Training: SCADEMY

CL-CCA Comprehensive C and C++ secure coding (ARM)

TRAINING GOALS:

As a developer, your duty is to write bulletproof code. However...

What if we told you that despite all of your efforts, the code you have been writing your entire career is full of weaknesses you never knew existed? What if, as you are reading this, hackers were trying to break into your code? How likely would they be to succeed?

This advanced course will change the way you look at code. A hands-on training during which we will teach you all of the attackers' tricks and how to mitigate them, leaving you with no other feeling than the desire to know more.

It is your choice to be ahead of the pack, and be seen as 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
- Have a practical understanding of cryptography
- Understand essential security protocols
- Understand some recent attacks against cryptosystems
- Learn about XML security
- Learn about typical coding mistakes and how to avoid them
- Be informed about recent vulnerabilities in various platforms, frameworks and libraries
- 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, software architects and testers

Compendium Centrum Edukacyjne Spółka z o.o.
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

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
  - Classification of security flaws
    - Landwehr’s taxonomy
    - The Seven Pernicious Kingdoms
- ARM machine code, memory layout and stack operations
  - ARM Processors – main registers
  - ARM Processors – most important instructions
  - ARM Processors – flags and condition fields
  - ARM Processors – control instructions
  - ARM Processors – stack handling instructions
  - Understanding complex ARM instructions
  - The function calling mechanism in ARM
  - The local variables and the stack frame
  - Function calls – prologue and epilogue of a function (ARM)
  - 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
    - Exercise BOFShellcode
  - Protection against stack overflow
    - Specific protection methods
    - Protection methods at different layers
    - The protection matrix of software security

Compendium Centrum Edukacyjne Spółka z o.o.
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

- Stack overflow – Prevention (during development)
- Stack overflow – Detection (during execution)
- Fortify compiler option (FORTIFY_SOURCE)
- Exercise BOFShellcode – Using the Fortify compiler option

- Stack smashing protection
  - Stack smashing protection variants
  - Stack smashing protection in GCC
  - Exercise BOFShellcode – Stack smashing protection
  - Effects of stack smashing protection – prologue
  - Effects of stack smashing protection – epilogue
  - Bypassing stack smashing protection – an example
  - Overwriting arguments – Mitigation
- Address Space Layout Randomization (ASLR)
  - Randomization with ASLR
  - Practical weaknesses and limitations to ASLR
  - Circumventing ASLR: NOP sledding
- Non executable memory areas – the NX bit
  - Access control on memory segments
  - The Never eXecute (NX) bit
  - Exercise BOFShellcode – Enforcing NX memory segments

- Buffer overflow
  - Return oriented programming (ROP)
    - Circumventing memory execution protection
    - Return-to-libc attack in ARM
    - ROP gadget - Register fill with constants
    - ROP gadget – Memory write
    - Combining the ROP gadgets
    - Real ROP attack scenarios
    - ROP mitigation
      - Mitigation techniques of ROP attack
  - Heap overflow
    - Memory allocation managed by a doubly-linked list
    - Buffer overflow on the heap
    - Steps of freeing and joining memory blocks
    - Freeing allocated memory blocks
    - Case study – Heartbleed

Compendium Centrum Edukacyjne Spółka z o.o.
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

- TLS Heartbeat Extension
- Heartbleed – information leakage in OpenSSL
- Heartbleed – fix in v1.0.1g
- Protection against heap overflow

- Practical cryptography
  - Rule #1 of implementing cryptography
  - Cryptosystems
    - Elements of a cryptosystem
  - Symmetric-key cryptography
    - Providing confidentiality with symmetric cryptography
    - Symmetric encryption algorithms
    - Modes of operation
    - Symmetric encryption with OpenSSL: encryption
    - Symmetric encryption with OpenSSL: decryption
  - Other cryptographic algorithms
    - Hash or message digest
    - Hash algorithms
    - SHAttered
    - Hashing with OpenSSL
    - Message Authentication Code (MAC)
    - Providing integrity and authenticity with a symmetric key
    - Random number generation
      - Random numbers and cryptography
      - Cryptographically-strong PRNGs
      - Weak PRNGs in C and C++
      - Stronger PRNGs in C
      - Generating random numbers with OpenSSL
      - Hardware-based TRNGs
  - Asymmetric (public-key) cryptography
    - Providing confidentiality with public-key encryption
    - Rule of thumb – possession of private key
    - The RSA algorithm
      - Introduction to RSA algorithm
      - Encrypting with RSA
      - Combining symmetric and asymmetric algorithms
      - Digital signing with RSA

Compendium Centrum Edukacyjne Spółka z o.o.
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

