Python for Network and Systems Automation

Duration: 5 Days

Course Summary

This 5-day personalised 1-on-1 training programme is designed for professionals transitioning into network and systems automation. Starting from Python fundamentals and building progressively toward real-world automation of network devices and infrastructure systems, the programme is entirely hands-on and tailored to the participant's own background and pace. By the end of the week, the participant will have built a working personal automation tool and a clear 30-day roadmap for continuing their automation journey independently. All tools used are open source.

Target Audience

This programme is designed for:

- Network and systems engineers moving into automation roles
- Infrastructure professionals with no prior Python experience
- Individuals seeking a structured, personalised path into scripting
- Anyone wanting to automate repetitive network and systems tasks

Day 1: Python Fundamentals for Automation

Module 1: Python Basics — Variables, Data Types and Control Flow

- Variables and data types
- Strings and numbers
- Lists and dictionaries
- if/else conditions
- for and while loops

Lab Exercises:

1. Write a Python script that stores device names and IP addresses in a dictionary and prints a formatted inventory report.
2. Write a script using loops and conditions that reads a list of hostnames and flags any that do not follow a defined naming convention.

Module 2: Functions, Modules and File Handling

- Defining functions
- Arguments and return values
- Importing modules
- Reading and writing files
- Working with CSV and JSON

Lab Exercises:

3. Write a function that reads a CSV file of network devices and returns a filtered list of devices matching a given site or device type.
4. Write a script that reads a JSON configuration file, extracts key device parameters, and writes a formatted summary to a text file.

Day 2: Connecting to Network Devices with Python

Module 3: SSH and Telnet Automation with Netmiko

- SSH basics for automation
- Netmiko overview
- Connecting to devices
- Sending commands
- Parsing command output

Lab Exercises:

5. Use Netmiko (open source) to SSH into a simulated network device, run show commands, and save the output to a file.
6. Write a script that connects to multiple devices from a list, runs the same command on each, and compiles results into a single report.

Module 4: Automating Configuration with Netmiko and Paramiko

- Sending config commands
- Multi-device configuration
- Paramiko basics
- Error handling in SSH
- Logging connections

Lab Exercises:

7. Write a script using Netmiko that pushes a VLAN configuration to multiple simulated switches from a provided device list.
8. Add error handling and logging to the script so failed connections are recorded to a log file without stopping the rest of the run.

Day 3: REST APIs and Network Automation Frameworks

Module 5: Working with REST APIs in Python

- REST API concepts
- HTTP methods
- requests library
- JSON parsing
- Authentication headers

Lab Exercises:

9. Use the Python requests library to call a public network device API, retrieve interface data, and print a formatted summary.
10. Write a script that authenticates to a REST API, retrieves device status, and flags any devices reporting a down state.

Module 6: NAPALM and Nornir for Multi-Vendor Automation

- NAPALM overview
- Multi-vendor support
- Getting and setting config
- Nornir task framework
- Inventory management

Lab Exercises:

11. Use NAPALM (open source) to retrieve and compare the running vs candidate configuration of a simulated network device.
12. Use Nornir (open source) to run a show command across a multi-vendor inventory in parallel and save per-device output to separate files.

Day 4: Systems Infrastructure Automation

Module 7: Linux and Windows Systems Automation with Python

- os and subprocess modules
- Running system commands
- File and directory automation
- Service management
- Log file parsing

Lab Exercises:

13. Write a Python script that checks disk usage, CPU load, and running services on a Linux host and sends an alert if any threshold is exceeded.
14. Write a script that scans a directory of log files, extracts lines containing ERROR or CRITICAL, and writes a consolidated report.

Module 8: Automation with Fabric and Ansible (Python-Driven)

- Fabric for remote execution
- SSH task automation
- Ansible with Python
- Playbook triggering from Python
- Inventory scripting

Lab Exercises:

15. Use Fabric (open source) to remotely execute a sequence of system health check commands across multiple Linux hosts from a single Python script.
16. Write a Python script that dynamically generates an Ansible inventory file from a provided device database and triggers a playbook run.

Day 5: Advanced Automation and Applied Project

Module 9: Scheduling, Alerting and Automation Pipelines

- schedule library
- Cron and task scheduling
- Email and Slack alerts
- Pipeline design
- Automation best practices

Lab Exercises:

17. Write a scheduled Python script using the schedule library that runs a network health check every 30 minutes and sends an email alert on failure.
18. Build a simple automation pipeline that collects device data, checks it against defined thresholds, and posts a summary notification to a Slack channel using a webhook.

Module 10: Applied Project — Personal Network Automation Tool

- Project scoping
- Device inventory script
- Config backup automation
- Health check automation
- 30-day learning roadmap

Lab Exercises:

19. Build a personal network automation tool that connects to a set of simulated devices, backs up their running configurations, and saves each to a timestamped file.
20. Extend the tool to run a health check, compare current config against a known-good baseline, flag any differences, and produce a per-device summary report.