Welcome to the (ISC)² Certified Information Systems Security Professional (CISSP) Training Course

Course Agenda

Domain 1: Security and Risk Management
Domain 2: Asset Security
Domain 3: Security Architecture and Engineering
Domain 4: Communication and Network Security
Domain 5: Identity and Access Management (IAM)
Domain 6: Security Assessment and Testing

Course Agenda (continued)

Domain 7: Security Operations

Domain 8: Software Development Security

Domain 8

Software Development Security

Domain Objectives

1. Understand development methodologies.
2. Explain how maturity models such as the Capability Maturity Model (CMM) can help organizations address software development properly.
3. Understand operations and maintenance.
4. Understand change management and how it applies to software development.
5. Understand the value of integrated product teams (IPTs), including DevOps.

Domain Objectives (continued)

6. Understand secure coding standards and guidelines.
7. Explain the evolution of programming languages and how this relates to security.
8. Explain the benefits of libraries and toolsets.
9. Understand the value of integrated development environments and runtime systems.
10. Understand security weaknesses and vulnerabilities at the source-code level.
11. Explain how to secure application programming interfaces (API) and secure coding practices.

Domain Objectives (continued)

12. Understand security and how it is applied in software environments.
13. Explain the importance of protecting code repositories.
14. Understand the importance of configuration management as an aspect of secure coding.
15. Understand the importance of auditing and logging all changes to software.
16. Understand how risk analysis and mitigation is applied to software security.
17. Explain how to assess security impact of acquired software.

Domain Agenda

- Security in the Software Development Lifecycle (SDLC)
- Secure Coding Guidelines and Standards
- Security Controls in Development Environments
- The Effectiveness of Software Security
- Domain Review

Module 1

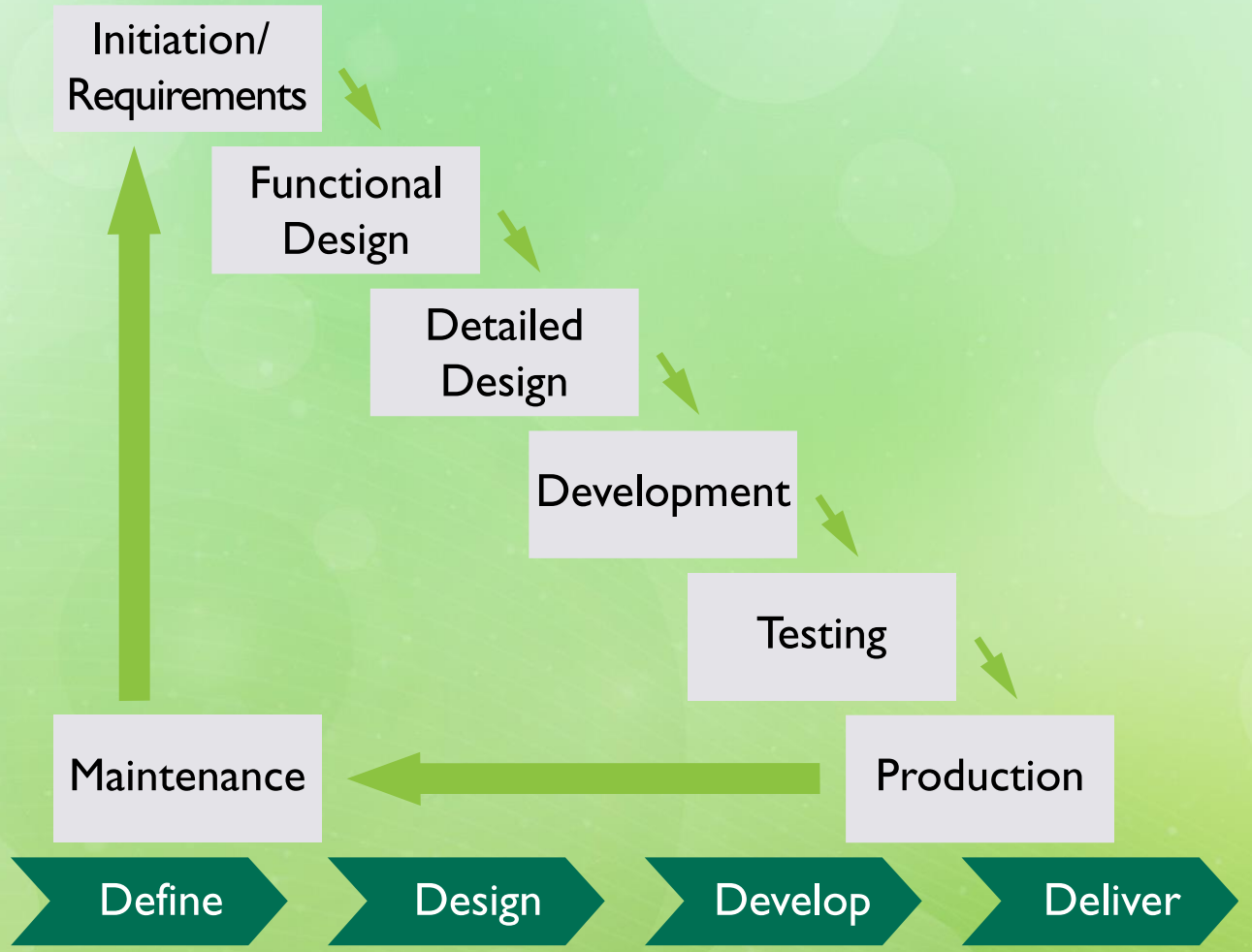
Security in the Software Development Lifecycle (SDLC)

Module Objectives

1. Understand development methodologies.
2. Explain how maturity models such as Capability Maturity Model (CMM) can help organizations address software development properly.
3. Understand operations and maintenance.
4. Understand change management and how it applies to software development.
5. Understand the value of Integrated Product Teams (IPTs), including DevOps.

Typical Phases of the System Lifecycle (SLC)

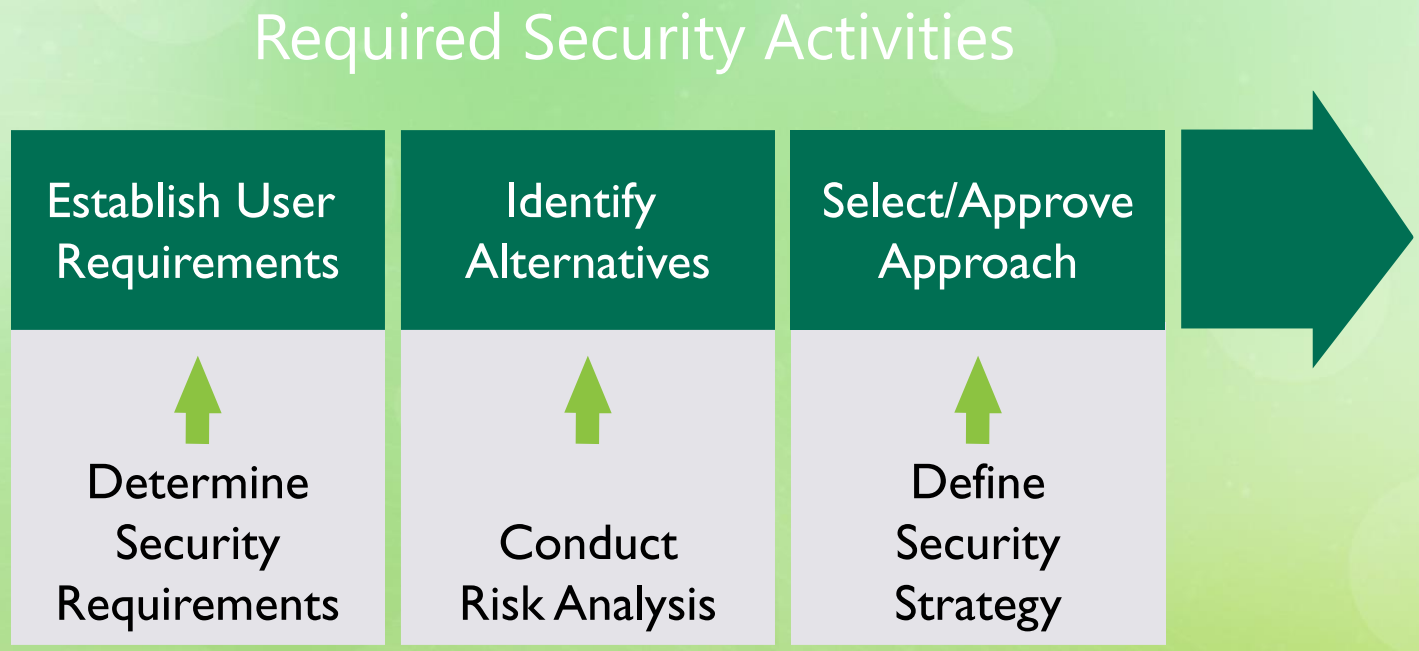
Software Development Lifecycle (SDLC)



