Document Generated: 11/14/2026

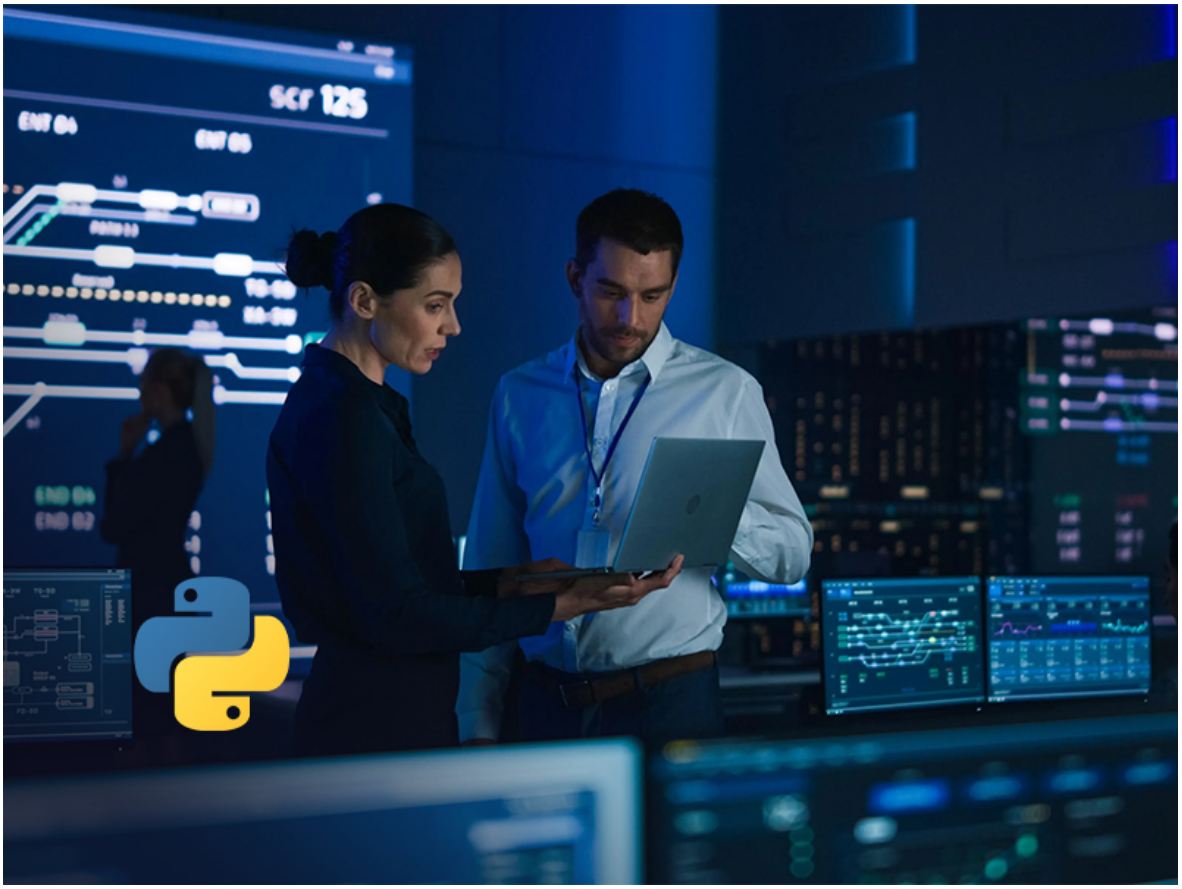
Learning Style: Online Instructor-Led

Technology:

Difficulty: Intermediate

Course Duration: 3 Days

Python Essentials for Networking & Systems Administration (TTPS4824)



About This Course:

Python Essentials for Networking & Systems Administration / SysAdmin is tailored for IT professionals, systems administrators, and network engineers who want to harness the power of Python to simplify and automate everyday tasks across distributed systems. Whether you're new to scripting or looking to expand your skillset, this course provides the perfect opportunity to build essential Python

expertise and apply it to real-world scenarios. Working in a hands-on lab environment, you'll start with foundational Python scripting essentials like file operations, regular expressions, and working with binary data, then progress to leveraging network-focused modules such as SSH, Git, and RESTful services. With a strong emphasis on practical application, this course ensures you're not just learning syntax but mastering the tools to solve real challenges in your role.

Course Objectives:

- Automate Networking and Administrative Tasks: Gain the ability to use Python scripts to automate repetitive tasks, such as configuring systems, managing files, and performing network diagnostics across distributed environments.
- Work with Python's Networking Libraries: Develop proficiency in leveraging Python's built-in and third-party libraries, including modules like Paramiko for SSH, requests for RESTful services, and socket programming for custom protocols.
- Handle Data Efficiently: Master the techniques for processing, manipulating, and storing data using Python, including working with text files, binary data, JSON, XML, and CSV formats.
- Develop Secure and Scalable Scripts: Learn best practices for creating efficient and secure Python scripts that can be easily scaled or integrated into existing workflows, ensuring reliability in production environments.
- Build Real-World Python Solutions: Apply the skills learned to complete practical projects, such as fetching web content, automating email tasks, and implementing multi-threaded scripts for improved system performance.

