app-resource-profile-custom)# memory 512
Device(config-app-resource-profile-custom)# end

Device# guestshell enable
Device# guestshell run python
```

Example: Configuring the AppGigabitEthernet Interface for Guest Shell

Note The following task is applicable only to Catalyst switches that have the AppGigabitEthernet interface. All other Catalyst switches use the management port.

The following example shows how to configure an AppGigabitEthernet interface for Guest Shell. Here, VLAN 4094 creates a Network Address Translation (NAT) this is used for Guest Shell. VLAN 1 is an external interface.

```
Device> enable
Device# configure terminal
Device(config)# ip nat inside source list NAT_ACL interface vlan 1 overload
Device(config)# ip access-list standard NAT_ACL
Device(config-std-nacl)# permit 192.168.0.0 0.0.255.255
Device(config-std-nacl)# exit
Device(config)# vlan 4094
Device(config-vlan)# exit
Device(config)# interface vlan 4094
Device(config-if)# ip address 192.168.2.1 255.255.255.0
Device(config-if)# ip nat inside
Device(config-if)# exit
Device(config)# interface vlan 1
Device(config-if)# ip nat outside
Device(config-if)# exit
Device(config)# ip routing
Device(config)# ip route 0.0.0.0 0.0.0.0 209.165.201.1
Device(config)# interface AppGigabitEthernet 1/0/1
Device(config-if)# switchport mode trunk
Device(config-if)# exit
Device(config)# app-hosting appid guestshell
Device(config-app-hosting)# app-vnic AppGigEthernet trunk
Device(config-config-app-hosting-trunk)# vlan 4094 guest-interface 0
Device(config-config-app-hosting-vlan-access-ip)# guest-ipaddress 192.168.2.2 netmask
255.255.255.0
Device(config-config-app-hosting-vlan-access-ip)# exit
Device(config-config-app-hosting-trunk)# exit
Device(config-app-hosting)# app-default-gateway 192.168.2.1 guest-interface 0
Device(config-app-hosting)# name-server0 172.16.0.1
Device(config-app-hosting)# name-server1 198.51.100.1
Device(config-app-hosting)# end
Device# guestshell enable
```

Example: Enabling Guest Shell on the Management Interface

This example is applicable to Cisco Catalyst 9200 Series Switches, Cisco Catalyst 9300 Series Switches, Cisco Catalyst 9400 Series Switches, Cisco Catalyst 9500 Series Switches, and Cisco Catalyst 9600 Series Switches.

```
Device> enable
Device# configure terminal
Device(config)# app-hosting appid guestshell
Device(config-app-hosting)# app-vnic management guest-interface 0
Device(config-app-hosting-mgmt-gateway)# end
```

```
Device# guestshell enable
```

Example: Guest Shell Usage

From the Guest Shell prompt, you can run Linux commands. The following example shows the usage of some Linux commands.

```
[guestshell@guestshell~]$ pwd
/home/guestshell

[guestshell@guestshell~]$ whoami
guestshell

[guestshell@guestshell~]$ uname -a
Linux guestshell 5.4.85 #1 SMP Tue Dec 22 10:50:44 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
```

Cisco 4000 Series Integrated Services Routers use the **dohost** provided by CentOS Linux release 7.1.1503.

Note The **dohost** command requires the **ip http server** command to be configured on the device.

Example: Guest Shell Networking Configuration

The following is a sample Guest Shell networking configuration.

- Configure Domain Name System (DNS)
- Configure proxy settings
- Configure YUM or PIP to use proxy settings

Sample DNS Configuration for Guest Shell

The following is a sample DNS configuration for Guest Shell:

```
[guestshell@guestshell ~]$ cat/etc/resolv.conf
nameserver 192.0.2.1

Other Options:
[guestshell@guestshell ~]$ cat/etc/resolv.conf
domain cisco.com
search cisco.com
nameserver 192.0.2.1
search cisco.com
nameserver 198.51.100.1
nameserver 172.16.0.6
domain cisco.com
```

```
nameserver 192.0.2.1
nameserver 172.16.0.6
nameserver 192.168.255.254
```

Example: Configuring Proxy Environment Variables

If your network is behind a proxy, configure proxy variables in Linux. If required, add these variables to your environment.

The following example shows how to configure your proxy variables:

```
[guestshell@guestshell ~]$cat /bootflash/proxy_vars.sh
export http_proxy=http://proxy.example.com:80/
export https_proxy=http://proxy.example.com:80/
export ftp_proxy=http://proxy.example.com:80/
export no_proxy=example.com
export HTTP_PROXY=http://proxy.example.com:80/
export HTTPS_PROXY=http://proxy.example.com:80/
export FTP_PROXY=http://proxy.example.com:80/
guestshell ~] source /bootflash/proxy_vars.sh
```

Example: Configuring Yum and PIP for Proxy Settings

The following example shows how to use Yum for setting proxy environment variables:

```
cat /etc/yum.conf | grep proxy
[guestshell@guestshell~]$ cat/bootflash/yum.conf | grep proxy
proxy=http://proxy.example.com:80/
```

PIP install picks up environment variable used for proxy settings. Use sudo with -E option for PIP installation. If the environment variables are not set, define them explicitly in PIP commands as shown in following example:

```
sudo pip --proxy http://proxy.example.com:80/install requests
sudo pip install --trusted-bost pypi.example.com --index-url
http://pypi.example.com/simple requests
```

The following example shows how to use PIP install for Python:

```
Sudo -E pip install requests
[guestshell@guestshell ~]$ python
Python 2.17.11 (default, Feb 3 2017, 19:43:44)
[GCC 4.7.0] on linux2
Type "help", "copyright", "credits" or "license" for more information
>>>import requests
```

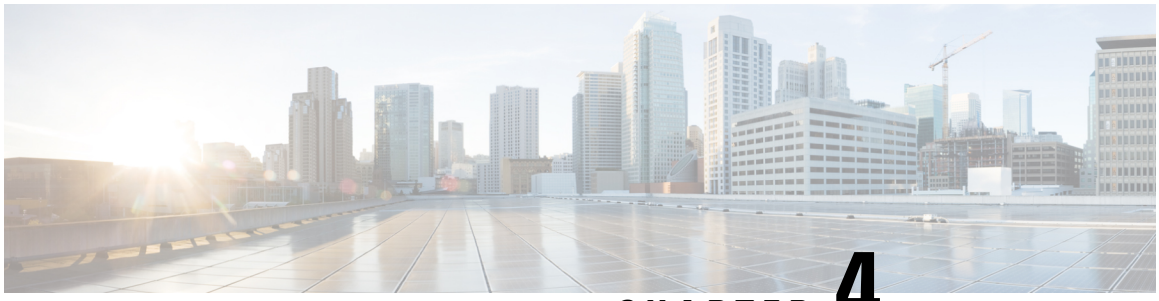
Additional References for Guest Shell

Related Documents

Related Topic	Document Title
Python module	CLI Python Module
Zero-Touch Provisioning	Zero-Touch Provisioning

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



CHAPTER 4

Python API

Python programmabililty supports Python APIs.

Feature Information for Python API

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 9: Feature Information for the CLI Python Module

Feature Name	Release	Feature Information
CLI Python Module	Cisco IOS XE 17.18.1	Python programmabilty provides a Python module that allows users to interact with IOS using CLIs. In Cisco IOS XE 17.18.1, this feature was implemented on the following platforms: • Cisco Catalyst 9350 Series Switches

About Python

The Cisco IOS XE devices support Python Version 2.7 in both interactive and non-interactive (script) modes within the Guest Shell. The Python scripting capability gives programmatic access to a device's CLI to perform various tasks and Zero Touch Provisioning or Embedded Event Manager (EEM) actions.

Cisco Python Module

Cisco provides a Python module that provides access to run EXEC and configuration commands. You can display the details of the Cisco Python module by entering the **help()** command. The **help()** command displays the properties of the Cisco CLI module.

The following example displays information about the Cisco Python module:

```
Device# guestshell run python

Python 2.7.5 (default, Jun 17 2014, 18:11:42)
[GCC 4.8.2 20140120 (Red Hat 4.8.2-16)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> >>> from cli import cli,clip,configure,configurep, execute, executep
>>> help(configure)
Help on function configure in module cli:

configure(configuration)
Apply a configuration (set of Cisco IOS CLI config-mode commands) to the device
and return a list of results.

configuration = '''interface gigabitEthernet 0/0
no shutdown'''

# push it through the Cisco IOS CLI.
try:
results = cli.configure(configuration)
print "Success!"
except CLIConfigurationError as e:
print "Failed configurations:"
for failure in e.failed:
print failure

Args:
configuration (str or iterable): Configuration commands, separated by newlines.

