Cisco Devnet Automation journey kickstart

Getting your basics right

Siddesh Vinod Nayak
Customer Delivery Architect,CCIE-SP

DEVNET-1714

Cisco Webex App

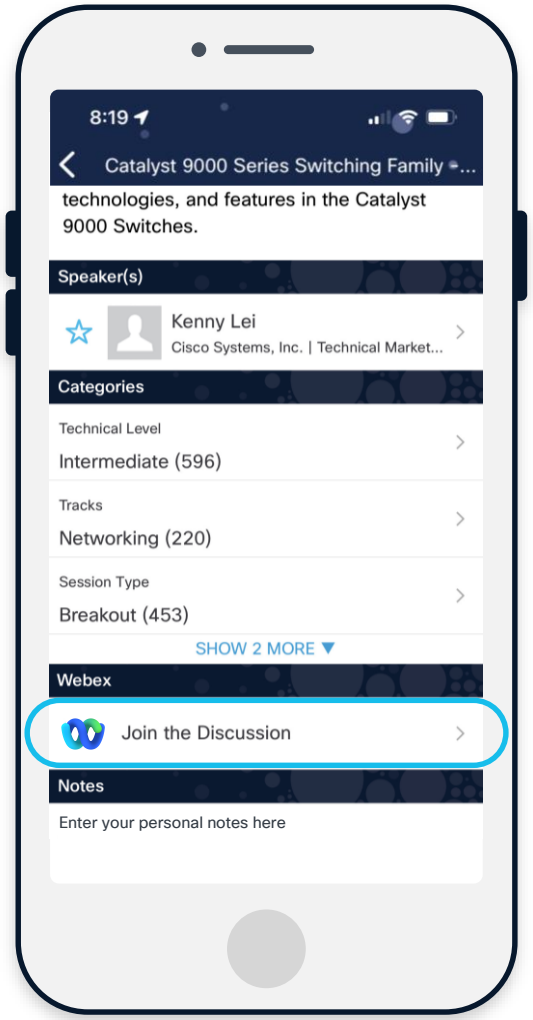
Questions?

Use Cisco Webex App to chat
with the speaker after the session

How

1. Find this session in the Cisco Live Mobile App
2. Click “Join the Discussion”
3. Install the Webex App or go directly to the Webex space
4. Enter messages/questions in the Webex space

Webex spaces will be moderated by the speaker until June 13, 2025.



https://ciscolive.ciscoevents.com/**#DEVNET-1714**

© 2025 Cisco and/or its affiliates. All rights reserved.

Agenda

01 Automation essentials and core concepts.

02 Data encoding formats and Yang modelling.

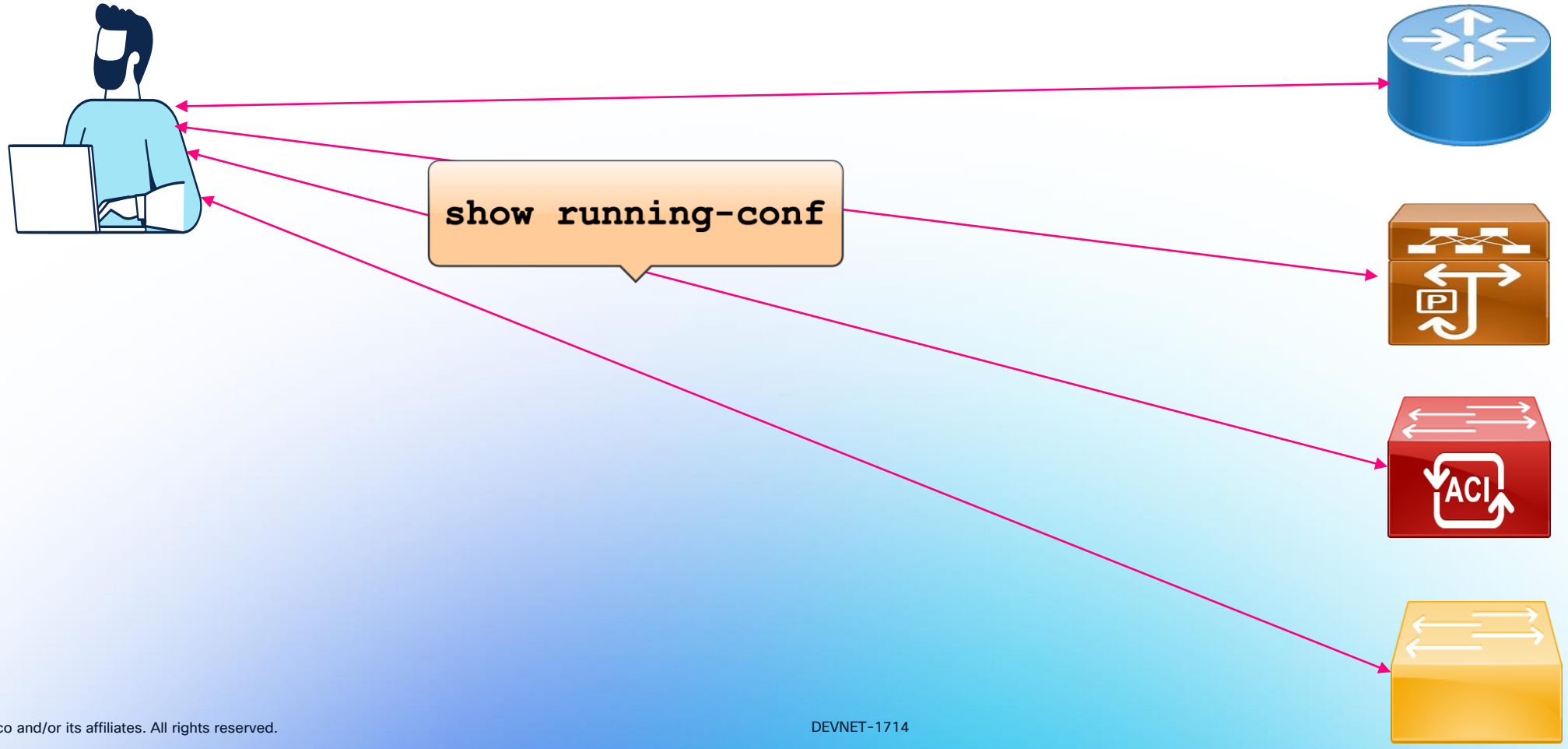
03 Automation – Netconf,RestConf &Rest API.

04 Code check-in process using GIT.

05 Navigating Cisco DevNet documentation.

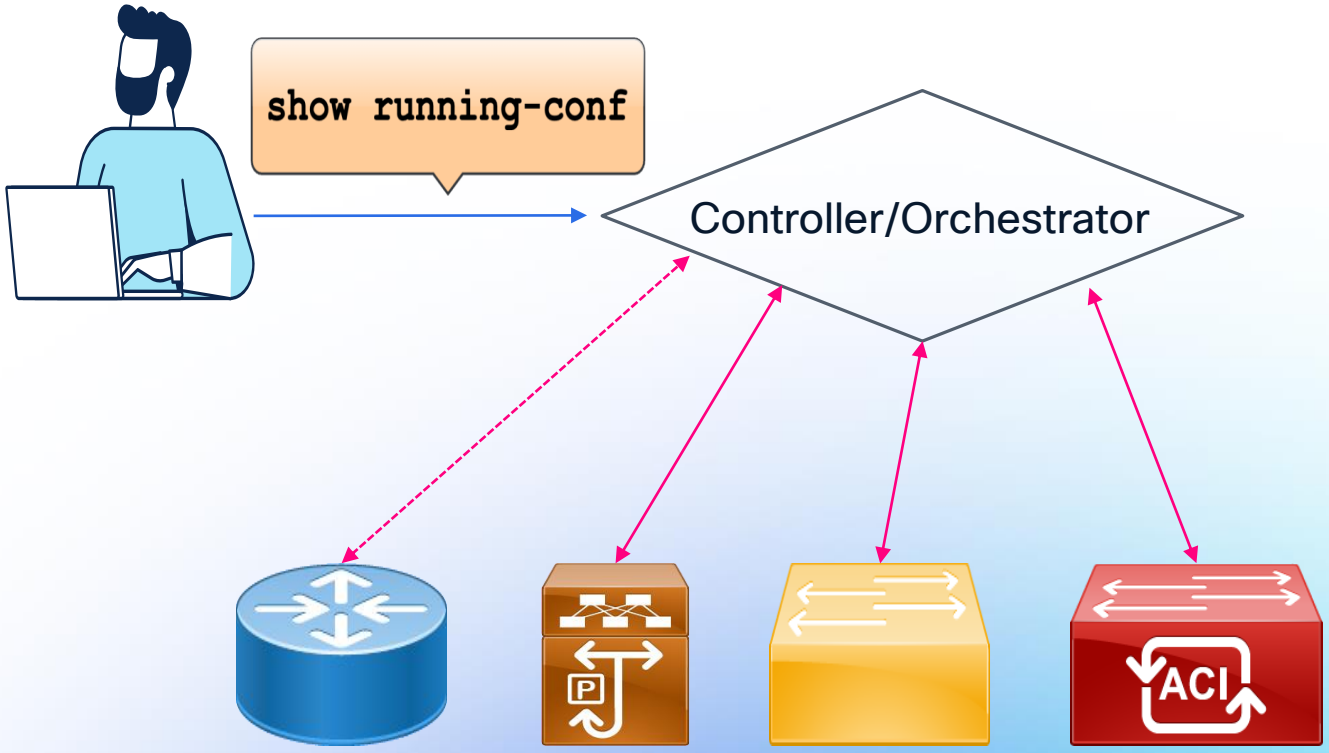
© 2025 Cisco and/or its affiliates. All rights reserved.

Controller-Level vs Device-Level Management and Programmability



© 2025 Cisco and/or its affiliates. All rights reserved.

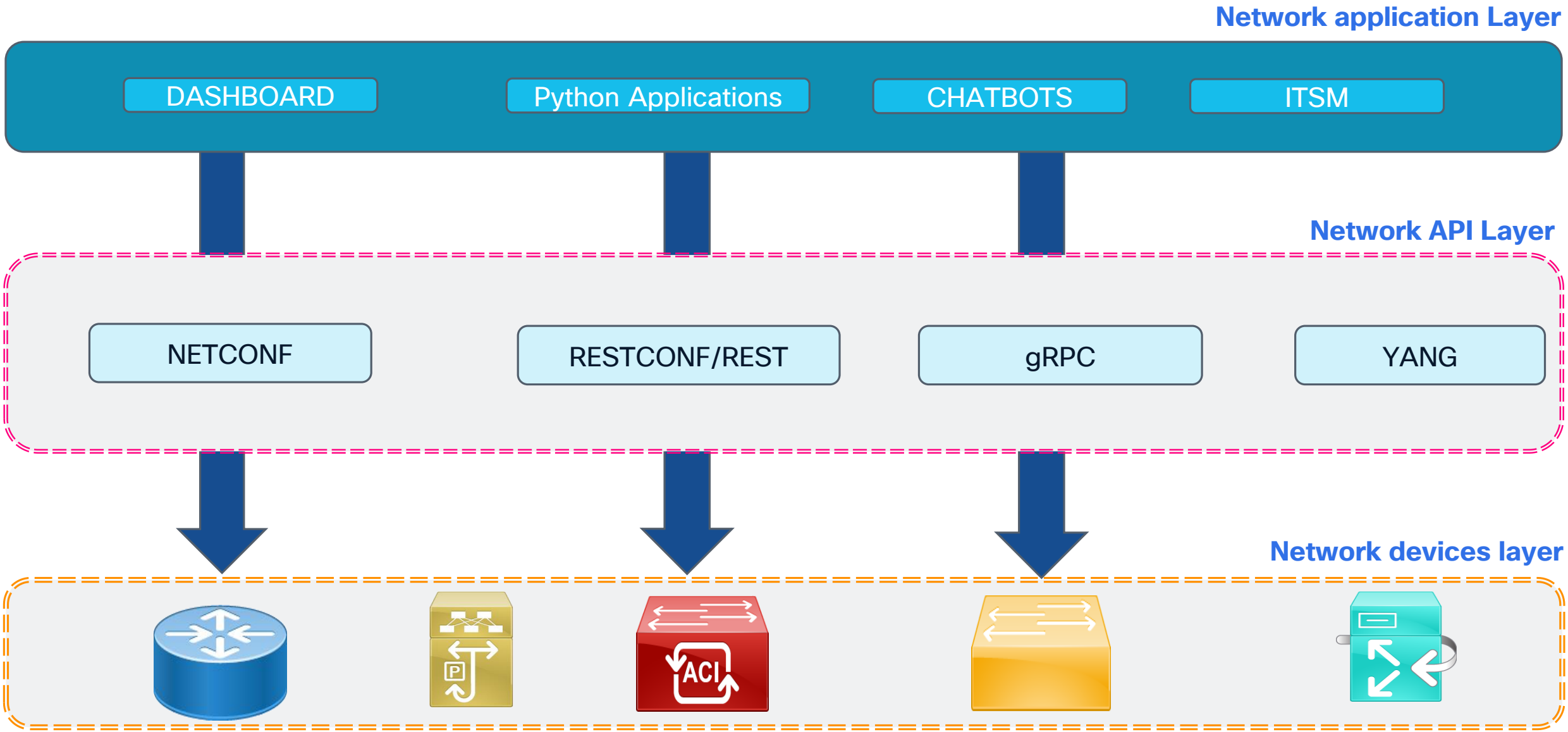
Controller-Level vs Device-Level Management and Programmability



© 2025 Cisco and/or its affiliates. All rights reserved.

Network automation key components

Network Automation key components



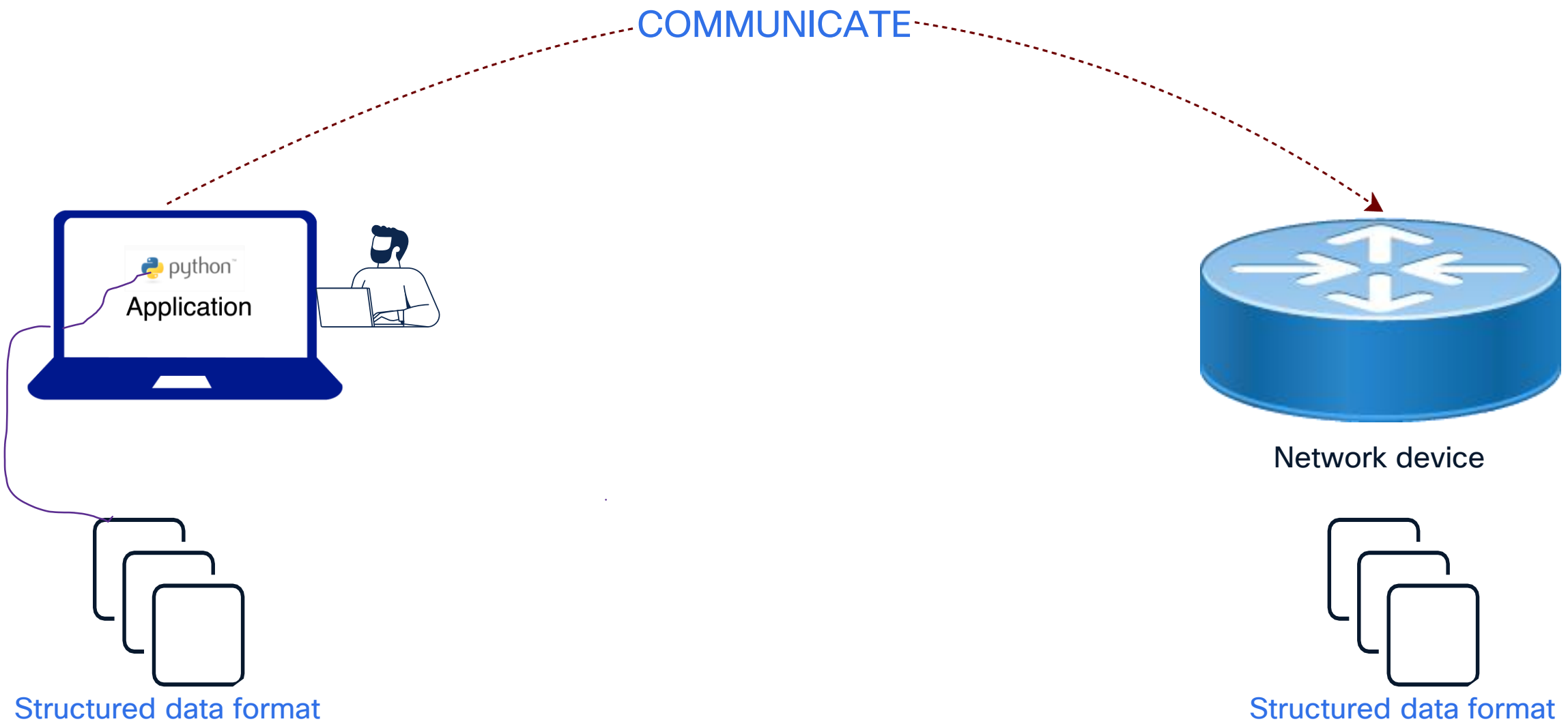
© 2025 Cisco and/or its affiliates. All rights reserved.