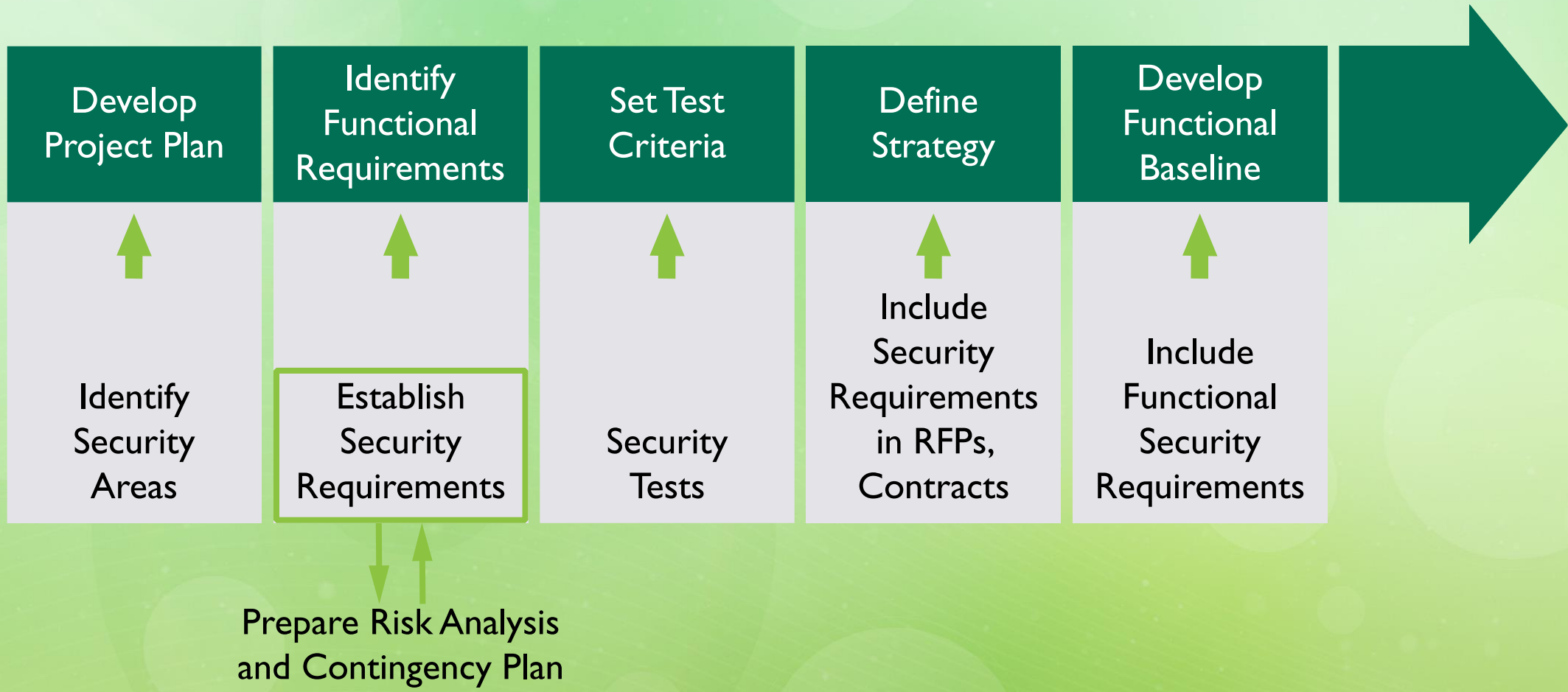
SDLC vs. SLC

Project initiation and planning (+ Management buy-in)	Functional requirements definition	System design specifications
Development and implementation	Documentation	Testing
Transition to production	Maintenance and use	Decommissioning and disposal

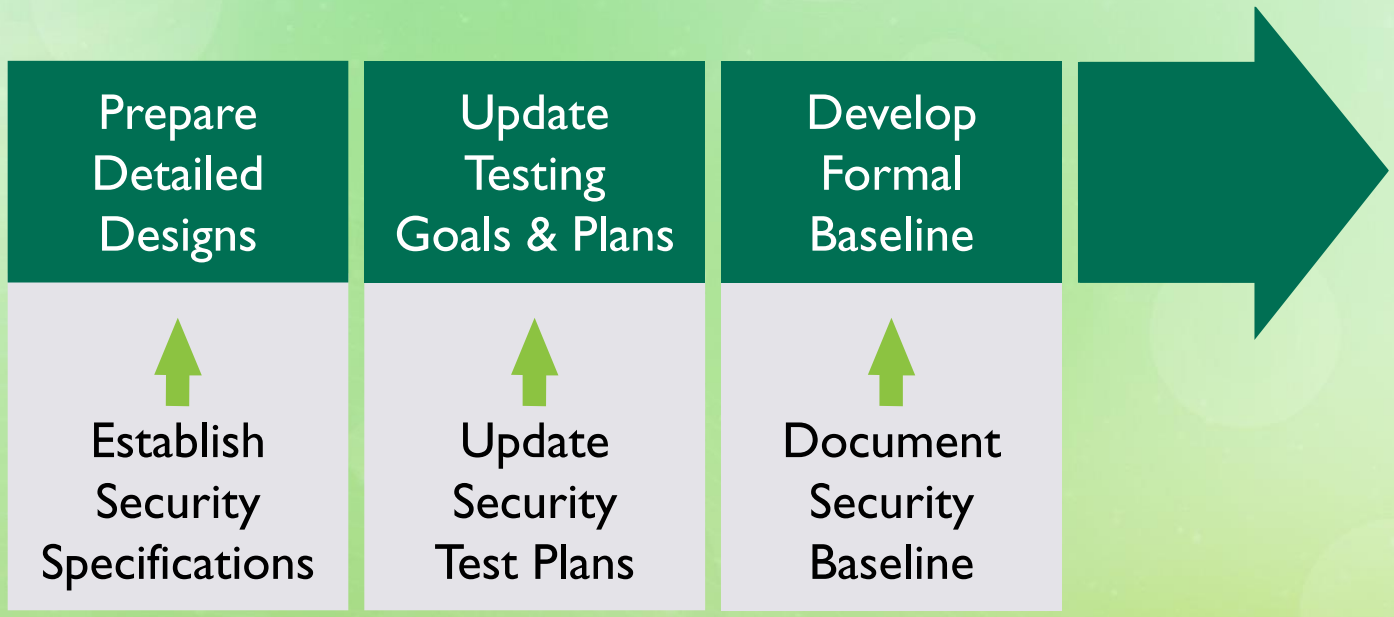
Project Initiation and Planning Security Activities