Returns:
list(ConfigResult): A list of results, one for each line.

Raises:
CLISyntaxError: If there is a syntax error in the configuration.

>>> help(configurep)
Help on function configurep in module cli:

configurep(configuration)
Apply a configuration (set of Cisco IOS CLI config-mode commands) to the device
and prints the result.

configuration = '''interface gigabitEthernet 0/0
no shutdown'''

# push it through the Cisco IOS CLI.
configurep(configuration)

Args:
configuration (str or iterable): Configuration commands, separated by newlines.
>>> help(execute)
Help on function execute in module cli:

execute(command)
```

```
Execute Cisco IOS CLI exec-mode command and return the result.

command_output = execute("show version")

Args:
command (str): The exec-mode command to run.

Returns:
str: The output of the command.

Raises:
CLISyntaxError: If there is a syntax error in the command.
```

```
>>> help(executep)
Help on function executep in module cli:

executep(command)
Execute Cisco IOS CLI exec-mode command and print the result.

executep("show version")

Args:
command (str): The exec-mode command to run.
```

```
>>> help(cli)
Help on function cli in module cli:

cli(command)
    Execute Cisco IOS CLI command(s) and return the result.

    A single command or a delimited batch of commands may be run. The
    delimiter is a space and a semicolon, " ;". Configuration commands must be
    in fully qualified form.

    output = cli("show version")
    output = cli("show version ; show ip interface brief")
    output = cli("configure terminal ; interface gigabitEthernet 0/0 ; no shutdown")

    Args:
        command (str): The exec or config CLI command(s) to be run.

    Returns:
        string: CLI output for show commands and an empty string for
            configuration commands.

    Raises:
        errors.cli_syntax_error: if the command is not valid.
        errors.cli_exec_error: if the execution of command is not successful.
```

```
>>> help(clip)
Help on function clip in module cli:

clip(command)
    Execute Cisco IOS CLI command(s) and print the result.

    A single command or a delimited batch of commands may be run. The
    delimiter is a space and a semicolon, " ;". Configuration commands must be
    in fully qualified form.

    clip("show version")
    clip("show version ; show ip interface brief")
    clip("configure terminal ; interface gigabitEthernet 0/0 ; no shutdown")
```

```
Args:
    command (str): The exec or config CLI command(s) to be run.
```

Cisco Python Module to Execute IOS CLI Commands

Note Guest Shell must be enabled for Python to run. For more information, see the *Guest Shell* chapter.

The Python programming language uses six functions that can execute CLI commands. These functions are available from the Python CLI module. To use these functions, execute the **import cli** command.

Arguments for these functions are strings of CLI commands. To execute a CLI command through the Python interpreter, enter the CLI command as an argument string of one of the following six functions:

- **cli.cli(command)**—This function takes an IOS command as an argument, runs the command through the IOS parser, and returns the resulting text. If this command is malformed, a Python exception is raised. The following is sample output from the **cli.cli(command)** function:

```
>>> import cli
>>> cli.clip('configure terminal; interface loopback 10; ip address
10.10.10.10 255.255.255.255')
*Mar 13 18:39:48.518: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback10, changed
 state to up
>>> cli.clip('show clock')
'\n*18:11:53.989 UTC Mon Mar 13 2017\n'
>>> output=cli.cli('show clock')
>>> print(output)
*18:12:04.705 UTC Mon Mar 13 2017
```

- **cli.clip(command)**—This function works exactly the same as the **cli.cli(command)** function, except that it prints the resulting text to *stdout* rather than returning it. The following is sample output from the **cli.clip(command)** function:

```
>>> cli
>>> cli.clip('configure terminal; interface loopback 11; ip address
10.11.11.11 255.255.255.255')
*Mar 13 18:42:35.954: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback11, changed
 state to up
*Mar 13 18:42:35.954: %LINK-3-UPDOWN: Interface Loopback11, changed state to up
>>> cli.clip('show clock')
*18:13:35.313 UTC Mon Mar 13 2017
>>> output=cli.clip('show clock')
*18:19:26.824 UTC Mon Mar 13 2017
>>> print (output)
None
```

- **cli.execute(command)**—This function executes a single EXEC command and returns the output; however, does not print the resulting text No semicolons or newlines are allowed as part of this command. Use a Python list with a for-loop to execute this function more than once. The following is sample output from the **cli.execute(command)**

function:

```
>>> cli.execute("show clock")
'15:11:20.816 UTC Thu Jun 8 2017'
>>>
>>> cli.execute('show clock'; 'show ip interface brief')
  File "<stdin>", line 1
    cli.execute('show clock'; 'show ip interface brief')
                            ^
SyntaxError: invalid syntax
>>>
```

- **cli.executep(command)**—This function executes a single command and prints the resulting text to *stdout* rather than returning it. The following is sample output from the **cli.executep(command)** function:

```
>>> cli.executep('show clock')
*18:46:28.796 UTC Mon Mar 13 2017
>>> output=cli.executep('show clock')
*18:46:36.399 UTC Mon Mar 13 2017
>>> print(output)
None
```

- **cli.configure(command)**—This function configures the device with the configuration available in commands. It returns a list of named tuples that contains the command and its result as shown below:

```
[Think: result = (bool(success), original_command, error_information)]
```

The command parameters can be in multiple lines and in the same format that is displayed in the output of the **show running-config** command. The following is sample output from the **cli.configure(command)** function:

```
>>>cli.configure(["interface GigabitEthernet1/0/7", "no shutdown",
"end"])
[ConfigResult(success=True, command='interface GigabitEthernet1/0/7',
line=1, output='', notes=None), ConfigResult(success=True, command='no shutdown',
line=2, output='', notes=None), ConfigResult(success=True, command='end',
line=3, output='', notes=None)]
```

- **cli.configurep(command)**—This function works exactly the same as the **cli.configure(command)** function, except that it prints the resulting text to *stdout* rather than returning it. The following is sample output from the **cli.configurep(command)** function:

```
>>> cli.configurep(["interface GigabitEthernet1/0/7", "no shutdown",
"end"])
Line 1 SUCCESS: interface GigabitEthernet1/0/7
Line 2 SUCCESS: no shut
Line 3 SUCCESS: end
```

Python Scripts Overview

Python run in a virtualized Linux-based environment, Guest Shell. For more information, see the *Guest Shell* chapter. Cisco provides a Python module that allows user's Python scripts to run IOS CLI commands on the host device.

Using Python

Additional References for Python API

Related Documents

Related Topic	Document Title
Guest Shell	Guest Shell
EEM Python Module	Python Scripting in EEM

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

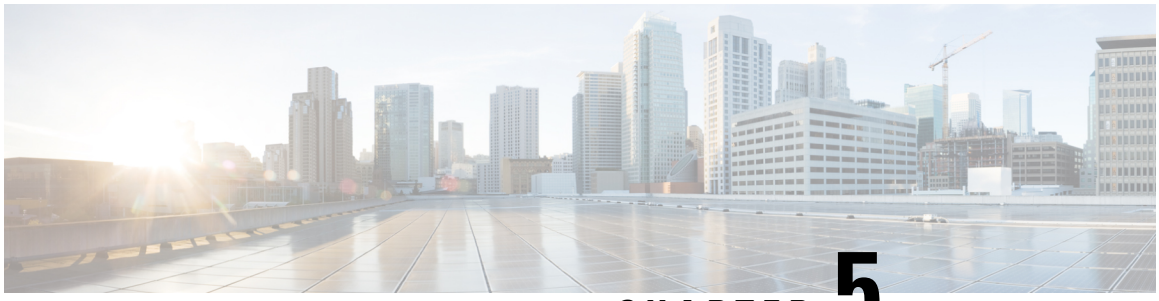
Additional References for Python API

Related Documents

Related Topic	Document Title
Guest Shell	Guest Shell
EEM Python Module	Python Scripting in EEM

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



CHAPTER 5

EEM Python Module

Embedded Event Manager (EEM) policies support Python scripts. Python scripts can be executed as part of EEM actions in EEM applets.

Feature Information for EEM Python Module

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 10: Feature Information for EEM Python Module

Feature Name	Release	Feature Information
EEM Python Module	Cisco IOS XE 17.18.1	This feature supports Python scripts as EEM policies. No new commands were introduced. In Cisco IOS XE 17.18.1, this feature was implemented on the following platforms: • Cisco Catalyst 9350 Series Switches

Prerequisites for the EEM Python Module

Guest Shell must be working within the container. Guest Shell is not enabled by default. For more information see the *Guest Shell* feature.

How to Configure the EEM Python Policy

Python Scripting in EEM

Embedded Event Manager (EEM) policies support Python scripts. You can register Python scripts as EEM policies, and execute the registered Python scripts when a corresponding event occurs. The EEM Python script has the same event specification syntax as the EEM TCL policy.

Configured EEM policies run within the Guest Shell. Guest Shell is a virtualized Linux-based environment, designed to run custom Linux applications, including Python for automated control and management of Cisco devices. The Guest Shell container provides a Python interpreter.

EEM Python Package

The EEM Python package can be imported to Python scripts for running EEM-specific extensions.

Note The EEM Python package is available only within the EEM Python script (The package can be registered with EEM, and has the EEM event specification in the first line of the script.) and not in the standard Python script (which is run using the Python script name).

The Python package includes the following application programming interfaces (APIs):

- Action APIs—Perform EEM actions and have default parameters.
- CLI-execution APIs—Run IOS commands, and return the output. The following are the list of CLI-execution APIs:
  - eem_cli_open()
  - eem_cli_exec()
  - eem_cli_read()
  - eem_cli_read_line()
  - eem_cli_run()
  - eem_cli_run_interactive()
  - eem_cli_read_pattern()
  - eem_cli_write()
  - eem_cli_close()
- Environment variables-accessing APIs—Get the list of built-in or user-defined variables. The following are the environment variables-accessing APIs:
  - eem_event_reqinfo ()-Returns the built-in variables list.
  - eem_user_variables()-Returns the current value of an argument.

Python-Supported EEM Actions

The Python package (is available only within the EEM script, and not available for the standard Python script) supports the following EEM actions:

- Syslog message printing
- Send SNMP traps
- Reload the box
- Switchover to the standby device
- Run a policy
- Track Object read
- Track Object Set
- Cisco Networking Services event generation

The EEM Python package exposes the interfaces for executing EEM actions. You can use the Python script to call these actions, and they are forwarded from the Python package via Cisco Plug N Play (PnP) to the action handler.

EEM Variables

An EEM policy can have the following types of variables:

- Event-specific built-in variables—A set of predefinied variables that are populated with details about the event that triggered the policy. The eem_event_reqinfo () API returns the builtin variables list. These variables can be stored in the local machine and used as local variables. Changes to local variables do not reflect in builtin variables.
- User-defined variables—Variables that can be defined and used in policies. The value of these variables can be referred in the Python script. While executing the script, ensure that the latest value of the variable is available. The eem_user_variables() API returns the current value of the argument that is provided in the API.

EEM CLI Library Command Extensions

The following CLI library commands are available within EEM for the Python script to work:

- eem_cli_close()—Closes the EXEC process and releases the VTY and the specified channel handler connected to the command.
- eem_cli_exec—Writes the command to the specified channel handler to execute the command. Then reads the output of the command from the channel and returns the output.
- eem_cli_open—Allocates a VTY, creates an EXEC CLI session, and connects the VTY to a channel handler. Returns an array including the channel handler.
- eem_cli_read()—Reads the command output from the specified CLI channel handler until the pattern of the device prompt occurs in the contents read. Returns all the contents read up to the match.

- eem_cli_read_line()—Reads one line of the command output from the specified CLI channel handler. Returns the line read.
- eem_cli_read_pattern()—Reads the command output from the specified CLI channel handler until the pattern that is to be matched occurs in the contents read. Returns all the contents read up to the match.
- eem_cli_run()—Iterates over the items in the *clist* and assumes that each one is a command to be executed in the enable mode. On success, returns the output of all executed commands and on failure, returns error.
- eem_cli_run_interactive()—Provides a sublist to the *clist* which has three items. On success, returns the output of all executed commands and on failure, returns the error. Also uses arrays when possible as a way of making things easier to read later by keeping expect and reply separated.
- eem_cli_write()—Writes the command that is to be executed to the specified CLI channel handler. The CLI channel handler executes the command.

How to Configure the EEM Python Policy

For the Python script to work, you must enable the Guest Shell. For more information, see the *Guest Shell* chapter.

Registering a Python Policy

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager directory user policy** *path*
4. **event manager policy** *policy-filename*
5. **exit**
6. **show event manager policy registered**
7. **show event manager history events**

DETAILED STEPS

Procedure

	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.

	Command or Action	Purpose
Step 3	**event manager directory user policy** *path* **Example:** `Device(config)# event manager directory user policy flash:/user_library`	Specifies a directory to use for storing user library files or user-defined EEM policies. **Note** You must have a policy in the specified path. For example, in this step, the eem_script.py policy is available in the flash:/user_library folder or path.
Step 4	**event manager policy** *policy-filename* **Example:** `Device(config)# event manager policy eem_script.py`	Registers a policy with EEM. • The policy is parsed based on the file extension. If the file extension is .py, the policy is registered as Python policy. • EEM schedules and runs policies on the basis of an event specification that is contained within the policy itself. When the **event manager policy** command is invoked, EEM examines the policy and registers it to be run when the specified event occurs.
Step 5	**exit** **Example:** `Device(config)# exit`	Exits global configuration mode and returns to privileged EXEC mode.
Step 6	**show event manager policy registered** **Example:** `Device# show event manager policy registered`	Displays the registered EEM policies.
Step 7	**show event manager history events** **Example:** `Device# show event manager history events`	Displays EEM events that have been triggered.

Example

The following is sample output from the **show event manager policy registered** command:

```
Device# show event manager policy registered

No.  Class     Type    Event Type          Trap  Time Registered           Name
1    script    user    multiple            Off   Tue Aug 2 22:12:15 2016   multi_1.py
 1: syslog: pattern {COUNTER}
 2: none: policyname {multi_1.py} sync {yes}
 trigger delay 10.000
  correlate event 1 or event 2
  attribute tag 1 occurs 1
 nice 0 queue-priority normal maxrun 100.000 scheduler rp_primary Secu none


2    script    user    multiple            Off   Tue Aug 2 22:12:20 2016   multi_2.py
 1: syslog: pattern {COUNTER}
 2: none: policyname {multi_2.py} sync {yes}
```

```
 trigger
  correlate event 1 or event 2
 nice 0 queue-priority normal maxrun 100.000 scheduler rp_primary Secu none


3    script    user    multiple            Off   Tue Aug 2 22:13:31 2016   multi.tcl
 1: syslog: pattern {COUNTER}
 2: none: policyname {multi.tcl} sync {yes}
 trigger
  correlate event 1 or event 2
  attribute tag 1 occurs 1
 nice 0 queue-priority normal maxrun 100.000 scheduler rp_primary Secu none
```

Running Python Scripts as Part of EEM Applet Actions

Python Script: eem_script.py

An EEM applet can include a Python script with an action command. In this example, an user is trying to run a standard Python script as part of the EEM action, however; EEM Python package is not available in the standard Python script. The standard Python script in IOS has a package named *from cli import cli,clip* and this package can be used to execute IOS commands.

```
import sys
from cli import cli,clip,execute,executep,configure,configurep

intf= sys.argv[1:]
intf = ''.join(intf[0])


print ('This script is going to unshut interface %s and then print show ip interface
brief'%intf)

if intf == 'loopback55':
configurep(["interface loopback55","no shutdown","end"])
else :
cmd='int %s,no shut ,end' % intf
configurep(cmd.split(','))

executep('show ip interface brief')
```

This following is sample output from the **guestshell run python3** command.

