scademy
secure coding academy

Crypto Library Programming with OpenSSL

CL-CLP | Virtual classroom | 3 days

Variant: x86

Audience: System architects, software and firmware engineers, safety engineers
Preparedness: General C/C++ development
Exercises: Hands-on

Today, cryptography protects the confidentiality and integrity of data in all states. Modern cryptographic functions are provided by well-established libraries, one of which is OpenSSL. It is imperative to understand how the pieces fit together, as misusing them can completely nullify the protections applied.

This course explores practical cryptography from the ground up, combining theory, practice using OpenSSL, and real-life case studies. First, we define secure-communication requirements (e.g. confidentiality, integrity) and transition into cryptographic primitives and their properties. Participants learn about symmetric encryption modes, hashing, message authentication codes (MACs), and correct usage. Next, asymmetric encryption is introduced, exploring RSA, DSA, and ECC. We then see how these building blocks construct the Public Key Infrastructure (PKI), certificates, and root of trust. At last, we build the TLS protocol and examine some of its vulnerabilities.

The course is invaluable for anyone working directly or indirectly with cryptographic functions. Following the practices and recommendations in this course ensures that cryptography applied will genuinely serve its function and protect data as intended.

Outline:

- IT security and secure coding
- Requirements of secure communication
- Practical cryptography
- Asymmetric (public-key) cryptography
- Public Key Infrastructure (PKI)
- Improper use of cryptography
- Security protocols
- Cryptographic vulnerabilities
- Principles of security and secure coding
- Knowledge sources

scademy
secure coding academy

Participants attending this course will:

Understand basic concepts of security, IT security and secure coding
Have a practical understanding of cryptography
Understand the requirements of secure communication
Understand essential security protocols
Understand some recent attacks against cryptosystems
Get sources and further readings on secure coding practices

Related courses:

- CL-ATP - Advanced TPM Security (Virtual classroom, 3 days)
- CL-CPI - C and C++ secure coding (x86) (Onsite / Virtual classroom, 3 days)
- CL-CPA - C and C++ secure coding (ARM) (Onsite / Virtual classroom, 3 days)
- CL-CCI - Comprehensive C and C++ secure coding (x86) (Onsite / Virtual classroom, 4 days)
- CL-CCA - Comprehensive C and C++ secure coding (ARM) (Onsite / Virtual classroom, 4 days)
- CL-CMI - C and C++ security master course (x86) (Onsite / Virtual classroom, 5 days)
- CL-CMA - C and C++ security master course (ARM) (Onsite / Virtual classroom, 5 days)
- CL-CTS - Security testing native code (Onsite / Virtual classroom, 3 days)

scademy
secure coding academy

Detailed table of contents

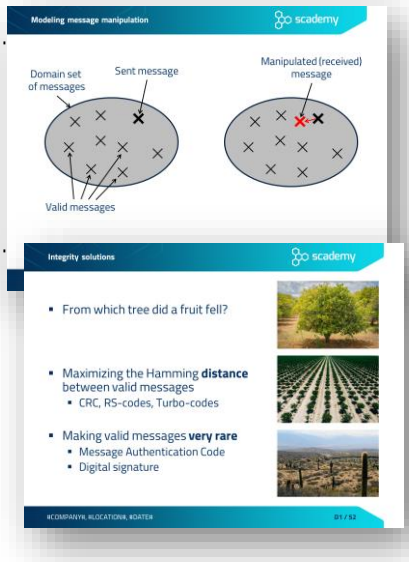
Day 1

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
  - From an infected computer to targeted attacks