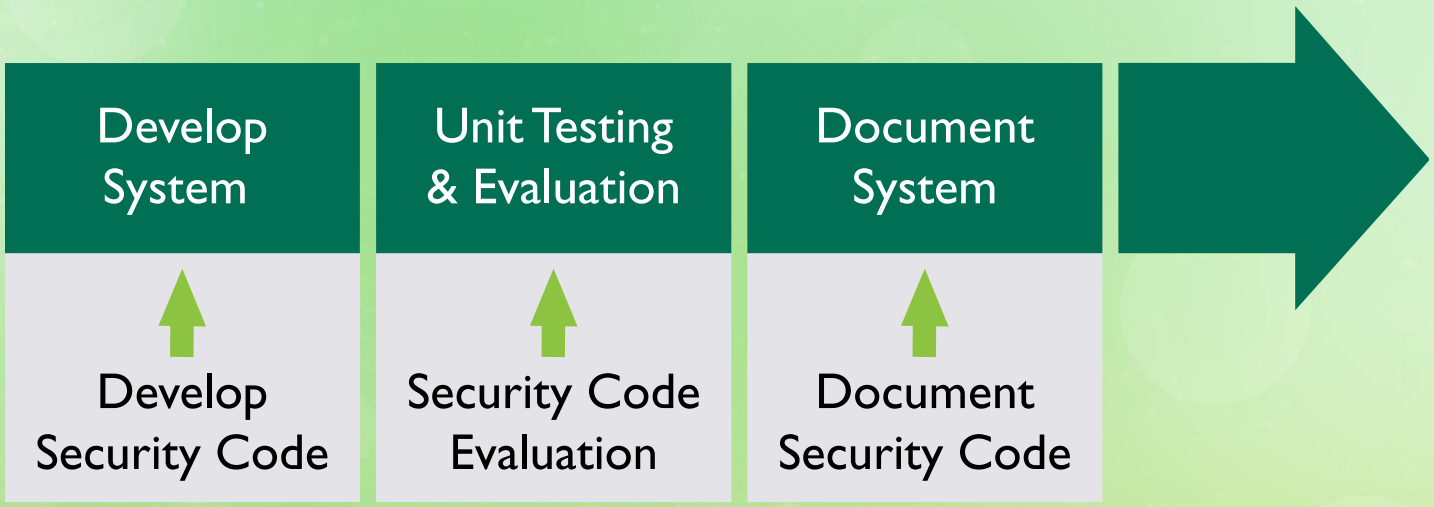
Functional Requirements Specifications
Security Activities



Detailed Design Specifications Security Activities



Development and Documentation Security Activities



Testing and Evaluation Controls

Test data should include the following:

- Data at the ends of the acceptable data ranges
- Various points in between
- Data beyond expected/allowable data points

Test with:

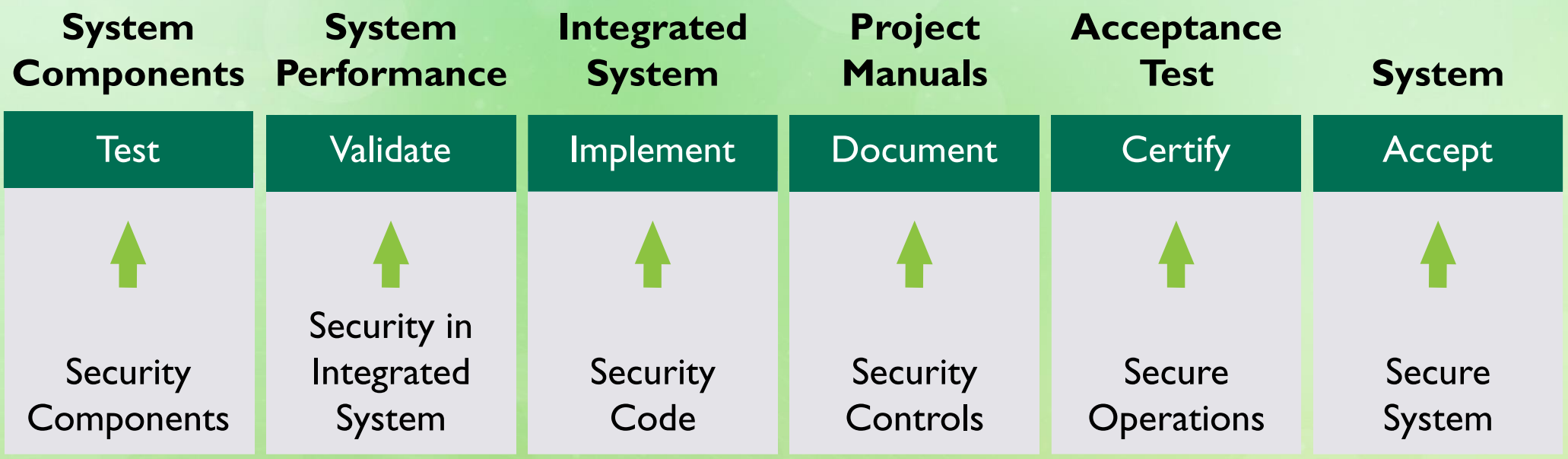
- Known good data
- Never live production data
- Sanitized data

Certification and Accreditation

Certification

Accreditation

Testing , Acceptance, and Transition into Production Security Activities



Decommissioning/Disposal

When an asset is being taken out of production and is decommissioned or retired, the owner is accountable for ensuring the following:

- Recovery requirements
- Media sanitization / destruction
- Asset disposal

Revisions and System Replacement

- Periodic evaluations and audits
- Changes must follow SDLC and be recorded

Operation and Maintenance

- Monitor the performance of the system
- Ensure continuity of operations
- Detect defects or weaknesses
- Manage and prevent system problems
- Recover from system problems
- Implement system changes

Software Development Methods – Primary Models

- Waterfall
- Structured Programming Development
- Agile
- Spiral Method
- Cleanroom

Iterative Development

- Prototyping
- Modified Prototype Model (MPM)
- Rapid Application Development (RAD)
- Joint Analysis Development (JAD)
- Exploratory Model

Other Methods and Models

Computer-Aided Software Engineering (CASE)

Component-Based Development

Reuse Model

Extreme Programming

Software Development Methods

- Waterfall – each phase at a time
  - Easy updates but does not scale to large, complex projects
- Spiral – combination of waterfall and prototype
  - Risk assessment at each phase, with Go/No Go decision
- Iterative Development – multiple waterfall approach
  - Successive refinements in requirements and design
- Joint Analysis Development – users and developers
  - Focus on team of experts; used for mainframe systems development

Software Development Methods (continued)

- Prototyping – build simple version first, then refine
  - 4 steps: concept, design/build, refine, complete and release
- Rapid Application Development (RAD) – rapid prototype
  - Strict time limits imposed to allow quick development
- Modified Prototype Model (MPM) – dynamic model that changes over time as organization needs change
- Exploratory Model – research used to enhance existing model
- Reuse Model – object-oriented

Software Development Methods (continued)

- Cleanroom – zero defect approach
- Computer Aided Software Engineering (CASE) – for large, complex projects
- Component-Based Development – standardized building block approach
- Structured Programming Development – modular development, high quality
- Extreme Programming – 80% function in 20% of the time allotted, using small teams to keep it simple

Model Choice Considerations and Combinations

- Organizations are combining models.
- Security must be included in methodologies.