```
Device#  guestshell run python3 /flash/eem_script.py loop55

This script is going to unshut interface loop55 and then print show ip interface brief
Line 1 SUCCESS: int loop55
Line 2 SUCCESS: no shut
Line 3 SUCCESS: end
Interface IP-Address OK? Method Status Protocol
Vlan1 unassigned YES NVRAM administratively down down
GigabitEthernet0/0 5.30.15.37 YES NVRAM up up
GigabitEthernet1/0/1 unassigned YES unset down down
GigabitEthernet1/0/2 unassigned YES unset down down
GigabitEthernet1/0/3 unassigned YES unset down down
GigabitEthernet1/0/4 unassigned YES unset up up
GigabitEthernet1/0/5 unassigned YES unset down down
GigabitEthernet1/0/6 unassigned YES unset down down
```

```
GigabitEthernet1/0/7 unassigned YES unset down down
GigabitEthernet1/0/8 unassigned YES unset down down
GigabitEthernet1/0/9 unassigned YES unset down down
GigabitEthernet1/0/10 unassigned YES unset down down
GigabitEthernet1/0/11 unassigned YES unset down down
GigabitEthernet1/0/12 unassigned YES unset down down
GigabitEthernet1/0/13 unassigned YES unset down down
GigabitEthernet1/0/14 unassigned YES unset down down
GigabitEthernet1/0/15 unassigned YES unset down down
GigabitEthernet1/0/16 unassigned YES unset down down
GigabitEthernet1/0/17 unassigned YES unset down down
GigabitEthernet1/0/18 unassigned YES unset down down
GigabitEthernet1/0/19 unassigned YES unset down down
GigabitEthernet1/0/20 unassigned YES unset down down
GigabitEthernet1/0/21 unassigned YES unset down down
GigabitEthernet1/0/22 unassigned YES unset down down
GigabitEthernet1/0/23 unassigned YES unset up up
GigabitEthernet1/0/24 unassigned YES unset down down
GigabitEthernet1/1/1 unassigned YES unset down down
GigabitEthernet1/1/2 unassigned YES unset down down
GigabitEthernet1/1/3 unassigned YES unset down down
GigabitEthernet1/1/4 unassigned YES unset down down
Te1/1/1 unassigned YES unset down down
Te1/1/2 unassigned YES unset down down
Te1/1/3 unassigned YES unset down down
Te1/1/4 unassigned YES unset down down
Loopback55 10.55.55.55 YES manual up up

Device#
Jun 7 12:51:20.549: %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback55,
changed state to up
Jun 7 12:51:20.549: %LINK-3-UPDOWN: Interface Loopback55, changed state to up
```

The following is a sample script for printing messages to the syslog. This script must be stored in a file, copied to the file system on the device, and registered using the event manager policy file.

```
::cisco::eem::event_register_syslog tag "1"    pattern     COUNTER    maxrun 200

import eem
import time

eem.action_syslog("SAMPLE SYSLOG MESSAGE","6","TEST")
```

The following is sample script to print EEM environment variables. This script must be stored in a file, copied to the file system on the device, and registered using the event manager policy file.

```
::cisco::eem::event_register_syslog tag "1"    pattern     COUNTER    maxrun 200

import eem
import time

c = eem.env_reqinfo()


print "EEM Environment Variables"
for k,v in c.iteritems():
      print "KEY : " + k + str(" ---> ") + v

print "Built in Variables"
for i,j in a.iteritems() :
```

```
print "KEY : " + i + str(" ---> ") + j
```

Adding a Python Script in an EEM Applet

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **event manager applet** *applet-name*
4. **event** [**tag** *event-tag*] **syslog pattern** *regular-expression*
5. **action** *label* **cli command** *cli-string*
6. **action** *label* **cli command** *cli-string* [**pattern** *pattern-string*]
7. **end**
8. **show event manager policy active**
9. **show event manager history events**

DETAILED STEPS

Procedure

	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	**event manager applet** *applet-name* **Example:** `Device(config)# event manager applet interface_Shutdown`	Registers an applet with the Embedded Event Manager (EEM) and enters applet configuration mode.
Step 4	**event** [**tag** *event-tag*] **syslog pattern** *regular-expression* **Example:** `Device(config-applet)# event syslog pattern "Interface Loopback55, changed state to administratively down"`	Specifies a regular expression to perform the syslog message pattern match.
Step 5	**action** *label* **cli command** *cli-string* **Example:** `Device(config-applet)# action 0.0 cli command "en"`	Specifies the IOS command to be executed when an EEM applet is triggered.

	Command or Action	Purpose
Step 6	**action** *label* **cli command** *cli-string* [**pattern** *pattern-string*] **Example:** `Device(config-applet)# action 1.0 cli command "guestshell run python3 /bootflash/eem_script.py loop55"`	Specifies the action to be specified with the **pattern** keyword. • Specify a regular expression pattern string that will match the next solicited prompt.
Step 7	**end** **Example:** `Device(config-applet)# end`	Exits applet configuration mode and returns to privileged EXEC mode.
Step 8	**show event manager policy active** **Example:** `Device# show event manager policy active`	Displays EEM policies that are executing.
Step 9	**show event manager history events** **Example:** `Device# show event manager history events`	Displays the EEM events that have been triggered.

What to do next

The following example shows how to trigger the Python script configured in the task:

```
Device(config)# interface loopback 55
Device(config-if)# shutdown
Device(config-if)# end
Device#

