

Class Five

Topics:

- **When software goes bad**
 - Malicious software
 - Security flaws in software
- **The System Development Process**
 - System development lifecycle
 - Capability Maturity Model (CMM)
 - Software QA and testing
 - Provably secure software
 - Code auditing
- **Internal Controls and Application Software**
 - Change Control
 - Testing & Quality Assurance
 - Separation of Duties
- **Database Management Security Issues**

8/24/2007

Class 5

1

Application Software Security Issues

- Does the application do what it is supposed to?
- Is the application reliable?
- Are users given access only to necessary functions?
- Is the organization's security policy supported?
- Is the best use made of security features provided by the hardware, operating system, and database?
- Has the application been implemented and is it being operated in a manner consistent with the organization's security policy?

8/24/2007

Class 5

2

Malicious Software

- Virus
- Worm
- Trojan Horse
- RAT (Remote Access Trojan)
- Spyware
- Logic bomb



8/24/2007

Class 5

3

Security Flaws in Software

- Not all software which compromises security is malicious:
 - Poor programming
 - Poor design
 - Poor implementation
 - Incorrect use
- Can all cause security problems without any intentionally "malicious" code
- Insecure software is often used by attackers to gain entry to systems

8/24/2007

Class 5

4

Security Flaws

- Validation Flaw – not checking input
 - Buffer overflow
 - SQL injection in Web forms
- Domain Flaw – weakness in protecting environments
 - Object Reuse (memory allocation, swap space, etc.)
- Serialization Flaws – events don't happen in the expected order
 - TOCTTOU (Time Of Check To Time Of Use)
 - Race Conditions
- Covert Channels - hidden information leakage

8/24/2007

Class 5

5

Engineering Secure Software

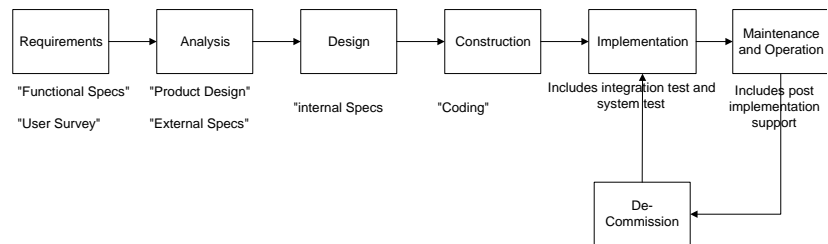
- Development methodology
 - Classic “waterfall” System Development Life Cycle
 - Spiral SDLC variation
 - Other methodologies
- Built-in quality and assurance
- Secure coding standards

8/24/2007

Class 5

6

The System Development Lifecycle



8/24/2007

Class 5

7

Security Considerations at Each Step of the Lifecycle

- All specifications are clear and complete
- Requirements meet business needs
- Requirements include business needs for appropriate security, including internal controls and availability
- Design includes:
 - Error checking
 - Audit trails
 - Support for separation of duties
 - Support for specific need-to-know and sensitivity classification requirements
 - Full use of operating system security mechanisms, including authentication and access controls
- Construction environment is separate from production. All programs are reviewed and authorized before being promoted to production.
- Implementation includes proper testing, and removal of any development-specific access
- Maintenance and operation supports security features of software.

8/24/2007

Class 5

8

Capability Maturity Model

- Software quality depends on the quality of the software development process
- The Carnegie Mellon University Software Engineering Institute (SEI) formalized process quality into a five level hierarchy:
 - Level 1 – Initiating: Ad hoc processes
 - Level 2 – Repeatable: Project management
 - Level 3 – Defined: Engineering processes, documented standards and processes
 - Level 4 – Managed, quantitative process control using statistical measurements of success. Improve the product.
 - Level 5 – Optimizing, continuous improvement in processes. Improve the process.
- Levels of process maturity
- NSA has specified a Systems Security Engineering (SSE) CMM.
 - Required for some government contracts
 - More info at <http://www.sse-cmm.org/index.html>

8/24/2007

Class 5

9

Application Software Security and Integrity Checks

- Error checking
- Ability to control user access to functions
- Full use of operating system security features
- NO bypassing of operating system or hardware security controls (e.g., no "tricks" to permit use of privileged instructions)
- Integration of application-level authentication and access control with operating system features
- Auditability of transactions
- Audit trails for any master file change
- Protection of sensitive information
- Full accountability for individual actions
- Least privilege (e.g., no assuming "superuser" power unless ABSOLUTELY required)

8/24/2007

Class 5

10

Methods of Testing

- Scope:
 - Unit (single module)
 - String (related modules)
 - System (entire system)
- Strategies
 - Parallel testing of entire systems vs. existing production environment
 - Compliance to specs vs. attempts to “break” software through “out of spec” inputs
 - White Box vs. Black Box (also called Openbox vs. Closedbox testing)
- Test Tools:
 - Code execution coverage – How much code is exercised during a test?
 - Keystroke replay and response recording – Simulate and automate manual user input
 - “Fuzzing”, generating erroneous and unexpected input to see if application breaks
 - Code audits and code scanning – Automated code review to find common errors

8/24/2007

Class 5

11

Test Data

- Should NOT be the production database!
- Should not be the production database copied into a test environment
- Should be synthetic, designed to test all possible logical paths in the software
- Should test both correct and error conditions
- Should include even "nonsense" conditions