Capability Maturity Model (CMM) for Software or Software Capability Maturity Model (SW-CMM)

- Focuses on quality management processes
- Five maturity levels

Software Capability Maturity Model (SW-CMM) Levels

Operation and Maintenance

- Monitor the performance of the system
- Ensure continuity of operations
- Detect defects or weaknesses
- Manage and prevent system problems
- Recover from system problems
- Implement system changes

Change Management

Successful change management requires the following:

- Benefits management and realization
- Effective communication
- Effective education, training
- Counter resistance
- Monitoring of the implementation

Typical Change Management Process Phases

Integrated Product and Process Development (IPPD)

Management technique that simultaneously integrates all essential acquisition activities through multidisciplinary teams

DevOps


Software
Development
Quality
Assurance
DevOps
Operations

DevOps

Addresses lack of accountability and disconnect in traditional development

Bridges gap between all major functions involved

Cooperation is facilitated to promote faster and better deployments

Emphasizes collaboration between development, QA, and Operations to ensure close alignment with business objectives

Module 2

Secure Coding Guidelines and Standards

Module Objectives

1. Understand secure coding standards and guidelines.
2. Explain the evolution of programming languages and how this relates to security.
3. Explain the benefits of libraries and toolsets.
4. Understand the value of integrated development environments and runtime systems.
5. Understand security weaknesses and vulnerabilities at the source-code level.
6. Explain how to secure application programming interfaces (API) and secure coding practices.

Secure Coding Guidelines and Standards

Can be used by organizations to encourage developers to follow a standard set of rules

Prevents traditional development where developers coded based on preference or familiarity

Can be used to enforce proper security requirements

Several coding guidelines and standards exist in the industry

The Software Environment

This environment begins with the standard model of hardware resources, with items such as the following:

- Central processing unit (CPU)
- Memory
- Input/output (I/O) requests
- Storage devices

Programming Languages

A programming language is a set of rules telling the computer what operations to perform.

Programming Language Generations

First generation

Second generation

Third generation

Fourth generation

Fifth generation

The Programming Procedure

Assembler:
translates assembly language into machine language

Compiler:
translates high-level language into machine language

Interpreter:
translates into machine language each time

Object-Oriented Technology and Programming

Encapsulation

Inheritance

Polymorphism

Polyinstantiation

- Allows different versions of the same information to exist at different classification levels
- May be useful in preventing inference

Distributed Object-Oriented Systems

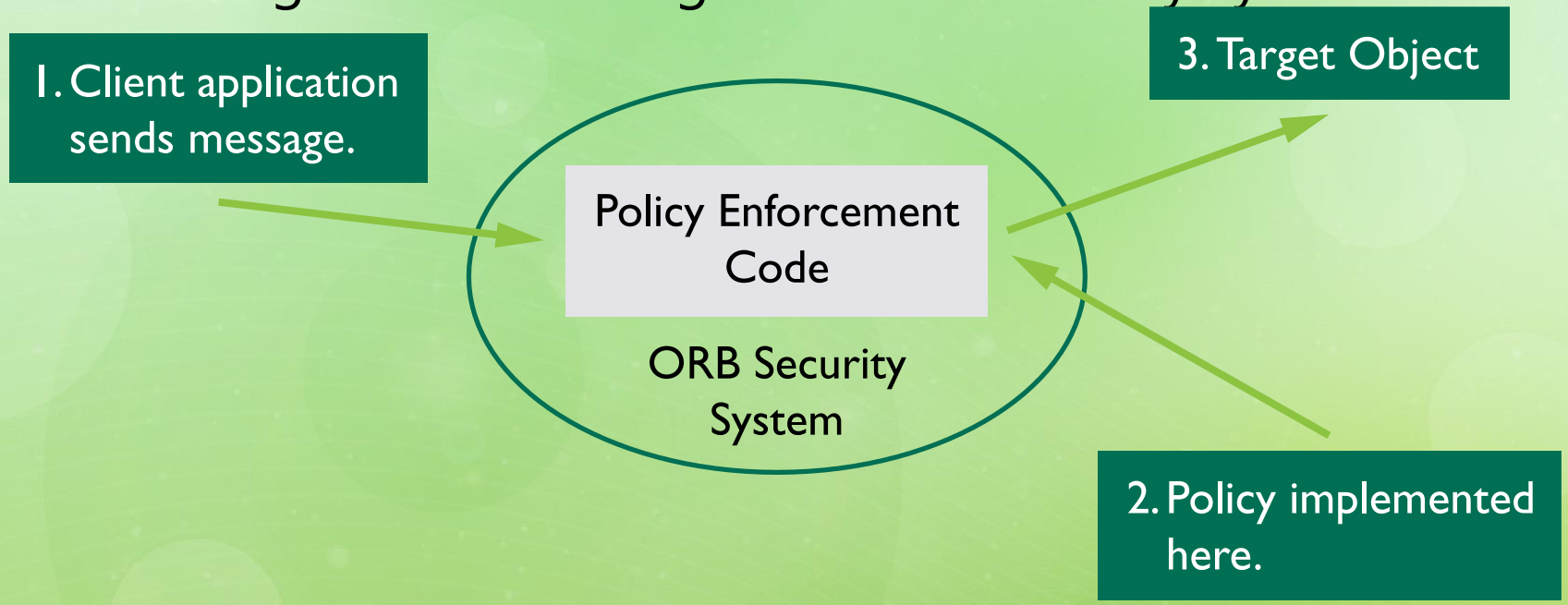
Allow applications to be divided into components that can exist in different locations

Common Object Request Broker Architecture (CORBA)

A set of standards that addresses the need for interoperability between hardware and software

CORBA

- Client sends a message to another object.
- The message is sent through the ORB security system.



CORBA Implementations

- Supported CORBA security features
- CORBA security
- Administration
- Access control mechanisms
- Tools for capturing and reviewing audit logs
- Any technical evaluations

Libraries and Toolsets

A software library consists of pre-written code, classes, procedures, scripts, and configuration data.

Library Benefits

Increased dependability

Reduced process risk

Effective use of specialists

Standards compliance

Accelerated development

Standard Libraries

A standard library in computer programming is the library made available across implementations of a programming language.

Common Programming Language Libraries

The C standard library

The C++ standard library

The Framework Class Library (FCL)

The Java Class Library (JCL)

The Ruby standard library

Programming Tools/Toolsets

A program or application that software developers use to create, debug, maintain, or otherwise support other programs and applications.

Integrated Development Environments (IDEs)

- Combine the features of many tools
- Maximize programmer productivity

Runtime

A runtime system is the collection of hardware and software components of a system that allows a program to run, regardless of the programming language used.

Security Weaknesses at Source Code Level and Secure Coding Practices

There are a number of vulnerabilities at the source code level that need to be addressed properly.

