Course Summary

Description

Learn how to develop secure applications with Python.

The Python programming class provides both the contextual and technical details necessary to design and code secure applications and systems.

The CIA Triad is an essential concept in cybersecurity and describes the pillars of any secure application: confidentiality, integrity, and availability. In this class, you will learn how to implement the CIA Triad.

Because of the internet, applications are becoming more interconnected across corporate and personal boundaries. This creates a massive opportunity for exploitation and breaches of security. You will learn techniques from securing network communication, such as cryptography and token authentication.

The course presents secure coding techniques; randomization with entropy, not accepting raw input, canonization, and more.

During class, you will review common vulnerabilities, attacks, and mitigation. Injection attacks, buffer overflow, cross-site scripting, DLL preloading attack, and other attacks, are presented to provide students practical security experience.

Cryptography is a consistent topic throughout the class, such as hashing to assure integrity and encryption to provide secrecy. You will learn cryptography using OpenSSL. Various crypto algorithms are introduced, such as SHA-256 and AES, and scenarios where they are applied. The final module is about token authentication using Python Flask. Token authentication is effective for network security in a stateless environment. You will develop a Python application using tokens in a walkthrough.

Objectives

At the end of this course, students will be able to:

- Learn cryptography terminology and theory
- Learn how to implement random numbers with entropy
- Learn about Alice and Bob documentation
- Learn various hashing algorithms and implementation
- Learn various encryption algorithms and implementation
- Learn the implementation details of symmetric versus asymmetric encryption
- Learn the basis of X.509 certificates
- Learn how to implement token authentication with Python Flask
- Learn the steps required to create a digital signature

Topics

- Basics
- Hashing
- Encryption
- Digital Signatures
- Certificates
- Advanced Concepts

Audience & Prerequisites

The audience for this course is Python developers with 3 to 6 months' experience.

Duration

Two days

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically

Course Outline

Course Outline

I. Basics

Learn cryptography concepts and key tools. Cryptography is most often used to protect vulnerabilities. STRIDE is investigated as a means of classifying vulnerabilities, including where cryptography may be appropriate. You will learn about the pillars of cryptography: confidentiality, integrity, and availability.

A. Confidentiality
B. Integrity
C. Availability
D. Randomization
E. Personal Identifiable Information
F. STRIDE
G. MITRE
H. CVSS

II. Hashing

Learn how to prevent data tampering with hashing. Integrity is the primary benefit of hashing data. In this course, investigate various hashing algorithms and standards. Learn the components of hashing, including random numbers, padding, and other concepts.

A. Hashing algorithms
B. Keyed hashes
C. Check sums
D. Common attacks on hashes

III. Encryption

Learn to protect secrets and implement confidentiality using symmetric or asymmetric encryption. In this class, you will learn the benefits of each.

A. Theory
B. Encryption algorithms
C. Create keypairs
D. Implement symmetric encryption
E. Implement asymmetric encryption
F. Common attacks on hashes

IV. Digital Signatures

Learn how digital signatures ensure nonrepudiation and create RSA digital signatures using the Python cryptography module.

A. Overview of digital of signatures
B. Contrast and compare digital signature algorithms
C. Steps for creating a digital signature
D. Implementation
E. Verifying a digital signature

V. Certificates

Lean the about digital certificates, which is an application of cryptography. In this class, we will explore the latest version of X.509 certificates. This includes various scenarios where certificates are useful. You will also learn the meaning of the various certificate attributes. Leveraging self-signed before deployment or release is another important topic included in this module.

A. X509 certificates
B. Certificate attributes
C. Tools for certificates
D. Self-signed certificates
E. Scenarios
F. TCP/TLS
G. HTTPS
H. Secure boot

VI. Advanced Concepts

Learn some of the various advanced concepts of security using Python. You will learn about block chaining modes and padding, which are essential components of cryptography. Master OAuth2 and the ability to authenticate users via social media platforms. Review modern algorithms, such as ECC.

A. Securing passwords
B. Block chaining modes
C. Padding
D. OAuth2
E. ECC
F. More

Due to the nature of this material, this document refers to numerous hardware and software products by their trade names. References to other companies and their products are for informational purposes only, and all trademarks are the properties of their respective companies. It is not the intent of ProTech Professional Technical Services, Inc. to use any of these names generically