Training: SCADEMY
CL-CTS Security testing native code

TRAINING GOALS:

To put it bluntly, writing C/C++ code can be a minefield for reasons ranging from memory management or dealing with legacy code to sharp deadlines and code maintainability. Yet, beyond all that, what if we told you that attackers were trying to break into your applications right now? How likely would they be to succeed?

After getting familiar with the common weaknesses and their consequences that can allow hackers to attack your system, participants learn about the general approach and the methodology for security testing, and the techniques that can be applied to reveal specific vulnerabilities. Security testing should start with information gathering about the system (ToC, i.e. Target of Evaluation), then a thorough threat modeling should reveal and rate all threats, arriving to the most appropriate risk analysis-driven test plan.

Security evaluations can happen at various steps of the SDLC, and so we discuss design review, code review, reconnaissance and information gathering about the system, testing the implementation and the testing and hardening the environment for secure deployment. Many different security testing techniques are introduced in details, like taint analysis and heuristics-based code review, static code analysis or fuzzing. Various types of tools are introduced that can be applied in order to automate security evaluation of software products, which is also supported by a number of exercises, where we execute these tools to analyze the already discussed vulnerable code.

This course prepares testers and QA staff to adequately plan and precisely execute security tests for applications written in C or C++, select and use the most appropriate tools and techniques to find even hidden security flaws, and thus gives essential practical skills that can be applied on the next day working day.

Having secure applications will give you a distinct edge over your competitors. It is your choice to be ahead of the pack – take a step and be a game-changer in the fight against cybercrime.

Participants attending this course will:

- Understand basic concepts of security, IT security and secure coding
- Realize the severe consequences of unsecure buffer handling
- Understand the architectural protection techniques and their weaknesses
- Learn about typical coding mistakes and how to avoid them
- Be informed about recent vulnerabilities in various platforms, frameworks and libraries
- Learn about denial of service attacks and protections

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Understand security testing approaches and methodologies
- Get practical knowledge in using security testing techniques and tools
- Learn how to set up and operate the deployment environment securely
- Get sources and further readings on secure coding practices

Audience:

C and C++ developers testers

CONSPECT:

- IT security and secure coding
  - Nature of security
  - What is risk?
  - IT security vs. secure coding
  - From vulnerabilities to botnets and cybercrime
    - Nature of security flaws
    - Reasons of difficulty
    - From an infected computer to targeted attacks
- x86 machine code, memory layout and stack operations
  - Intel 80x86 Processors – main registers
  - Intel 80x86 Processors – most important instructions
  - Intel 80x86 Processors – flags
  - Intel 80x86 Processors – control instructions
  - Intel 80x86 Processors – stack handling and flow control
  - The memory address layout
  - The function calling mechanism in C/C++ on x86
  - Calling conventions
  - The local variables and the stack frame
  - Function calls – prologue and epilogue of a function
  - Stack frame of nested calls
  - Stack frame of recursive functions
- Buffer overflow
  - Stack overflow
    - Buffer overflow on the stack
    - Overwriting the return address
    - Exercises – introduction
    - Exercise BOFIntro

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70

BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Exercise BOFShellcode
- Protection against stack overflow
  - Specific protection methods
  - Protection methods at different layers
  - The protection matrix of software security
  - Stack overflow – Prevention (during development)

- Common coding errors and vulnerabilities
  - Time and state problems
    - Time and state related problems
    - Serialization errors
    - Exercise TOCTTOU
    - Best practices against TOCTTOU
- Code quality problems
  - Dangers arising from poor code quality
  - Poor code quality – spot the bug!
  - Unreleased resources
  - Type mismatch – Spot the bug!
  - Exercise TypeMismatch
  - Memory allocation problems
    - Smart pointers
    - Zero length allocation
    - Double free
    - Mixing delete and delete[]
- Common coding errors and vulnerabilities
  - Input validation
    - Input validation concepts
    - Integer problems
      - Representation of negative integers
      - Integer ranges
      - Integer overflow
      - Integer problems in C/C++
      - The integer promotion rule in C/C++
      - Arithmetic overflow – spot the bug!
      - Exercise IntOverflow
      - What is the value of abs(INT_MIN)?
      - Signedness bug – spot the bug!