Social Engineering

Uses deception and intimidation to get someone to provide information that they shouldn't

Activity: Security Weaknesses and Vulnerabilities at the Source Code Level and Secure Coding Practices

INSTRUCTIONS:

Review each of the security weaknesses/threats on your own and be able to explain them to someone else in the class. Understand how security needs to be involved to mitigate the following vulnerabilities.

- Buffer overflow
- Citizen programmers
- Covert channel
- Malformed input attacks
- Memory reuse (object reuse)
- Executable content/mobile code
- Time of Check vs Time of Use (TOCTOU)
- Between-the-lines attack
- Trapdoor/ backdoor

Source Code Analysis Tools

- Designed to analyze source code to help find security flaws
- Used in software development phase so that issues are addressed before implementation

Strengths of Source Code Analysis Tools

- Scale well
- Output is good for developers

Weaknesses of Source Code Analysis Tools

Many security vulnerabilities are difficult to find automatically

False positives

Frequently cannot find configuration issues

Difficult to prove actual vulnerability

Difficulty analyzing code that cannot be compiled

Security of Application Programming Interfaces (APIs)

Application Programming Interfaces (APIs)

- Are the “messengers” that carry information between different applications and systems so they can talk to each other
- Are the connectors for the Internet of Things (IoT), allowing our devices to speak to each other
- Are the “unknown, unseen force”
- Must be protected or wrong messages could be carried to components of architectures

Representational State Transfer (REST)

- A means of expressing specific entities in a system by URL path elements
- Allows interaction with a web-based system via simplified URLs

REST-based API Security Recommendations

- Employ the same security mechanisms for your APIs as any web application your organization deploys.
- Do not create and implement your own security solutions.
- Unless your API is a free, read-only public API, do not use single key-based authentication.
- Do not pass unencrypted static keys.
- Use HMAC.

Authentication Options

Basic authentication w/TLS

OAuth 1.0a

OAuth 2

OWASP REST Security Cheat Sheet

“RESTful web services should use session-based authentication, either by establishing a session token via a POST or using an API key as a POST body argument or as a cookie. Usernames and passwords, session tokens, and API keys should not appear in the URL, as this can be captured in web server logs and makes them intrinsically valuable….”

Secure Coding Practices

The following are ways to address vulnerabilities at the source code level

Trusted Computing Bases (TCBs)

Collection of all of the hardware, software, and firmware within a computer system

Reference Monitors

Ensures any subject attempting to access any object has the appropriate rights to do so

Protects the object from unauthorized access attempts by bad actors

Security Kernels

Made up of all of the components of the TCB, and it is responsible for implementing and enforcing the reference monitor

Processor Privilege States

Protect the processor and the activities that it performs

Privilege levels are typically referenced in a ring structure

Security Controls for Buffer Overflows

A buffer overflow:

- Is caused by improper bounds checking on input to a program
- Must be corrected by the programmer or by directly patching system memory

Controls for Incomplete Parameter Check and Enforcement

- The lack of parameter checking can lead to buffer overflow attacks
- Operating systems should offer some type of buffer management

Process Isolation and Memory Protection

Ensure that multiple processes do not attempt to access the same system resources at the same time

Interrupts

Interrupts allow the operating system to ensure that a process is given enough time to access the CPU when necessary to carry out its required functions

Process Encapsulation

Encapsulating a process means that no other process is able to understand or interact with the internal programming code of the process

Time Multiplexing

Allows the operating system to provide structured access to processes that need to use resources according to a tightly managed schedule

Naming Distinctions

Ensure that each process is assigned a unique identity within the context of the operating system

Virtual Address Memory Mapping

- Allows each process to have access to its own memory space as it executes
- Enforced through the operating system's use of the memory manager

Memory Management

Provides an abstraction level for programmers

Maximizes performance with the limited amount of memory available

Protects the operating system and applications once they are loaded into memory

Memory Manager Responsibilities

Relocation

Protection

Sharing

Logical organization

Physical organization

Covert Channel Controls

Communication channel allowing two cooperating processes to transfer information in a way that violates the system's security policy

Covert Channel Controls (continued)

Identify potential covert channels

Analyze these channels to determine whether a channel actually exists

Manual inspection and appropriate testing techniques to verify if the channel creates security concerns

Address them properly through security control implementation

Cryptography

Cryptographic techniques protect the confidentiality and integrity of information

Password Protection Techniques

Operating system and application software use passwords as a convenient mechanism to authenticate users

Inadequate Granularity of Controls

If there is not enough granularity of security, users may be able to gain more access permission than needed

Control and Separation of Environments

Development environment

Quality assurance environment

Application (production) environment

Race Conditions vs. Time of Check vs. Time of Use (TOCTOU) Attacks

- If there are multiple threads of execution occurring at the same time, a TOCTOU attack is possible
- Takes advantage of the dependency on the timing of events that take place in a multitasking operating system
- To avoid TOCTOU attacks, the operating system should use software locking

Social Engineering

These are some of the ways attackers can try to use social influence over users:

- Subtle intimidation
- Bluster
- Pulling rank
- Exploiting guilt
- Pleading for special treatment
- Exploiting a natural desire to be helpful
- Deception

Backup Controls

- Backing up operating system and application software ensures productivity in the event of a system crash
- Operational copies of software should be available in the event of a system crash

Software Forensics

Analysis of program code to determine infringements related to patent, trade secret, or copyright

Mobile Code Controls

Examples of threats to resources include:

- Disclosure of information
- Denial of service (DoS) attacks
- Damaging or modifying data
- Annoyance attacks

Sandbox

Provides a protective area for program execution

Activity: Sandbox Applet Operations

INSTRUCTIONS:

Use the list below to determine if sandbox applets can perform each of the following operations or not by noting “can” or “cannot” next to each one.

1. _____ read secure properties
2. _____ access printing functions
3. _____ access client printer resources
4. _____ save files on the client
5. _____ load native libraries

Activity: Sandbox Applet Operations – Answers

1. *Can* read secure properties
2. *Can* access printing functions
3. *Cannot* access client printer resources
4. *Can* save files on the client
5. *Cannot* load native libraries

Programming Language Support

Type-safe language:

- Method of providing safe execution of programs
- Ensures that arrays stay in bounds, the pointers are always valid, and code cannot violate variable typing

Module 3

Security Controls in Development Environments

Module Objectives

1. Understand security and how it is applied in software environments.
2. Explain the importance of protecting code repositories.
3. Understand the importance of configuration management as an aspect of secure coding.

Security of the Software Environment

The objective of information security is to make sure that the:

- System and its resources are available when needed
- Integrity of the processing of the data and the data itself are ensured
- Confidentiality of the data is protected

Current Software Environment

- More distributed
- Substantial increase in open protocols, interfaces, and source code
- Increased sharing requires increased protection
- More complex

Open Source

Linus’s law: with sufficiently many eyeballs looking at the code, all bugs will become apparent

Database Management System (DBMS) Architecture

- A suite of application programs that typically manages large, structured sets of persistent data
- Stores, maintains, and provides access to data using ad hoc query capabilities

Elements of a DBMS

The database engine itself

The hardware platform

Application software

Users

Database Models

The relationship between the data elements and provides a framework for organizing the data:

- Transaction persistence
- Fault tolerance and recovery
- Sharing by multiple users
- Security controls

Hierarchical Database Management Model

- Oldest of the database models
- Stores data in a series of records that have field values attached
- Collects all the instances of a specific record together as a record type
- Uses parent/child relationships through the use of trees

Network Database Management Model

- Represents its data in the form of a network of records and sets that are related to each other
- Records are the equivalent of rows in the relational model
- Record types are sets of records of the same type

Relational Database Management Model

Data structured in tables

Linkages to other tables allows relationships to be analyzed

Elements of the Relational Mode

Tables or relations

Integrity rules

Data manipulation agents

Attributes of a Table

Attributes

Tuple

Primary keys

Foreign key value

Integrity Constraints in Relational Databases

To solve the problems of concurrency and security within a database, the database must provide some integrity:

- Entity integrity
- Referential integrity

Structured Query Language (SQL)

- Language in which users may issue commands
- These are the main components of a database using SQL:
  - Schemas
  - Tables
  - Views

SQL Sublanguage

Data Definition Language (DDL)

Data Manipulation Language (DML)

Data Control Language (DCL)

Object-Oriented (OO) Database Model

One of the most recent database models

Stores data as objects

Activity: Database Model Review

INSTRUCTIONS:

Match the database model with the correct description.