A common language, standard data encoding format and identify each other.

– *Makes network automation possible*

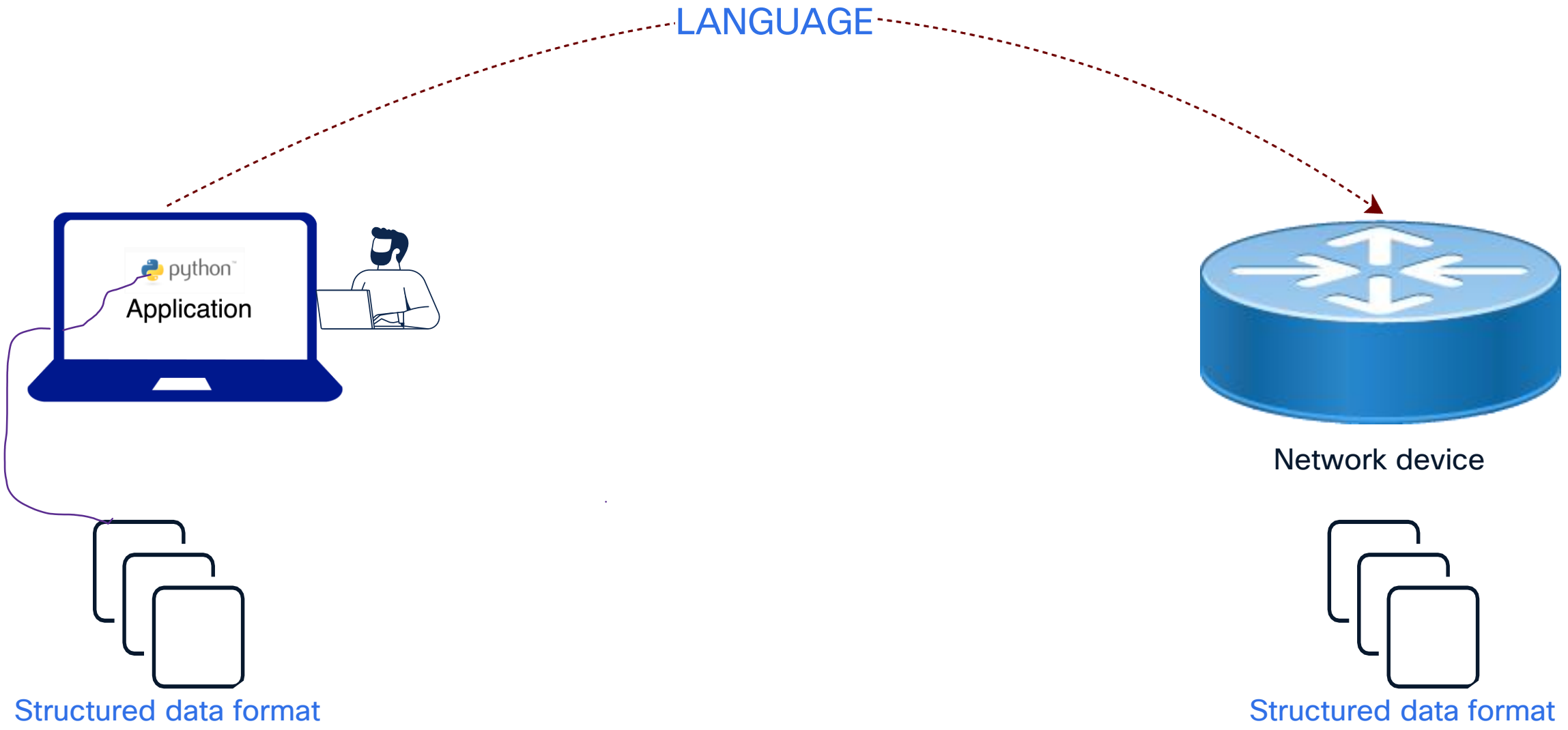
© 2025 Cisco and/or its affiliates. All rights reserved.

Communication with network devices



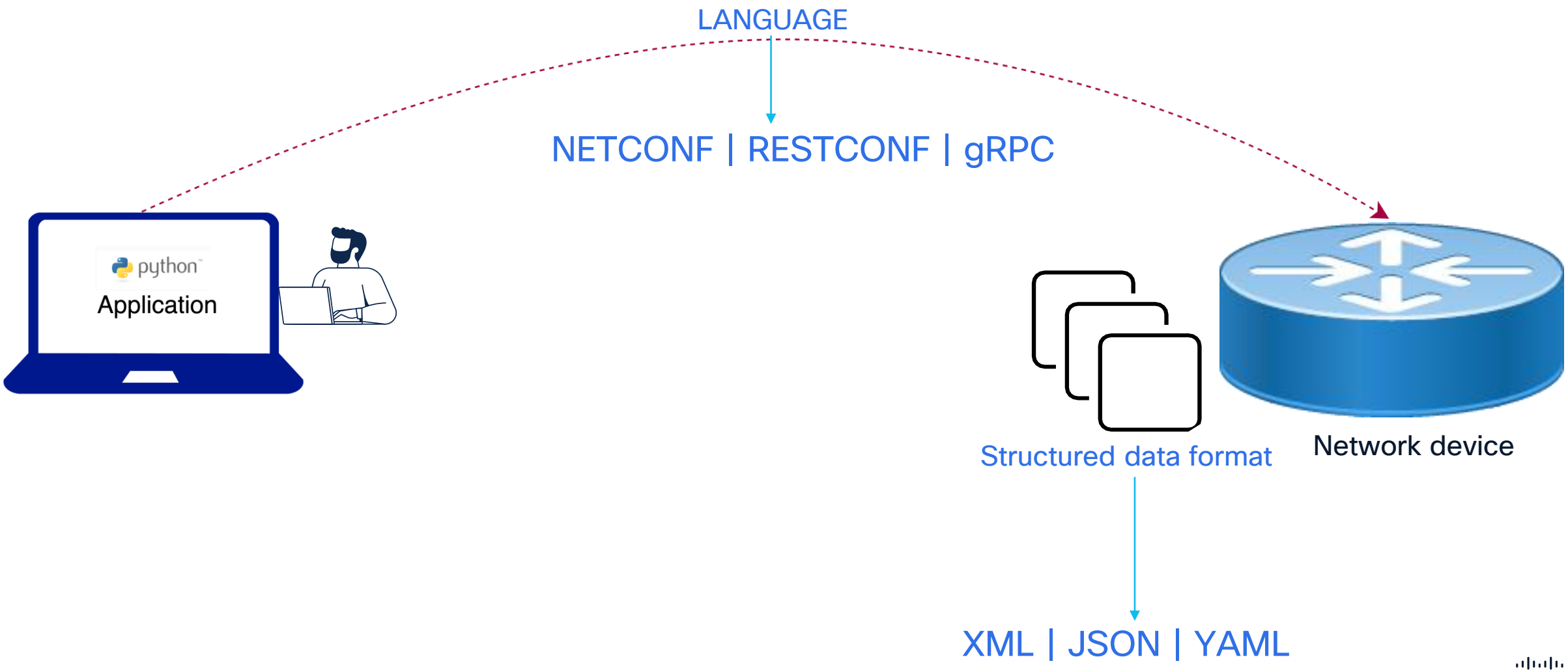
© 2025 Cisco and/or its affiliates. All rights reserved.

Communication with network devices



© 2025 Cisco and/or its affiliates. All rights reserved.

Communication with network devices



© 2025 Cisco and/or its affiliates. All rights reserved.

Python essentials

Python essentials

Python Data Types



LIST ➔ Collection of elements of different types, enclosed in SQUARE brackets

Example: [“Cisco8000”, ”24.3.3”, “ASR9K”, “7.9.2”]

Index[0] Index[1] Index[2] Index[3]

Accesing an element of a “list” data type in python:

Example code snippet

```
devices_versions=["Cisco8000","24.3.3","ASR9K","7.9.2"]
print(devices_versions[0],"\n",devices_versions[1])
Cisco8000
 24.3.3
```

© 2025 Cisco and/or its affiliates. All rights reserved.

Python essentials

Python Data Types



DICTIONARY ➔ Collection of KEY-VALUE pair, enclosed in CURLY brackets

Example: { “Name”:”Siddesh”, ”Age”:”18”, ”Emp_id”:”100” }

Key Value Key Value

Accesing an element of a “dictionary” data type in python:

Example code snippet

```
employee={"Name":"Siddesh","age":"18","emp_id":"20"}
print(employee["Name"],"\n",employee["age"])
Siddesh
 18
```

Key Key

© 2025 Cisco and/or its affiliates. All rights reserved.

Python essentials – Libraries, modules, packages

Python Modules and Libraries.

Software Library – Collection of pre-written re-usable code.



© 2025 Cisco and/or its affiliates. All rights reserved.

Python essentials – Libraries, modules, packages

Python Modules and Packages



MODULE



PACKAGES

Importing a module inside python code

```
import netmiko
      or
from netmiko import ConnectHandler
```

Module name

Class name

Importing a module inside a package in python

```
from network_utils.protocol import data_parser

# Using the imported modules
data_parser.process_data("config data")
```

Package name

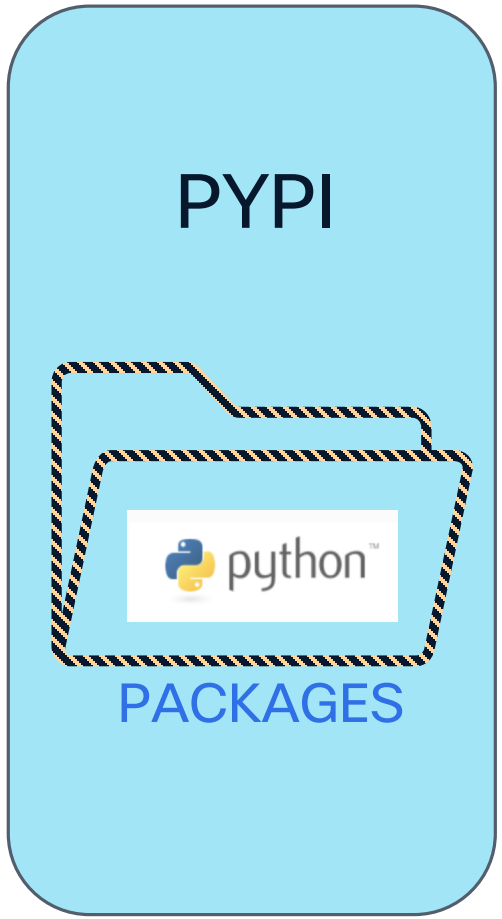
Module name

Class name

© 2025 Cisco and/or its affiliates. All rights reserved.

Python essentials – PYPI

Python package management




```
pip search <package-name>

pip install <package-name>

pip install <package-name> --upgrade

pip uninstall <package-name>

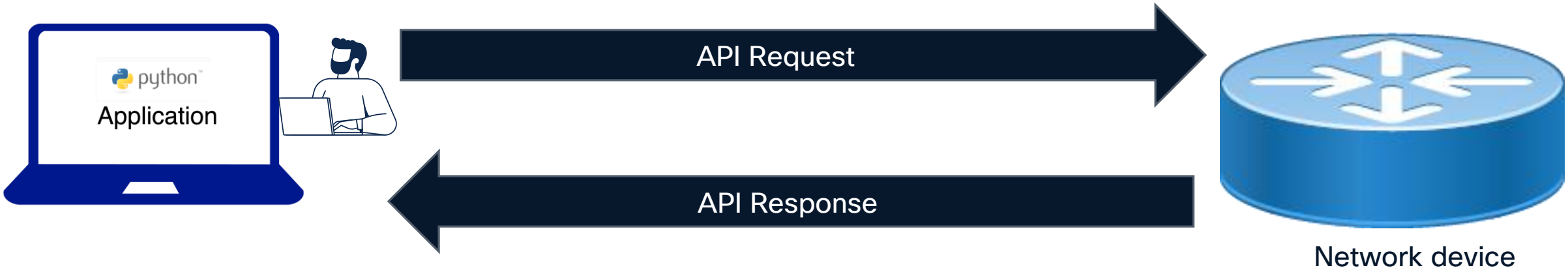
pip list

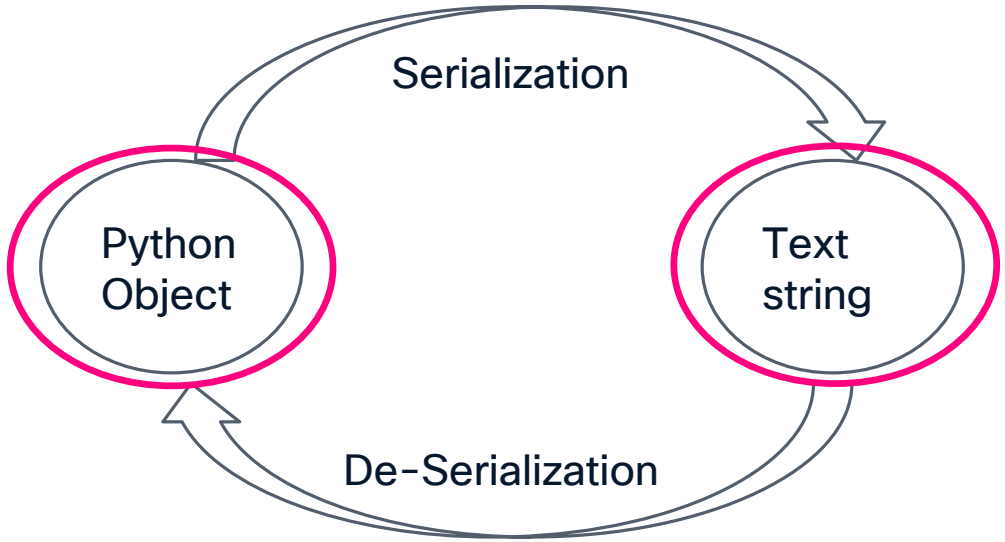
pip freeze > requirements.txt
```

© 2025 Cisco and/or its affiliates. All rights reserved.

Serialization and Deserialization of Data

Data serialization/deserialization





© 2025 Cisco and/or its affiliates. All rights reserved.

Example :Serialization/De-serialization in python

Json encoding serialization example

```
>>> device_inventory =  {
      "inventory": {
          "device1": {
              "name": ”ASR9K_lab1",
              “ip_address: : “192.168.0.100
          }
          “device 2: {
               ”name": ”ASR9K_lab2",
               “ip_address: : “192.168.0.200
          }
      }
  }
>>> type(device_inventory)
<class ‘dict’>
```

```
>>> device_inventory_json=json.dumps(device_inventory)
>>> type(device_inventory_json)
<class ‘str’>
```

© 2025 Cisco and/or its affiliates. All rights reserved.

Serialization/De-serialization in python

Python modules for json,xml and yaml

```
json.dumps()

dicttoxml.dicttoxml().decode()

yaml.dump()
```

Converts provided data type to string.

Accepts:
- Lists
- Dictionary

```
json.loads()

xmltodic.parse(xml_file.read(),attr_prefix='')

yaml.safe_load(yaml_data)
```

Converts provided string to valid python data type.

Returns:
- Lists
- Dictionary

© 2025 Cisco and/or its affiliates. All rights reserved.

A common language, standard data encoding format and identify each other.

– *Makes network automation possible*

© 2025 Cisco and/or its affiliates. All rights reserved.

JSON

JSON, data types

Json encoding syntax

DICTIONARY

Collection of Key-Value Pair

LIST

List of elements or attributes

© 2025 Cisco and/or its affiliates. All rights reserved.

JSON, example

Json encoding format example

Example : List

```
[
  {
    "device1":{
      "name":"ASR9K_lab1",
      "ip_address":"192.168.0.100"
    },
    "device2":{
      "name":"ASR9K_lab2",
      "ip_address":"192.168.0.200"
    }
  }
]
```

Example : Dictionary

```
'{
     "inventory": {
          "device1": {
               "name":”ASR9K_lab1",
               “ip_address:“192.168.0.100”
               },
          “device 2”: {
              ”name": ”ASR9K_lab2",
              “ip_address:“192.168.0.200”
          }
     }
 }'
```

© 2025 Cisco and/or its affiliates. All rights reserved.