Mar 13 10:53:22.358 EDT: %SYS-5-CONFIG_I: Configured from console by console
Mar 13 10:53:24.156 EDT: %LINK-5-CHANGED:  Line protocol on Interface Loopback55, changed
state to down
Mar 13 10:53:27.319 EDT: %LINK-3-UPDOWN:  Interface Loopback55, changed state to
administratively down
Enter configuration commands, one per line.  End with CNTL/Z.
Mar 13 10:53:35.38 EDT:  %LINEPROTO-5-UPDOWN: Line protocol on Interface Loopback55, changed
 state to up
*Mar 13 10:53:35.39 EDT %LINK-3-UPDOWN: Interface Loopback55, changed state to up
+++ 10:54:33 edi37(default) exec +++
show ip interface br
Interface              IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0/0   unassigned      YES unset  down                  down
GigabitEthernet0/0/1   unassigned      YES unset  down                  down
GigabitEthernet0/0/2   10.1.1.31       YES DHCP   up                    up
GigabitEthernet0/0/3   unassigned      YES unset  down                  down
GigabitEthernet0       192.0.2.1       YES manual up                    up
Loopback55             198.51.100.1    YES manual up                    up
Loopback66             172.16.0.1      YES manual up                    up
Loopback77             192.168.0.1     YES manual up                    up
Loopback88             203.0.113.1     YES manual up                    up
```

Additional References EEM Python Module

Related Documents

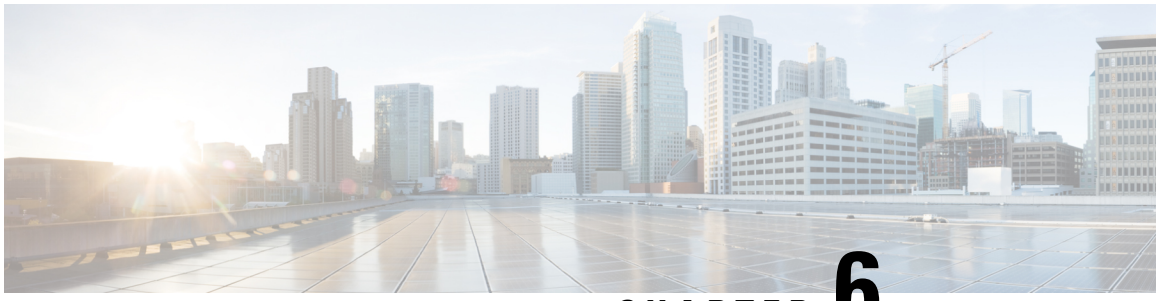
Related Topic	Document Title
EEM configuration	*Embedded Event Manager Configuration Guide*
EEM commands	*Embedded Event Manager Command Reference*
Guest Shell configuration	Guest Shell

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

PART III

Application Hosting



CHAPTER 6

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.

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 11: Feature Information for Application Hosting

Feature Name	Release	Feature Information
Application Hosting	Cisco IOS XE 17.18.1	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. This feature was implemented on Cisco C9350 Series Switches.

Feature Name	Release	Feature Information
Application Hosting: Autotransfer and Auto-Install of Apps from Internal Flash to SSD	Cisco IOS XE 17.18.1	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. This feature was introduced on the Cisco C9350 Series Switches.
Application Hosting: Front-Panel Network Port Access	Cisco IOS XE 17.18.1	Introduces datapath connectivity between the Application Hosting container and the front-panel network ports. Also enables ZTP functionality on the front-panel network. This feature was implemented on Cisco C9350 Series Switches.
Application Hosting: Front-Panel USB Port Access	Cisco IOS XE 17.18.1	Introduces datapath connectivity between the Application Hosting container and the front-panel USB port. This feature was implemented on Cisco C9350 Series Switches.
ERSPAN Support on the AppGigabitEthernet Port	Cisco IOS XE 17.18.1	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 17.18.1	Multicast traffic forwarding is supported on the AppGigabitEthernet interface. Applications can select thenetworks that allow multicast traffic. This feature was implemented on Cisco C9350 Series Switches.
Native Docker Container: Application Auto-Restart	Cisco IOS XE 17.18.1	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. This feature was implemented on Cisco C9350 Series Switches.

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 Cisco C9350 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).
- The front-panel Universal Serial Bus (USB) stick is not supported.

  Cisco C9350 Series Switches support only back-panel Cisco-certified USB SSD.

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 switch 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 C9300 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 C9350 Series Switches

This section describes application-hosting on Cisco C9350 Series Switches.

For application hosting, Cisco C9350 Series Switches support the management interface and front-panel ports.

The USB 3.0 SSD is enabled on Cisco C9350 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 C9350 Series Switches

Front-panel application hosting is enabled on Cisco C9350 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 12: 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.

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 13: 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 C9350 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 C9350 Series Switches.

Application Auto-Restart Scenarios

This section describes various application auto-restart scenarios:

Table 14: 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.

Scenario	Single Media in the Active Device	Media in the Active and Standby Devices
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 C9350 Series Switches

This section describes how application auto-restart works on Cisco C9350 Series Switches in a multimember stack:

On Cisco C9350 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 Switches.

Table 15: Supported Network Types

Network Type	Supported Platform and Release
Management port	Cisco C9350 Series Switches in Cisco IOS XE 17.18.1
Front-panel port (trunk and VLAN)	Cisco C9350 Series Switches in Cisco IOS XE 17.18.1 **Note** Cisco C9350 Series Switches support multiple AppGigabitEthernet ports.
Cisco IOS Network Address Translation (NAT)	Cisco C9350 Series Switches in Cisco IOS XE 17.18.1 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.

Network Type	Supported Platform and Release
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.

Multicast Routing on the AppGigabitEthernet Interface

Multicast traffic forwarding is supported on the AppGigabitEthernet interface. Applications can select the networks that allow multicast traffic. Multicast traffic forwarding is enabled through the IOS CLI and the package.yaml file.

If a platform supports multicast routing; but the network does not support multicast, an activation error message is displayed. On some platforms, to enable multicast routing, you must disable IGMP snooping.

If multicast traffic forwarding is enabled on an app in a network, you cannot activate another app on the same network, without enabling multicast on that app.

Note Multicast traffic forwarding is not supported on the management interface. However; when the management interface is used as an external AppGigabitEthernet interface, multicast traffic forwarding can be enabled on the interface.

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

Procedure

	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.
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 17.18.1 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

Procedure

	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.

	Command or Action	Purpose
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 17.18.1, 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 17.18.1 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

Procedure

	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.
Step 7	**app-hosting appid** *name* **Example:** `Device(config)# app-hosting appid iox_app`	Configures an application and enters application-hosting configuration mode.

	Command or Action	Purpose
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

Procedure

	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	**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.

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

Procedure

	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.

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

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

Procedure

	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.

	Command or Action	Purpose
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

Procedure

	Command or Action	Purpose
Step 1	**enable**	Enables privileged EXEC mode.

	Command or Action	Purpose
	Example: `Device> enable`	• 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

Procedure

	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 C9350 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 C9350 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.
Step 7	**app-hosting appid** *name* **Example:** `Device(config)# app-hosting appid iox_app`	Configures an application and enters application-hosting configuration mode.

	Command or Action	Purpose
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

Procedure

	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.
Step 6	**memory** *memory* **Example:** `Device(config-app-resource-profile-custom)# memory 2048`	Changes the default memory allocation.
Step 7	**vcpu** *number* **Example:**	Changes the virtual CPU (vCPU) allocation for the application.

	Command or Action	Purpose
	`Device(config-app-resource-profile-custom)# vcpu 2`	
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

Procedure

	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.

	Command or Action	Purpose
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

Procedure

	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.

Enabling Multicast Routing on the AppGigabitEthernet Interface

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **app-hosting appid** *name*
4. **app-vnic AppGigabitEthernet trunk**

5. **vlan** *vlan-ID* **guest-interface** *guest-interface-number*
6. **guest-ipaddress***ip-address* **netmask** *netmask*
7. **multicast**
8. **exit**
9. **exit**
10. **app-default-gateway** *ip-address* **guest-interface** *network-interface*
11. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enters 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	**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 5	**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.
Step 6	**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`	Configures a static IP address.
Step 7	**multicast** **Example:** `Device(config-config-app-hosting-vlan-access-ip)# multicast`	Enables multicast traffic forwarding on the AppGigabitEthernet interface.

	Command or Action	Purpose
Step 8	**exit** **Example:** `Device(config-config-app-hosting-vlan-access-ip)# exit`	Exits application-hosting VLAN-access IP configuration mode and returns to application-hosting trunk-configuration mode
Step 9	**exit** **Example:** `Device(config-config-app-hosting-trunk)# exit`	Exits application-hosting trunk-configuration mode and returns to application-hosting configuration mode.
Step 10	**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 11	**end** **Example:** `Device(config-app-hosting)# end`	Exits application-hosting 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

Procedure

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
```

Verifying the Application-Hosting Configuration

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

- **show iox-service**

  Displays the status of all the Cisco IOx services.

```
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
```

- **show app-hosting detail**

  Displays detailed information about the application.

```
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
```

- **show app-hosting device**

  Displays information about the USB device.

```
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
```

- **show app-hosting list**

  Displays the list of applications and their status.

```
Device# show app-hosting list

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

- **show interfaces trunk**

  Displays trunk interface information.

```
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
```

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

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

```
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 MacUnderrun frames
0 Pause frames
0 Cos 0 Pause frames
0 Cos 1 Pause frames
0 Cos 2 Pause frames
0 Cos 3 Pause frames
0 Cos 4 Pause frames
0 Cos 5 Pause frames
0 Cos 6 Pause frames
0 Cos 7 Pause frames
0 Oam frames
0 Oam frames
0 Minimum size frames
0 65 to 127 byte frames
0 128 to 255 byte frames
0 256 to 511 byte frames
0 512 to 1023 byte frames
0 1024 to 1518 byte frames
0 1519 to 2047 byte frames
0 2048 to 4095 byte frames
0 4096 to 8191 byte frames
0 8192 to 16383 byte frames
0 16384 to 32767 byte frame
0 > 32768 byte frames
0 Late collision frames
0 Excess Defer frames
0 Good (1 coll) frames
0 Good (>1 coll) frames
0 Deferred frames
0 Gold frames dropped
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
```

```
0 IpgViolation frames
0 MacOverrun frames
0 Pause frames
0 Cos 0 Pause frames
0 Cos 1 Pause frames
0 Cos 2 Pause frames
0 Cos 3 Pause frames
0 Cos 4 Pause frames
0 Cos 5 Pause frames
0 Cos 6 Pause frames
0 Cos 7 Pause frames
0 OamProcessed frames
0 OamDropped frames
0 Minimum size frames
0 65 to 127 byte frames
0 128 to 255 byte frames
0 256 to 511 byte frames
0 512 to 1023 byte frames
0 1024 to 1518 byte frames
0 1519 to 2047 byte frames
0 2048 to 4095 byte frames
0 4096 to 8191 byte frames
0 8192 to 16383 byte frames
0 16384 to 32767 byte frame
0 > 32768 byte frames
0 SymbolErr frames
0 Collision fragments
0 ValidUnderSize frames
0 InvalidOverSize frames
0 ValidOverSize frames
0 FcsErr frames
```

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 17.18.1 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

This example shows how to configure 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
```

