Web application security masterclass

CYDWeb5d | 5 days | On-site or online | Hands-on

Your application written in any programming language 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 -2^31? 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 Java, and extended by core programming issues, discussing security pitfalls of the Java language.

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

So that nothing unexpected happens.

Nothing.

Cyber security skills and drills

38 LABS

19 CASE STUDIES

Audience

Web developers

Group size

12 participants

Preparedness

General Web development

Outline

- Cyber security basics
- The OWASP Top Ten 2021
- Security testing
- Wrap up

Standards and references

OWASP, CWE and Fortify Taxonomy

What you'll have learned

- Getting familiar with essential cyber security concepts
- Understanding how cryptography supports security
- Understanding Web application security issues
- Detailed analysis of the OWASP Top Ten elements
- Putting Web application security in the context of any programming language
- Going beyond the low hanging fruits
- Input validation approaches and principles
- Managing vulnerabilities in third party components
- Getting familiar with security testing techniques and tools

Table of contents

Day 1

  - CSRF best practices
  - CSRF defense in depth
  - *Lab – CSRF protection with tokens*

A02 - Cryptographic Failures

- Information exposure
  - Exposure through extracted data and aggregation
  - *Case study – Strava data exposure*
  - Data exposure best practices
  - Data masking
  - Privacy violation
    - Privacy essentials
    - Related standards, regulations and laws in brief
    - Privacy violation and best practices
  - System information leakage
    - Leaking system information
    - Exposure through debug information
    - Exposure through files and directories
  - Information leakage through side channels
    - Exposure through side channels
    - Side channels and covert channels
  - Information exposure best practices
- Cryptography for developers
  - Cryptography basics
  - Elementary algorithms
    - Random number generation
    - Pseudo random number generators (PRNGs)
    - Cryptographically strong PRNGs
    - Seeding
    - Using virtual random streams
    - *Lab – Using random numbers*
    - True random number generators (TRNG)
    - Assessing PRNG strength
      - *Case study – Equifax credit account freeze*

Day 2

› The OWASP Top Ten 2021

A02 - Cryptographic Failures (continued)

- Cryptography for developers
  - Elementary algorithms

    - Hashing
    - Hashing basics
      - 🗎 *Case study - Shattered*
    - Common hashing mistakes
    - 🔬 *Lab – Hashing*
    - Hash algorithms for password storage
    - Password storage algorithms and considerations
    - Best practices when using password hashing algorithms
  - Confidentiality protection
    - Symmetric encryption
    - Stream ciphers
    - [Block ciphers]()
    - Modes of operation
    - Modes of operation and IV – best practices
    - Authenticated encryption
    - Authenticated encryption modes of operation
    - Best practices
    - Best practices – Using cryptographic storage
    - 🔬 *Lab – Symmetric encryption*
    - Asymmetric encryption
    - The RSA algorithm
    - Using RSA – best practices
    - Combining symmetric and asymmetric algorithms
    - Key exchange and agreement
    - Key exchange
    - Diffie-Hellman key agreement algorithm
    - Key exchange pitfalls and best practices
    - Some further key management challenges
- Certificates
  - Certificates and PKI
  - X.509 certificates
  - Chain of trust
  - PKI actors and procedures
  - PGP – Web of Trust
  - Certificate pinning
  - Certificate revocation
- Transport security
  - Transport security weaknesses
  - The TLS protocol
    - TLS basics
    - TLS features (changes in v1.3)
    - The handshake in a nutshell (v1.3)
    - TLS best practices
    - 🔬 *Lab – Using a secure socket*

- HTTP Strict Transport Security (HSTS)

A03 - Injection

- Input validation
  - Input validation principles
  - Denylists and allowlists
  - Data validation techniques
  - *Lab – Input validation*
  - What to validate – the attack surface
  - Where to validate – defense in depth
  - When to validate – validation vs transformations
  - Output sanitization
  - Encoding challenges
  - Unicode challenges
  - *Lab – Encoding challenges*
  - Validation with regex

Day 3

› The OWASP Top Ten 2021

A03 - Injection (continued)

- 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
  - *Lab – Using prepared statements*
  - Additional considerations
  - *Case study – Hacking Fortnite accounts*
  - SQL injection protection and ORM
- Parameter manipulation
  - CRLF injection
  - HTTP header manipulation

    - HTTP response splitting
- Code injection
  - OS command injection
    - OS command injection best practices
      - *Case study – Shellshock*
    - *Lab - Shellshock*
- HTML injection - 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
    - *Lab – XSS fix / stored*
    - *Lab – XSS fix / reflected*
    - Client-side protection principles
    - Additional protection layers – defense in depth

› The OWASP Top Ten 2021

A04 - Insecure Design

- Client-side security
  - Same Origin Policy
    - Simple request
    - Preflight request
    - Cross-Origin Resource Sharing (CORS)
  - Frame sandboxing
    - Cross-Frame Scripting (XFS) attacks
    - *Lab - Clickjacking*
    - Clickjacking beyond hijacking a click
    - Clickjacking protection best practices
    - *Lab – Using CSP to prevent clickjacking*
  - Content Security Policy
    - Directives in the HTTP response
    - Browser support
    - Resource control
    - Source whitelists
    - CSP best practices
  - JSON security