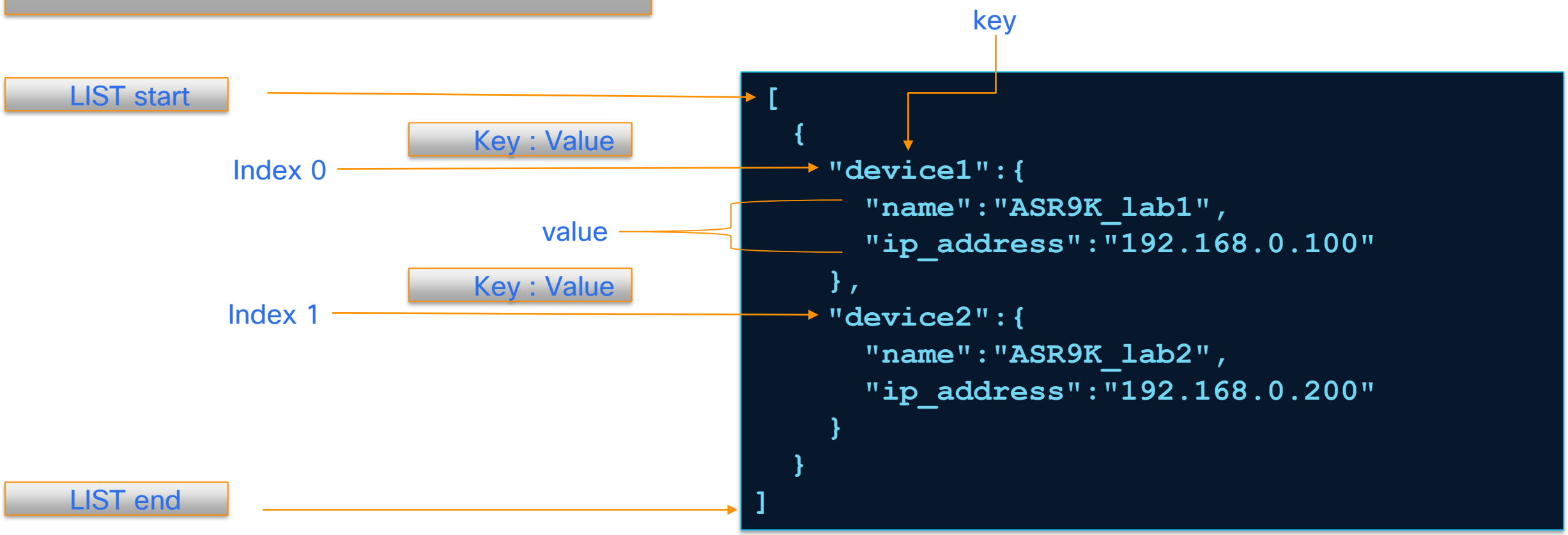
JSON, List data type

Json encoding syntax

Value → List of key-value pairs

LIST

‘[device1,device2]’



© 2025 Cisco and/or its affiliates. All rights reserved.

Example: Accessing a value in JSON data data type



`device_inventory_json=`

```
[
  {
    "device1":{
      "name":"ASR9K_lab1",
      "ip_address":"192.168.0.100"
    },
    "device2":{
      "name":"ASR9K_lab2",
      "ip_address":"192.168.0.200"
    }
  }
]
```

How would you print ”device1” ip_address” ?

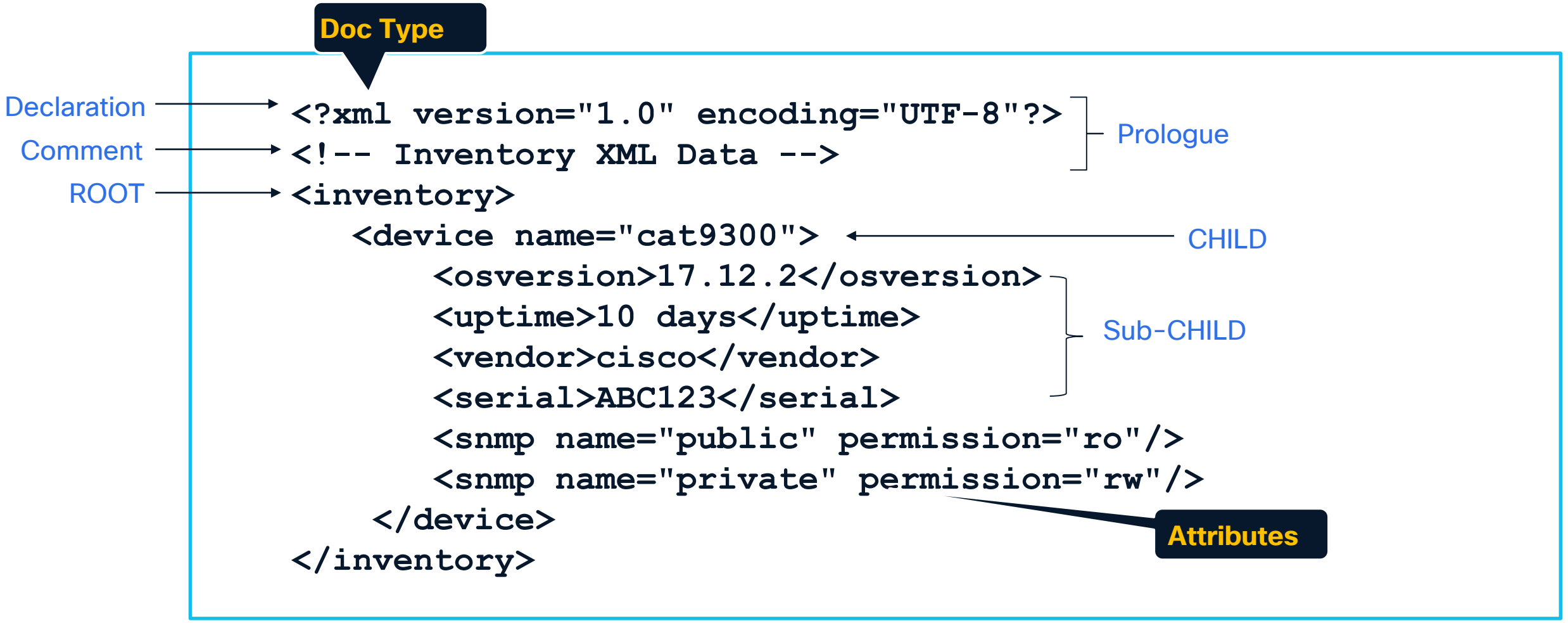
`device_inventory_json[0][device1][ip_address]`

© 2025 Cisco and/or its affiliates. All rights reserved.

XML

XML example

XML encoding format example



© 2025 Cisco and/or its affiliates. All rights reserved.

YAML

YAML example

YAML encoding format example

YAML

```
---inventory: csr1kv1
```

Document separator

JSON

```
{ "inventory": "csr1kv1“ }
```

YAML

```
---
- GigabitEthernet1
- GigabitEthernet2
- GigabitEthernet3
```

JSON

```
[
    “GigabitEthernet1”,
    “GigabitEthernet2",
    “GigabitEthernet3”
]
```

© 2025 Cisco and/or its affiliates. All rights reserved.

YAML example

YAML encoding format example

YAML

```
---
inventory:
  device:
    csr1kv1:
      vendor: cisco
```

JSON

```
{
    "inventory" : {
        "device" : {
            "csr1kv1": {
                "vendor": "cisco"
            }
        }
    }
}
```

```
>>>import yaml
>>> yaml_data = open('inventory.yml', 'r')
data = yaml.safe_load(yaml_data)
```

© 2025 Cisco and/or its affiliates. All rights reserved.

XML/JSON/YAML

Format	Readability	Use Case
XML	Verbose	Netconf
JSON	Lightweight	REST APIs
YAML	Human-readable	Ansible

© 2025 Cisco and/or its affiliates. All rights reserved.

A common language, standard data encoding format and identify each other.

– *Makes network automation possible*

© 2025 Cisco and/or its affiliates. All rights reserved.

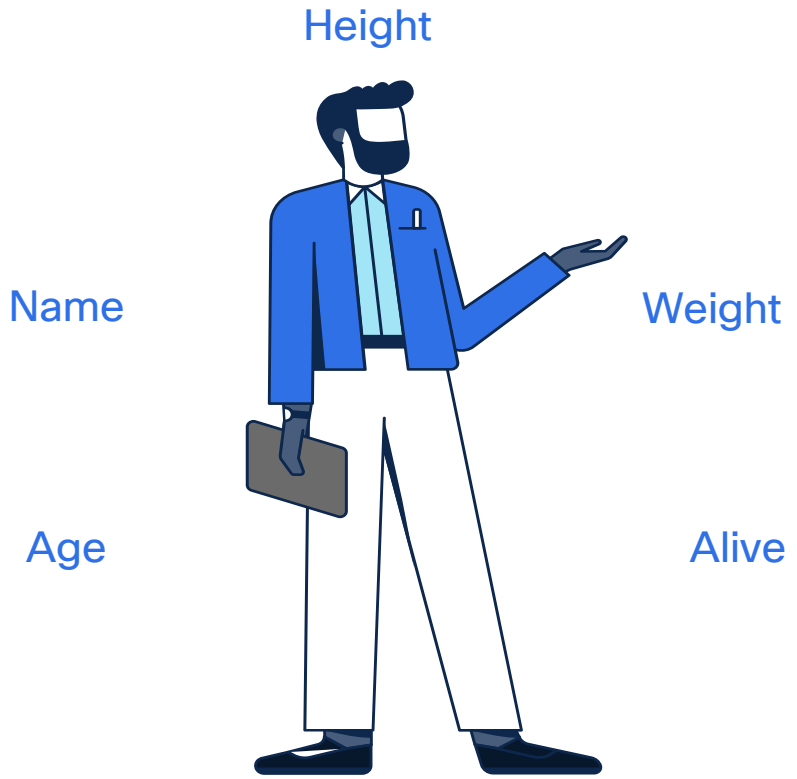
Data Modelling with YANG

What is a Data Model

© 2025 Cisco and/or its affiliates. All rights reserved.

Data model analogy

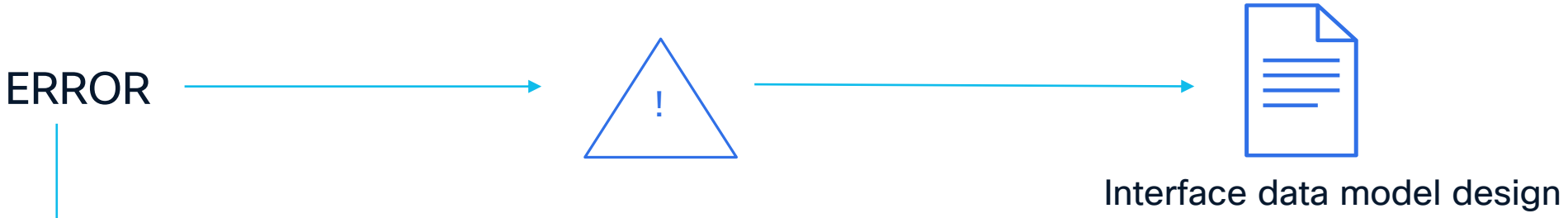
Data modelling



© 2025 Cisco and/or its affiliates. All rights reserved.

Data model analogy

Data modelling





```
interface GigabitEthernet0/0/0/0
  vrf CustomerVRF
  encapsulation dot1q Siddesh
% Invalid input detected at '^' marker. %

”siddesh" is not a valid VLAN ID; VLAN ID must be an integer between 1-4094
```

© 2025 Cisco and/or its affiliates. All rights reserved.

YANG MODEL Overview

- Yang is a data-modeling language defined in RFC 6020 designed for modelling the network data consumable by the networking device.
- Utilized by Netconf and Restconf protocols.
- Defines the structure, syntax, semantics and constraints for device configuration data and operational state data

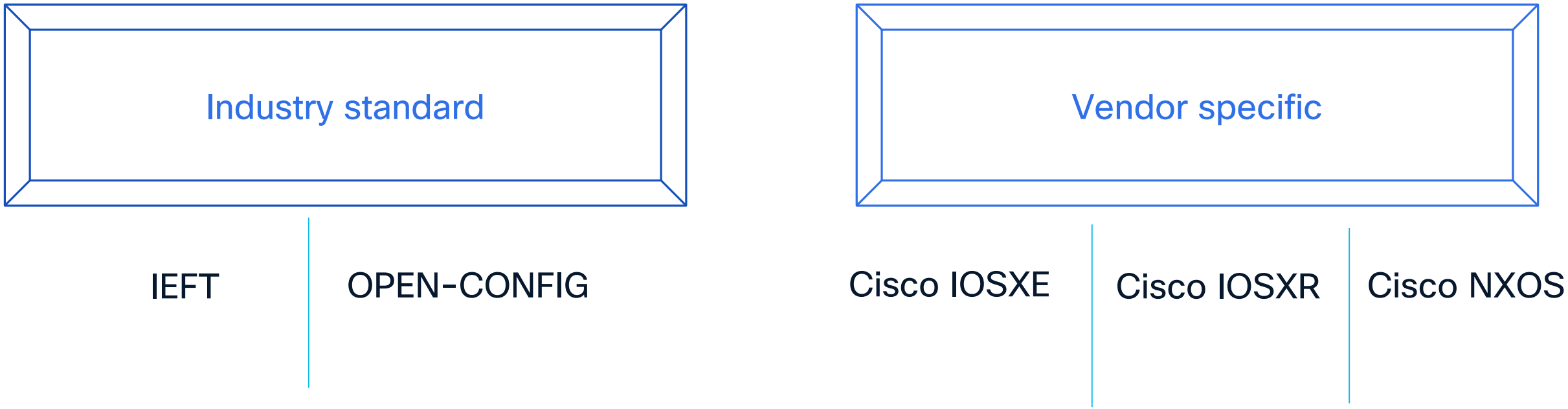
© 2025 Cisco and/or its affiliates. All rights reserved.

TYPES OF YANG MODEL

© 2025 Cisco and/or its affiliates. All rights reserved.

YANG Model types

Different types of available YANG models



© 2025 Cisco and/or its affiliates. All rights reserved.

YANG Model types

Different types of available YANG models

Cisco IOSXE Native

```
{
  "Cisco-IOS-XE-native:GigabitEthernet": {
    "name": "1",
    "ip": {
      "address": {
        "primary": {
          "address": "10.10.10.1",
          "mask": "255.255.255.0"
        }
      }
    },
    "mop": {
      "enabled": false,
      "sysid": false
    },
    "Cisco-IOS-XE-ethernet:negotiation": {
      "auto": true
    }
  }
}
```

COMPARE

OPEN-CONFIG

```
{
  "openconfig-interfaces:interface": {
    "name": "GigabitEthernet1",
    "config": {
      "enabled": true
    },
    "state": {
      "enabled": true,
      "admin-status": "UP",
      "oper-status": "UP",
      "counters": {
        "in-octets": "417083378",
        "in-unicast-pkts": "1488045"
      }
    },
    "subinterfaces": {
      "subinterface": [
        {
          "openconfig-if-ip:ipv4": {
            "addresses": {
              "address": [ ... ]
            }
          }
        }
      ]
    }
  }
}
```

https://github.com/openconfig

© 2025 Cisco and/or its affiliates. All rights reserved.

YANG Model types

Different types of available YANG models

Cisco IOSXE Native

```
{
  "Cisco-IOS-XE-native:GigabitEthernet": {
    "name": "1",
    "ip": {
      "address": {
        "primary": {
          "address": "10.10.10.1",
          "mask": "255.255.255.0"
        }
      }
    },
    "mop": {
      "enabled": false,
      "sysid": false
    },
    "Cisco-IOS-XE-ethernet:negotiation": {
      "auto": true
    }
  }
}
```

COMPARE

IEFT YANG Model

```
{
  "ietf-interfaces:interface": {
    "name": "GigabitEthernet1",
    "type": "iana-if-type:ethernetCsmacd",
    "enabled": true,
    "ietf-ip:ipv4": {
      "address": [
        {
          "ip": "10.10.10.1",
          "netmask": "255.255.255.0"
        }
      ]
    },
    "ietf-ip:ipv6": {}
  }
}
```

© 2025 Cisco and/or its affiliates. All rights reserved.

YANG Model types

Different types of available YANG models

Cisco IOSXE Native

```
{
  "Cisco-IOS-XE-native:GigabitEthernet": {
    "name": "1",
    "ip": {
      "address": {
        "primary": {
          "address": "10.10.10.1",
          "mask": "255.255.255.0"
        }
      }
    },
    "mop": {
      "enabled": false,
      "sysid": false
    },
    "Cisco-IOS-XE-ethernet:negotiation": {
      "auto": true
    }
  }
}
```

COMPARE

Cisco CLI

```
Lab_router#sho run interface GigabitEthernet1
Building configuration...

Current configuration : 118 bytes
!
interface GigabitEthernet1
 ip address 10.10.10.1 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
end
```

© 2025 Cisco and/or its affiliates. All rights reserved.

