Chapter 14 Notes

\*\*\*FIRST NOTES AFTER MIDTERM

Security Engineering is concerned with eh development and evolution of systems that can resist malicious attacks, which are intended to damage the system or its data. Software security engineering is part of the more general field of computer security.

When you consider security issues, you have to consider both the application software (The control system, the information system, etc.) and the infrastructure on which this system is built. The infrastructure for complex applications may include:

* An operating system platform, such as Linux or Windows
* Other generic application that run on that system, such as web browsers an email clients
* A database management system
* Middleware the supports distributed computing and database access
* Libraries of reusable components that are used by the application software

Application Security = a software engineering problem where software engineers should ensure that the system is designed to resist attacks

Infrastructure security = a management problem where system managers configure the infrastructure to resist attacks. System manager have to set up the infrastructure to make the most effective use of whatever infrastructure security features are available. They also have to repair infrastructure security vulnerabilities that come to light as the software is used.

System Security Management is not a single task but includes a range of activities:

* User and Permission Management
  + Includes adding and removing users from the system, ensuring that appropriate user authentication mechanisms are in place and setting up the permissions in the system so that users only have access to the resources that they need
* System Software Deployment and Maintenance
  + Includes installing system software and middleware and configuring these properly so that security vulnerabilities are avoided. It also involves updating this software regularly with new versions of patches, which repair security problems that have been discovered.
* Attack Monitoring, Detection, and Recovery
  + Includes activities which monitor the system for unauthorized access, detect, and put in place strategies for resisting attacks, and backup activities so that normal operation can be resumed after an external attack.

Security Management is usually not consider to be part of Application Security Engineering

**Security Risk Management (Section 14.1)**

* Risk management is concerned with assessing the possible losses that might ensue from attacks on assets in the system, and balancing these losses against the costs of security procedures that may reduce these losses.
* Risk Management is a business issue rather than a technical issue so software engineers should not decide what controls should be included in a system. It is up to senior management to decide whether or not to accept the cost of security or the exposure that results from a lack of security procedures. Rather, the role of software engineers is to provide informed technical guidance and judgment on security issues. They are, therefore, essential participants in the risk management process.
* Critical input to the Risk Assessment and Management process is the organizational security policy. The organizational security policy applies to all systems and should set out what should and should not be allowed. The security policy sets out conditions that should always be maintained by a security system and so helps to identify risks and threats that might arise. The security policy therefore defines what is and what is not allowed. In the Security Engineering process, you design the mechanisms to implement this policy.
* Risk Assesment starts before the decision to acquire the system has been made and should continue throught the system development process and after the system has gone into use.
* Stages of Risk Assessment
  + Preliminary Risk Assessment
    - At this stage, decisions on the detailed system requirements, the system design, or the implementation technology have not been made. The aim of this process is to decide if an adequate level of security can be achieved at a reasonable cost. If this is the case, you can then derive specific security requirements for the system. You do not have information about potential vulnerabilities in the system or the controls that are included in reused system component or middleware.
    - Focuses on deriving security requirements
  + Life-Cycle Risk Assessment
