Python Programming for Security Analysts & Professionals (TTPS4894)

Category: Software Development & Programming | **Vendor:** Trivera Tech

Duration: 32.00 hours (4 days) **26.0 CPD Hours** **Rating:** ⭐ 4.6 (5,878 reviews)

Course Information

Delivery Format: Instructor Led - Online

Course Overview

This course is for experienced security professionals new to Python. You will gain practical, hands-on knowledge that leads you student from the basics of writing and running Python scripts to more advanced features. This course is tailored specifically for Security Analysts and others who wish to use Python functionality for security-related tasks such as log manipulation or forensics. This course is essential for security professionals that are performing security reviews and audits of Python applications or are supporting development teams in implementing better defenses in Python.

About This Course

This course is for experienced security professionals new to Python. You will gain practical, hands-on knowledge that leads you student from the basics of writing and running Python scripts to more advanced features. This course is tailored specifically for Security Analysts and others who wish to use Python functionality for security-related tasks such as log manipulation or forensics. This course is essential for security professionals that are performing security reviews and audits of Python applications or are supporting development teams in implementing better defenses in Python.

Who Should Attend

This course is tailored specifically for Security Analysts and others new to Python, who wish to learn and use Python functionality for security-related tasks such as log manipulation or forensics. Students are required to have some basic programming experience and exposure prior to attending this course. Students should have basic development experience in any programming language, along with a working, user-level knowledge of Unix/Linux, Mac, or Windows.

Learning Outcomes

Upon successful completion of this course, participants will be able to:

This course is approximately 50% hands-on, combining expert lecture, real-world demonstrations and group discussions with machine-based practical labs and exercises. Our engaging instructors and mentors are highly experienced practitioners who bring years of current 'on-the-job' experience into every classroom. Throughout the hands-on course students will learn to write essential Python scripts using the most current and efficient skills and techniques.

Working in a hands-on learning environment, guided by our expert team, attendees will learn to:

Create working Python scripts following best practices

Use python data types appropriately

Read and write files with both text and binary data

Search and replace text with regular expressions

Get familiar with the standard library and its work-saving modules

Use lesser known but powerful Python data types

Create 'real-world', professional Python applications

Work with dates, times, and calendars

Know when to use collections such as lists, dictionaries, and sets

Understand Pythonic features such as comprehensions and iterators

Write robust code using exception handling

Write Secure Python Applications

Perform Log File Analysis

Work with Security Filters, Packet Analysis and related Analytics

Optional: Working with RESTful Services

Detailed Course Outline

An Overview of Python

- What is python
- 1 -- An overview of Python
- What is python
- Python Timeline
- Advantages/Disadvantages of Python
- Getting help with pydoc

The Python Environment

- Starting Python
- Using the interpreter
- Running a Python script
- Python scripts on Unix/Windows
- Editors and IDEs

Getting Started

- Using variables
- Builtin functions
- Strings
- Numbers
- Converting among types
- Writing to the screen
- Command line parameters

Flow Control

- About flow control
- White space
- Conditional expressions
- Relational and Boolean operators
- While loops
- Alternate loop exits

Sequences

- About sequences
- Lists and list methods
- Tuples
- Indexing and slicing
- Iterating through a sequence
- Sequence functions, keywords, and operators
- List comprehensions
- Generator Expressions
- Nested sequences

Working with files

- File overview
- Opening a text file
- Reading a text file
- Writing to a text file
- Reading and writing raw (binary) data
- Converting binary data with struct

Dictionaries and Sets

- About dictionaries
- Creating dictionaries
- Iterating through a dictionary
- About sets
- Creating sets
- Working with sets

Functions

- Defining functions
- Parameters
- Global and local scope
- Nested functions
- Returning values

Sorting

- The sorted() function
- Alternate keys
- Lambda functions
- Sorting collections

Errors and Exception Handling

- Syntax errors
- Exceptions
- Using try/catch/else/finally
- Handling multiple exceptions
- Ignoring exceptions