Clone YANG model

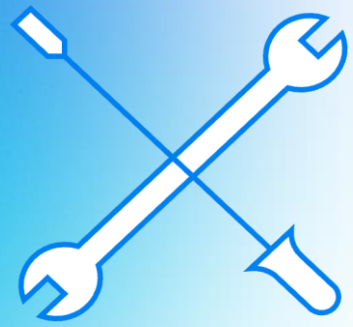
```
sinayak@SINAYAK-M-DG9Y OneDrive - Cisco % mkdir DEVNET
sinayak@SINAYAK-M-DG9Y OneDrive - Cisco % cd DEVNET
sinayak@SINAYAK-M-DG9Y DEVNET % ls
sinayak@SINAYAK-M-DG9Y DEVNET % pwd
/Users/sinayak/OneDrive - Cisco/DEVNET
sinayak@SINAYAK-M-DG9Y DEVNET % mkdir yang_clone
sinayak@SINAYAK-M-DG9Y DEVNET % cd yang_clone
sinayak@SINAYAK-M-DG9Y yang_clone % ls
sinayak@SINAYAK-M-DG9Y yang_clone % git clone https://github.com/YangModels/yang
Cloning into 'yang'...
remote: Enumerating objects: 60191, done.
remote: Counting objects: 100% (81/81), done.
remote: Compressing objects: 100% (52/52), done.
remote: Total 60191 (delta 47), reused 34 (delta 27), pack-reused 60110 (from 3)
Receiving objects: 100% (60191/60191), 157.61 MiB | 38.98 MiB/s, done.
Resolving deltas: 100% (47084/47084), done.
Updating files: 100% (141963/141963), done.
sinayak@SINAYAK-M-DG9Y yang_clone %
sinayak@SINAYAK-M-DG9Y yang_clone % ls yang
README.md experimental standard testall.sh vendor
sinayak@SINAYAK-M-DG9Y yang_clone % ls yang/vendor
README alliedtelesis ciena cisco fujitsu huawei juniper nokia readylinks
sinayak@SINAYAK-M-DG9Y yang_clone % ls yang/vendor/cisco
README.md check.sh nx svo xe xr
sinayak@SINAYAK-M-DG9Y yang_clone %
```

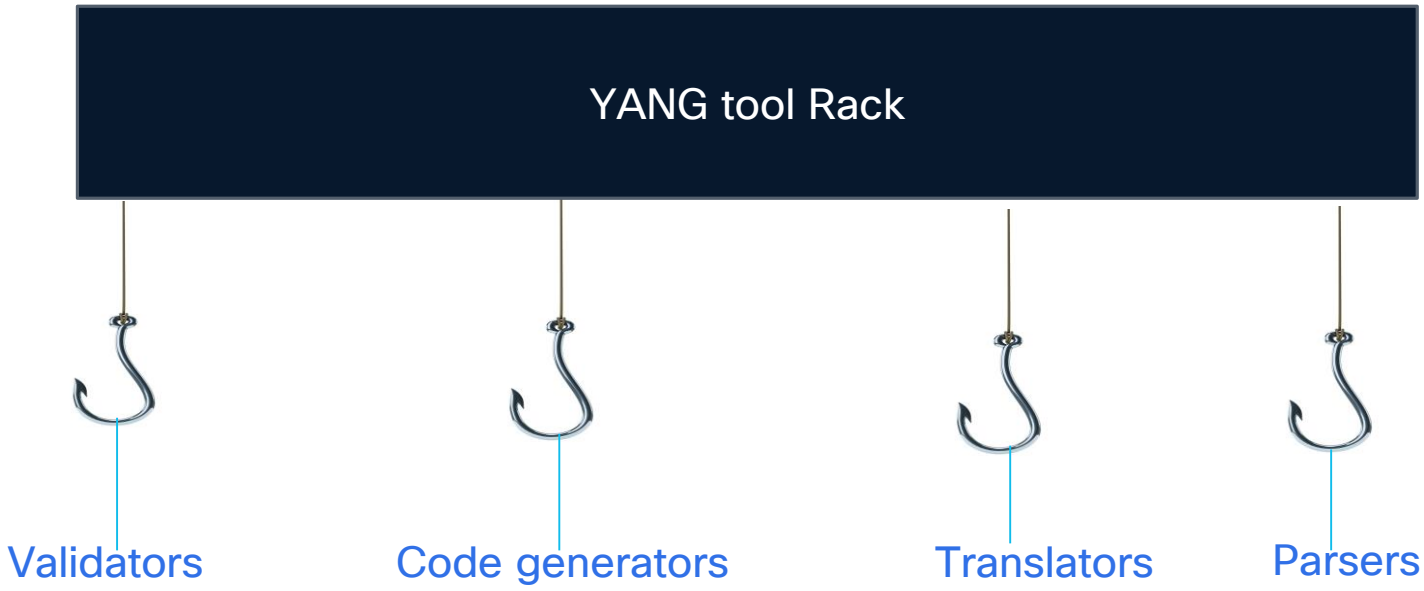
© 2025 Cisco and/or its affiliates. All rights reserved.

YANG TOOLS

© 2025 Cisco and/or its affiliates. All rights reserved.

YANG Tools





© 2025 Cisco and/or its affiliates. All rights reserved.

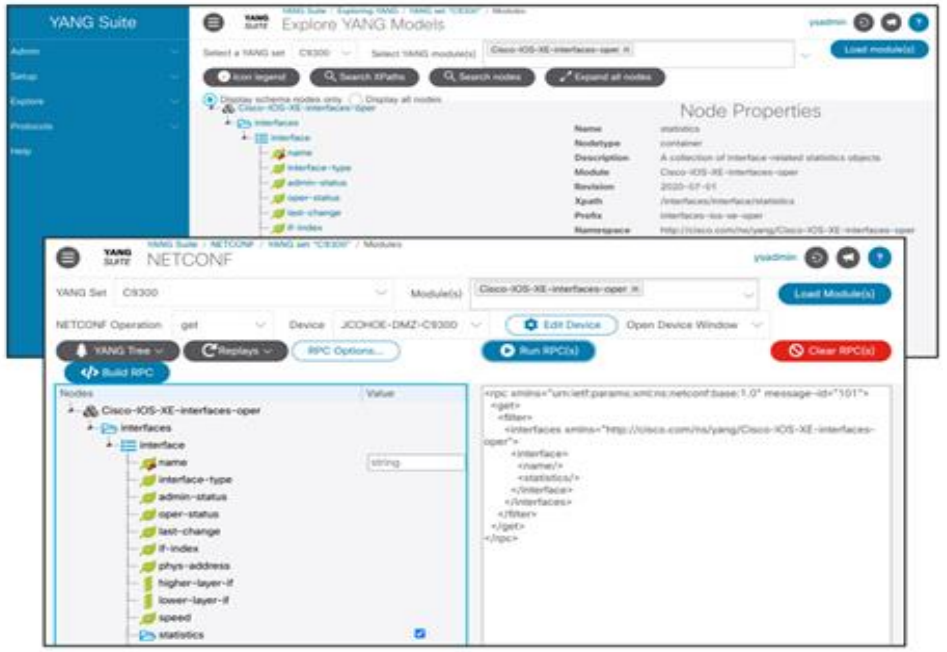
YANG Tool – Code generation and validation

Cisco YANG Suite

YANG API Testing and Validation Environment

Construct and test YANG based APIs over NETCONF, RESTCONF, gRPC and gNMI

IOS XE / IOS XR / NX OS platforms



Now Available !

developer.cisco.com/yangsuite

github.com/CiscoDevNet/yangsuite

© 2025 Cisco and/or its affiliates. All rights reserved.

YANG Tool - PYANG

Open-source yang validator, yang translator and code generator tool

```
(pyangenv) sinayak@SINAYAK-M-DG9Y %
```

o/p format | Path of *yang* model | yang model name | *Nodes* to display

```
pyang -f tree -p yang/vendor/cisco/xe/17121 yang/vendor/cisco/xe/17121/ietf-interfaces.yang --tree-depth=3
```

```
module: ietf-interfaces
  +--rw interfaces                                    <-- Node level - 1
  |  +--rw interface* [name]                          <-- Node level - 2
  |     +--rw name                        string
  |     +--rw description?                string
  |     +--rw type                        identityref
  |     +--rw enabled?                    boolean
  |     +--rw link-up-down-trap-enable?   enumeration {if-mib}?
  +--ro interfaces-state                              <-- Node level - 3
     +--ro interface* [name]
        +--ro name                string
        +--ro type                identityref
        +--ro admin-status        enumeration {if-mib}?
        +--ro oper-status         enumeration
        +--ro last-change?        yang:date-and-time
        +--ro if-index            int32 {if-mib}?
        +--ro phys-address?       yang:phys-address
        +--ro higher-layer-if*    interface-state-ref
        +--ro lower-layer-if*     interface-state-ref
        +--ro speed?              yang:gauge64
        +--ro statistics                            <-- Node level - 4
           ...
(pyangenv) sinayak@SINAYAK-M-DG9Y yang_clone %
```


© 2025 Cisco and/or its affiliates. All rights reserved.

cisco

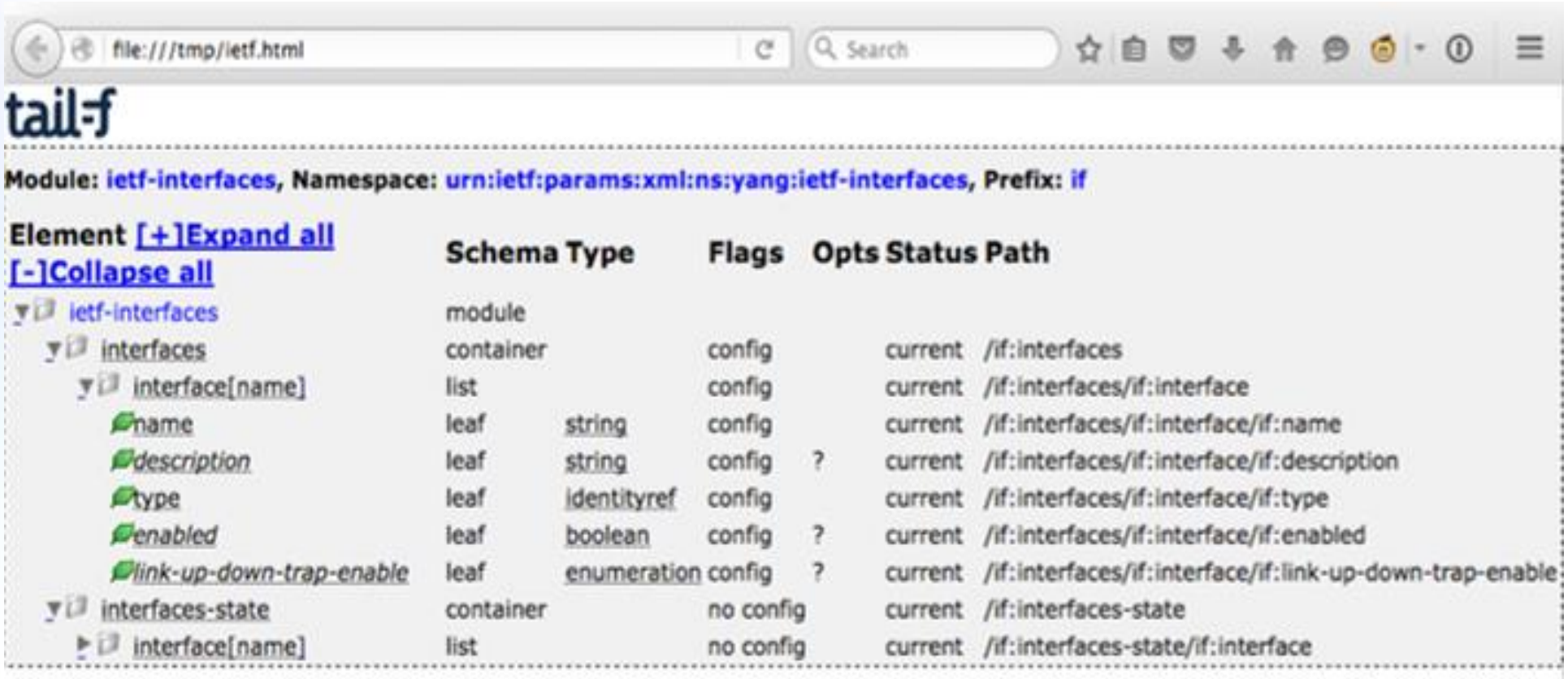


Yang tool

PYANG Web UI

To visualize any YANG model in a tree format over a Web UI using PYANG , simply execute pyang, pipe the output to HTML, and open the file in a browser.

```
pyang -f tree -p yang/vendor/cisco/xe/17121 yang/vendor/cisco/xe/17121/ietf-interfaces.yang > intf.html
```



© 2025 Cisco and/or its affiliates. All rights reserved.

A common language, standard data encoding format and identify each other.

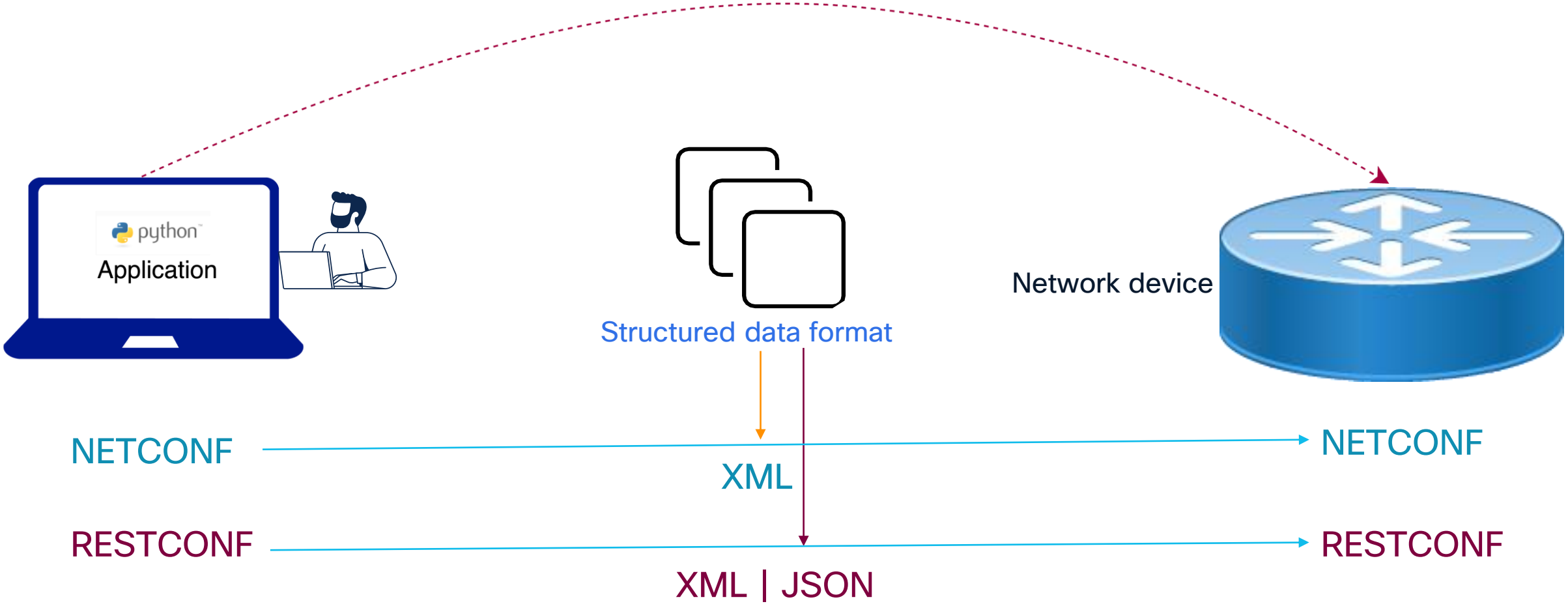
– *Makes network automation possible*

© 2025 Cisco and/or its affiliates. All rights reserved.

Network automation with NETCONF and RestConf

cisco Live !

Communication with network devices



© 2025 Cisco and/or its affiliates. All rights reserved.

Network Management using NETCONF





Network device

NETCONF ➔ SSH/TLS over TCP port 830

```
<rpc message-id="101"
     xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
   <get/>
</rpc>
```



Message encoding

© 2025 Cisco and/or its affiliates. All rights reserved.

