CISSP Domain 8 — Software Development Security

Domain 8 — Software Development Security

The **Software Development Lifecycle or SDLC** phases are:

- Project initiation and planning
- Functional requirements
- System design specifications
- Development
- Documentation
- Testing and evaluation
  - **Acceptance:** verify the system does what it's supposed to
  - **Certification:** formal process of evaluating the security capabilities, a comprehensive technical analysis
  - **Accreditation:** Management reviews the certification, authorizes the system to be implemented or deployed into production, can be provisional or full
- Implementation / deployment

Software Development Models

- **Waterfall:** Each phase is completed before the next one can begin, cannot revisit earlier phases.
- **Structured Programming Development:** Extensive use and reuse of subroutines and block structures.
- **Spiral:** A nested version of the original waterfall model, adds four sub-stages to each phase based on the Deming Plan: Plan, Do, Check, Act or PDCA. It's about continuous improvement.
- **Cleanroom:** Spend more time in design, get it right the first time more often.
- **Iterative Development:** Can go back and re-design as needed. This term isn't used these days, Agile, Scrum, Extreme Programming, etc,. are used instead.
- **Prototyping:** Build a simplified version of the application, release it for review, use feedback from stakeholders to build a second, much better version.
- **Rapid Application Development or RAD:** Strict time limits on each phase, relies on efficient tools to produce quality code quickly.
- **Joint Analysis Development or JAD:** Helps developers work directly with stakeholders.
- **Computer-Aided Software Engineering or CASE:** Today called IDE or Integrated Development Environment, like Visual Studio Team Edition.
- **Component-Based Development:** Use standardized building blocks to assemble, rather than write, the application code.
- **Reuse Model:** Built the application from existing and tested components.
- **Extreme Programming:** The team produces the software in a series of small, fully integrated releases intended to fulfill the owner-defined needs.

Component-Based Development and Reuse Model is where we are today.

Object-Oriented Design

- **Encapsulation:** a.k.a. data hiding
- **Inheritance:** Subclasses can share some or all characteristics of the main class.
- **Polymorphism:** Objects of different data types can be processed differently.
- **Polyinstantiation:** Different versions of the same information can exist at different classification levels. Users at a lower classification level don't know the higher level exists.

Distributed object-oriented systems can use CORBA, Enterprise JavaBean, and DCOM or Distributed Component Object Model. XML web services and cloud services are newer examples.

Code Weaknesses and Secure Coding Practices

- Buffer Overflow
- Covert Channel
- Malformed Input Attacks
- Memory Reuse
- Executable Content / Mobile Code
- Time of Check vs Time of Use or TOCTOU
- Between-the-Lines Attack
- Trapdoor / Backdoor