**Module 3 Architecture & Design**

**3.6 Application Security**

**Software Development Life-Cycle (SDLC) Models**

* Waterfall

1. Steps – requirements gathering, design, implementation/coding, testing/verification, deployment, & maintenance
2. Each stage completely self-contained & completed in order

* Agile

1. Works in cycles, with each cycle producing specific deliverables
2. A type of rapid prototyping through repeated processes
3. Methods – scrum, adaptive software development, crystal, feature-driven development, dynamic systems development method, lean software development, XP (Extreme Programming)

**Secure DevOps**

* AKA DevSecOps/Rugged DevOps
* Security integrated into all development operations, including database design, programming & infrastructure
* Having security practices integrated into entire software delivery cycle
* Address security concerns at beginning of projects
* Add automated security testing techniques
* Continuous Integration – security in every step with updates from centralised, controlled repository
* Security Automation – repeatable/scripted tasks
* Baselining – reference points that require completion & approval of set of predefined project requirements to prevent uncontrolled change & lesson vulnerabilities
* Immutable Systems

1. No changing to systems in place
2. Maintain a known, documented & repeatable configuration

* Infrastructure as Code (IaC)

1. Programmable infrastructure.
2. Infrastructure configuration included in application code

**Compiled vs. Runtime Code**

* Method for creating executable code
* Compiled code uses compiler program such as C/C++
* Runtime uses interpreters such as Java/.NET – generally faster, but less secure

**Change Management/Version Control**

* Go hand-in-hand
* Control & manage software changes – needed for quality & security
* Version Control (AKA Source Control)

1. Prevents tampering/changing source code/executable
2. Tracks software file changes/app code changes
3. Uses distributed storage for code (Git/Github/Subversion)

* Benefits

1. Historical data on changes to files
2. Branching & merging capabilities
3. Traceability

**Provisioning & Deprovisioning**

* Provisioning – creation/update of resource
* Deprovisioning – removal of resource
* Part of SDLC
* Generally automated where software packages made available to users through self-service portal

**Secure Coding Techniques**

* Authentication

1. Hard-coding credentials into code
2. Use of cookies

* Proper error handling

1. Errors should be generic/not divulge specific system/app information
2. Comments should not be visible in end-product

* Proper input validation

1. Scrub & validate input from outside/untrusted sources
2. Use of default values & character limitations

* Normalisation – conversion of data to its anticipated, simplest known form
* Stored procedures – associated with database queries/precompiled SQL statements
* Code reuse/dead code

1. Reusing existing software modules
2. Reused code should be validated for vulnerabilities
3. Dead code – no longer provides useful function, but not scrubbed

* Use of 3rd party libraries & SDKs (Software Development Kit)

1. Know where code comes from – trusted source
2. Check for CVE (Common Vulnerabilities & Exposures)

* Code Signing

1. Signing executable code using certificate-based digital signature
2. Proves author’s identity & provides code integrity

* Data Exposure – encryption of sensitive data at all times (in transit & at rest)
* Encryption

1. Standard encryption algorithms, hashing & digital signatures
2. TLS for data in transit

* Obfuscation/Camouflage

1. Hiding back-end code
2. Prevents code from being reverse-engineered

* Memory Management

1. Optimises performance by assigning blocks of memory to programs & processes
2. Attackers may exploit improper memory utilisation (Buffer Overflow)

* Server-side vs. client-side execution & validation

1. Client-side validation – entered data validated via script on user’s browser before form sent to server
2. Server-side validation – occurs on back-end of server housing application code. Protects against malicious attempts by user to bypass validation