Network Management using NETCONF





Network device



```
<rpc-reply message-id="101
     xmlns="urn:ietf:params:xml:ns:netconf:base:1.0>
   <data>
       <!-- ... entire set of data returned ... -->
   </data>
</rpc-reply>
```

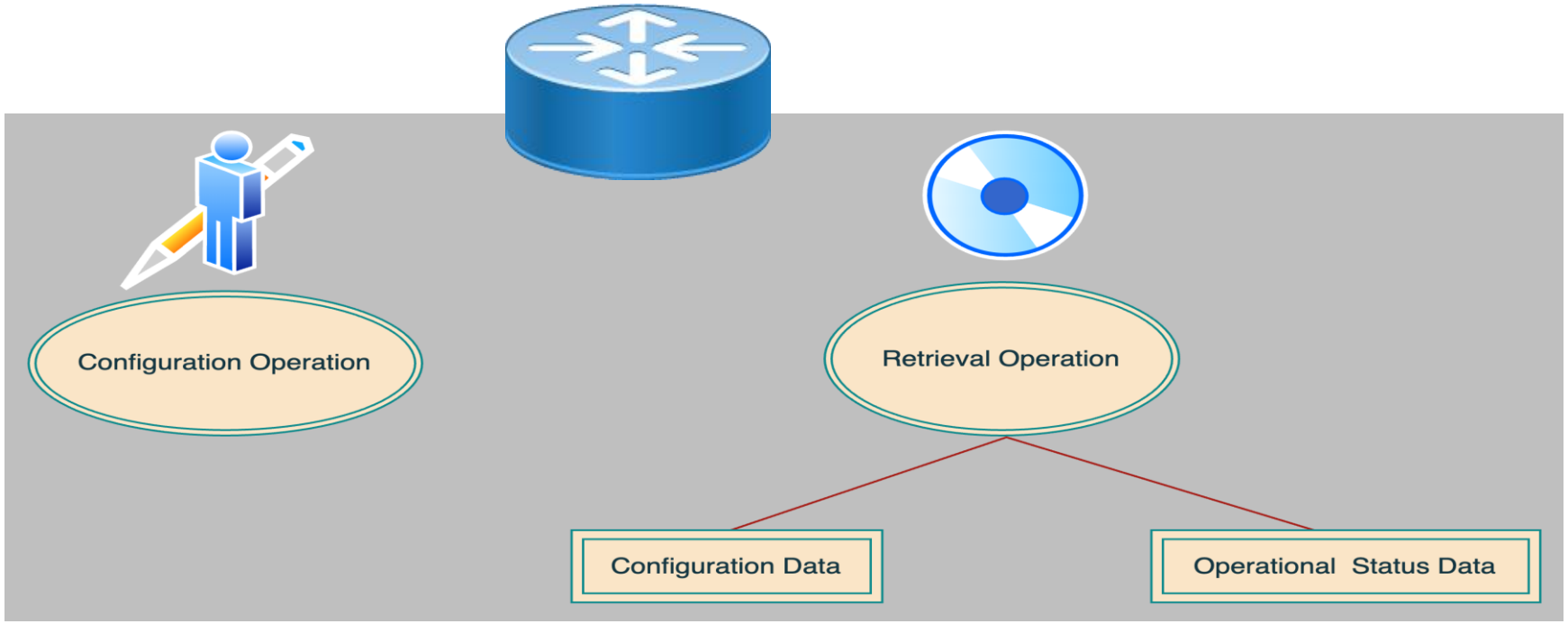


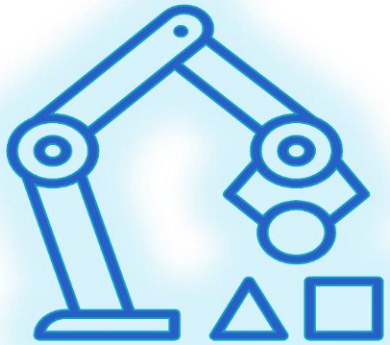
Message encoding

© 2025 Cisco and/or its affiliates. All rights reserved.

Network Management using NETCONF







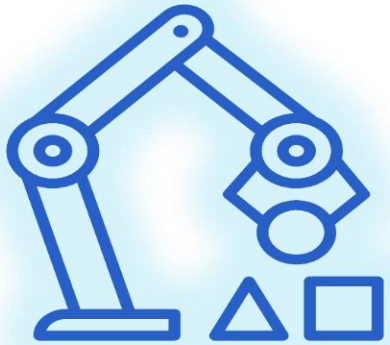
Operations

© 2025 Cisco and/or its affiliates. All rights reserved.

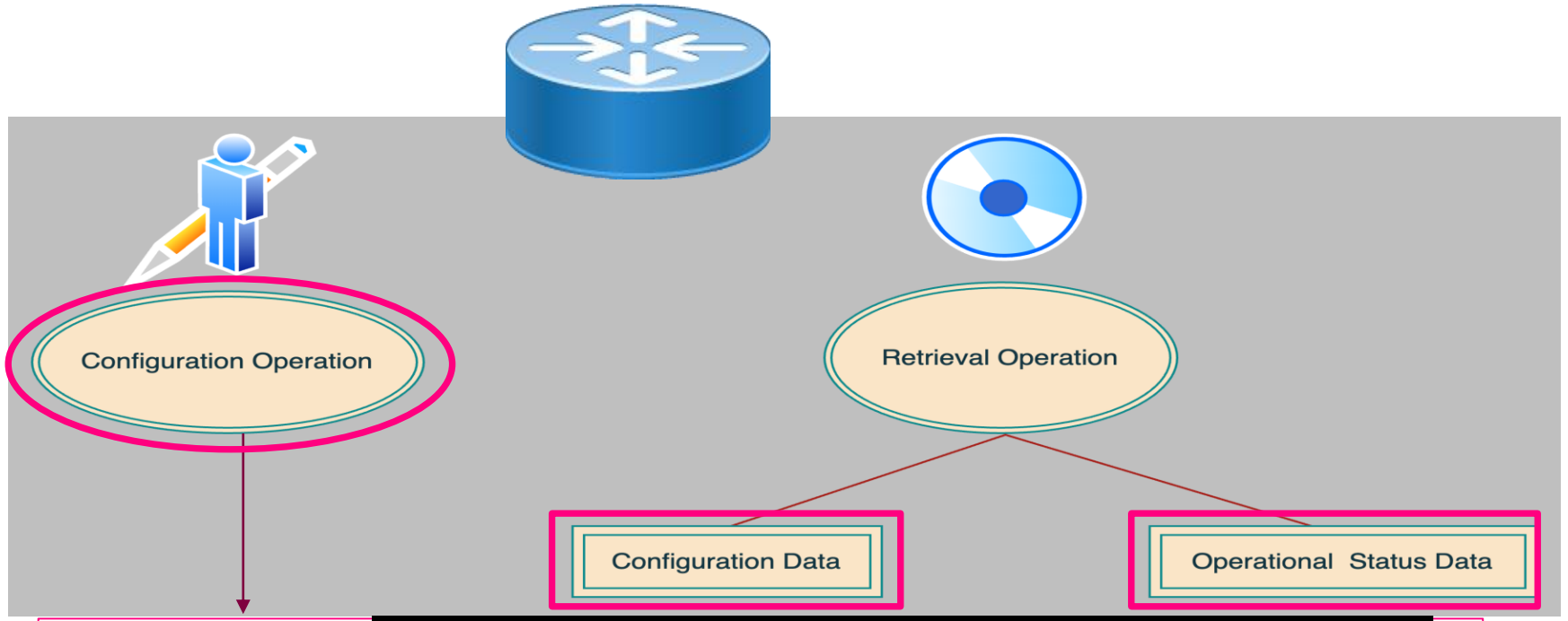
Network Management using NETCONF



NETCONF



Operations



<edit-config>, operation="create|"replace"|"merge"|"delete""

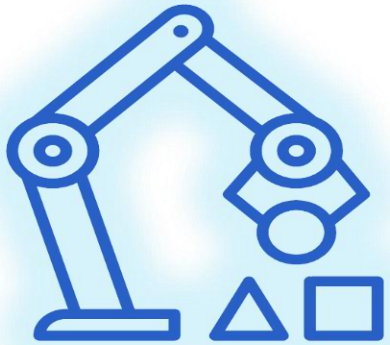
NETCONF RPC (Remote Procedure Call)

© 2025 Cisco and/or its affiliates. All rights reserved.

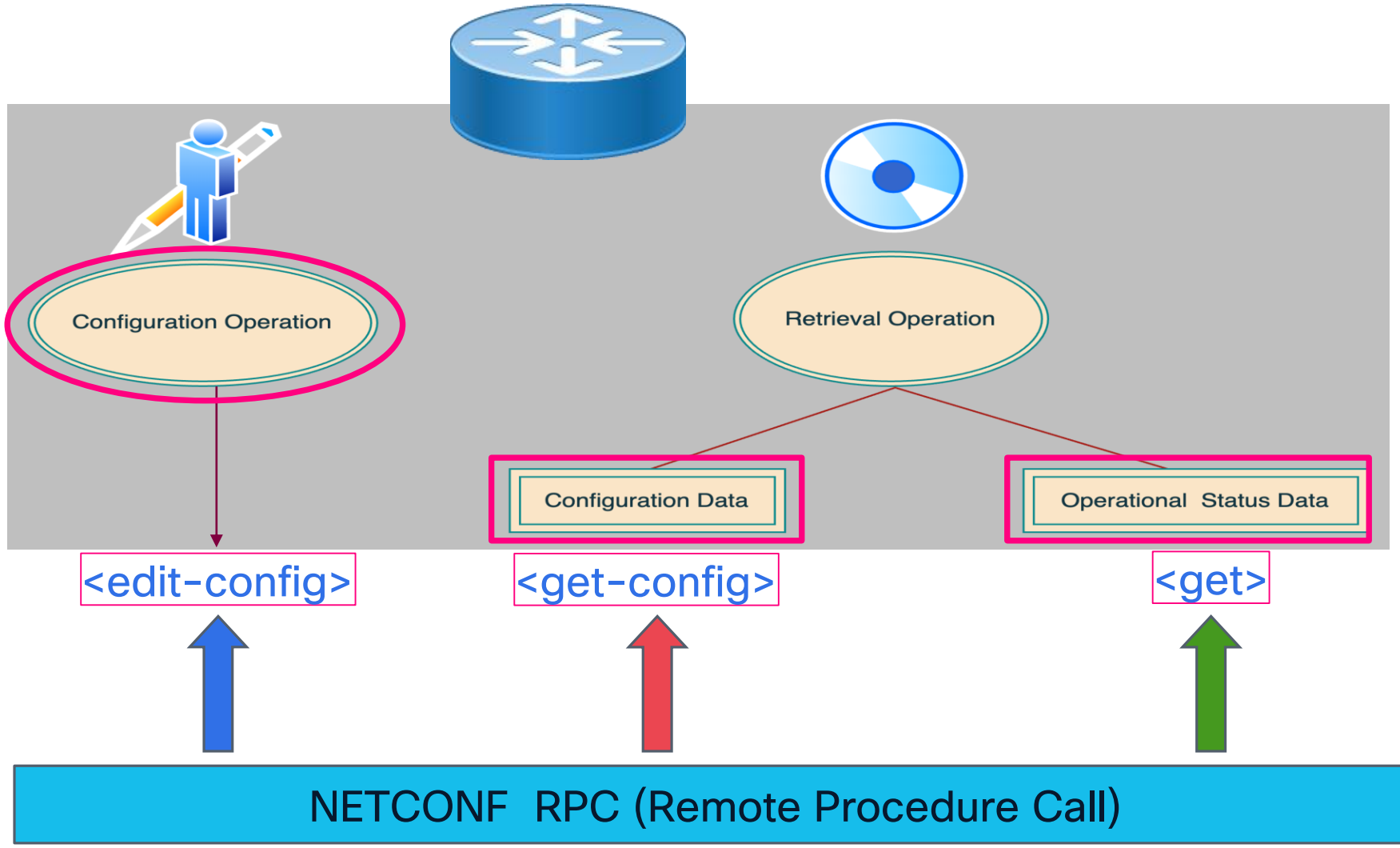
Network Management using NETCONF



NETCONF



Operations

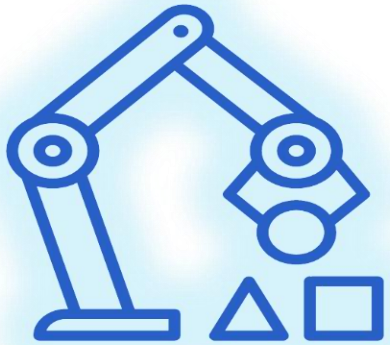


© 2025 Cisco and/or its affiliates. All rights reserved.

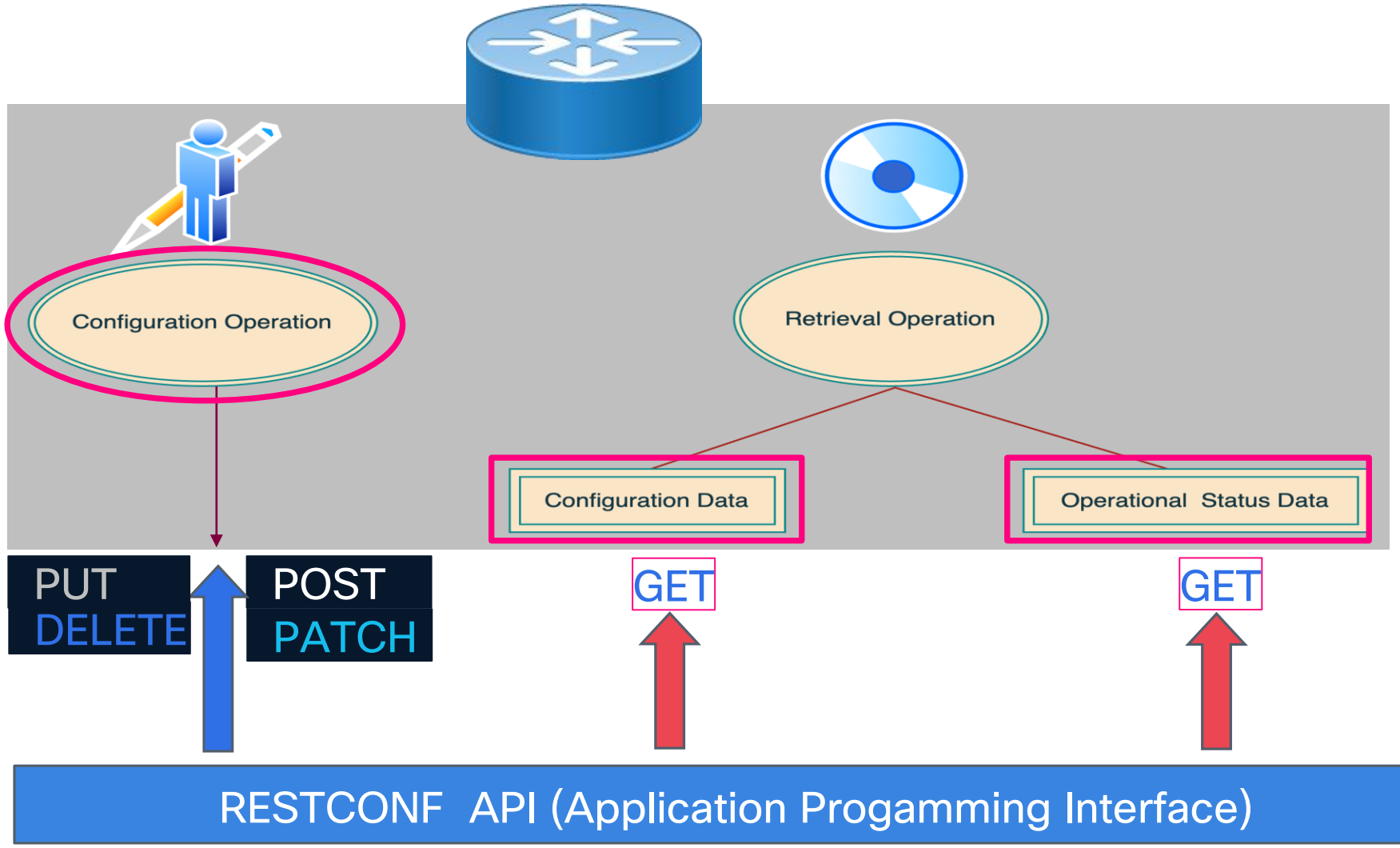
Network Management using RESTCONF



RESTCONF



Operations

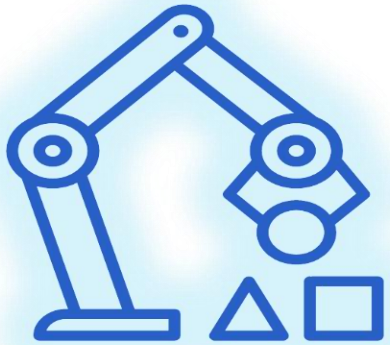


© 2025 Cisco and/or its affiliates. All rights reserved.