Audience:

- This introductory-level Python course is appropriate for advanced users, system administrators and web site administrators who want to use Python to support their server installations, as well as anyone else who wants to automate or simplify common tasks with the use of Python scripts. Students should have basic development experience in any programming language, along with a working, user-level knowledge of Unix/Linux, Mac, or Windows

Prerequisites:

- At least some prior hands-on experience with scripting or programming. You don't need to be an expert in either, but you should have had some exposure and should be coming from a technical background.

- Working with Unix or Linux, and familiarity with using the command line interface for simple tasks, such as file navigation and executing commands.

- Basic familiarity working with text editors like Notepad, or IDEs, would be helpful as the course includes hands-on lab sessions requiring code editing.

Course Outline:

1. The Python Environment

- Starting Python
- If the interpreter is not in your PATH
- Using the interpreter
- Trying out a few commands
- Running Python scripts
- Getting help
- Python Editors and IDEs

2. Variables and Values

- Using variables
- Keywords and Builtins
- Variable typing
- Strings
- String operators and methods
- Numeric literals
- Math operators and expressions
- Converting among types

3. Basic input and output

- Writing to the screen
- String Formatting
- Legacy String Formatting
- Command line parameters
- Reading from the keyboard

4. Flow Control

- About flow control
- What's with the white space?
- if and elif
- Conditional Expressions
- Relational Operators
- Boolean operators
- while loops
- Alternate ways to exit a loop

5. Array types

- Lists
- Tuples and unpacking
- Indexing and slicing
- Iterating through a sequence
- Functions for all sequences
- The range() function
- List comprehensions
- Generator Expressions

6. Working with Files

- Text file I/O
- Opening a text file
- The with block
- Reading a text file
- Writing to a text file

7. Dictionaries and sets

- When to use dictionaries?
- Creating dictionaries
- Getting dictionary values
- Iterating through a dictionary
- Reading file data into a dictionary
- Counting with dictionaries
- Creating Sets
- Working with sets

8. Functions, modules, packages

- Defining a function
- Returning values
- Function parameters
- Variable scope
- Creating Modules
- The import statement
- Where did __pycache__ come from?
- Module search path
- Packages

9. An Introduction to Python Classes

- About O-O programming
- Defining classes
- Constructors
- Instance methods
- Properties
- Class methods and data

- Static Methods
- Private methods
- Inheritance
- Untangling the nomenclature

10. Errors and Exception Handling

    - Syntax errors
    - Exceptions
    - Handling exceptions with try
    - Handling multiple exceptions
    - Handling generic exceptions
    - Ignoring exceptions
    - Using else
    - Cleaning up with finally

11. Efficient Scripting

    - Running external programs
    - Parsing arguments
    - Creating filters to read text files
    - Logging

12. Regular Expressions

    - Regular Expressions
    - RE Syntax Overview
    - Finding matches
    - RE Objects
    - Compilation Flags
    - Groups
    - Special Groups
    - Replacing text
    - Replacing with a callback
    - Splitting a string

13. Binary data

    - str vs bytes
    - Binary files
    - Structured binary data
    - Bitwise operations

14. Network Programming

    - Grabbing a web page
    - Consuming Web services
    - HTTP the easy way
    - sending e-mail
    - Email attachments

- Remote Access
- Copying files with Paramiko

15. Sockets

- Sockets
- Socket options
- Server concepts
- Client concepts
- Application protocols
- Forking servers

16. Multiprogramming

- Multiprogramming
- What Are Threads?
- The Python Thread Manager
- The threading Module
- Threads for the impatient
- Creating a thread class
- Variable sharing
- Using queues
- Debugging threaded Programs
- The multiprocessing module
- Using pools
- Alternatives to multiprogramming

17. Serializing Data: XML, XPath, JSON, CSV

- About XML
- Normal Approaches to XML
- Which module to use?
- Getting Started With ElementTree
- How ElementTree Works
- Elements
- Creating a New XML Document
- Parsing An XML Document
- Navigating the XML Document
- Using XPath
- About JSON
- Reading JSON
- Writing JSON
- Customizing JSON
- Reading CSV data
- Nonstandard CSV
- Using csv.DictReader
- Writing CSV Data

Bonus Chapters / Time Permitting

18. Sorting

- Sorting Overview
- The sorted() function
- Custom sort keys
- Lambda functions
- Sorting nested data
- Sorting dictionaries
- Sorting in reverse
- Sorting lists in place