Cisco - Designing and Implementing Cisco Network Programmability

Designing and Implementing Cisco Network Programmability v.1.0 (NPDESI) is a 5 Day course that addresses the evolving role of network engineers towards more programmability, automation and orchestration, enabling them to leverage the powerful level of abstraction provided by controller based architectures to create real added value. This course is aligned with the topics covered by the 300-550 NPDESI exam, leading to the Cisco Network Programmability Design and Implementation Specialist Certification. The goal of the course is to provide the student a foundation to get started with network programmability by learning how to minimize the amount of manual interactions ("CLI") with the network and increase the use of scripts and automation tools to drive down operational inefficiencies. Within these goals, the learner reviews network programmability fundamentals such as using Linux and Python, common automation protocols such as NETCONF and REST, how they relate to YANG data models, SDN controller northbound and southbound APIs, how to use device APIs on various Cisco platforms, and how to get started using automation tools such as Ansible and Puppet. To participate in the hands-on labs in this class, you need to bring a laptop computer with the following:

- We recommend using at least a Pentium 4 or better and 1 GB of RAM or more.
- We recommend running Windows XP Professional SP3 or greater (Vista & Windows 7/8).
- Mac & Linux machines are also supported.
- Browser Requirements: Internet Explorer 10 or greater or Mozilla Firefox. (Safari and Mozilla Firefox for Mac OSX)
- All students are required to have administrator rights to their PCs and cannot be logged in to a domain using any Group Policies that will limit their machine’s capabilities.
- If you do not have administrator rights to your PC, you at least need permissions to download, install, and run Cisco Any Connect Client.
- If you are participating in a WebEx event, it is highly recommended to take this class at a location that has bandwidth speeds at a minimum of 1 Mbps bandwidth speeds.

Skills Gained
Upon completing this course, the learner will be able to meet these overall objectives:

- Learn how to minimize the amount of manual interactions ("CLI") with the network, and increase the use of scripts and automation tools to drive down operational inefficiencies.
- Review network programmability fundamentals including Linux and Python, common automation protocols such as NETCONF and REST and how they relate to YANG data models.
- Learn about SDN controllers including APIC, APIC-EM and OSC, as well as how to use device-level APIs such as Cisco NXOS, IOS-XE, IOS-XR and ASA OS.
Who Can Benefit

- Network Engineer
- Field Engineer
- System Engineer
- Network Designer
- Network Operations
- Network Automation Engineer
- Network Programmer
- Network Developer
- Software Engineer
- Application Developer

Prerequisites

The learner is expected to have the following skills and knowledge before attending this course:

- Baseline Python knowledge
- CCNA/CCNP level network background

Course Details

Note

Note: Students registering for this course will be receiving their course kit in a digital format. To be able to view your digital kit, it is recommended that you bring a laptop PC and/or a compatible tablet to be able to view the course materials and labs on separate screens. The recommended system requirements and instructions to access the course kit content can be found at the following link: Digital Course Kit Requirements and Instructions. Please also be aware that this digital version is designed for online use, not for printing. You can print up to 10 pages only in each guide within a course. Please note that every time you click the Print button in the book, this counts as one page printed, whether or not you click OK in the Print dialog.

Module 1: Network Programmability Fundamentals

Lesson 1: Introduction to Network Programmability

- Understanding Software-Defined Networking
- Traditional versus Software-Defined Networks
- Current Industry Trends
- Open Source Software
- Programmable Infrastructure
- Software Defined Networking
- DevOps
- Cisco ACI
- Network Programmability and Automation
- Uses of Network Automation
Lesson 2: Linux Primer for Network Engineers

- Why Learn Linux?
- Navigating the Linux File System
- Paths and Directories
- Linux Distributions and Package Managers
- Package Management
- Working with Files and Directories
- File Permissions
- Linux Processes

