Training: SCADEMY
CL-CPI C and C++ secure coding (x86)

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 code right now? How likely would they be to succeed?

This course will change the way you look at your C/C++ code. We'll teach you the common weaknesses and their consequences that can allow hackers to attack your system, and – more importantly – best practices you can apply to protect yourself. We give you a holistic view on C/C++ programming mistakes and their countermeasures from the machine code level to virtual functions and OS memory management. We present the entire course through live practical exercises to keep it engaging and fun.

Writing secure code 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
- Have a practical understanding of cryptography
- 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 sources and further readings on secure coding practices

Audience:

C and C++ developers, software architects and testers

CONSPECT:

- IT security and secure coding
  - Nature of 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

- 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
    - 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)
    - Stack overflow – Detection (during execution)
    - Fortify compiler option (FORTIFY_SOURCE)
    - Exercise BOFShellcode – Using the Fortify compiler option
  - Stack smashing protection
    - Stack smashing protection variants

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 smashing protection in GCC
- Exercise BOFShellcode – Stack smashing protection
- Effects of stack smashing protection
- 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

- Buffer overflow
  - Return-to-libc attack – Circumventing the NX bit protection
    - Circumventing memory execution protection
    - Return-to-libc attack
  - Return oriented programming (ROP)
    - Exploiting with ROP
    - ROP gadgets
    - 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
      - 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

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

- 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
    - 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

- XML security
  - XML injection
    - Injection principles
    - Exercise – XML injection

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

- Protection through sanitization and XML validation
- XML parsing in C++

- Abusing XML Entity
  - XML Entity introduction
  - Exercise – XML bomb
  - XML bomb
  - 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
  - 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)
  - 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

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

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

- 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
    - 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
    - Path traversal vulnerability
      - Path traversal – weak protections
      - Path traversal – best practices
    - Log forging
      - Some other typical problems with log files
  - Improper use of security features

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

- 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++
  - Case study – the Ashley Madison data breach
  - Typical mistakes in password management
  - Exercise – Hard coded passwords
- Time and state problems
  - Time and state related problems
  - Serialization errors
  - Exercise TOCTTOU
  - Best practices against TOCTTOU
- 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).

TRAINER:

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

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