Web Application Security in Python

Category: Networking & Security | **Vendor:** Various

Duration: 24.00 hours (3 days) **19.5 CPD Hours** **Rating:** ⭐ 4.6 (5,878 reviews)

Course Information

Delivery Format: Instructor Led - Online

Course Overview

Your Web application written in Python works as intended, so you are done, right? But did you consider feeding in incorrect values? 16Gbs of data? A null? An apostrophe? Negative numbers, or specifically -1 or -231? Because that’s what the bad guys will do – and the list is far from complete.

Handling security needs a healthy level of paranoia, and this is what this course provides: a strong emotional engagement by lots of hands on labs and stories from real life, all to substantially improve code hygiene. Mistakes, consequences, and best practices are our blood, sweat and tears.

The curriculum goes through the common Web application security issues following the OWASP Top Ten but goes far beyond it both in coverage and the details.All this is put in the context of Python, and extended by core programming issues, discussing security pitfalls of the programming language.

So that you are prepared for the forces of the dark side.

So that nothing unexpected happens.

Nothing.

About This Course

- Cyber security basics
- The OWASP Top Ten
- Common software security weaknesses
- JSON security
- Wrap up

Who Should Attend

Python developers working on Web applications

Prerequisites & Entry Requirements

General Prerequisites:

General Python and Web development

Learning Outcomes

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

- Getting familiar with essential cyber security concepts
- Understanding Web application security issues
- Detailed analysis of the OWASP Top Ten elements
- Putting Web application security in the context of Python
- Going beyond the low hanging fruits
- Managing vulnerabilities in third party components
- Identify vulnerabilities and their consequences
- Learn the security best practices in Python
- Input validation approaches and principles

Detailed Course Outline

DAY 1

Cyber security basics

- What is security?
- Threat and risk
- Cyber security threat types
- Consequences of insecure software

The OWASP Top Ten

- OWASP Top 10 – 2017
- A1 – Injection

- Injection principles
- Injection attacks
- SQL injection

- SQL injection basics
- Lab – SQL injection
- Attack techniques
- Content-based blind SQL injection
- Time-based blind SQL injection
- SQL injection best practices

- Input validation
- Parameterized queries
- Additional considerations
- Lab – SQL injection best practices
- Case study – Hacking Fortnite accounts
- Code injection

- Code injection via input()
- OS command injection

- Lab – Command injection
- OS command injection best practices
- Avoiding command injection with the right APIs
- Lab – Command injection best practices
- Case study – Shellshock
- Lab – Shellshock
- Case study – Command injection via ping
- Script injection

- Server-side template injection (SSTI)
- Lab – Template injection

- General protection best practices

- A2 – Broken Authentication

- Authentication basics

- Multi-factor authentication

- Authentication weaknesses – spoofing

- Spoofing on the Web

- Case study – PayPal 2FA bypass

- Password management

- Inbound password management

- Storing account passwords

- Password in transit

- Lab – Is just hashing passwords enough?

- Dictionary attacks and brute forcing

- Salting

- Adaptive hash functions for password storage

- Password policy

- NIST authenticator requirements for memorized secrets

- Case study – The Ashley Madison data breach

- The dictionary attack

- The ultimate crack

- Exploitation and the lessons learned

- Password database migration

- (Mis)handling None passwords

DAY 2

The OWASP Top Ten

- A2 – Broken Authentication

- Password management

- Outbound password management

- Hard coded passwords
- Best practices
- Lab – Hardcoded password
- Protecting sensitive information in memory

- Challenges in protecting memory
- Session management

- Session management essentials
- Session ID best practices
- Why do we protect session IDs – Session hijacking
- Session fixation
- Session handling in Flask
- A3 – Sensitive Data Exposure

- Information exposure
- Exposure through extracted data and aggregation
- Case study – Strava data exposure
- Error and exception handling principles
- A4 – XML External Entities (XXE)

- DTD and the entities
- Entity expansion
- Lab – Billion laughs attack
- External Entity Attack (XXE)

- File inclusion with external entities
- Server-Side Request Forgery with external entities
- Lab – External entity attack
- Case study – XXE vulnerability in SAP Store
- Preventing XXE
- Lab – Using non-vulnerable parsers
- A5 – Broken Access Control