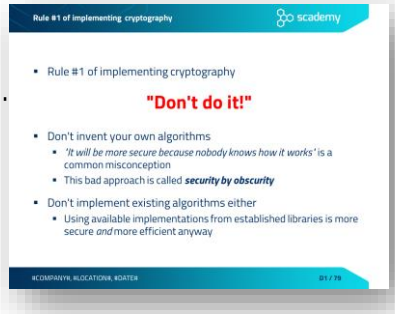
Requirements of secure communication

- Security levels
- Secure acknowledgment
  - Malicious message absorption
    - Feasibility of secure acknowledgment
    - The solution: Clearing Centers
  - Inadvertent message loss
- Integrity
  - Error detection - Inadvertent message distortion (noise)
    - Modeling message distortion
    - Detection of distortion – maximizing Hamming distances
    - Error detection and correction codes
  - Authenticity - Malicious message manipulation
    - Modeling message manipulation
    - Practical integrity protection (detection)
  - Non-repudiation
  - Summary
    - Detecting integrity violation
    - Integrity solutions
- Confidentiality
  - Model of encrypted communication
  - Encryption methods in practice
  - Strength of encryption algorithms
- Remote identification
  - Requirements of remote identification



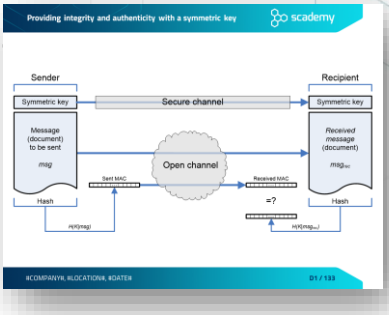
- Anonymity and traffic analysis
  - Model of anonymous communication
  - Traffic analysis
  - Theoretically strong protection against traffic analysis
  - Practical protection against traffic analysis
- Summary
  - Relationship between the requirements

Practical cryptography



- Rule #1 of implementing cryptography.....
- Cryptosystems
  - Elements of a cryptosystem
  - FIPS 140-2
- Symmetric-key cryptography
  - Providing confidentiality with symmetric cryptography
  - Symmetric encryption algorithms
  - Stream ciphers
  - Modes of operation
  - Comparing the modes of operation
  - Authenticated Encryption modes
    - Authenticated Encryption
    - CCM – Counter with CBC-MAC
    - GCM – Galois Counter Mode
    - GCM encryption
    - Symmetric encryption with OpenSSL: encryption
    - Symmetric encryption with OpenSSL: decryption
    - Decryption with OpenSSL
- Other cryptographic algorithms
  - Hash or message digest
  - Hash algorithms
  - SHAttered
  - Hashing with OpenSSL
  - Password management
    - Exercise – Weakness of hashed passwords
    - Password management and storage
    - Brute forcing
    - Special purpose hash algorithms for password storage
    - Case study – the Ashley Madison data breach
    - Case study – Equifax password management issues

- Other cryptographic algorithms (continued)
  - 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
    - Case study – Equifax account freeze PIN code generation



Day 2

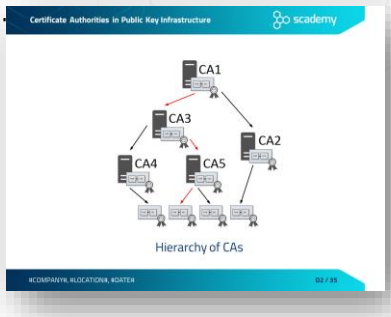
Asymmetric (public-key) cryptography

- 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
  - Group signature schemes
- The Digital Signature Algorithm (DSA)
  - Introduction to DSA algorithm
  - Digital signing with DSA
  - RSA and DSS approaches
- Elliptic Curve Cryptography (ECC)
  - Introduction to ECC
  - Elliptic Curve Discrete Logarithm Problem
  - Calculate ECC Addition
  - Windows CryptoAPI Spoofing Vulnerability

scademy
secure coding academy

Public Key Infrastructure (PKI)

- Root of Trust Concept
  - Man-in-the-Middle (MitM) attack
  - Digital certificates against MitM attack
  - Certificate Authorities in Public Key Infrastructure
  - X.509 digital certificate
  - Certificate Revocation Lists (CRLs)
  - Online Certificate Status Protocol (OCSP)
  - Storing Private Keys (PKCS #8)
  - Generate private key with OpenSSL
  - Storing Multiple Cryptographic Keys (PKCS #12)
  - X.509 File Extensions
  - Generate CA certificate
  - View PEM encoded certificate
  - Transform PEM to DER
  - Generate PKCS #12 key store with OpenSSL
  - Issuing a certificate
  - Generate CSR request with OpenSSL
  - Configure OpenSSL as a CA
  - Revoking a certificate
  - Enroll user certificate in a CA with OpenSSL
  - Cryptographic token interface (PKCS #11)
  - Cryptographic Message Syntax (CMS)