Network Management using RESTCONF



RESTCONF



Operations

GET Retrieves a resource

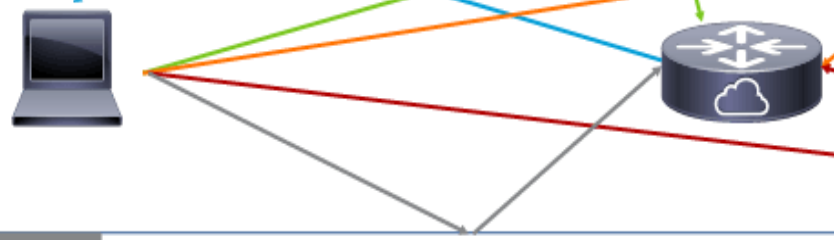
```
…
"GigabitEthernet": [
      {
        "name": "1",
        "shutdown": [null],
        "mop": {
          "enabled": false,
          "sysid": false
…
```

POST Creates a new resource

```
…
"ip"
    "address":{
        "primary": {
          "address":"10.0.10.1",
          "mask": "255.255.255.0"
        }
    }
…
```

PUT Updates an existing resource

```
. . .
"ip"
    "address":{
        "secondary": [
            {
                "secondary": "10.0.10.2",
                "mask": "255.255.255.0",
                "secondary": [
                    null
                ]
            }
        ],
        "primary": {
            "address":"10.0.10.1",
            "mask": "255.255.255.0"
        }
    }
. . .
```



DELETE Deletes a resource

PATCH Merges resources

```
…
"GigabitEthernet": [
      {
        "name": "1",
        "description" : "Not Used",
        "shutdown": [null],
…
```

```
…
"GigabitEthernet": [
      {
        "name": "1",
        "shutdown": [null],
        "ip": {
          "address": {
…
```

```
…
"GigabitEthernet": [
      {
        "name": "1",
        "description" : "Not Used",
        "shutdown": [null],
        "ip": {
          "address": {
…
```

RESTCONF API (Application Progamming Interface)

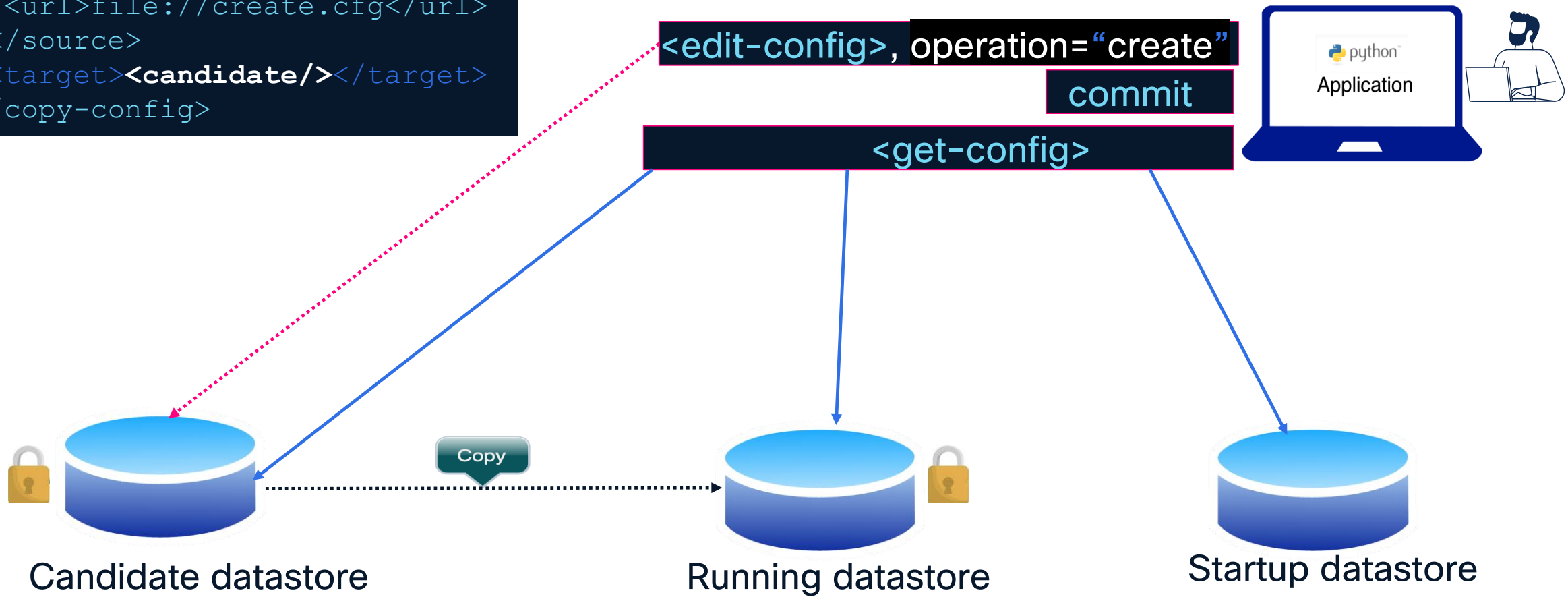
© 2025 Cisco and/or its affiliates. All rights reserved.

NETCONF Datastores

© 2025 Cisco and/or its affiliates. All rights reserved.

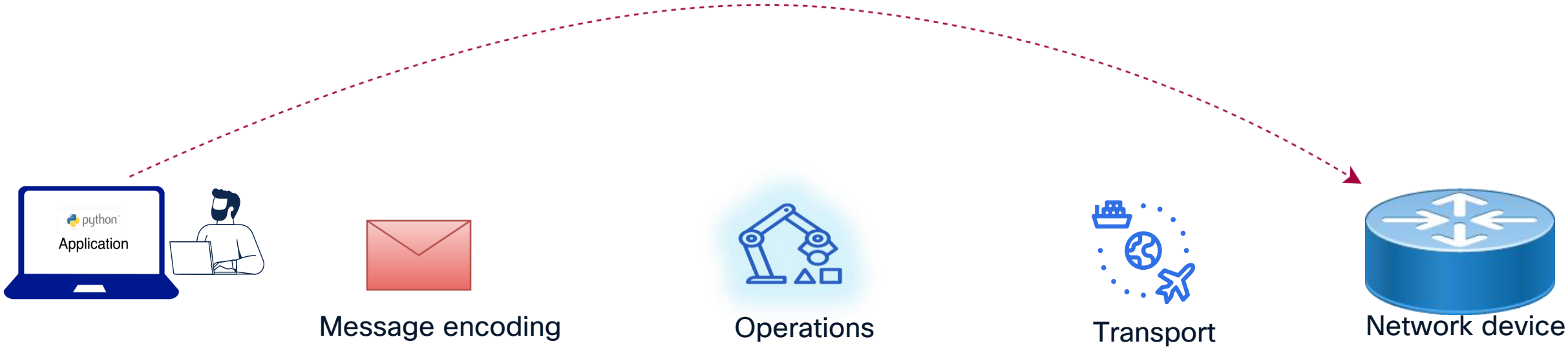
NETCONF Datastores

```
<copy-config>
  <source>
   <url>file://create.cfg</url>
  </source>
  <target><candidate/></target>
 </copy-config>
```



© 2025 Cisco and/or its affiliates. All rights reserved.

Network management with Netconf and Restconf



NETCONF	rpc, rpc-reply XML	get-config, edit-config, get	SSH	NETCONF
RESTCONF	API XML | JSON	Get,Post,Put,Patch,Delete	HTTPS	RESTCONF

© 2025 Cisco and/or its affiliates. All rights reserved.

Enabling NetConf and RestConf on Cisco devices.

CLI configuration for enabling Netconf and restconf

	NETCONF	RESTCONF

IOS-XE	`conf t`
	`netconf-yang`
	`aaa new-model`
	`aaa authentication login default local`
	`aaa authorization exec default local`
	`username netconf privilege 15`
		`enable`
		`configure terminal`
		`restconf`
		`ip http`

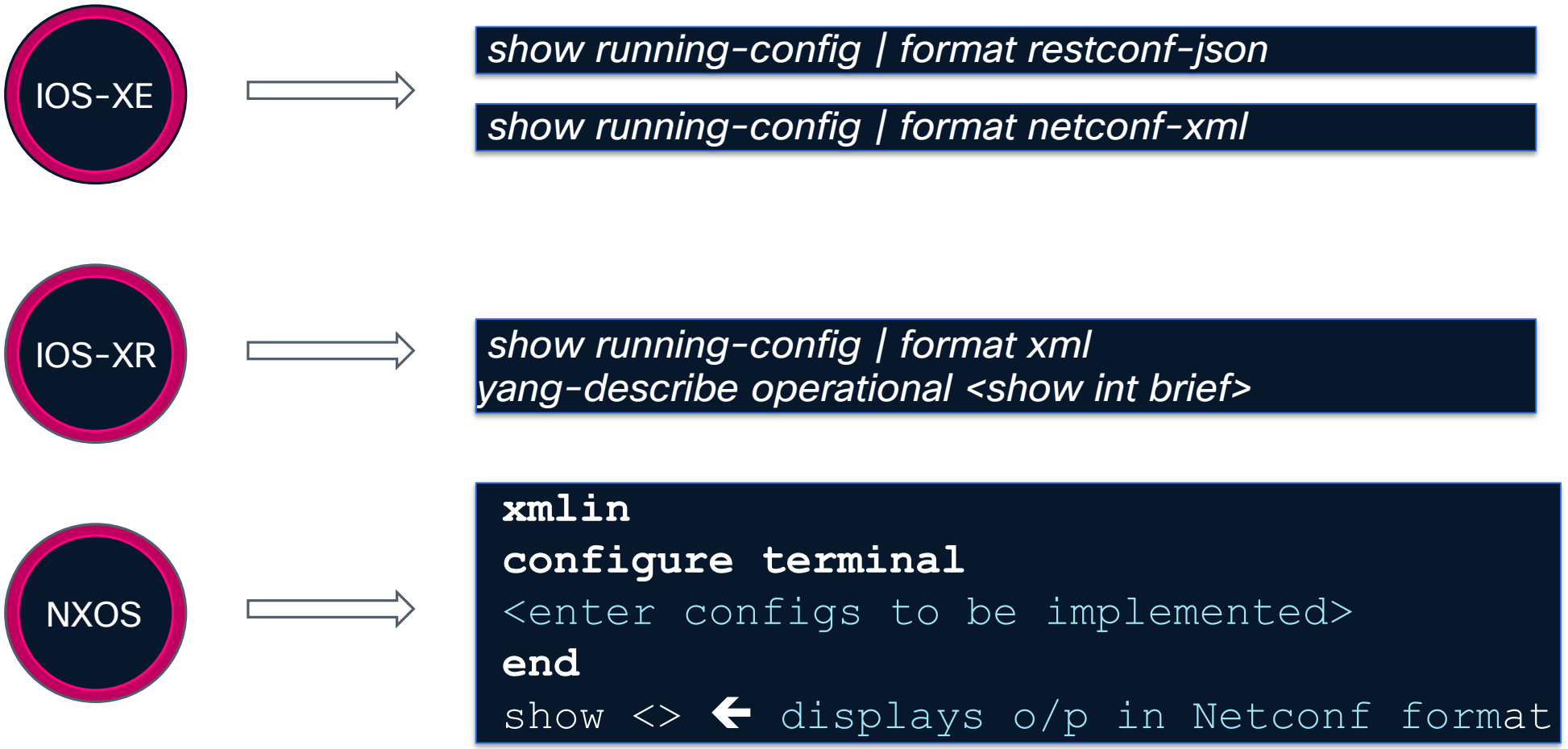
IOS-XR	`conf t`
	`ssh server v2`
	`ssh server netconf`
	`netconf agent tty`
	`netconf-yang agent ssh`
	mgbl k9sec
	Crypto keys
		NOT SUPPORTED

NXOS	`configure terminal`
	`feature netconf`
		`feature restconf`
		`nxapi http port 80`
		`nxapi https port 443`

© 2025 Cisco and/or its affiliates. All rights reserved.

CLI to XML and JSON on Cisco devices.

Translating Cisco CLI running config to XML and JSON formats



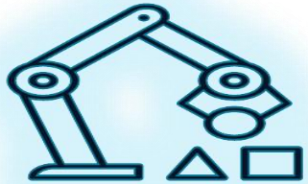
© 2025 Cisco and/or its affiliates. All rights reserved.

Demo Flow – Overview

© 2025 Cisco and/or its affiliates. All rights reserved.

Demo - Overview

Tools 

Operation mode 

Implementation 

NETCONF DEMO →  **YangSuite** → *<edit-config>* Configure bgp neighbors on IOS-XE

REST CONF DEMO →  **YangSuite** → *<GET>* Retrieve routing table information.

© 2025 Cisco and/or its affiliates. All rights reserved.

Demo – Netconf with YangSuite

© 2025 Cisco and/or its affiliates. All rights reserved.

Building your first RestConf/Rest api python code

© 2025 Cisco and/or its affiliates. All rights reserved.

RestConf/Rest API code – API request

URL

```
url = "https://sandboxdnac2.cisco.com/dna/intent/api/v1/network-device"
```

Hostname

API Path

Header

```
api_header = {
                    "Accept": "application/json",
                   "Content-Type": "application/json",
                   ”X-Auth-Token": “<token>”
                  }
```

API call

```
api_response = requests.request(‘GET’
                            url,
                  headers=api_header,
                    data=api_payload,
                  verify=False
                            )
```


© 2025 Cisco and/or its affiliates. All rights reserved.

cisco

RestConf/Rest API code – API Response

De-serialization

```
api_response_json=api_response.json()
```

Status code

```
status_code=api_response.statuscode()
```


© 2025 Cisco and/or its affiliates. All rights reserved.

cisco

RestConf/Rest API code – Full picture

```
import requests

url = "https://sandboxdnac2.cisco.com/dna/intent/api/v1/network-device"

api_header = {
                    "Accept": "application/json",
                    "Content-Type": "application/json",
                    ”X-Auth-Token": “<token>”
                  }

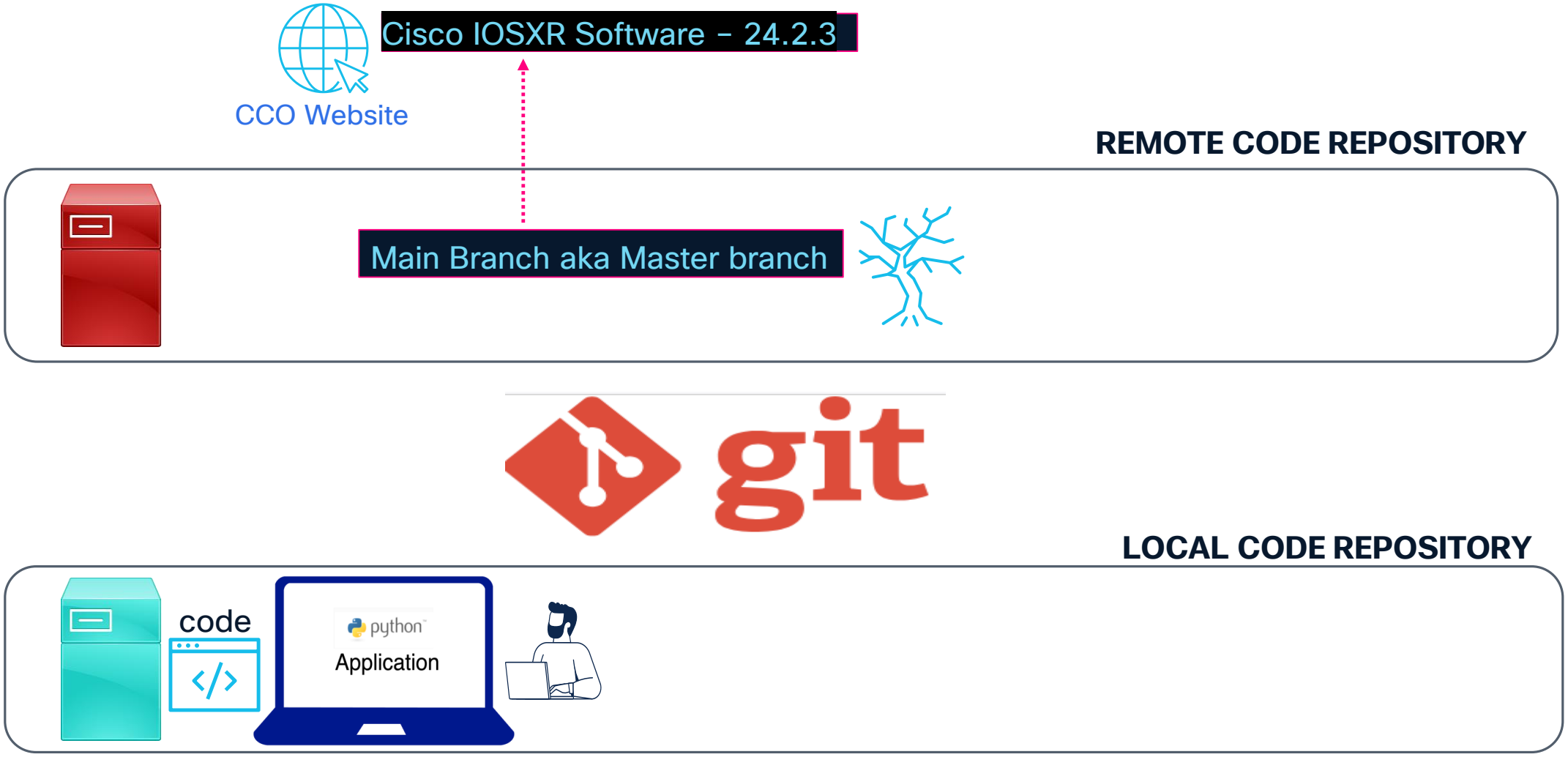
api_response = requests.request(‘GET’
                        url,
              headers=api_header,
                data=api_payload,
              verify=False
                        )
api_response_json=api_response.json()
status_code=api_response.statuscode()
print(status_code)
```