8/24/2007

Class 5

12

Software Engineering – Design for Security

- Economy of Mechanism
- Fail Safe Default
- Complete Mediation by Security Mechanism
- Open Design (esp. with vendor-provided software)
- Separation of Privilege
- Least Privilege
- Least Common Mechanism (as little functionality in lowest layers as required)
- Psychological Acceptability, User Friendly

8/24/2007

Class 5

13

Software Engineering – Formal Methods

- “Informal Methods”
 - In many cases, informal methods can still produce very good software
- Formal Methods
 - Mathematical proof of software correctness
 - Automated verification tools in formal proof of security
- Formal Methods in information security
 - Define security policy of software as a set of assertions
 - Attempt to prove assertions cannot be violated
- Required for higher level TCSEC (“orange book”) certification
- More current efforts involve modeling cryptographic protocols

8/24/2007

Class 5

14

Software Engineering - Code Auditing

- Origins in software engineering practices
 - Code walkthrough
 - Peer review
- Detailed manual review of program code for errors
- Used in many open source products for assurance
- While not provably secure, these products can achieve a high level of reliability
- The OpenBSD operating system uses code auditing extensively
- Automated tools may assist:
 - RATS
 - FlawFinder

8/24/2007

Class 5

15

Internal Controls and Application Software

- Application software records business transactions
- Internal controls ensure the integrity of business records
- Software design and operation are thus essential to maintaining the integrity of business records.
- Application software should support internal control objectives, and do so in a reliable fashion:
 - completeness
 - accuracy
 - authorization
 - validityof all transactions during application processing.

8/24/2007

Class 5

16

Components of Application Software Internal Controls

- Separation of duties
 - Between application development and application operations
 - Between different incompatible application user roles
- Configuration and change management
 - State of software is known
 - Software is only changed via authorized, documented procedures
- Secure development standards
 - Input validation
 - User authentication and access controls
 - Application level audit trails
- Secure operations procedures
 - Input controls
 - Processing controls
 - Output controls

8/24/2007

Class 5

17

Separation of Duties in Application Development

- Required to minimize the risk of fraud, minimize the probability of introducing malicious code, to enforce accountability, and to ensure a properly secure configuration is maintained.
- Areas to be separated:
 - End Users
 - Application Development
 - Operations
 - Technical Administration
- Where separation of duties is not practical, compensating controls should be adopted.
 - Audit trails to establish post-facto accountability
 - Use of least privilege to prevent unnecessary access
 - Outside verification that transactions are authorized and correctly recorded

8/24/2007

Class 5

18

Separation of Duties - Development and Production

- Fites - "separating incompatible duties to strengthen internal controls"
- Why?
 - Ensure quality of applications
 - Ensure accountability for production environment
 - Ensure ALL production changes are properly authorized
 - Ensure production system fulfills business requirements
 - Prevent malicious code
- How?
 - ACLs to prevent programmer update/write access to production environment
 - Separate machines for production and development (if feasible)
 - User approval of all specifications for application change
 - User approval of all completed changes before promotion
 - Code review
- *Configuration and Change Management*

8/24/2007

Class 5

19

Configuration and Change Management

- Change management is the configuration of system changes
- Configuration management includes change management as well as definition of the initial system state
- The initial system state and subsequent changes must be documented and approved by the data owners
- Change management ensures that changes to a production system are
 - Properly authorized
 - Tested to ensure they are correct per specifications, and are in compliance with applicable standards
 - Properly documented
- Librarian function involves promotion and tracking of changes
- Emergency changes
 - Define "emergency"
 - Permit violating other rules as long as proper accountability and post-facto review ensured
 - **The only time when an application developer should directly work in a production environment**

8/24/2007

Class 5

20

What is a Database Management System (DBMS)?

- A logical method of organizing data on a computer, such that the organization corresponds to the process or entity about which the data represents
- The data structure reflects the real world, and not the conveniences of the computer
- Most modern databases are relational, which means:
 - Data is stored in tables, based on a row/column format
 - Tables may be searched, or rows individually accessed based on a column called a key
 - Tables may be linked to other tables, based on a common key
- Relational databases include programming-like features, such as triggers and stored procedures
- Structured Query Language (SQL) is a common language for defining and accessing relational databases
- Common DBMS software includes:
 - Oracle
 - Microsoft SQLServer and Sybase
 - IBM DB2
 - Microsoft Access

8/24/2007

Class 5

21

DBMS Security Issues

- The DBMS becomes a programming language, subject to the same abuses
 - Poor code
 - Malicious code
- The DBMS may (should!) provide access control at a finer granularity than the Operating System
- DBMS supports transactional integrity
- Integration of DBMS, application software, and operating system security systems can become problematic.

8/24/2007

Class 5

22

DBMS Security Issues

- **Transactional Databases**
 - Transactional integrity
 - DBMS access controls should support separation of duties
- **Statistical Databases**
 - Inference – of confidential individual fact from statistical aggregates
 - Aggregation – inverse of inference, where the aggregate is confidential
 - Trade off between value of disclosed data vs. protection of confidentiality
- **Multi-Level (confidential, classified) Databases**
 - Apply classification attributes to data attributes
 - User's view of data is different depending on their clearance
 - Polyinstantiation – multiple records or tuples referring to same real world fact, but appear different for different user classification levels

8/24/2007

Class 5

23

DBMS Security Issues

- Typical integrity mechanisms:
 - Encapsulated update, consistency, atomicity
 - Redundancy and recovery
 - Fine grain access control
 - Transaction control and layered update
 - Auditing
 - Authentication
 - Fail-safe defaults and human factors
- It either happens completely and correctly or not at all

8/24/2007

Class 5

24