Lesson 3: Linux Networking

- Basic Linux Networking Commands
- Viewing the Routing and ARP Tables
- Persistent Network Configuration

Lesson 4: Python Foundations for Network Engineers – Part 1

- Understanding Python
- Executing Python Code
- Python Helper Utilities and Function
- Writing Idiomatic Python
- Common Python Data Types
- Variable Assignment
- Data Types: Strings
- Printing Strings
- Concatenating Strings
- String Built in Methods
- Data Type: Numbers
- Data Type: Booleans
- Conditionals
- Challenge
  - Challenge 3: Python Foundations—Part 1
  - Task 1: Create Strings
  - Task 2: Navigate the Built-In Help Features
  - Task 3: Use Built-in Methods of Strings
  - Task 4: Use Conditionals

Lesson 5: Python Foundations for Network Engineers – Part 2

- Lists
- Dictionaries
- Loops
- Function
- Working with Files
- Challenge
  - Challenge 4: Python Foundations – Part 2
  - Task 1: Working with Lists
  - Task 2: Working with Dictionaries
  - Task 3: Using a For Loop
  - Task 4: Working with Functions

Lesson 6: Writing and Troubleshooting Python Scripts

- Writing Scripts
- Executing Scripts
- Analyzing Code
- Error Handling
- Challenge
  - Challenge 5: Writing and Troubleshooting Python Scripts
  - Task 1: Writing a Hello World Script
  - Task 2: Writing a Network Script
  - Task 3: Analyzing and Troubleshooting a Script

Lesson 7: Python Libraries

- Python Libraries
Module 2: APIs and Automation Protocols
Lesson 1: Introduction to Network APIs and Protocols

- Evolution of Device Management and Programmability
- Model-Driven Programmability Stack
- Data Encoding Formats
- JSON
  - Discovery 1: Working with JSON Objects in Python
  - Task 1: Working with JSON Objects in Python
- XML
  - Discovery 2: Using XML in Python
  - Task 1: Learning Lab Procedure
- Data Models
- YANG
  - Model-Driven Programmability
  - Model-Driven Programmability Stack
- REST
- NETCONF
  - Discovery 3: NETCONF Capabilities Exchange
  - Task 1: Learning Lab Procedure
- RESTCONF
- gRPC
- Challenge

Lesson 2: Cisco ASA REST API

- Cisco ASA REST API Overview
- REST API Agent Pre-requisites
- User Credentials
- Cisco ASA REST API Documentation and Console
- Cisco ASA REST API Examples
- Challenge
  - Challenge 7: Interacting with Cisco ASA REST API
  - Task 1: Retrieve Interface Statistics Using Postman
Lesson 3: NX-OS Programmability

- Nexus Programmability Overview
- NX-API CLI—Part 1
- Discovery 4: NX-API Discovery
- Task 1: Learning Lab Procedure
- NX-API CLI—Part 2
- NETCONF
- Python on Box
- NX-API REST
- Challenge
  - Challenge 8: Consume NX-API with Python
  - Task 1: Consume NX-API with Python
  - Challenge 9: Consuming NETCONF on Nexus
  - Task 1: Using the xmlagent Service
  - Task 2: Consuming NETCONF in Python
  - Challenge 10: Using Python on the Nexus Switch
  - Task 1: Cisco CLI APIs
  - Task 2: Executing Python Script on Box

Lesson 4: Cisco IOS XE APIs

- Cisco IOS XE APIs Overview
- IOS XE RESTCONF API
- IOS XE NETCONF API
- Challenge
  - Challenge 11: Interacting with the Cisco IOS XE RESTCONF API
  - Task 1: Gather Data Using Postman
  - Task 2: Making Configuration Changes Using Postman
  - Task 3: Troubleshoot Python Script Consuming RESTCONF API
  - Challenge 12: Interacting with the Cisco IOS XE NETCONF API
  - Task 1: Consume NETCONF with Python

