Summary

TDT4237 - Software Security

What should I be able to answer on my exam?

By Marte Løge

Chapter 1

Security Engineering

,

1.1 Usability and Pshycology

- Authentication vs Identification
- Why do I as a developer want to add insecure code?
- Pretexting
- Phishing
- What does the brain do better than a computer?
- What does a computer do better than the brain?
- How do people act under under uncertainty?
- Social psychology - Authority
- Humans memory and password policy
- What is social engineering?
- Eavesdropping (”avlytting”)
- CAPTHA

1.2 Protocols

- Simple authentication vs. Two-factor authentication
- Basic key management
- Needham-Schroeder protocol
- Kerberos
- BAN logic (not in details, but know what it is)

1.3 Access Control

- What is access control?
- Access Control Lists (ACL)
- Capabilities
- Groups vs. roles in (access system)
- Sandboxing
- Proof carrying code

1.4 Cryptography

- One-way function

1.5 Multilevel security

- What is a security model?
- Bell LaPadula security model
- The simple security property
- Classification levels
- MAC (Mandatory Acces Control) vs DAC (Discretionary Access Control)
- High/low-water-mark
- The Biba model
- (No) Write up/down and (No) read up/down

1.6 Multilateral Security

- Multilevel security vs Multilateral security
- The lattice model
- The Chineese wall

Chapter 2

OWASP

2.1 Information Gathering

- Spiders, robots and crawlers → wget tool
- Search engine discovery → Google search
- Identify application entry points → map attack suface
- Web application fingerprints → get web server info
- Analysis of error codes

2.2 Configuration Management

- Encryptet communication channel → SSL/TLS → HTTPS
- DB listner testing → DB network deamon, port 1521
- Infrastructure
- Application configuration management → comments, error pages, server overload, logging, etc
- File extension management
- Backup, old files and forgotten files → login.asp, login.asp.old
- Admin pages
- HTTP Methods → GET, POST, PUT, DELETE, TRACE, HEAD, OPTIONS, CONNECT

2.3 Authentication

"Confirming that something or someone is authentic"

"Authenticate a person to verifi identity"

- Encrypted channel
  - HTTP and SSL/TLS → HTTPS
  - Encryption Algorithm used → Safe ?
  - Use POST over GET → GET can log sensitive data
- User enumeration
  - "The password is invalid", "The username is invalid"
  - Testing possible combinations of username and password
  - URI probing
  - Automatic generated username/ID → Sequence? Predictable?
  - Guessing and predictability
- Guessable user account
  - Stored default password
  - Weak password policy
  - Information leak in comments (code)
- Bruteforce
  - Dictionary, Search, rule-based
  - "All possible combinations"
  - Rainbow tables → MD5 algorithm
- Bypassing
  - Direct page request
  - Parameter modification
  - SessionID prediction
  - SQL injection
- Forgotten password/Reset password/Remembering password
  - Questions → "What is your mothers name?"
  - Browser cache

  - Forgotten password → Password sended in clear-text to mail? → Unsafe!! (like norwegain, hihi:))

- Browser cache → Logout, timed logout, cached pages, sesstion handling and server checks.
- CAPTCHA
- Multiple factor authentication: → Something you have and something you know
- Race conditions → Account and money transfer

2.4 Session Management

- Session management
  - used to identify users accross requests
  - used to store actions → Shopping cart → Items added
  - Can be tampered with → Webstore → Checkout → Price
  - Analyse → Collect cookies → Algorithm, patterns, etc
  - Coocie overflow
- Cookie Analysis
  - How many cookies is stores? → What information do they contain?
  - Where (sites at webapp) is a new cookie generated?
  - What parts of a website needs a cookie?
  - Token structure → Patterns? Static parts? Clear text? Hash function?
  - Predictability, randomness and guessability
  - Secure flag
  - Expiration
  - Tamering possible?
  - Brute force
  - Cookie overflow
  - Attributes → Secure, HTTPOnly, Domain, Path, expires
- CSRF
  - One-click attack/session riding
  - XSS → Exploits the trust a user have for a particular site
  - CSRF → Exploits the trust that a site has in a user's browser (cookies)
  - Example: Eve: Hello Alice! Look here: <img src:"http://bank.example.com/withdraw?account=Alice&Amount=100000000&For=Eve" >

2.5 Business Logic

- Business rules
- negative amount (e.g price)
- Roles in a system → Access
- Sequential ID generation
- Workflow

2.6 Data Validation

- XSS
  - Reflected XSS (non-persistant) → URI
  - Stored XSS (persistant) → Input fields, forms, etc
  - DOM-based XSS → Poor javascript, metacharacters, etc
- SQL injections (SQLi)
  - Input from client to app
  - Inband → Uses the same channel
  - Out-of-band → Email or other channels
  - Inferial → No transfer of data, reconstruct info by DB behaviour
  - Blind → App hides error details
  - SQLi detection → the use of metacharacters
  - UNION query SQLi → get access to data/structure
  - Stored SQLi → do input validation!
- XML injection → Discovery → Metacharacters
- Buffer overflow → Heap overflow and stack overflow

2.7 Denail of Service (DoS)

- SQL
  - Force database to carry out CPU intensive queries
  - LIKE operator → Often CPU intensive
- Locking custimer/admin accounts → to many login attempts
- Buffer overflow
- User input as a loop
- Failure to release resources

Chapter 3

various topics

- 7 Touchpoints. List.
- Principle of least privilege
- Attack surface
- What is a security requirement?
- Zero-day vulnerability
- Attack tree
- Page map
- Data Flow diagram (DFD)
- Use and misuse cases
- How to calculate risk
- Risk analysis vs. risk assessment vs. risk management