© 2025 Cisco and/or its affiliates. All rights reserved.

cisco

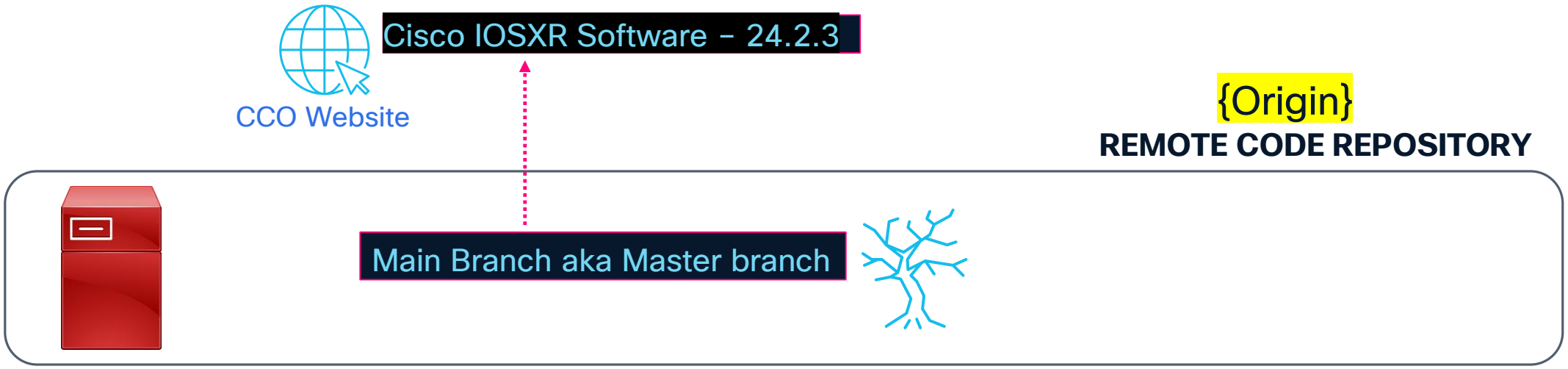
Managing your code with GIT

Need for version control system



© 2025 Cisco and/or its affiliates. All rights reserved.

Version control system - Taxonomy



Origin ➔ Default name GIT uses to refer to the remote repository.
It's a shorthand alias for the URL of the remote repository.

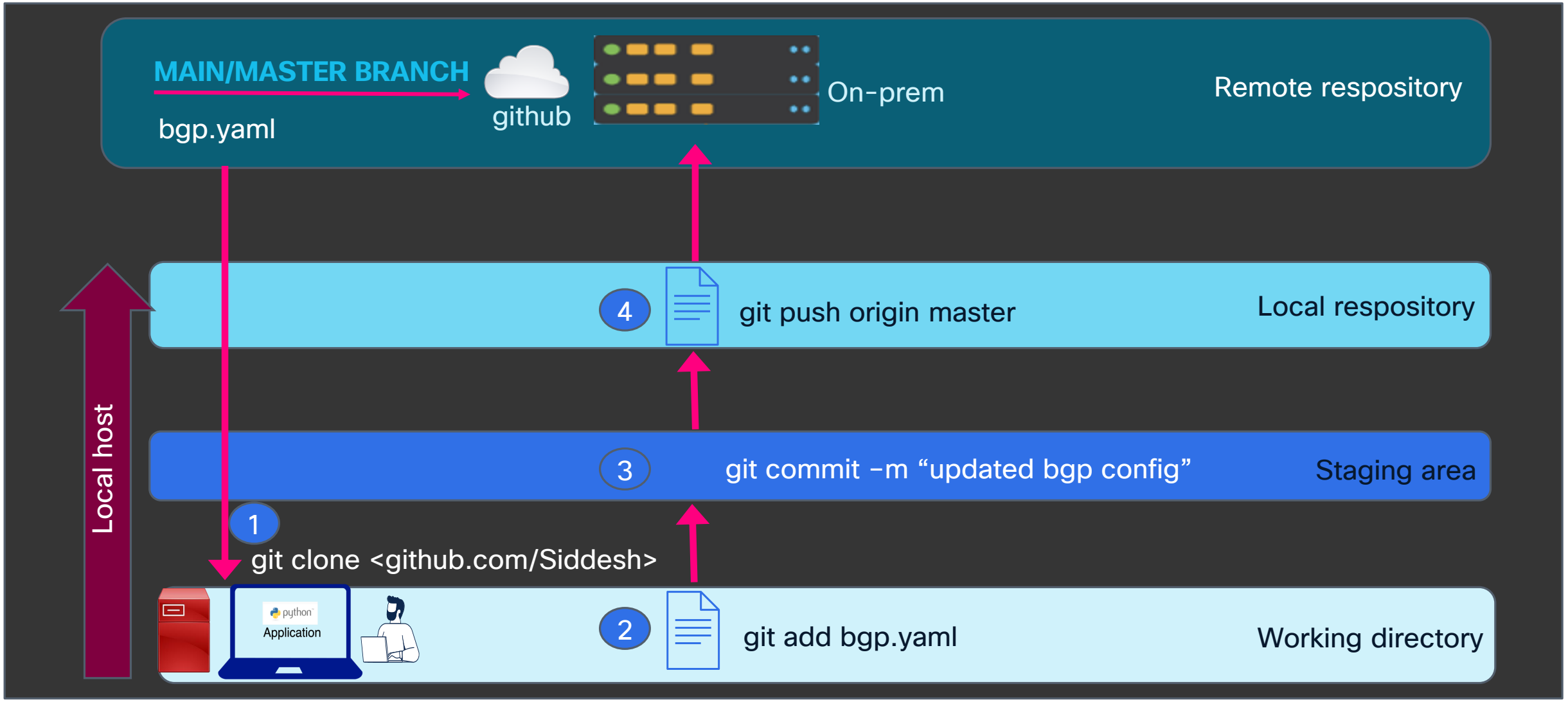
Main/Master ➔ Primary branch from which you want to push/pull the latest code changes.

© 2025 Cisco and/or its affiliates. All rights reserved.

GIT Architecture

© 2025 Cisco and/or its affiliates. All rights reserved.

GIT architecture – Workflow

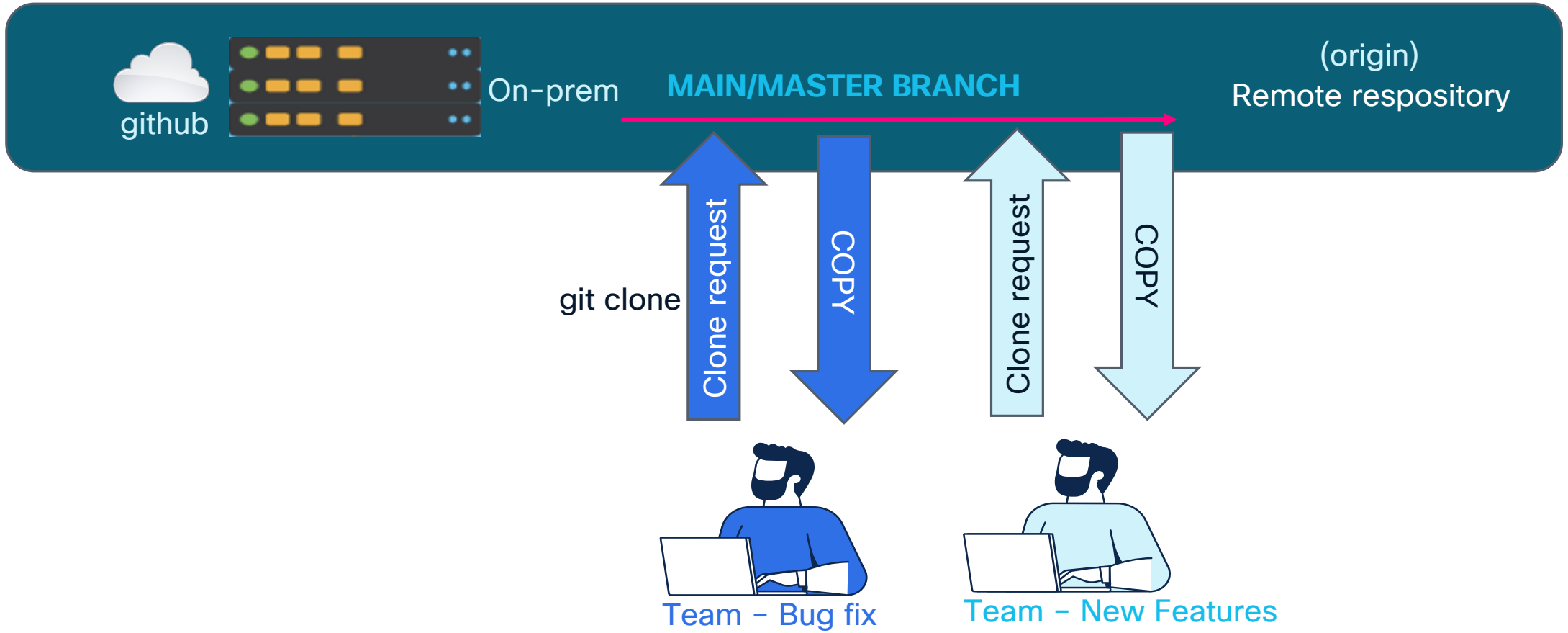


© 2025 Cisco and/or its affiliates. All rights reserved.

GIT Branches

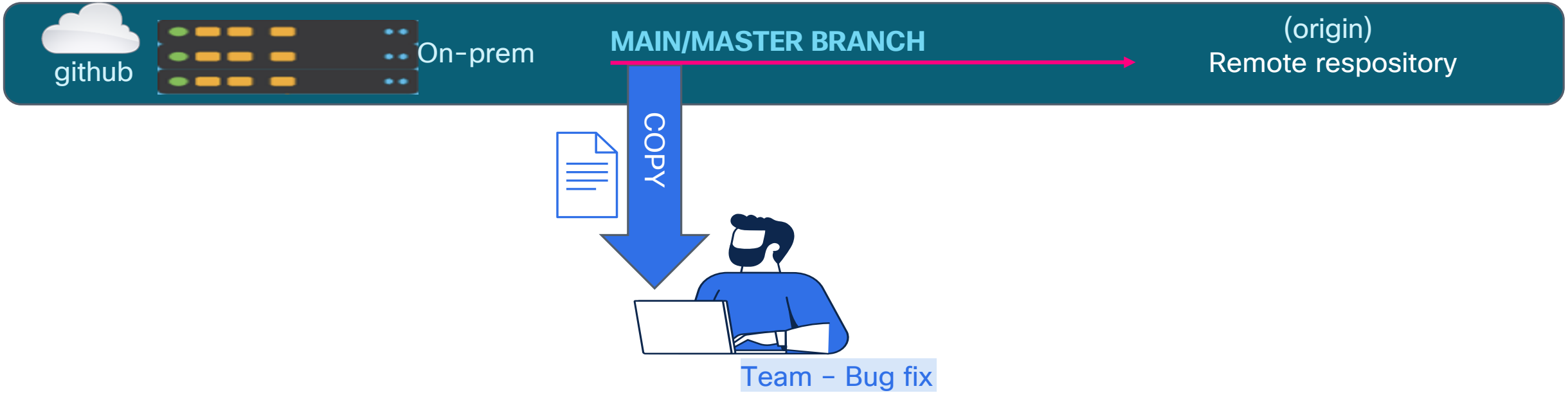
© 2025 Cisco and/or its affiliates. All rights reserved.

GIT branches



© 2025 Cisco and/or its affiliates. All rights reserved.

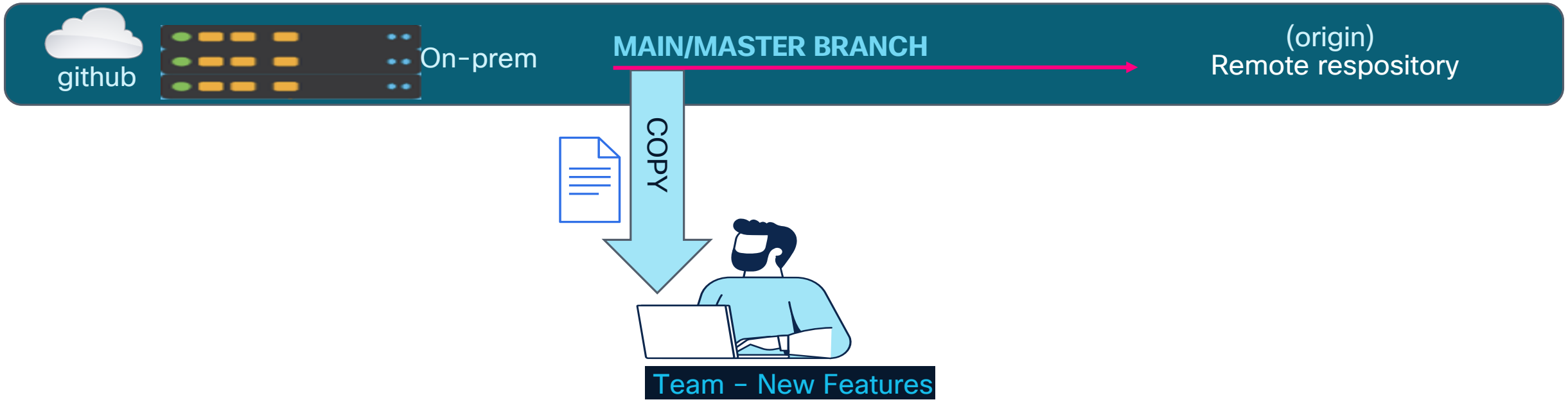
GIT branches




```
cisco@cisco-vm:~$ git checkout -b bug _fix
 Switched to a new branch 'bug_fix'
cisco@cisco-vm :~$ git branch
* bug_fix
  main
```

© 2025 Cisco and/or its affiliates. All rights reserved.