1. _____ Stores data in a series of records that have field values attached. It collects all the instances of a specific record together as a record type.
2. _____ Allows data to be structured in a series of tables that have columns representing the variables and rows that contain specific instances of data.

a. Hierarchical Database Model
b. Network Database Management Model
c. Relational Database Management Model
d. Object-Orientated Database Model

Activity: Database Model Review (continued)

3. _____ One of the most recent database models.

4. _____ Represents data in the form of a network of records and sets that are related to each other, forming a network of links.

a. Hierarchical Database Model
b. Network Database Management Model
c. Relational Database Management Model
d. Object-Orientated Database Model

Activity: Database Model Review – Answers

INSTRUCTIONS:

Match the database model with the correct description.

1. ___a___ Stores data in a series of records that have field values attached. It collects all the instances of a specific record together as a record type.
2. ___c___ Allows data to be structured in a series of tables that have columns representing the variables and rows that contain specific instances of data.

a. Hierarchical Database Model
b. Network Database Management Model
c. Relational Database Management Model
d. Object-Orientated Database Model

Activity: Database Model Review – Answers (continued)

3. ___d___ One of the most recent database models.
4. ___b___ Represents data in the form of a network of records and sets that are related to each other, forming a network of links.

a. Hierarchical Database Model
b. Network Database Management Model
c. Relational Database Management Model
d. Object-Orientated Database Model

Database Interface Languages

- Open Database Connectivity (ODBC)
- Java Database Connectivity (JDBC)
- Extensible Markup Language (XML)
- Object Linking and Embedding Database (OLE DB)
- ActiveX Data Objects (ADO)

Activity: Database Interface Languages Review

- What is a markup language?
- What is Object Linking and Embedding (OLE)?
- What is the protocol that allows OLE to work?
- What is JDBC?

Activity: Database Interface Languages Review – Answers

1. What is a markup language?

   A system of symbols and rules to identify structures (format) in a document.

2. What is Object Linking and Embedding (OLE)?

   A Microsoft technology that allows an object, such as an Excel spreadsheet, to be embedded or linked to the inside of another object, such as a Word document.

Activity: Database Interface Languages Review – Answers (continued)

3. What is the protocol that allows OLE to work?

   The Component Object Model (COM).

4. What is JDBC?

   An API from Sun Microsystems used to connect Java programs to databases

Application Programming Interfaces (APIs)

API security issues include the following:

- Authentication of users
- Authorizations of users
- Encryption
- Protection of the data from unauthorized entry, accountability, and auditing
- Availability of current data

Tiered Application Approach

- There can be any number of layers
- Three-tier approach is most typical:
  - Presentation layer
  - Business logic layer
  - Data layer

ActiveX Data Objects (ADO)

- Microsoft high-level interface for all kinds of data
- No configurable restrictions on its access to the underlying system
- Newer browsers implement sandboxing and stronger ActiveX controls to help mitigate this vulnerability

Metadata

Metadata is useful because it provides:

- Valuable information about the unseen relationships between data
- The ability to correlate data that was previously considered unrelated
- The keys to unlocking critical or highly important data inside the data warehouse

Online Analytical Processing (OLAP)

OLAP technologies provide an analyst with the ability to formulate queries and then define further queries

Activity: Database Vulnerabilities and Threats

INSTRUCTIONS:

Working with a partner, review your assigned threats and prepare to introduce them to the rest of the class.

Aggregation and inference	Bypass attacks	Compromising database views	Concurrency	Data contamination
Deadlocking	Denial of Service (DoS)	Improper modification of information	Inference	Interception of data
Query attacks	Server access	TOCTOU	Web security	Unauthorized access

DBMS Controls

As a first line of security to prevent unauthorized users from accessing the system, the DBMS should use the following:

- Identification
- Authentication
- Authorization
- Other forms of access controls

Lock Controls

Locks are used for read and write access to specific rows of data in relational systems or objects in object-oriented systems

- Atomicity
- Consistency
- Isolation
- Durability

Other DBMS Access Controls

- View-based access controls
- Grant and revoke access controls
- Security for object-oriented (OO) databases
- Metadata controls
- Data contamination controls

Online Transaction Processing (OLTP)

- Data processing system facilitating and managing transaction-oriented applications
- Two security concerns for OLTP systems:
  - Concurrency
  - Atomicity

Knowledge Management

A key feature of knowledge management is application of artificial intelligence techniques to support business intelligence

Knowledge Discovery in Databases (KDD)

Mathematical, statistical, and visualization method of identifying valid and useful patterns in data

Security Controls in KDD

Protecting the knowledge base

Routinely verifying decisions

Changes to the rules must go through a change control process

Additional and different queries to verify the information

Making risk management decisions

Developing a baseline of expected performance from the analytical tool

Web Application Environment

- Most attacks are conducted at the application level
- There are many exploits and vulnerabilities that exist

Factors that Make Websites Vulnerable

- Designed to be widely accessible
- Usually heavily advertised
- Administrators turn off logging of traffic
- Not well suited for firewalls and intrusion detection systems

Web Application Threats and Protection

- Particular assurance sign-off process for web servers
- Harden operating system of such servers
- Extend web and network vulnerability scans prior to deployment
- Passively assess IDS and IPS technology
- Use application proxy firewalls
- Disable unnecessary documentation and libraries

Web Application Threats and Protection (continued)

- Remove or appropriately secure administrative interfaces
- Only allow access from authorized hosts or networks
- Do not hard code the authentication credentials
- Use account lockout and extended logging and audit
- Ensure the interface is at least as secure as the rest of the application

Open Web Application Security Project (OWASP) Framework

Development Guide

Code Review Guide

Testing Guide

Top Ten Web Application Security Vulnerabilities

OWASP Mobile

Malicious Software (Malware)

- Can compromise programs and data to the point where they are no longer available
- Generally uses the resources of the system it has attacked
- Viruses are the largest class of malware

Viruses

A **virus** is a program written with functions and intent to copy and disperse itself without the knowledge and cooperation of the owner or user of the computer.

Types of Viruses

- File infectors
- Boot sector infectors
- System infectors
- Companion virus
- Email virus
- Multipartite
- Macro virus
- Script virus

Malware Types

- Worms
- Hoaxes
- Trojans
- DDoS zombies
- Logic bombs
- Spyware and adware
- Pranks
- Botnets

Case: WannaCry Ransomware – 2017

What are best practices for protecting against ransomware?