Example: Enabling Multicast Routing on the AppGigabitEthernet Interface

```
Device> enable
Device# configure terminal
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 192.168.0.2 netmask
255.255.255.0
Device(config-config-app-hosting-vlan-access-ip)# multicast
Device(config-config-app-hosting-vlan-access-ip)# exit
Device(config-config-app-hosting-trunk)# exit
Device(config-app-hosting)# app-default-gateway 172.19.0.23 guest-interface 0
Device(config-app-hosting)# end
```

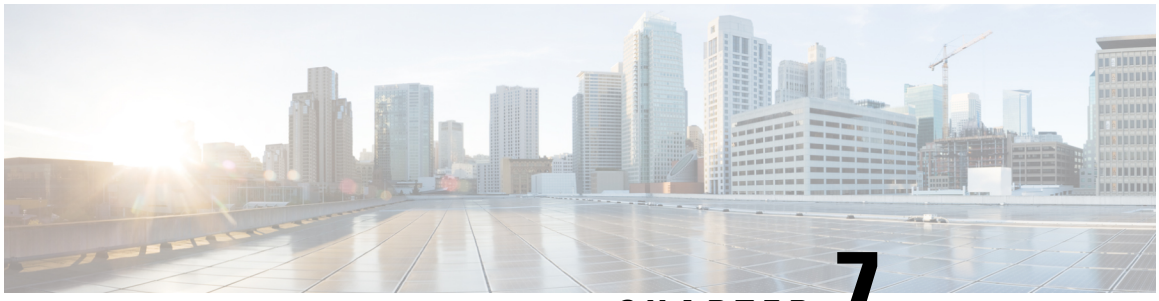
Additional References

Related Documents

Related Topic	Document Title
Programmability commands	*Programmability Command Reference*
DevNet	https://developer.cisco.com/docs/app-hosting/
USB3.0 SSD on Cisco C9350 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



CHAPTER 7

ThousandEyes Enterprise Agent

ThousandEyes Enterprise Agent is an enterprise network-monitoring tool that provides you an end-to-end view across networks and services that impact your business. This module describes how to download and install the Enterprise Agent.

Feature Information for ThousandEyes Enterprise Agent

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 16: Feature Information for Application Hosting

Feature Name	Release	Feature Information
ThousandEyes Integration	Cisco IOS XE 17.18.1	ThousandEyes is a cloud-ready, enterprise network-monitoring tool that provides an end-to-end view across networks and services. • In Cisco IOS XE 17.18.1, this feature was implemented on Cisco Catalyst 9350 Series Switches.

Prerequisites for the ThousandEyes Enterprise Agent

- The ThousandEyes Enterprise Agent image available at the ThousandEyes site must be signed by the same certificate authority (CA) that is used by www.cisco.com for HTTPS downloads; without an username or a password.
- Installation of the Enterprise Agent requires Internet connectivity, or a proxy server. For more information, see the *ThousandEyes documentation* at: https://docs.thousandeyes.com/product-documentation/enterprise-agents.
- The Enterprise Agent application can only be used after the user's license privileges are validated.
- Only Docker-based applications are supported.
- 1:1 stack mode is a must for ThousandEyes Stateful Switchover (SSO) support.

  1:1 mode is when the active and standby roles are assigned to specific devices in a stack. This overrides the traditional N+1 role selection algorithm, where any device in the stack can be the active or the standby.

Information About ThousandEyes Enterprise Agent

ThousandEyes Enterprise Agent Overview

ThousandEyes Enterprise Agent is an enterprise network-monitoring tool that provides you an end-to-end view across networks and services that impact your business. It monitors the network traffic paths across internal, external, carrier, and Internet networks in real time, to provide network performance data. Enterprise Agents are commonly installed in branch sites and data centers to provide a detailed understanding of WAN and Internet connectivity.

In previous Cisco IOS XE releases, ThousandEyes was supported as a third-party Kernel-based Virtual Machine (KVM) appliance on the SSD.

In Cisco IOS XE 17.18.1, a new version of the ThousandEyes Enterprise Agent, Version 3.0 is introduced. This is an embedded Docker-based application that runs on Cisco devices using the application-hosting capability. The Enterprise Agent is available on both the SSD and bootflash, and it supports all tests except browser tests (page load and transaction). The browser tests are available in Cisco IOS XE 17.18.1 and later releases with Enterprise Agent Version 4.0.

The ThousandEyes Enterprise Agent provides the following:

- Benchmarking the performance of networks and applications.
- Detailed hop-by-hop metrics.
- End-to-end path visualization from branch or campus to data center or cloud.
- Outage detection and resolution.
- User-experience analysis.
- Visualization of the traffic-flow pattern.

ThousandEyes Enterprise Agent Version 4.0 available in Cisco IOS XE 17.18.1, supports the following additional features that are not available in the ThousandEyes Agent Version 3.0:

- BrowserBot support when back-panel SSD is available.
- DNAC app icon and description.
- Docker health monitoring.
- The **app-hosting upgrade URL** command to upgrade the ThousandEyes Enterprise Agent.

Resources Required for the ThousandEyes Enterprise Agent

This table describes the required resources for installing the ThousandEyes Enterprise Agent:

Table 17: Resources Required for the ThousandEyes Enterprise Agent

App Media	Maximum Resource	Supported Release
SSD **Note** Only 120G SSD is supported.	• CPU: 2 vCPU • Memory: 2G RAM • Storage: No limit on SSDs	Cisco IOS XE 17.18.1 • Cisco Catalyst 9350 Series Switches
Flash	• CPU: 2 vCPU • Memory: 2G RAM • Storage: 1G for persistent logging by applications, out of the 4G partition in the flash file system. The storage is shared with the IOx metadata.	Cisco IOS XE 17.18.1 • Cisco Catalyst 9350 Series Switches

In Cisco IOS XE 17.18.1, add-on mode is supported on Cisco Catalyst 9350 Series Switches.

ThousandEyes Enterprise Agent Download

BrownField and GreenField are two types of ThousandEyes Enterprise Agents. For existing devices, you can download the Brownfield version from the ThousandEyes website. However, new devices are shipped with the Greenfield application loaded in the bootflash.

This table lists the download options available for the agents.

Table 18: ThousandEyes Enterprise Agent Download Options

BrownField	GreenField
• Download the file from the *Installing Enterprise Agents on Cisco Switches with Docker* page. The file is signed by the same certificate authority (CA) that is used by www.cisco.com for HTTPS downloads; without an username or a password. • Use the **install** command to download and deploy the application.	• Available in the bootflash under the */apps* folder. Shipped with the device. • Use the **install** command to download and deploy the application.

This section describes the maximum resources required for the agent to run:

- CPU: 2 vCPUs
- Memory: 2G
- Storage: 1G for persistent logging by applications, out of the 4G partition in the flash file system. This storage is shared by the IOx metadata.
- Media storage:
  - 120G SSD for Cisco Catalyst 9350 Series Switches in Cisco IOS XE 17.18.1.

After the download of the Enterprise Agent, it initiates a call to create a secure channel to the ThousandEyes cloud-based portal that provides the required application configuration, and gathers application data. The link to the TE portal is https://app.thousandeyes.com.

ThousandEyes BrowserBot

ThousandEyes Enterprise Agent Version 4.0 provides a BrowserBot for transaction scripting test. The BrowserBot is a component of the Enterprise Agent that manages page load and transaction tests. The BrowserBot allows you to enable customized JavaScript tests which mimic the actions of your web browser on the ThousandEyes Cloud Portal. To protect the host operating system from any errant JavaScript operations, the ThousandEyes agent creates sandbox containers to run your JavaScript.

If an unrestricted disk is used by the application, the ThousandEyes agent will dynamically install the BrowserBot package during initialization that permits portal transaction scripting tests to be configured.

Note The BrowserBot support is not available in ThousandEyes Agent Version 3.0.