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Integer truncation – spot the bug!
- Integer problem – best practices
- Case study – Android Stagefright

- Printf format string bug
  - Printf format strings
  - Printf format string bug – exploitation
  - Exercise Printf
  - Printf format string exploit – overwriting the return address
- Printf format string problem – best practices
- Some other input validation problems
  - Array indexing – spot the bug!
  - Off-by-one and other null termination errors
  - The Unicode bug
- Log forging
  - Some other typical problems with log files

- Improper use of security features
  - Typical problems related to the use of security features
  - Password management
    - Exercise – Weakness of hashed passwords
    - Password management and storage
    - Special purpose hash algorithms for password storage
    - Argon2 and PBKDF2 implementations in C/C++
    - bcrypt and scrypt implementations in C/C++
    - Password audit
    - Exercise – using John the Ripper
    - Case study – the Ashley Madison data breach
    - Typical mistakes in password management
    - Exercise – Hard coded passwords
  - Sensitive information in memory
    - Protecting secrets in memory
    - Sensitive info in memory - minimize the attack surface
    - Your secrets vs. dynamic memory
    - Zeroisation
    - Zeroisation vs. optimization – Spot the bug!
    - Copies of sensitive data on disk
    - Core dumps

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Disabling core dumps
- Swapping
- Memory locking - preventing swapping
- Problems with page locking
- Best practices

- Denial of service
  - DoS introduction
  - Asymmetric DoS
  - Regular expression DoS (ReDoS)
    - Exercise ReDoS
    - ReDoS mitigation
    - Case study – ReDos in Stack Exchange
  - Hashtable collision attack
    - Using hashtables to store data
    - Hashtable collision
    - Hashtable collision in Java
- Security testing
  - Functional testing vs. security testing
  - Security vulnerabilities
  - Prioritization – risk analysis
  - Security assessments in various SDLC phases
  - Security testing methodology
    - Steps of test planning (risk analysis)
    - Scoping and information gathering
      - Stakeholders
      - Assets
      - Security objectives for testing
    - Threat modeling
      - Attacker profiles
      - Threat modeling
      - Threat modeling based on attack trees
      - Threat modeling based on misuse/abuse cases
      - Misuse/abuse cases – a simple example
      - SDL threat modeling
      - The STRIDE threat categories
      - Diagramming – elements of a DFD

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Data flow diagram – example
- Threat enumeration – mapping STRIDE to DFD elements
- Risk analysis – classification of threats
- The DREAD risk assessment model

- Testing steps
  - Deriving test cases
  - Accomplishing the tests
  - Processing test results
  - Mitigation concepts
  - Standard mitigation techniques of MS SDL
  - Review phase

- Security testing techniques and tools
  - General testing approaches
  - Source code review
    - Code review for software security
    - Taint analysis
    - Heuristic-based
    - Static code analysis
      - Static code analysis
      - Exercise – Static code analysis using FlawFinder
  - Testing the implementation
    - Manual vs. automated security testing
    - Penetration testing
    - Stress tests
    - Binary and memory analysis
      - Exercise – Binary analysis with strings
    - Instrumentation libraries and frameworks
      - Exercise – Using Valgrind
    - Fuzzing
      - Automated security testing - fuzzing
      - Challenges of fuzzing
      - Exercise – Fuzzing with AFL (American Fuzzy Lop)

- Deployment environment
  - Assessing the environment
  - Configuration management
    - Configuration management

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

- Hardening
  - Hardening
  - Network-level hardening
  - Hardening the deployment – server administration
  - Hardening the deployment – access control
- Patch and vulnerability management
  - Patch management
  - Vulnerability repositories
  - Vulnerability attributes
  - Common Vulnerability Scoring System – CVSS
  - Vulnerability management software
- Principles of security and secure coding
  - Matt Bishop’s principles of robust programming
  - The security principles of Saltzer and Schroeder
- Knowledge sources
  - Secure coding sources – a starter kit
  - Vulnerability databases
  - Recommended books – C/C++

REQUIREMENTS:

General C/C++ development, QA and testing

Difficulty level

CERTIFICATE:

The participants will obtain certificates signed by SCADEMY (course completion).

TRAINER:

Authorized SCADEMY Trainer

ADDITIONAL INFORMATION:

Training come with a number of easy-to-understand exercises providing live hacking fun. By

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007

accomplishing these exercises with the lead of the trainer, participants can analyze vulnerable code snippets and commit attacks against them in order to fully understand the root causes of certain security problems. All exercises are prepared in a plug-and-play manner by using a pre-set desktop virtual machine, which provides a uniform development environment.

SCADEMY together with online application security educational platform AVATAO (more about AVATAO www.avatao.com) for each of participant SCADEMYs authorized training adds the 30 days business AVATAO trial holds the following package:

- 30-day customized free trial

Compendium Education Center Ltd.
ul. Tatarska 5, 30-103 Kraków, tel.: (12) 298 47 77
ul. Postępu 18B, 02-676 Warszawa, tel.: (22) 417 41 70
BNP Paribas Bank Polska S.A.
EUR: PL 79 1600 1462 1853 6451 3000 0006
USD: PL 52 1600 1462 1853 6451 3000 0007