- Always keep your security software up to date to protect yourself against them, patch often and as necessary.
- Keep your operating system and other software updated to the latest versions. Software updates issued by Microsoft and other operating system vendors will frequently include patches for newly discovered security vulnerabilities that could be exploited by ransomware attackers.
- Email is thought to be one of the main infection methods. Be wary and very careful of unexpected emails especially if they contain links and attachments.

Case: WannaCry Ransomware – 2017 (continued)

- Be extremely wary of any Microsoft Office email attachment that requires you to enable macros to view its content. Unless you are absolutely sure that this is a genuine email from a trusted source, do not enable macros and delete the email instead.
- Backing up your most important data regularly is the single most effective way of combating ransomware. Attackers have leverage over their victims by encrypting valuable files and leaving them inaccessible. If the victim has backup copies, they can restore their files once the infection has been cleaned up. Organizations should ensure that backups are appropriately protected or stored off-line so that attackers cannot delete them.

Case: WannaCry Ransomware – 2017 (continued)

- In today's environments, using cloud services could help mitigate ransomware infection, since many retain previous versions of files, allowing you to roll back to the unencrypted form.

Malware Protection: Training and Policies

Do not double-click on attachments	Describe the content of attachments	Do not blindly use the most widely used products as a company standard
Disable Windows Script Host, ActiveX, VBScript, and JavaScript	Do not send HTML-formatted email	Use more than one scanner, and scan everything

Malware Protection: Tools

Scanners	Heuristic scanners	Activity monitors
Change detection	Reputation monitoring/zero-day/zero-hour	Anti-malware policies

Activity: Malware Protection Tools

1. Which tool is known to generate a lot of false alarms?
2. Which tool looks for search strings whose presence is characteristic of a known virus?
3. What is the period of time from when a new malware hosting website is created until it is recognized as malicious?
4. What tool watches for suspicious activity?

Activity: Malware Protection Tools – Answers

1. Which tool is known to generate a lot of false alarms?

   Heuristic

2. Which tool looks for search strings whose presence is characteristic of a known virus?

   Scanner

Activity: Malware Protection Tools – Answers (continued)

3. What is the period of time from when a new malware hosting website is created until it is recognized as malicious?

   Zero-day/Zero-hour

4. What tool watches for suspicious activity?

   An activity monitor

Security of Code Repositories

Goal is to ensure safety of the source code in the development environment, but also while in production for inspection and modification through change control

Configuration Management (CM)

Goal is to guarantee integrity, availability, and usage of the correct version of all system components and how changes are applied

Configuration Management Plans

The set of artifacts (configuration items) under the jurisdiction of CM

How artifacts are named

How artifacts enter and leave the controlled set

How an artifact under CM is allowed to change

How different versions of an artifact under CM are made available

How CM tools are used to enable and enforce CM

Information Protection Management

Protect shared software from unauthorized modification with policies, developmental controls, and lifecycle controls

Module 4

The Effectiveness of Software Security

Module Objectives

1. Understand the importance of auditing and logging all changes to software.
2. Understand how risk analysis and mitigation is applied to software security.
3. Explain how to assess security impact of acquired software.

Effectiveness of Software Security

- Evaluating the effectiveness of software security is important to organizations and having an efficient and secure process for developing applications needs to be in place
- Testing and assessing methods may include:
  - Meaningful metrics
  - Use cases
  - Auditing and logging
  - Certification and accreditation
  - Risk management
  - Change control
  - Testing and verification

Certification and Accreditation

Certification:
The comprehensive technical analysis of something to make sure it meets requirements

Accreditation:
The management decision to accept a solution

NIST SP 800-37 R1

The revised process extends certification and accreditation to emphasize:

- Building information security capabilities as part of the development process
- Maintaining awareness to all stakeholders
- Providing essential information to senior leaders to drive the correct decision

Risk Management Framework (RMF)

The risk management process changes the traditional focus of C&A as a static, procedural activity to a more dynamic approach to effectively manage information system-related risks

RMF Characteristics

- Encourages the use of automation
- Integrates information security
- Emphasizes selection, implementation, assessment, and monitoring of security controls
- Links risk management processes at the information system level to risk management processes at the organization level
- Establishes responsibility and accountability for security controls
- http://csrc.nist.gov/groups/SMA/fisma/framework.html

Certification for Private Organizations

Why private organizations may choose certification:

- Control framework
- Low overhead
- Use of standards
- Includes all aspects of a system's security

Auditing and Logging of Changes

Systems and network device reporting is important to the overall health and security of systems

Logs

- Logs are records of actions and events that have taken place on a computer system
- They provide a clear view of who owns a process, what action was initiated, when it was initiated, where the action occurred, and why the process ran
- They are primary record keepers of system and network activity

Auditing

The enterprise should have auditing policies in place that effectively and efficiently collect information regarding critical events in the form of logs and to manage them appropriately

Change Management

Organizations need to be able to plan for change, manage it through a well-defined lifecycle, approve changes, document change, and roll it back if required

Information Integrity, Accuracy, and Auditing

Information integrity	Information accuracy	Character checks
Relationship checks	Transaction limits	Information auditing

Risk Analysis and Mitigation

Risk: an event that has a probability of occurring and could have either a positive or negative impact to a project should that risk occur

Risk Management

- An ongoing process that continues through the life of a project
- Includes processes for:
  - Risk management planning
  - Identification
  - Analysis
  - Monitoring
  - Control

Testing and Verification

- When mitigations are implemented, they must be tested
- Development environments are supported with testing teams and quality assurance

Testing and Verification Roles

- Security findings should be addressed the same as any other change request
- The developer or system owner does not declare the risk mitigated without concurrence of an independent verification and validation (IV&V)

Code Signing

- Code signing
  - A technique that can be used to:
    - Ensure code integrity
    - Determine who developed a piece of code
    - Determine the purposes for which a developer intended a piece of code to be used
- Certificates
  - Digital certificates that will help protect users from downloading compromised files or applications

Code Signature Component

Seal

Digital signature

Unique identifier

Code Signature Limitation

Cannot guarantee a piece of code is free of security vulnerabilities

Cannot guarantee an app will not load unsafe or altered code during execution

Is not a DRM or copy protection technology

Regression and Acceptance Testing

Whenever developers change or modify their software, even a small tweak can have unexpected consequences

Regression Testing

- Tests existing software applications to make sure that a change or addition has not broken any existing functionality
- Catches bugs that may have been accidentally introduced into a new build or release candidate
- Ensures that previously eradicated bugs continue to stay dead

Acceptance Testing

A formal test conducted to determine whether a system satisfies its acceptance criteria and to enable the customer to determine whether to accept the system

Assess Security Impact of Acquired Software

Software assurance is the level of confidence that software is free from vulnerabilities, either intentionally designed into the software or accidentally inserted at any time during its lifecycle, and that it functions in the intended manner.

Software Assurance in the Phases of Acquiring Software

Planning

Contracting

Monitoring and Acceptance

Follow-on

Planning Phase

Needs determination:

- Develop software requirements
- Create an acquisition strategy
- Develop evaluation criteria and an evaluation plan

Contracting Phas

Create/issue the solicitation or RFP

Evaluate supplier proposals

Finalize contract negotiation

Monitoring and Acceptance Phas

Establish and consent to the contract work schedule

Implement change control procedures

Review and accept software deliverables

Follow-on

Sustainment

Disposal or decommissioning

Software Assurance Policy

Ensure a well-documented software assurance policy and process is in place in the enterprise

Risks Associated with Software Vulnerabilities

- Unintentional errors
- Intentional insertion of malicious code
- Theft of vital information
- Theft of personal information
- Changed product
- Inserted agents
- Corrupted information

Acquisition Process

System and software assurance focuses on the management of risk and assurance of safety, security, and dependability within the context of system and software lifecycles

Module 5

Domain Review

Domain Summary

- Understand the Software Development Lifecycle (SDLC) and how to apply security to it.
- Identify which security control(s) are appropriate for the development environment.
- Assess the effectiveness of software security.
- To protect applications and the functions they provide, we need to involve security at the beginning of the SDLC.
- Organizations can choose the correct methodologies for applications development, but the development methodology needs to involve security as part of the process.