- Access control basics
- Failure to restrict URL access
- Confused deputy

- Insecure direct object reference (IDOR)
- Lab – Insecure Direct Object Reference
- Authorization bypass through user-controlled keys
- Case study – Authorization bypass on Facebook
- Lab – Horizontal authorization
- File upload

- Unrestricted file upload
- Good practices
- Lab – Unrestricted file upload
- A6 – Security Misconfiguration

- Configuration principles
- Python configuration best practices

- Configuring Flask
- A7 – Cross-site Scripting (XSS)

- Cross-site scripting basics
- Cross-site scripting types

- Persistent cross-site scripting
- Reflected cross-site scripting
- Client-side (DOM-based) cross-site scripting
- Lab – Stored XSS
- Lab – Reflected XSS
- Case study – XSS in Fortnite accounts
- XSS protection best practices

- Protection principles – escaping
- XSS protection APIs in Python
- XSS protection in Jinja2
- Lab – XSS fix / stored
- Lab – XSS fix / reflected
- Additional protection layers
- Client-side protection principles
- A8 – Insecure Deserialization

- Serialization and deserialization challenges
- Deserializing untrusted streams
- Deserialization with pickle
- Lab – Deserializing with Pickle
- PyYAML deserialization challenges
- Deserialization best practices

DAY 3

The OWASP Top Ten

- A9 – Using Components with Known Vulnerabilities

- Using vulnerable components

- Assessing the environment
- Hardening
- Untrusted functionality import
- Malicious packages in Python
- Importing JavaScript
- Lab – Importing JavaScript
- Case study – The British Airways data breach
- Vulnerability management

- Patch management
- Vulnerability databases
- A10 – Insufficient Logging & Monitoring

- Logging and monitoring principles
- Insufficient logging
- Plaintext passwords at Facebook
- Logging best practices
- Monitoring best practices
- Web application security beyond the Top Ten

- Client-side security
- Same Origin Policy

- Lab – Same-origin policy demo
- Tabnabbing
- Lab – Reverse tabnabbing
- Frame sandboxing

- Cross-Frame Scripting (XFS) attack
- Lab – Clickjacking
- Clickjacking beyond hijacking a click
- Clickjacking protection best practices

- Lab – Using CSP to prevent clickjacking

Common software security weaknesses

- Input validation

- Input validation principles

- Blacklists and whitelists
- Data validation techniques
- Lab – Input validation
- What to validate – the attack surface
- Where to validate – defense in depth
- How to validate – validation vs transformations
- Output sanitization
- Encoding challenges
- Lab – Encoding challenges
- Validation with regex
- Regular expression denial of service (ReDoS)
- Lab – Regular expression denial of service (ReDoS)
- Dealing with ReDoS
- Files and streams

- Path traversal
- Path traversal-related examples
- Lab – Path traversal
- Additional challenges in Windows
- Virtual resources
- Path traversal best practices
- Format string issues
- Unsafe native code

- Native code dependence

- Lab – Unsafe native code

- Best practices for dealing with native code

JSON security

- JSON injection

- Dangers of JSONP

- JSON/JavaScript hijacking

- Best practices

- Case study – ReactJS vulnerability in HackerOne

Wrap up

- Secure coding principles

- Principles of robust programming by Matt Bishop

- Secure design principles of Saltzer and Schröder

- And now what?

- Software security sources and further reading

- Python resources

Skills You Will Learn:

- Getting familiar with essential cyber security concepts
- Understanding Web application security issues
- Detailed analysis of the OWASP Top Ten elements
- Putting Web application security in the context of Python
- Going beyond the low hanging fruits
- Managing vulnerabilities in third party components
- Identify vulnerabilities and their consequences
- Learn the security best practices in Python
- Input validation approaches and principles

Frequently Asked Questions

Q: What delivery options are available for Web Application Security in Python?

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 3-day Web Application Security in Python course provides up to 19.5 CPD hours of structured learning. CPD certificates can be provided upon request.

Q: What is the duration of the Web Application Security in Python training?

The training takes place over 3 day(s), with each day lasting approximately 24.00 hours including breaks for lunch and refreshments.

Q: Do you provide corporate training for Web Application Security in Python?

Yes, we provide corporate training, dedicated training, and closed classes for Web Application Security in Python. 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: Web Application Security in Python, Web Application Security, WASEC-PY

Q: Why choose Nexus Human for Web Application Security in Python?

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 Web Application Security in Python 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 14/11/2026.