Lesson 5: Cisco IOS XR APIs

- IOS XR NETCONF APIs
- Challenge
  - Challenge 13: Interacting with the Cisco IOS XR NETCONF API
  - Task 1: Consume NETCONF with Python
Lesson 6: Securing the Management Plane

- Management Plane
- Access Control Lists
- Challenge
- Challenge 14: Securing the Management Plane
- Task 1: Create Access Control Lists

Module 3: Data Models
Lesson 1: YANG Data Modeling

- YANG Overview
- YANG Module
- YANG Module Header
- YANG Leaf Statement
- YANG Leaf-List Statement
- YANG List Statements
- Putting Things Together
- YANG Types
- YANG Typedef Statement
- YANG Choice Statement
- YANG Grouping Statement
- YANG Miscellaneous Statements
- YANG RPC Statement
- YANG Imports and Includes
- YANG Modules and Submodules
- YANG Model Examples
- XML
- JSON
- CLI
- Challenge

Lesson 2: YANG Tools

- YANG Validator
- Discovery 5: Validating YANG Models Using yangvalidator
- Task 1: Using yangvalidator
- Challenge 15: Writing a Custom YANG Model
- Task 1: Validate Device Configurations
- pyang
- Discovery 6: Viewing and Validating YANG Models with pyang
- Task 1: Viewing and Validating YANG Models with pyang
- YANG Development Kit
Module 4: SDN Controllers

Lesson 1: Introduction to Controller Networking

- Origins of Controller Based Networking
- OpenFlow
- OpenFlow Deployment Models
- Challenge

Lesson 2: OpenDaylight

- OpenDaylight Overview
- OpenDaylight Architecture
- OpenDaylight Use Cases
- Challenge

Lesson 3: Cisco APIC-EM

- APIC-EM Overview
- APIC-EM Web User Interface Dashboard
- APIC-EM Platform Architecture
- Supported Devices
- APIC-EM Common Tasks
- Discovery 10: Network Discovery Configuration
- Task 1: Performing a Network Discovery
- Performing APIC-EM Tasks
- APIC-EM Applications
- APIC-EM APIs
- Using Postman with APIC-EM
- Challenge
- Challenge 16: APIC-EM Network Discovery and RBAC
- Task 1: Create Saved SNMP String Values in Preparation for the Discovery
Lesson 4: Cisco Application Centric Infrastructure

- Task 2: Create Saved CLI Credentials in Preparation for the Discovery
- Task 3: Perform a New Network Discovery
- Task 4: Creating Internal Users
- Challenge 17: Consuming the APIC-EM API
- Task 1: Postman Basic Tasks
- Task 2: Generating Python Code in APIC-EM

Before Getting Started
- Task 1: Consuming the APIC REST API with Postman
- Task 2: Construct Python Scripts Using Requests

ACI Overview

Discovery 11: ACI Fabric Discovery
- Task 1: ACI Fabric Discovery

Discovery 12: Creating Objects with APIC GUI
- Task 1: Creating Objects with APIC GUI

ACI Object Model

Discovery 13: Navigating the Object Model
- Task 1: Navigating the Object Model

APIC REST API

Discovery 14: Using API Inspector
- Task 1: Learning Lab Procedure

Using Postman REST Client

Cobra SDK

Arya

Discovery 15: Using ARYA
- Task 1: Using ARYA

ACI Toolkit

ACI Toolkit Applications — CLI

Discovery 16: CLI Emulation
- Task 1: CLI Emulation

ACI Toolkit Applications — Diagrams Tool

Discovery 17: ACI Diagram Tool
- Task 1: ACI Diagram Tool

ACI Toolkit Applications — Lint

Discovery 18: ACI Toolkit — Lint
- Task 1: ACI Toolkit — Lint

ACI Toolkit Applications — Cable Plan

ACI Toolkit Applications — Event Feeds

ACI Toolkit Applications — Fake APIC

Challenge 18: Using the APIC REST API
Lesson 5: Cisco Element and Domain Managers