- Asymmetric encryption with OpenSSL
- Digital signatures with OpenSSL

- Public Key Infrastructure (PKI)
  - Man-in-the-Middle (MitM) attack
  - Digital certificates against MitM attack
  - Certificate Authorities in Public Key Infrastructure
  - X.509 digital certificate

- Security protocols
  - Secure network protocols
  - Specific vs. general solutions
  - The TLS protocol
    - SSL and TLS
    - Usage options
    - Security services of TLS
    - SSL/TLS handshake

- Cryptographic vulnerabilities
  - Protocol-level vulnerabilities
    - BEAST
  - Padding oracle attacks
    - Adaptive chosen-ciphertext attacks
    - Padding oracle attack
    - CBC decryption
    - Padding oracle example
    - POODLE

- XML security
  - Introduction
  - XML parsing
  - XML injection
    - Injection principles
    - Exercise – XML injection
    - Protection through sanitization and XML validation
    - XML parsing in C++
  - Abusing XML Entity
    - XML Entity introduction
    - Exercise – XML bomb
    - XML bomb

Compendium Centrum Edukacyjne Spółka z o.o.
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

- XML external entity attack (XXE) – resource inclusion
- Exercise – XXE attack
- Preventing entity-related attacks
- Case study – XXE in Google Toolbar

- 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!
      - Integer truncation – spot the bug!
      - Integer problem – best practices
      - Case study – Android Stagefright
    - Injection
      - Injection principles
      - SQL Injection exercise
      - Typical SQL Injection attack methods
      - Blind and time-based SQL injection
      - SQL Injection protection methods
      - Command injection
      - Command injection exercise – starting Netcat
      - Case study - Shellshock
    - Printf format string bug
      - Printf format strings
      - Printf format string bug – exploitation
      - Exercise Printf
    - Printf format string problem – best practices
    - Some other input validation problems
      - Array indexing – spot the bug!

Compendium Centrum Edukacyjne Spółka z o.o.
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

- Off-by-one and other null termination errors
- The Unicode bug
- Path traversal vulnerability
  - Path traversal – weak protections
  - Path traversal – best practices
- Log forging
  - Some other typical problems with log files

- Common coding errors and vulnerabilities
  - Improper error and exception handling
    - Typical problems with error and exception handling
    - Empty catch block
    - Overly broad catch
    - Exercise ErrorHandling – spot the bug!
    - Exercise – Error handling
    - Case study – "e;#iamroot"e; authentication bypass in macOS
      - Authentication process in macOS (High Sierra)
      - Incorrect error handling in opendirectoryd
      - The #iamroot vulnerability (CVE-2017-13872)
      - Information leakage through error reporting
  - 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[]
    - Use after free
      - Use after free – Instance of a class
      - Spot the bug
      - Use after free – Dangling pointers
    - Case study - WannaCry
      - The WannaCry ransomware

Compendium Centrum Edukacyjne Spółka z o.o.
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

- The vulnerability behind WannaCry – spot the bug!
- Lessons learned

- Common coding errors and vulnerabilities
  - 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
      - Brute forcing
      - Special purpose hash algorithms for password storage
      - Argon2 and PBKDF2 implementations in C/C++
      - bcrypt and scrypt implementations in C/C++
      - 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
      - Disabling core dumps
      - Swapping
      - Memory locking - preventing swapping
      - Problems with page locking
      - Best practices
    - Insufficient anti-automation
      - Captcha
      - Captcha weaknesses
  - Time and state problems
    - Time and state related problems
    - Serialization errors
    - Exercise TOCTTOU
    - Best practices against TOCTTOU

Compendium Centrum Edukacyjne Spółka z o.o.
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

- Problems with temp files
- Requirements for creating temp files
- Requirements explained
- Creating temp files on POSIX systems
- Creating temp files portably
- Deleting temp files

- 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
    - Assessing the environment
    - Searching for online devices with SHODAN
    - Exercise – using SHODAN
    - Finding weaknesses with search engines
    - Exercise – Finding weaknesses with search engines
    - Password audit
    - Exercise – using John the Ripper

Compendium Centrum Edukacyjne Spółka z o.o.
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

- Testing random number generators
- Exercise – Testing random number generators

- Configuration management
  - Configuration management
- 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
  - Software identification through CPE and SWID
  - 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

Difficulty level



CERTIFICATE:

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

Compendium Centrum Edukacyjne Spółka z o.o.
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

TRAINER:

Authorized SCADEMY Trainer.

ADDITIONAL INFORMATION:

Training come with a number of easy-to-understand exercises providing live hacking fun. By 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 Centrum Edukacyjne Spółka z o.o.
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