    - JSON validation
    - JSON injection
    - Dangers of JSONP
    - JSON/JavaScript hijacking
    - Best practices
      - *Case study – ReactJS vulnerability in HackerOne*
  - XML security
    - XML validation
    - XML injection
    - XPath injection
    - Blind XPath injection

Day 4

› The OWASP Top Ten 2021

A05 - Security Misconfiguration

- Configuration principles
- Server misconfiguration
- Cookie security
  - Cookie security best practices
  - Cookie attributes
- XML entities
  - DTD and the entities
  - Attribute blowup
  - Entity expansion
  - 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*
    - *Lab – Prohibiting DTD*

A06 - Vulnerable and Outdated Components

- Using vulnerable components
- Assessing the environment
- Hardening
- Untrusted functionality import
- Vulnerability management
  - Patch management
  - Vulnerability management

  - Vulnerability databases
  - Vulnerability rating – CVSS
  - Bug bounty programs
  - [DevOps, the build process and CI / CD]

A07 - Identification and Authentication Failures

- Authentication
  - Authentication basics
  - Multi-factor authentication
  - Time-based One Time Passwords (TOTP)
  - Authentication weaknesses
  - [Spoofing on the Web]
  - *Case study – PayPal 2FA bypass*
  - Authentication bypass by capture / replay
  - *Case study – Hacking the Mitsubishi Outlander PHEV hybrid*
  - User interface best practices
  - *Lab – On-line password brute forcing*
  - Impersonation
- Session management
  - Session management essentials
  - Why do we protect session IDs – Session hijacking
  - Session fixation
  - Session invalidation
  - Session ID best practices
  - Using tokens
    - Token-based authentication
- 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]
    - Password hardening
    - Using passphrases
    - Password change
    - Password recovery issues
    - Password recovery best practices
    - *Lab – Password reset weakness*
      - *Case study – The Ashley Madison data breach*

- 📑 *The dictionary attack*
- 📑 *The ultimate crack*
- 📑 *Exploitation and the lessons learned*
- Password database migration
- (Mis)handling null passwords

- Outbound password management
  - Hard coded passwords
  - Best practices
  - 🔬 *Lab – Hardcoded password*
  - Protecting sensitive information in memory
  - Challenges in protecting memory

Day 5

› The OWASP Top Ten 2021

A08 - Software and Data Integrity Failures

- Integrity protection
  - Authenticity and non-repudiation
  - Message Authentication Code (MAC)
    - 🔬 *Lab – Calculating MAC*
  - Digital signature
    - Digital signature with RSA
    - Elliptic Curve Cryptography
    - ECC basics
    - ECC curves and pitfalls
    - Digital signature with ECC
    - 🔬 *Lab – Digital signature with ECDSA*
- Subresource integrity
  - Importing JavaScript
  - 🔬 *Lab – Importing JavaScript*
  - 📑 *Case study – The British Airways data breach*
- Insecure deserialization
  - Serialization and deserialization challenges
  - Integrity – deserializing untrusted streams
  - Integrity – deserialization best practices
  - Property Oriented Programming (POP)
    - Creating a POP payload
    - 🔬 *Lab – Creating a POP payload*
    - 🔬 *Lab – Using the POP payload*
    - Summary – POP best practices

A09 - Security Logging and Monitoring Failures

- Logging and monitoring principles
- Insufficient logging
- *Case study – Plaintext passwords at Facebook*
- Log forging
  - Log forging – best practices
  - *Case study – Log interpolation in log4j*
  - *Case study – The Log4Shell vulnerability (CVE-2021-44228)*
  - *Case study – Log4Shell follow-ups (CVE-2021-45046, CVE-2021-45105)*
  - *Lab – Log4Shell*
- Logging best practices
- Monitoring best practices
- Firewalls and Web Application Firewalls (WAF)
  - Intrusion detection and prevention
  - *Case study – The Marriott Starwood data breach*

A10 - Server-Side Request Forgery (SSRF)

- Server-side Request Forgery (SSRF)
- *Case study – SSRF and the Capital One breach*

Web application security beyond the Top Ten

- Denial of service
  - Flooding
  - Resource exhaustion
  - Sustained client engagement
  - Infinite loop
  - Economic Denial of Sustainability (EDoS)
  - Amplification
    - Other amplification examples
  - Algorithm complexity issues
    - Regular expression denial of service (ReDoS)
    - *Lab – ReDoS*
    - Dealing with ReDoS
    - Hash table collision
    - How do hash tables work?
    - Hash collision in case of hash tables

› Security testing

Security testing techniques and tools

- Code analysis
  - Security aspects of code review

  - The OWASP Code Review methodology
  - Static Application Security Testing (SAST)
- Dynamic analysis
  - Security testing at runtime
  - Penetration testing
  - Stress testing
  - Dynamic analysis tools
    - Dynamic Application Security Testing (DAST)
    - Web vulnerability scanners
    - *Lab – Using web vulnerability scanners*
    - SQL injection tools
    - *Lab – Using SQL injection tools*
  - Fuzzing
- Metasploit
  - Metasploit basics
  - Using Metasploit modules
- Password cracking
  - Using password cracking tools
  - *Lab – Password cracking with John the Ripper*
- Proxies and sniffing
  - Proxy servers and sniffers
  - Sniffing – tools and considerations
  - *Lab – Using a proxy*

› Wrap up

Secure coding principles

- Principles of robust programming by Matt Bishop

And now what?

- Software security sources and further reading