BrowserBot consumes a large amount of hardware resources. 2GB system memory and 2 VCPU loads are the maximum IOx system memory and CPU load allocated for all IOx apps. To allow multiple apps to concurrently run in the bootflash, lower the default package.yaml BrowserBot resources before activating the agent. Use the **app-resource profile custom** command to override the default package.yaml settings:

- CPU:1850 CPU units (1/4 VCPU)
- Memory: 500MB

For more information on transaction scripting, see the following links:

- https://docs.thousandeyes.com/product-documentation/tests/transaction-scripting-guide
- https://docs.thousandeyes.com/product-documentation/tests/transaction-scripting-reference

For examples of transaction scripting, see https://github.com/thousandeyes/transaction-scripting-examples.

ThousandEyes Agent Upgrade and Downgrade

ThousandEyes Agent Upgrade

Agent 4.0 is available in Cisco IOS XE 17.18.1, and the agent auto-upgrade updates to the latest Agent 4.0 binary on startup. No upgrade is available for Agent 4.0 at present.

Application upgrades can be done using the following methods:

- ThousandEyes agent auto-upgrade: Happens automatically when an application starts up. The agent binary within the running container is upgraded, but the application package is not upgraded.
- Using the **app-hosting upgrade** command.
- DNAC app upgrades.

ThousandEyes Agent Downgrade

Agent 4.0 available in Cisco IOS XE 17.18.1 can be downgraded to Agent 3.0 available. No other downgrade is possible.

When downgrading, if the application does not come to the same state as the previous release, deactivate or uninstall the application, and install or restart it.

How to Install the ThousandEyes Enterprise Agent

To install the Enterprise Agent, follow these steps:

1. Configure IOx. For more information, see the "Enabling Ciso IOx" section.
2. Configure AppHosting.
3. Configure the AppGigabitEthernet port.
4. Install the ThousandEyes Enterprise Agent.

Configuring AppHosting for the ThousandEyes Enterprise Agent

SUMMARY STEPS

1. **enable**
2. **configure terminal**
3. **app-hosting appid** *application-name*
4. **app-vnic AppGigabitEthernet trunk**

5. **vlan** *vlan-ID* **guest-interface** *guest-interface-number*
6. **guest-ip** *ip-address* **netmask** *netmask*
7. **exit**
8. **exit**
9. **app-default-gateway** *ip-address* **guest-interface** *network-interface*
10. **nameserver#** *ip-address*
11. **app-resource docker**
12. **run-opts** *options*
13. **prepend-pkg-opts**
14. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enters privileged EXEC mode.
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 appid 1keyes`	Configures an application and enters application-hosting configuration mode.
Step 4	**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 5	**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.
Step 6	**guest-ip** *ip-address* **netmask** *netmask* **Example:** `Device(config-config-app-hosting-vlan-access-ip)# guest-ipaddress 172.19.0.24 netmask 255.255.255.0`	Configures a static IP address for the guest interface.

	Command or Action	Purpose
Step 7	**exit** **Example:** `Device(config-config-app-hosting-vlan-access-ip)# exit`	Exits application hosting VLAN-access IP configuration mode and returns to application-hosting trunk-configuration mode.
Step 8	**exit** **Example:** `Device(config-config-app-hosting-trunk)# exit`	Exits application-hosting trunk-configuration mode and returns to application hosting configuration mode.
Step 9	**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 10	**nameserver#** *ip-address* **Example:** `Device(config-app-hosting)# name-server0 10.2.2.2`	Configures the DNS server.
Step 11	**app-resource docker** **Example:** `Device(config-app-hosting)# app-resource docker`	Enters application-hosting docker-configuration mode to specify application resource updates.
Step 12	**run-opts** *options* **Example:** `Device(config-app-hosting-docker)# run-opts 1 "-e TEAGENT_ACCOUNT_TOKEN=[account-token]"`	Specifies the Docker run time options.
Step 13	**prepend-pkg-opts** **Example:** `Device(config-app-hosting-docker)# prepend-pkg-opts`	Merges the package options with the Docker runtime options. • Any duplicate variable is overwritten.
Step 14	**end** **Example:** `Device(config-app-hosting-docker)# end`	Exits application-hosting docker-configuration mode and returns to privileged EXEC mode.

Configuring AppGigabitEthernet Interface for the ThousandEyes Enterprise Agent

SUMMARY STEPS

1. **enable**
2. **configure terminal**

3. **interface appgigabitethernet** *number*
4. **switchport trunk allowed vlan** *vlan-ID*
5. **switchport mode trunk**
6. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enters privileged EXEC mode.
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	**end** **Example:** `Device(config-if)# end`	Exits interface configuration mode and returns to privileged EXEC mode.

Installing the ThousandEyes Enterprise Agent

Before you begin

You can install the ThousandEyes Enterprise Agent either from the URL given below or from the flash filesystem.

SUMMARY STEPS

1. **enable**
2. **app-hosting install appid** *application-name* **package** *package-path*
3. **app-hosting start appid** *application-name*
4. **end**

DETAILED STEPS

Procedure

	Command or Action	Purpose
Step 1	**enable** **Example:** `Device> enable`	Enters privileged EXEC mode.

Step 2	**app-hosting install appid** *application-name* **package** *package-path*
	Example:
	`Device# app-hosting install 1keyes https://downloads.thousandeyes.com/enterprise-agent/thousandeyes-enterprise-agent-3.0.cat9k.tar`
	Or
	`Device# app-hosting install appid 1keyes package flash:/apps/[greenfield-app-tar]`
		Installs an application from the specified location.

Step 3	**app-hosting start appid** *application-name* **Example:** `Device# app-hosting start appid 1keyes`	(Optional) Starts the application.
Step 4	**end** **Example:** `Device# end`	Exits application hosting configuration mode and returns to privileged EXEC mode.

The following is sample output from the **show app-hosting list** command:

```
Device# show app-hosting list

App id                                   State
---------------------------------------------------------
1keyes                                   RUNNING
```

Configuration Examples for ThousandEyes Enterprise Agent

Example: Installing ThousandEyes Enterprise Agent

This example shows how to:

- Enable IOx.
- Configure AppHosting.
- Configure the AppGigabitEthernet port.
- Install the ThousandEyes Enterprise Agent.

The following example shows how to enable IOx:

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

The following example shows how to configure AppHosting:

```
Device> enable
Device# configure terminal
Device(config)# app-hosting appid appid 1keyes
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 172.19.0.24
netmask 255.255.255.0
Device(config-config-app-hosting-vlan-access-ip)# exit
Device(config-config-app-hosting-trunk)# exit
Device(config-app-hosting)# app-default-gateway 172.19.0.23
guest-interface 0
Device(config-app-hosting)# name-server0 10.2.2.2
Device(config-app-hosting)# app-resource docker
Device(config-app-hosting-docker)# run-opts 1
"-e TEAGENT_ACCOUNT_TOKEN=[account-token]"
Device(config-app-hosting-docker)# prepend-pkg-opts
Device(config-app-hosting-docker)# end
```

The following example shows how to configure the Appgigabitethernet interface:

```
Device> enable
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)# end
```

The following example shows how to install the ThousandEyes Enterprise Agent.

Note You can either download the BrownField application from the ThousandEyes website or install the prepackaged Greenfield application from the flash filesystem.

```
Device> enable
Device# Device# app-hosting install 1keyes https://downloads.thousandeyes.com/
enterprise-agent/thousandeyes-enterprise-agent-3.0.cat9k.tar
OR
Device# app-hosting install appid 1keyes package flash:/apps/[greenfield-app-tar]
Device# app-hosting start appid 1keyes
Device# end
```