Modules and Packages

- The import statement
- Module search path
- Creating modules and Using packages
- Function and Module aliases

Working with Classes

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

Regular Expressions

- RE syntax overview
- RE Objects
- Searching and matching
- Compilation flags
- Groups and special groups
- Replacing text
- Splitting strings

The standard library

- The sys module
- Launching external programs
- The string module
- Reading CSV data

Dates and times

- Working with dates and times
- Translating timestamps
- Parsing dates from text

Working with the file system

- Paths, directories, and filenames
- Checking for existence
- Permissions and other file attributes
- Walking directory trees
- Creating filters with fileinput
- Security and File Access

Network services

- Grabbing web content
- Detecting Malformed Input

Writing secure Python applications

- Parsing command-line options
- Getting help with pydoc
- Safely handling untrusted data
- Managing eval() permissions
- Potential insecure packages
- Embedding code snippets in Python
- Embedding authentication data in Python
- Potentially dangerous operations:
- File access
- Operating system access
- Calls to external services
- Called to external data sources
- Static analysis tools such as Bandit

Log File Analysis

- Raw log file manipulation
- Fail2Ban
- Customizing Fail2Ban with Python

Security Filters

- SQL-Injection Detection
- ModSecurity CRS filtering

Packet Analysis

- Packet Sniffing in Python

Analytics

- Security Logging and Analytics
- Attack Detection and Defense
- Python and Spark High-Level Overview

RESTful Web Services

- What is Flask
- Developing a Flask Web service
- Mapping resources using URLs
- Mapping resources using HTTP
- Negotiating data content

, An Overview of Python

What is python

Python Timeline

Advantages/Disadvantages of Python

Getting help with pydoc

The Python Environment

Starting Python

Using the interpreter

Running a Python script

Python scripts on Unix/Windows

Editors and IDEs

Getting Started

Using variables

Builtin functions

Strings

Numbers

Converting among types

Writing to the screen

Command line parameters

Flow Control

About flow control

White space

Conditional expressions

Relational and Boolean operators

While loops

Alternate loop exits

Sequences

About sequences

Lists and list methods

Tuples

Indexing and slicing

Iterating through a sequence

Sequence functions, and operators

List comprehensions

Generator Expressions

Nested sequences

Working with files

File overview

Opening a text file

Reading a text file

Writing to a text file

Reading and writing raw (binary) data

Converting binary data with struct

Dictionaries and Sets

About dictionaries

Creating dictionaries

Iterating through a dictionary

About sets

Creating sets

Working with sets

Functions

Defining functions

Parameters

Global and local scope

Nested functions

Returning values

Sorting

The sorted() function

Alternate keys

Lambda functions

Sorting collections

Errors and Exception Handling

Syntax errors

Exceptions

Using try/catch/else/finally

Handling multiple exceptions

Ignoring exceptions

Modules and Packages

The import statement

Module search path

Creating modules and Using packages

Function and Module aliases

Working with Classes

About o-o programming

Defining classes

Constructors

Methods

Instance data

Properties

Class methods and data

Regular Expressions

RE syntax overview

RE Objects

Searching and matching

Compilation flags

Groups and special groups

Replacing text

Splitting strings

The standard library

The sys module

Launching external programs

The string module

Reading CSV data

Dates and times

Working with dates and times

Translating timestamps

Parsing dates from text

Working with the file system

Paths, and filenames

Checking for existence

Permissions and other file attributes

Walking directory trees

Creating filters with fileinput

Security and File Access

Network services

Grabbing web content

Detecting Malformed Input

Writing secure Python applications

Parsing command-line options

Getting help with pydoc

Safely handling untrusted data

Managing eval() permissions

Potential insecure packages

Embedding code snippets in Python

Embedding authentication data in Python

Potentially dangerous operations:

File access

Operating system access

Calls to external services

Called to external data sources

Static analysis tools such as Bandit

Log File Analysis

Raw log file manipulation

Fail2Ban

Customizing Fail2Ban with Python

Security Filters

SQL-Injection Detection

ModSecurity CRS filtering

Packet Analysis

Packet Sniffing in Python

Analytics

Security Logging and Analytics

Attack Detection and Defense

Python and Spark High-Level Overview

Bonus Content / Time Permitting

RESTful Web Services

What is Flask

Developing a Flask Web service

Mapping resources using URLs

Mapping resources using HTTP

Negotiating data content

Python application security

OWASP 2021 Top Ten Overview

Python Code Access Control

Options for Protecting Data

Injection and Python

Python and Data Validation

Python and XML Processing

Python and Known Vulnerable Components

Python and Serialization/Deserialization

Additional Course Details

Nexus Humans Python Programming for Security Analysts & Professionals (TTPS4894) training program is a workshop that presents an invigorating mix of sessions, lessons, and masterclasses meticulously crafted to propel your learning expedition forward.

This immersive bootcamp-style experience boasts interactive lectures, hands-on labs, and collaborative hackathons, all strategically designed to fortify fundamental concepts.

Guided by seasoned coaches, each session offers priceless insights and practical skills crucial for honing your expertise. Whether you're stepping into the realm of professional skills or a seasoned professional, this comprehensive course ensures you're equipped with the knowledge and prowess necessary for success.

While we feel this is the best course for the Python Programming for Security Analysts & Professionals (TTPS4894) course and one of our Top 10 we encourage you to read the course outline to make sure it is the right content for you.

Additionally, private sessions, closed classes or dedicated events are available both live online and at our training centres in Dublin and London, as well as at your offices anywhere in the UK, Ireland or across EMEA.

Frequently Asked Questions

Q: What delivery options are available for Python Programming for Security Analysts & Professionals (TTPS4894)?

We offer multiple delivery formats:

- Live Instructor-Led Classroom Online (Virtual/Live Online)
- Traditional Instructor-Led Classroom Training (ILT)
- On-site delivery at your offices anywhere in United Kingdom
- Private dedicated courses customized for your team

Q: How many CPD hours does this course provide?

The 4-day Python Programming for Security Analysts & Professionals (TTPS4894) course provides up to 26.0 CPD hours of structured learning. CPD certificates can be provided upon request.

Q: What is the duration of the Python Programming for Security Analysts & Professionals (TTPS4894) training?

The training takes place over 4 day(s), with each day lasting approximately 32.00 hours including breaks for lunch and refreshments.

Q: Do you provide corporate training for Python Programming for Security Analysts & Professionals (TTPS4894)?

Yes, we provide corporate training, dedicated training, and closed classes for Python Programming for Security Analysts & Professionals (TTPS4894). Training can take place anywhere in United Kingdom including London, Manchester, Birmingham, Edinburgh, or live online allowing teams from across United Kingdom or internationally to attend.

Q: What related terms do people search for?

Popular related searches include: Python; Security

Q: Why choose Nexus Human for Python Programming for Security Analysts & Professionals (TTPS4894)?

Nexus Human is recognized as one of the leading training providers. Our trainers have won multiple awards including:

- Small Firms Best Trainer Award
- National Training Partner of the Year (Ireland) - Multiple Years
- Global Top 30 Instructor Awards (2012, 2019, 2021)
- Tech Excellence Award Nominations
- Learning Performance Institute (LPI) External Training Provider Sponsor 2024

Q: Are there any discount codes available?

Yes! Use discount code **PENPAL5** when booking your Python Programming for Security Analysts & Professionals (TTPS4894) training. Please note that only one discount code can be used per booking and cannot be combined with other special offers.

Nexus Human

Professional Training & Development

Email: info@nexushuman.com

Website: www.nexushuman.com

Phone: +353 1 889 8700 (Ireland) | +44 20 8142 8961 (UK)

© 2026 Nexus Human. All rights reserved. This brochure was generated on 13/11/2026.