Improper use of cryptography

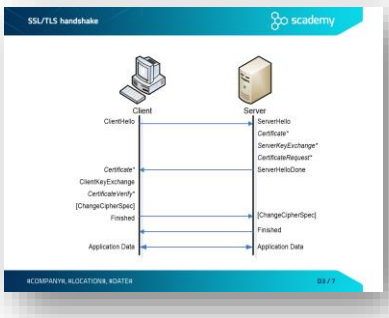
- Time and state problems
  - Time and state related problems
  - Serialization errors
  - Best practices against TOCTTOU
    - Signature schemes
    - RSA signature
    - Bleichenbacher signature forgery
    - Bleichenbacher signature forgery attack
- Sensitive information in memory
  - Protecting secrets 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
- Case study – XSW on XAdES signatures
  - Case study – the original signed document
  - Case study – signed document modification
- Protections against power analysis attacks
  - Protection measures against power analysis
  - Hiding techniques
  - Hiding: adding noise, desynchronization
  - Hiding: dual-rail precharge logic (DRP)
  - Hiding, DRP: sense amplifier based logic (SABL)
  - Hiding: current mode logic (CML)
  - Masking: masked dual rail precharge logic (MDPL)
  - Limitations of PA resisting logic styles
- Fault injection attacks
  - Fault injection attacks (FIA)
  - Fault injection models
  - Practical attacks

Day 3

Security protocols

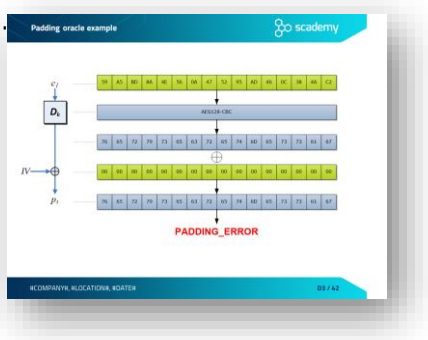
- The TLS protocol
  - SSL and TLS
  - Usage options
  - Security services of TLS
  - SSL/TLS handshake
  - Exercise – SSL/TLS with OpenSSL
    - Datagram Transport Layer Security (DTLS)
    - DTLS Handshake
    - DTLS Challenges
    - DTLS attacks



scademy
secure coding academy

Cryptographic vulnerabilities

- Protocol-level vulnerabilities
  - BEAST
  - CRIME
  - TIME
  - TIME without MitM
  - BREACH
  - Protecting against CRIME/TIME/BREACH
  - FREAK
  - FREAK – attack against SSL/TLS
  - Logjam attack
- Padding oracle attacks
  - Adaptive chosen-ciphertext attacks
  - Padding oracle attack
  - CBC decryption
  - Padding oracle example
  - Exercise: Padding oracle exploit in practice ()
  - Lucky Thirteen
  - POODLE
- RSA timing attack
  - Implementation of encoding/decoding in RSA
  - Fast exponentiation
  - Differences in execution times
  - RSA timing attack
  - Measurements
  - RSA timing attack – principles
  - Correlation of total and partial execution times
  - RSA timing attack – in practice
  - Example – RSA measurements
  - Example – Partial execution times are calculated for first i bits
  - Example – Candidates are ordered according to the variance
  - The RSA timing attack algorithm
  - Practical exploitation using the RSA timing attack
  - Attacking SSL/TLS servers



- Protection against timing attacks
  - Hiding: RSA timing attack countermeasures
  - Masking: using blind signature
  - Real RSA implementations
  - Summary – lessons learnt from side channel attacks
  - Security evaluation of side-channel protections

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++