Sample Configuration for ThousandEyes Enterprise Agent

The following is sample output from the **show app-hosting detail** command:

```
Device# show app-hosting detail

App id                 : 1keyes
Owner                  : iox
State                  : RUNNING
Application
  Type                 : docker
  Name                 : thousandeyes/enterprise-agent
  Version              : 3.0
  Description          :
  Path                 : flash:thousandeyes-enterprise-agent-3.0.cat9k.tar
  URL Path             :
Activated profile name : custom

Resource reservation
  Memory               : 0 MB
  Disk                 : 1 MB
  CPU                  : 1850 units
  CPU-percent          : 25 %
  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:c0:a2:ab
   IPv4 address        : 10.0.0.110
   IPv6 address        : ::
   Network name        : mgmt-bridge-v14


Docker
------
Run-time information
  Command              :
  Entry-point          : /sbin/my_init
```

```
  Run options in use   : -e TEAGENT_ACCOUNT_TOKEN=TOKEN_NOT_SET --hostname=$(SYSTEM_NAME)
--cap-add=NET_ADMIN
                         --mount type=tmpfs,destination=/var/log/agent,tmpfs-size=140m
                       --mount type=tmpfs,destination=/var/lib/te-agent/data,tmpfs-size=200m

                          -v $(APP_DATA)/data:/var/lib/te-agent -e TEAGENT_PROXY_TYPE=DIRECT

                         -e TEAGENT_PROXY_LOCATION= -e TEAGENT_PROXY_USER= -e
TEAGENT_PROXY_AUTH_TYPE=
                         -e TEAGENT_PROXY_PASS= -e TEAGENT_PROXY_BYPASS_LIST= -e
TEAGENT_KDC_USER=
                         -e TEAGENT_KDC_PASS= -e TEAGENT_KDC_REALM= -e TEAGENT_KDC_HOST=
-e TEAGENT_KDC_PORT=88
                         -e TEAGENT_KERBEROS_WHITELIST= -e TEAGENT_KERBEROS_RDNS=1 -e
PROXY_APT=
                         -e APT_PROXY_USER= -e APT_PROXY_PASS= -e APT_PROXY_LOCATION= -e
TEAGENT_AUTO_UPDATES=1
                         -e TEAGENT_ACCOUNT_TOKEN=r3d29srpebr4j845lvnamwhswlori2xs
                         --hostname=cat9k-9300-usb --memory=1g
  Package run options  : -e TEAGENT_ACCOUNT_TOKEN=TOKEN_NOT_SET --hostname=$(SYSTEM_NAME)
--cap-add=NET_ADMIN
                         --mount type=tmpfs,destination=/var/log/agent,tmpfs-size=140m
                       --mount type=tmpfs,destination=/var/lib/te-agent/data,tmpfs-size=200m

                          -v $(APP_DATA)/data:/var/lib/te-agent -e TEAGENT_PROXY_TYPE=DIRECT

                         -e TEAGENT_PROXY_LOCATION= -e TEAGENT_PROXY_USER= -e
TEAGENT_PROXY_AUTH_TYPE=
                         -e TEAGENT_PROXY_PASS= -e TEAGENT_PROXY_BYPASS_LIST= -e
TEAGENT_KDC_USER=
                         -e TEAGENT_KDC_PASS= -e TEAGENT_KDC_REALM= -e TEAGENT_KDC_HOST=
                         -e TEAGENT_KDC_PORT=88 -e TEAGENT_KERBEROS_WHITELIST= -e
TEAGENT_KERBEROS_RDNS=1
                         -e PROXY_APT= -e APT_PROXY_USER= -e APT_PROXY_PASS= -e
APT_PROXY_LOCATION=
                         -e TEAGENT_AUTO_UPDATES=1
Application health information
  Status               : 0
  Last probe error     :
  Last probe output    :
```

The following sample output from the **show running-configuration** command displays the static IP address configuration:

```
Device# show running-config | section app-hosting

app-hosting appid 1keyes
 app-vnic AppGigabitEthernet trunk
  vlan 14 guest-interface 0
   guest-ipaddress 10.0.0.110 netmask 255.255.255.0
app-default-gateway 10.0.0.1 guest-interface 0
app-resource docker
  prepend-pkg-opts
  run-opts 1 "-e TEAGENT_ACCOUNT_TOKEN=r3d29srpebr4j845lvnamwhswlori2xs"
  run-opts 2 "--hostname=cat9k-9300-usb --memory=1g"
name-server0 10.0.0.1
start
```

The following sample output from the **show running-configuration** command displays the static IP address configuration and the proxy server information:

```
Device# show running-config | section app-hosting

app-hosting appid 1keyes
 app-vnic AppGigabitEthernet trunk
  vlan 14 guest-interface 0
   guest-ipaddress 172.27.0.137 netmask 255.240.0.0
 app-default-gateway 172.27.0.129 guest-interface 0
 app-resource docker
  run-opts 1 "-e TEAGENT_ACCOUNT_TOKEN=r3d29srpebr4j845lvnamwhswlori2xs"
 run-opts 3 "-e TEAGENT_PROXY_TYPE=STATIC"
 run-opts 4 "-e TEAGENT_PROXY_LOCATION='proxy-wsa.esl.cisco.com:80'"
 prepend-pkg-opts
 name-server0 172.16.0.2
 start
```

The following is sample output from running the app-resource Docker package merged with the Docker runtime options:

```
// Example of "prepend-package-opts" merging
app-hosting appid TEST
app-vnic management guest-interface 3
app-resource docker
prepend-package-opts !!!
run-opts 1 "--entrypoint '/bin/sleep 1000000'"
run-opts 2 "-e TEST=1 "

# Specify runtime and startup
startup:
runtime_options: "--env MYVAR2=foo --cap-add=NET_ADMIN"

Merged docker run-opts passed to CAF's activation payload:
{"auto_deactivate": false, "resources": {"profile": "custom", "cpu":
"1000", "memory": "1024", "rootfs_size": "0", "vcpu": 1, "disk": 10,"network":
[{"interface-name": "eth3", "network-name": "mgmt-bridge100"}, {"interface-name":
"eth4", "network-type": "vlan", "mode": "static", "ipv4": {"ip": "10.2.0.100",
"prefix": "24", "default": false, "gateway": "" },"network-info": { "vlan-id": "10" },
"mac_forwarding": "no", "mirroring": "no"}, {"interface-name": "eth0",
"network-type": "vlan", "network-info": { "vlan-id": "12" }, "mac_forwarding": "no",
"mirroring": "no"}, {"interface-name": "eth2", "network-type": "vlan", "networkinfo":
{"vlan-id": "22" }, "mac_forwarding": "no", "mirroring": "no"},
{"interface-name
": "eth1", "network-type": "vlan", "network-info": {"vlan-id": "all" },
"mac_forwarding": "no", "mirroring": "no"}]},

"startup":{"runtime_options":"--env MYVAR2=foo --cap-add=NET_ADMIN --
entrypoint'/bin/sleep 1000000' -e TEST=1"}}

// Example of no "prepend-package-opts" which is the current behavior since
16.12 where pkg.yml default runoptions are ignored.
app-hosting appid TEST
app-vnic management guest-interface 3
app-resource docker !!!
run-opts 1 "--entrypoint '/bin/sleep 1000000'"
run-opts 2 "-e TEST=1 "

# Specify runtime and startup
startup:
runtime_options: "--env MYVAR2=foo --cap-add=NET_ADMIN"

Merged docker run-opts passed to CAF's activation payload:
{"auto_deactivate": false, "resources": {"profile": "custom", "cpu":
"1000", "memory": "1024", "rootfs_size": "0", "vcpu": 1, "disk": 10,"network":
[{"interface-name": "eth3", "network-name": "mgmt-bridge100"}, {"interface-name":
"eth4", "network-type": "vlan", "mode": "static", "ipv4": {"ip": "10.2.0.100",
```

```
"prefix": "24", "default": false, "gateway": "" },"network-info": { "vlan-id": "10" },
"mac_forwarding": "no", "mirroring": "no"}, {"interface-name": "eth0",
"network-type": "vlan", "network-info": { "vlan-id": "12" }, "mac_forwarding": "no",
"mirroring": "no"}, {"interface-name": "eth2", "network-type": "vlan", "networkinfo":
{"vlan-id": "22" }, "mac_forwarding": "no", "mirroring": "no"},
{"interface-name": "eth1", "network-type": "vlan", "network-info": {"vlan-id": "all" },
"mac_forwarding": "no", "mirroring": "no"}]},

"startup":{"runtime_options":"--entrypoint '/bin/sleep 1000000' -e
TEST=1"}}


// Config 1 : default behavior when "app-resource docker" is not
configured.
app-hosting appid TEST
app-vnic management guest-interface 3

// Config 2: no docker run-opts specified
app-hosting appid TEST
app-vnic management guest-interface 3
app-resource docker
prepend-package-opts
```

Additional References

Related Topic	Document Title
ThousandEyes URL	https://app.thousandeyes.com

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