    - Takes place during the system development life cycle and is informed by the technical system design and implementation decision. The results of the assessment may lead to changes to the security requirements and the addition of new requirements. Known and potential vulnerabilities are identified and this knowledge is used to inform decision making about the system functionality and how it is to be implemented, tested, and deployed
  + Operational Risk Assessment
    - After a system has been deployed and put into use, risk assessment should continue to take account of how the system is used and proposals for new and changed requirements. Assumptions about the operating requirement made when the system was specified may be incorrect. Organizational changes may mean that the system is used in different ways from those originally planned. Operational Risk Assessment therefore leads to new security requirements that have to be implemented as the system evolves.
* To carry out a risk assessment, you need to identify the possible threats to a system. One way to do this is to develop a set of “Misuse Cases”.
  + They are similar to Use Cases which show typical interactions with a system…Use Cases are used to derive system requirements.
  + Misuse Cases are scenarios are scenarios that represent malicious interactions with a system. You can use these to discuss and identify possible threats and, therefore also determine the system’s security requirements. They can be use alongside Use Cases when deriving the system requirements
  + 4 Heading for Classifying Threats (for identifying possible misuse cases)
    - Interception Threats
      * Allow an attacker to gain access to an asset
    - Interruption Threats
      * Allow an attacker to make part of the system unavailable. Therefore, a possible misuse case might be a denial of service attack on a system database server.
    - Modification Threats
      * Allow an attacker to tamper with a system asset.
    - Fabrication Threats
      * Allow an attacker to insert false information into a system.
  + Misuse Cases are also used for security analysis in Life-Cycle Risk Analysis and Operational Risk Analysis.
* Life-Cycle Risk Assessment (Section 14.1.1)
  + Preliminary Risk Assessment should identify the most important security requirements for a system. These reflect how the security policy should be implemented in that application, identify the assets to be protected, and decide what approach should be used to provide that protection. However, it is impossible for the initial security requirements to take all details that affect security into account
  + Life-Cycle Risk Assessment identifies the design and implementation details that affect security.
    - This is the most important distinction between life-cycle and preliminary. Life-cycle assessment affects the interpretation of existing security requirements, generates new requirements, and influences the overall design of the system
  + At this stage, you should now much more about the system, its vulnerabilities, and what needs to be protected. Some of the vulnerabilities will be the result will be inherent in the design choices made.
  + Security Risk Assessment should be part of all SW processes from Requirements Engineering to System Deployment.
  + A difference between Preliminary and Life-Cycle is that in Life-Cycle you now have knowledge about information representation and distribution and the database organization for important assets that need protection.
  + Important phases of Life-Cycle Risk Assessment
    - Asset Representation and Organization
    - Asset Value Assessment (Input is Asset Representation and Organization)
    - Exposure Assessment (Input is Asset Value Assessment)
    - Design and Requirements Changes ( Input is Exposure Assessment and Attack Assessment)
    - Threat Identification (Input is Asset Representation and Organization, and Technology Choices)
    - Attack Assessment (Input is Technology Choices)
    - Control Identification (input is Available Controls and Technology Choices)
  + When you develop a system by reusing an existing system, you have to accept design decision made by the developers of that system.
  + Once vulnerabilities have been identified, you then have to make a decision on what steps that you can take to reduce the associated risks.
* Operational Risk Assessment (Section 14.1.2)
  + Security Risk Assessment should continue through the lifetime of the system to identify emerging risks and system changes that may be required to cope with these risks. This process is called Operational Risk Assessment.
  + The process is similar to that of Life-Cycle Risk Assessment, but it has further information about the environment in which the system is used. The environment is important because characteristics of the environment can lead to new risks to the system.