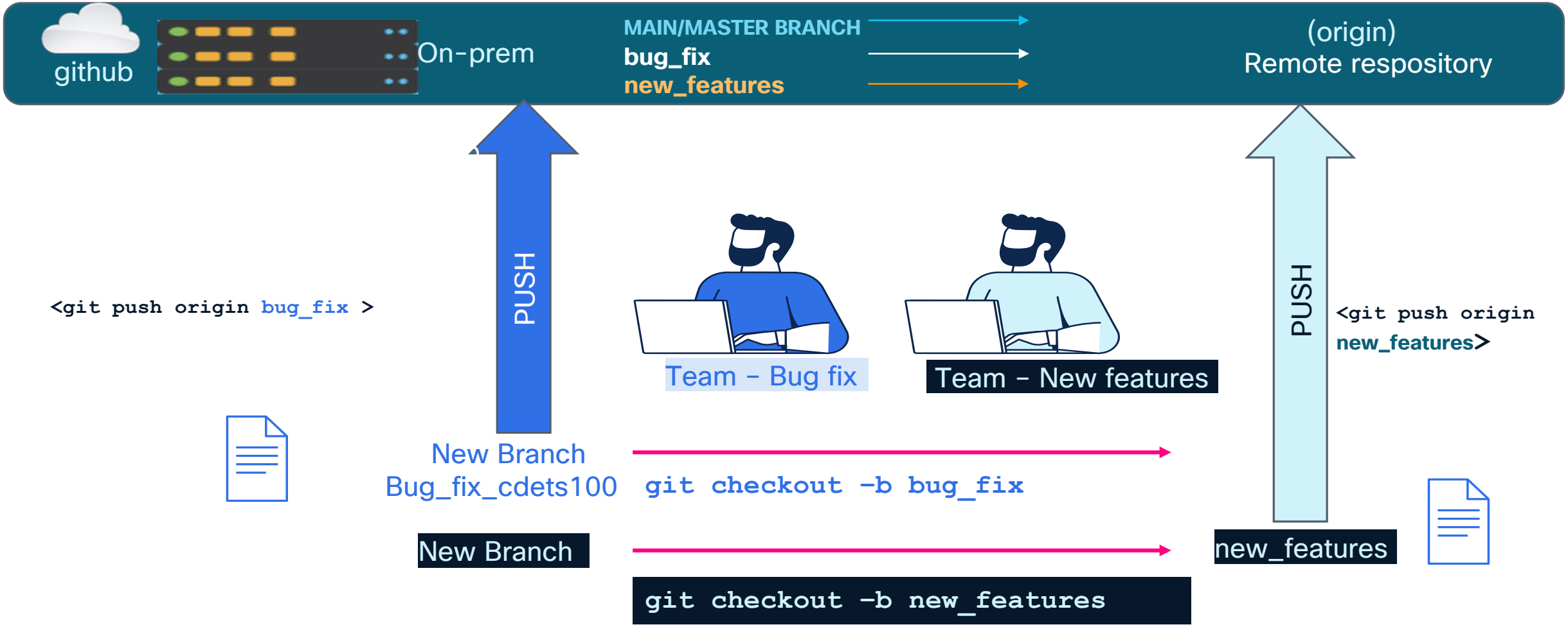
GIT branches




```
cisco@cisco-vm:~$ git checkout -b new_features
Switched to a new branch 'new_feature2'
cisco@cisco-vm :~$ git branch
* new_feature2
  main
```

© 2025 Cisco and/or its affiliates. All rights reserved.

GIT branches



© 2025 Cisco and/or its affiliates. All rights reserved.

Start your first GIT code checkin

© 2025 Cisco and/or its affiliates. All rights reserved.

GIT Code Check-in visualization

1 Create a username and email to associate with the GIT application.

```
student@student-vm:~/<--dir path-->$
student@student-vm:~/<-dir path->$git config--global user.email "student@cisco.com"
student@student-vm:~/<-dir path->$ git config -global user. name "cisco student"
student@student-vm:~/<-dir path->$ git config --list
user.email=student@cisco.com
user.name=cisco student
student@student-vm:~/labs/lab05$
```

2 Create a new directory which can be used as our working directory.

```
student@student-vm:~/<-dir path->$
student@student-vm:~/<-dir path->$ mkdir working_directory
student@student-vm:~/<-dir path->$ cd working_directory
student@student-vm:~/<-dir path->/working_directory$
```

© 2025 Cisco and/or its affiliates. All rights reserved.

GIT Code Check-in visualization (Ctnd...)

3 Inside your new working directory,Clone the remote repository where we want to make new changes

```
student@student-vm:~/<-dir path->/working_directory$
student@student-vm:~/<-dir path->/working_directory$ git clone
/home/student/git/backups.git
Cloning into 'backups'
...
done.
student@student-vm:~/<-dir path->/working_directory$
```

4 Verify the remote repository which “origin” is pointing in your local host.

```
student@student-vm:~/<-dir path->/working_directory/backups$
student@student-vm:~/<-dir path->/working_directory/backups$ git remote -v
origin /home/student/git/backups.git (fetch)
origin /home/student/git/backups.git (push)
student@student-vm:~/<-dir path->/working_directory/backups$
```

© 2025 Cisco and/or its affiliates. All rights reserved.

GIT Code Check-in visualization (Ctnd...)

5 Verify status of git tracker before and after making any new changes in the working directory

```
student@student-vm:~/<-dir path->/working_directory/backups$ git status
On branch master
Your branch is up to date with 'origin/master'.
nothing to commit, working tree clean
student@student-vm:~/<-dir path->/working_directory/backups$|
```

BEFORE

```
student@student-vm:~/<-dir path->/working_directory/backups/configs$ git status
On branch master
Your branch is up to date with 'origin/master'
Changes not staged for commit:
(use "git add <file>..." to update what will be committed)
(use "git restore <file>..." to discard changes in working directory)
      modified: r1.cfg
Untracked files:
(use "git add <files..." to include in what will be committed)
      r4.cfg
no changes added to commit (use "git add" and/or "git commit -a")
```

AFTER

© 2025 Cisco and/or its affiliates. All rights reserved.

GIT Code Check-in visualization (Ctnd...)

6 Staging your files which needs to be tracked by GIT and verify git tracking the files.

```
student@student-vm:~/<-dir path->/working_directory/backups/configs$
student@student-vm:~/<-dir path->/working_directory/backups/configs$ git add --all
student@student-vm:~/<-dir path->/working_directory/backups/configs$ git status
On branch master
Your branch is up to date with 'origin/master'
Changes to be committed:
(use "git restore --staged < file>..." to unstage)
        modified:    r1.cfg
        new file:    r4.cfg
student@student-vm:~/<-dir path->/working_directory/backups/configs$
```

7 Commit the new changes from your working directory to git local repository.

```
student@student-vm:~/<-dir path->$ git commit -m "Updated r1.cfg and file r4.cfg"
[master 4be3a74] Updated r1.cfg and file r4.cfg
2 files changed, 10 insertions+)
create mode 100644 configs/r4.cfg
student@student-vm:~/<-dir path->/working_directory/backups/configs$
```

© 2025 Cisco and/or its affiliates. All rights reserved.

GIT Code Check-in visualization (Ctnd...)

8 Verify your latest commit to the git tracked local repository.

```
student@student-vm:~/<-dir path->/working_directory/backups/configs$ git log -1
commit 4be3a74beb7f655b54e4a647013d808e0f1f2241 (HEAD -> master)
Author: cisco student «student@cisco.com›
Date:
Wed Apr 23 01:15:50 2025 +0000
I
Updated r1.cfg and file r4.cfg
student@student-vm:~/<-dir path->/working_directory/backups/configs$
```

9 Verify git status of the local repository post commit operation w.r.t remote repository

```
student@student-vm:~/<-dir path->/working_directory/backups/configs$
student@student-vm:~/<-dir path->/working_directory/backups/configs$ git status
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)
nothing to commit, working tree clean
student@student-vm:~/<-dir path->/working_directory/backups/configs$
```

© 2025 Cisco and/or its affiliates. All rights reserved.

GIT Code Check-in visualization (Ctnd...)

10 Push your new changes from local repository to master/main branch in the remote repository.

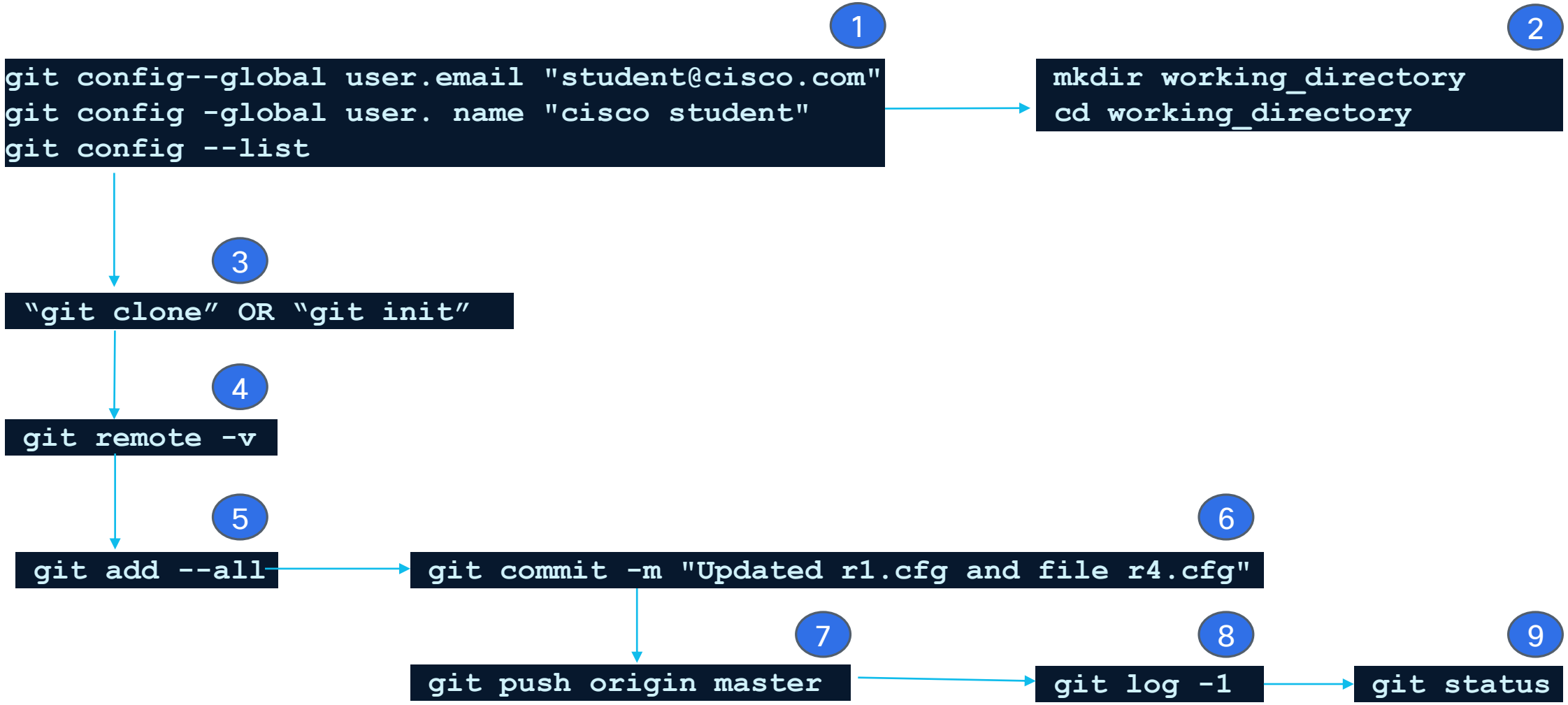
```
student@student-vm:~/<-dir path->/working_directory/backups/configs$
student@student-vm:~/<-dir path->$ git push origin master
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 4 threads Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 606 bytes | 606.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0)
I
To /home/student/git/backups.git
d02af60..4be3a74 master -> master
```

11 Verify git status of the local repository post "push" operation w.r.t remote repository

```
student@student-vm:~/<-dir path->/working_directory/backups/configs$ git status
On branch master
Your branch is up to date with 'origin/master'.
nothing to commit, working tree clean
student@student-vm:~/<-dir path->/working_directory/backups/configs$
```

© 2025 Cisco and/or its affiliates. All rights reserved.

GIT Code Check-in visualization



© 2025 Cisco and/or its affiliates. All rights reserved.

Demo – Navigating Devnet documentation

© 2025 Cisco and/or its affiliates. All rights reserved.

Call To Action – Next 24 hrs

© 2025 Cisco and/or its affiliates. All rights reserved.

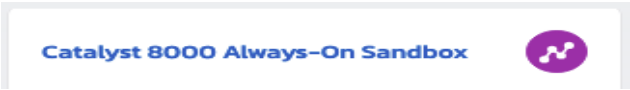
Call to action – Next Steps

Milestone 1: Push your first basic script developed on VSCode to GitHub.
Milestone 2: Execute your first NetConf and RestConf RPC on YangSuite on Cisco learning Labs
Milestone 3: Execute your first Rest API on Postman to perform a GET operation on Catalyst center sandbox

Milestone 1: Push your first basic script developed on VSCode to GitHub.

1. Download VS code and execute a basic script to print your name.
2. Create a github account.
3. Push your basic script to your master branch in your newly created GitHub repository using the suggested steps explained in the git process.

Milestone 2: Execute your first NetConf and RestConf RPC on YangSuite on Cisco learning Labs.

1. Go to YangSuite learning labs ➔Yangsuite learning lab – Hyperlink (Login using gmail/CCO/GitHub)
2. Reserve the Catalyst 8Kv always on Sandbox – Hyperlink for YangSuite learning lab 

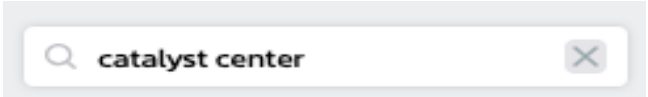
3. Follow steps in YangSuite learning labs to implement your first NetConf RPC and generate your first NetConf python code on YangSuite learning lab.

© 2025 Cisco and/or its affiliates. All rights reserved.

Call to action – Next Steps

Milestone 3: Execute your first Rest API on Postman to perform a GET operation on Catalyst center sandbox.

1. Reserve the Devnet Sandbox - Hyperlink Hyperlink and search “catalyst center”



3. Go to www.postman.com and signup and sign in.
4. Top left navigate to “Workspace ➔ My workspace ➔ New ➔ HTTP.
5. Go to Catalyst Center API doc https://developer.cisco.com/docs/dna-center/2-3-7-9/

   Navigate to “API Reference ➔ API ➔ Authentication API”
6. On Postman, build your HTTP header, URL and Body. Hostname in url is the sandbox dnac hostname.
7. Catalyst Center API doc, navigate to “API Reference ➔ API ➔ Devices ➔ Get Device list” https://developer.cisco.com/docs/dna-center/2-3-7-9/get-device-list/
8. On Postman, build your HTTP header, URL and Body following API doc and perform GET to retrieve device list from the Catalyst center.

© 2025 Cisco and/or its affiliates. All rights reserved.

REFERENCES

© 2025 Cisco and/or its affiliates. All rights reserved.

References – Yangsuite and Yang



Yangsuite Official documentation

https://pubhub.devnetcloud.com/media/yang-suite/docs/index.html

https://developer.cisco.com/docs/yangsuite/welcome-to-cisco-yang-suite/

Cisco devnet learning lab - YangSuite

Yangsuite learning lab - Hyperlink

Yang Models – Official github link to clone yang models across all vendors and standards

Yang Models - Github - Hyperlink

Advanced NetConf Explorer(ANX) - Official GitHub - Open source YangSuite alternative for NetConf

Advanced NetConf Explorer - Official GitHub - Hyperlink

Pyang – Official GitHub Link – Read me to explore available features and capabilities of PYANG tool

PYANG - Official GitHub link

© 2025 Cisco and/or its affiliates. All rights reserved.

References – CCO Documentation, Whitepapers, Articles



CCO Documentation for enabling Netconf and RestConf on IOS-XE, IOS-XR and NX-OS

CCO Documentation - NetConf IOSXR- Hyperlink
CCO Documentation - NetConf RestConf IOS-XE - Hyperlink
CCO Documentation - NetConf RestConf NXOS - Hyperlink

CCO - Master White Paper - Model driven programmability – NetConf,RestConf,gNMI

CCO - Master White Paper - Model driven programmability - Hyperlink

XR Docs article - YangSuite

XR Docs article - YangSuite - Hyperlink



Video Tutorials – Devnet Official Youtube channel - Playlists

Devnet YouTube playlist - Hyperlink

© 2025 Cisco and/or its affiliates. All rights reserved.

References – API, Postman collection and python code

Python code samples for NetConf and RestConf implementation.

Python code samples - GitHub - Hyperlink

Postman Collection Cisco devnet public Workspaces

Postman Collection - Cisco Devnet. - Hyperlink

Devnet Official API documentation for all products

Devnet API Documentation- Hyperlink

CMSE

Cisco Metrics Search Engine tool – Netconf,RestConf,gNMI for all Cisco platform

CMSE Tool - Hyperlink

© 2025 Cisco and/or its affiliates. All rights reserved.

Complete Your Session Evaluations



Complete a minimum of 4 session surveys and the Overall Event Survey to be entered in a drawing to win 1 of 5 full conference passes to Cisco Live 2026.



Earn 100 points per survey completed and compete on the Cisco Live Challenge leaderboard.



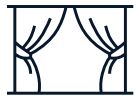
Level up and earn exclusive prizes!



Complete your surveys in the Cisco Live mobile app.

© 2025 Cisco and/or its affiliates. All rights reserved.

Continue your education



Visit the Cisco Showcase for related demos



Book your one-on-one Meet the Engineer meeting



Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs



Visit the On-Demand Library for more sessions at www.CiscoLive.com/on-demand

Contact me at: sinayak@cisco.com

© 2025 Cisco and/or its affiliates. All rights reserved.

Thank you

cisco Live !


CISCO