- Cisco Virtual Topology System
- Cisco Nexus Data Broker
- Cisco Network Service Orchestrator
- WAN Automation Engine
- Cisco UCS Manager
- Cisco UCS Director
- Challenge

Module 5: Network Operations
Lesson 1: Software Development Methodologies

- Software is Everywhere
- Waterfall
- Lean
- Agile
- Challenge
- Reference

Lesson 2: Introduction to DevOps

- Dev and Ops – The Problem
- DevOps Demystified
- DevOps Tools and Technologies
- Challenge

Lesson 3: Version Control

- Version Control Systems
- Overview of Git
- Git Architecture
- Git Commands
- Git Workflow
- Git Branches
- Using Git
- Collaborating with GitHub
- Discovery 19: GitHub Pull Request
- Task 1: Learning Lab Procedure
- GitHub Pull Request: Fork and Pull
- Changing Views
- Challenge
Lesson 4: Automated Testing

- Network Test Infrastructure
- Network Function Virtualization
- VRL
- DevNet
- DevNet Sandbox
- DevNet Learning Labs
- DevNet GitHub
- Discovery 20: DevNet Sandbox
- Task 1: DevNet Sandbox
- Discovery 21: DevNet Learning Labs
- Task 1: DevNet Learning Labs
- Discovery 22: DevNet GitHub
- Task 1: DevNet GitHub
- Network Testing
- Unit Tests
- Challenge
- Challenge 20: Integration Testing
- Task 1: Validate Device Configurations
- Reference

Lesson 5: Continuous Integration

- Introduction to Continuous Integration
- Travis CI
- Challenge

Lesson 6: Configuration Management and Automation Tools

- Configuration Management
- Ansible Overview
- Ansible Base Modules
- NXOS Features Modules
- Puppet
- Puppet Components
- Puppet Server
- Puppet Manifests
- Puppet Resources
- Puppet Node and Agent
- Puppet Agent Installation
- Puppet Installation with OAC
- Puppet Installation with Guestshell
- Puppet Installation with Bash-shell
- Final Steps: Puppet Agent Setup
- Managing Interfaces
- Managing OSPF
- Challenge
- Challenge 21: Compliance Checks with Ansible
- Task 1: Verify Config Register and IOS Version
- Challenge 22: Tenant Provisioning with Ansible
- Task 1: Deploy Tenants with Ansible
- References

### Schedule (as of 3 )

<table>
<thead>
<tr>
<th>Date</th>
<th>Location</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sep 9, 2019 – Sep 13, 2019</td>
<td>Virtual</td>
<td>GTR</td>
<td>Enroll</td>
</tr>
<tr>
<td>Aug 26, 2019 – Aug 30, 2019</td>
<td>Virtual</td>
<td></td>
<td>Enroll</td>
</tr>
<tr>
<td>Sep 23, 2019 – Sep 27, 2019</td>
<td>Virtual</td>
<td></td>
<td>Enroll</td>
</tr>
<tr>
<td>Oct 14, 2019 – Oct 18, 2019</td>
<td>Virtual</td>
<td></td>
<td>Enroll</td>
</tr>
<tr>
<td>Oct 28, 2019 – Nov 1, 2019</td>
<td>Virtual</td>
<td></td>
<td>Enroll</td>
</tr>
<tr>
<td>Nov 18, 2019 – Nov 22, 2019</td>
<td>Virtual</td>
<td></td>
<td>Enroll</td>
</tr>
<tr>
<td>Nov 18, 2019 – Nov 22, 2019</td>
<td>Virtual</td>
<td></td>
<td>Enroll</td>
</tr>
<tr>
<td>Dec 16, 2019 – Dec 20, 2019</td>
<td>Virtual</td>
<td></td>
<td>Enroll</td>
</tr>
</tbody>
</table>