**Design for Security (Section 14.2)**

* You need to take security issues into account during the Design process because it is very difficult to add security to a system after it has been implemented.
* The use of redundancy and diversity may mean that a system can resist and recover from attacks that target specific design or implementation characteristics.
* Designing a system to be secure involves compromises. For example, measure taken to make the system more secure will use up more computation power.
* Architectural Design (Section 14.2.1)
  + The choice of software architecture can have profound effects on the emergent properties of a system. If an inappropriate architecture is used, it may be difficult maintain security, availability, etc.
  + In designing a system architecture that maintains security, you need to consider 2 fundamental issues:
    - Protection – How should the system be organized so that critical assets can be protected against external attack?
    - Distribution – How should system assets be distributed so that the effects of a successful attack are minimized?
    - Examples – If you place all assets in one locate with multiple layers of protection, then, if all layers fail, all of the assets are exposed. If you place assets in different location, this is expensive to secure. But, you won’t suffer a total loss of assets as in the first case.
  + Ex. Patient Record System
    - Use a Centralized Data Architecture with a Layered Architecture for security
      * The Layers are Platform-level Protection, Application-level Protection, and Record-level Protection.
        + Platform-level Protection – Top level controls access to platform. Involves a user signing on to a particular computer and maintaining the integrity of files on the system, backups, etc.
        + Application-level Protection – This level is built into the application itself. Involves user accessing the application, being authenticated and getting authorization to take actions such as viewing or modifying data.
        + Record-level Protection – This level is invoked when access to specific records is required, and involves checking that a user is authorized to carry out the requested operations on that record. This layer may also encrypt records.
  + If the protection of data is a critical requirement, then a client-server architecture should be used, with the protection mechanisms built into the server.
  + If you think of DoS attacks as a major risk, then you may use a Distributed Object Architecture where assets are distributed across a number of different platforms, with separate protection mechanisms used for each of these.
* Design Guidelines (Section 14.2.2)
  + Guidelines
    - Base Security decisions on an explicit security policy
      * Security policy = high-level statement that sets out fundamental security conditions for an organization.
      * The policy should NOT define the mechanisms to be used to provide and enforce security.
      * Usually all aspects of the policy should be reflected in the system requirements.
    - Avoid a single point of failure
      * You should not rely on a single mechanism to ensure security.
    - Fail securely
      * When the system fails, you should NOT use fallback procedures that are less secure than the system itself.
      * The system failure should not mean that an attacker can access data that would not normally be allowed.
    - Balance security and usability
      * Demands of security and usability are often contradictory so you need to find a balance
    - Log user actions
      * This log should at least, record who did what, the assets used, and the time and date of the action. This gives you the option of “replaying” the log to recover from failures.
      * These logs can also help understand how an attack happened.
      * Acts as a deterrent because people won’t like knowing their actions are logged
    - Use redundancy and diversity to reduce risk
      * Use redundancy and diversity so that a platform or technology vulnerability will not affect all versions and so lead to a common failure.
    - Validate all inputs
      * Prevent buffer overflows, SQL injections, and other attacks
    - Compartmentalize your assets
      * Compartmentalizing means that you should not provide all-or-nothing access to information in system
      * You should organize the information in a system into compartments.
      * Users should only have access to the information that they need, rather than to all of the information in a system.
    - Design for deployment
      * Always design your system so that facilities are included to simplify deployment in the customer’s environment and to check for potential configuration errors and omissions in the deployed system.
    - Design for recoverability
      * Always design your system with the assumption that a security failure could occur.
      * Think about how to recover form possible failures and restore the system to a secure operational state.
* Design for Deployment (Section 14.2.3)
  + Deployment of a system involves configuring the software to operate in an operational environment, installing the system on the computers in that environment, and then configuring the installed system for these computers.
  + Stages
    - Understand and Define the SW Operational Environment
    - Configure SW with Environment Details
    - Install SW on Computer where it will operate
    - Configure SW with computer details
  + 4 Ways to incorporate deployment support in a system
    - Include support for viewing and analyzing configurations
      * If an administrator can get a complete picture of a configuration, they are more likely to sport errors and omissions in configuration.
    - Minimize default privileges
      * Design software so that the default configuration of a system provides minimum essential privileges.
      * For example, the dealt system administrator authentication should only allow access to a program that enables an administrator to set up credentials.
    - Localize configuration settings
      * Should ensure that everything in a configuration that affects the same part of a system is set up in the same place.
    - Provide easy ways to fix security vulnerabilities
      * Include straightforward mechanisms for updating the system to repair security vulnerabilities that have been discovered.

System Survivability (Section 14.3)

* No matter how much attention is paid to security, it cannot be guaranteed that a system will be able to resist external attacks. You should assume penetration is possible and that the integrity of the system cannot be guaranteed. You should therefore think about how to make the system resilient so that it survives to deliver essential services to users.
* Survivability/Resilience = property of a system as a whole, rather than a property of individual components, which may not themselves be survivable.
* Survivability reflects a system’s ability to continue to deliver services to legitimate users while it is under attack of after part of the system has been damaged.
* Maintaining the availability of critical services is the essence of survivability. You must know:
  + The system services that are the most critical for a business
  + The minimal quality of service that must be maintained
  + How these services might be compromised
  + How these services can be protected
  + How you can recover quickly if the services become unavailable
* How to achieve Survivability:
  + Resistance
    - Building capabilities into the system to repel attacks
  + Recognition
    - Detecting problems by building capabilities into the system to detect attacks and failures and assess the resultant damage.
  + Recovery
    - Building capabilities into the system to deliver essential services while under attack, and to recover full functionality after an attack.
* Survivable Systems Analysis (used to assess vulnerabilities in system to support the design of system architectures and features that promote system survivability)
  + 4 Stage Process
    - System Understanding
      * Review the foals of the system, the system requirements, and the system architecture
    - Critical Service Identification
      * Services that must always be maintained and the components that are required to maintain these services are identified
    - Attack Simulation
      * Scenarios or Use Cases for possible attacks are identified along with the system components that would be affected by these attacks
    - Survivability Analysis
      * Components that are both essential and compromisable by an attack are identified and survivability strategies based on resistance, recognition, and recovery are identified.