Domain Summary (continued)

- There are maturity models and other methods that can be used by organizations to mature and get better in software development and get security is involved.
- Change management is useful in allowing changes to anything that is already running in production, including applications and systems. Security needs to be a part of the entire change control process.

Domain Summary (continued)

- Organizations need to understand the benefits of integrating traditionally separate environments in software. Integrating the development area together with the quality assurance function and the operations environment provides a better way to understand and address goals and objectives.
- Organizations need to standardize on using secure coding guidelines and standards that exist in the industry. These can provide guidance on how to develop secure code in applications by using toolsets, program languages, libraries, and other methods.

Domain Summary (continued)

- It is important to address weaknesses at the source code level. These include making sure we protect APIs and addressing known vulnerabilities that exist in application software environments such as buffer overflows, escalation of privileges, and data validation.
- Applications are instrumental in providing access and control of database environments. Protecting against vulnerabilities and exploits and risk in the database environment needs special attention. Using controls related to concurrency, integrity protection and inference and aggregation becomes very important.

Domain Summary (continued)

- Protecting the web application environment is very challenging to organizations and needs to be done in a structured and layered defense model. Data validation is one of the most important focuses in web environments.
- Malicious software are applications that are written to do something malicious. Protecting against all of the different flavors of malware requires a consistent and effective malware protection program within the organization.

Domain Summary (continued)

- On a regular basis, it is important to measure and provide assurance related to the effectiveness of software security. Having software assurance policies and procedures and assessment methods is how we address this need.
- Risk management processes need to be applied in the software environment, and it becomes important to provide assurance for any software that is acquired and purchased through vendors and third parties.

Domain Review Questions

1. The Software Engineering Institute's Capability Maturity Model (CMM) Integration focuses on:

A. Software development methodologies

B. Systems integration

C. Process management

D. Software testing and evaluation

Answer

The correct answer is C.

CMM is a process improvement methodology to allow organizations to mature to better levels in relation to process improvement.

Domain Review Questions (continued)

2. Two cooperating processes that simultaneously compete for a shared resource, in such a way that they violate the system's security policy is commonly known as:

A. Denial of service (DoS)

B. Race condition

C. Object reuse

D. Overt channel

Answer

The correct answer is B.

Race condition occurs when two processes need to carry out their tasks against one resource. The processes, however, need to execute in the correct order, process 1 first, process 2 second. If that order can be disrupted by an attacker, then the attacker can manipulate the output of the results of the combined action of the two processes and potentially create a different outcome than the one intended.

Domain Review Questions (continued)

3. Programmed procedures which ensure that valid transactions are processed accurately are referred to as:

A. Data installation

B. Application controls

C. Operations controls

D. Physical controls

Answer

The correct answer is B.

Key word is the word “programmed” that indicates they are applications. Plus valid transactions would need to be ensured as part of the application controls.

Domain Review Questions (continued)

4. Buffer overflow and boundary condition errors are subsets of:

A. Race condition errors

B. Access validation errors

C. Exceptional conditional handling errors

D. Input validation errors

Answer

The correct answer is D.

Inadequate input, or data validation, is the problem that relates to most attacks and conditions related to application problems. Validating input properly is the best control to avoid many attacks and buffer overflow conditions.

Domain Review Questions (continued)

5. Copies of essential application programs, documentation, and electronic data should be:

A. Stored with the computer system

B. Licensed by users

C. Maintained by the developers

D. Stored at a backup site

Answer

The correct answer is D.

Several key words here, such as “copies” or even “essential” that tell us we are talking about valuable assets that need to be stored at a backup site.

Domain Review Questions (continued)

6. A property that ensures only valid or legal transactions that do not violate any user-defined integrity constraints in DBMS technologies is known as:

A. Durability

B. Isolation

C. Consistency

D. Atomicity

Answer

The correct answer is C.

Consistency as part of the ACID test ensures that transactions that are applied do not affect the integrity of the database and its contents. The integrity of the database needs to be the same as it was before the transaction was applied.

Domain Review Questions (continued)

7. The ability to combine non-sensitive data from separate sources to create possibly more sensitive information is referred to as:

A. Concurrency

B. Inference

C. Polyinstantiation

D. Aggregation

Answer

The correct answer is D.

Combining smaller things together to possibly come up with the ability to infer sensitive information is referred to as aggregation, in fact, the word itself means “combining things together.” Inference is the ability to deduce more sensitive information.

Domain Review Questions (continued)

8. The purpose of polyinstantiation is to prevent:

A. Low-level users from inferring the existence of higher level data

B. Low-level users from inferring the existence of data in other databases

C. Low-level users from accessing low-level data

D. High-level users from inferring the existence of data at lower levels

Answer

The correct answer is A.

Polyinstantiation allows different versions of the same information to exist at different classification levels to prevent inference of more sensitive information that exists at higher levels.

Domain Review Questions (continued)

9. Which virus type changes some of its characteristics as it spreads?

A. Boot sector infector

B. Macro

C. Stealth

D. Polymorphic

Answer

The correct answer is D.

The word polymorphism means many changes. Polymorphic viruses change something about themselves as they infect to try and hide from detection programs.

Domain Review Questions (continued)

10. Which of the following BEST describes a logic bomb?

A. Functions triggered by a specified condition

B. Cause the execution of unanticipated functions

C. Used to remove data or copies of data from the computer

D. Used to move assets from one system to another

Answer

The correct answer is A.

A logic bomb is defined as malware that waits for a specific condition to exist before its negative (damaging) payload is triggered. The condition can be related to time, or specific parameters that exist in the system.

Computerized Adaptive Testing (CAT)

- **Exam language availability:** English
- **Length of exam:** 3 hours
- **Number of questions:** 100-150
- **Question format:** Multiple choice and advanced innovative questions
- **Passing grade:** 700 out of 1,000 points
- **Testing Center:** ISC)^2 Authorized PPC and PVTC Select Pearson VUE Testing Centers
- For additional information, go to www.isc2.org/certificatons/CISSP-CAT

CISSP Linear Examination Information

- **Exam language availability:** French, German, Brazilian Portuguese, Spanish, Japanese, Simplified Chinese, Korean
- **Length of exam:** 6 hours
- **Number of questions:** 250
- **Question format:** Multiple choice and advanced innovative questions
- **Passing grade:** 700 out of 1000 points
- **Testing center:** ISC)^2 Authorized PPC and PVTC Select Pearson VUE Testing Centers

Pearson VUE

- Ready to sign up for the exam? Visit the Pearson VUE website to create an account and book your exam.
- What to expect in a Pearson VUE test center:
  - At Pearson VUE, your security matters to us. You will experience some - or all - of the security measures featured in this video. Be prepared; for specific security requirements, please check the relevant documentation/website/FAQ information related to your test program
  - https://youtu.be/T6tK_tY2AQQ