Data vs Information

- Computer security is about controlling access to information and resources
- Controlling access to information can sometimes be quite elusive and it is often replaced by the more straightforward goal of controlling access to data
- The distinction between data and information is subtle but it is also the root of some of the more difficult problems in computer security
- *Data* represents information. *Information* is the (subjective) interpretation of data

Data vs Information

- Protecting information means to protect not only the data directly representing the information
- Information must be protected also against transmissions through:
  - Covert channels: communication channels that can be exploited by a process to transfer information in a manner that violates the systems security policy
  - Inference
    - It is typical of database systems
    - It refers to the derivation of sensitive information from non-sensitive data

Inference - Example

Name	Sex	Programme	Units	Grade Ave
Alma	F	MBA	8	63
Bill	M	CS	15	58
Carol	F	CS	16	70
Don	M	MIS	22	75
Errol	M	CS	8	66
Flora	F	MIS	16	81
Gala	F	MBA	23	68
Homer	M	CS	7	50
Igor	M	MIS	21	70

Inference - Example

- Assume that there is a policy stating that the average grade of a single student cannot be disclosed; however statistical summaries can be disclosed
- Suppose that an attacker knows that Carol is a female CS student
- By combining the results of the following legitimate queries:
  - Q1: SELECT Count (*) FROM Students WHERE Sex ='F' AND Programme = 'CS'
  - Q2: SELECT Avg (Grade Ave) FROM Students WHERE Sex ='F' AND Programme = 'CS'

  The attacker learns from Q1 that there is only one female student so the value 70 returned by Q2 is precisely her average grade

Security Policies and Procedures

- A security policy is a set of security objectives for a company that includes rules of behavior for users and administrators and specifies system requirements. These objectives, rules, and requirements collectively ensure the security of a network, the data, and the computer systems within an organization.
- Standards help an IT staff maintain consistency in operating the network. Standards provide the technologies that specific users or programs need in addition to any program requirements or criteria that an organization must follow.
- Guidelines are a list of suggestions on how to do things more efficiently and securely. They are similar to standards, but are more flexible and are not usually mandatory. Guidelines define how standards are developed and guarantee adherence to general security policies.
- Procedure documents are longer and more detailed than standards and guidelines. Procedure documents include implementation details that usually contain step-by-step instructions and graphics.

Example Policy:

Employees must use strong passwords on their accounts. Passwords must be changed regularly and protected against disclosure.

Example Standards:

- Passwords must be at least 10 characters long and incorporate at least one lowercase letter, one uppercase letter, one numerical digit (0–9), and one special character permitted by our system (&%$#@!). Passwords must be changed every 90 days, and must not be written down or stored on insecure media.

Example Practice:

- The practice identifies other reputable organizations and agencies that offer recommendations the organization may have adopted or adapted.
- Use a minimum password length of 15 characters for administrator accounts.
  - Require the use of alphanumeric passwords and symbols.
  - Enable password history limits to prevent the reuse of previous passwords.
  - Prevent the use of personal information as passwords, such as phone numbers and dates of birth.

Example Guidelines:

- Guidelines provide examples and recommendations to assist users in complying with the new policy.
- In order to create strong yet easy-to-remember passwords, consider the following recommendations from NIST SP 800-118: Guide to Enterprise Password Management (Draft), April 2009:

Policy and Mechanism

- A security policy is a statement of what is, and what is not, allowed.
  - Policies may be presented mathematically, as a list of allowed (secure) and disallowed (nonsecure) states
  - Described in English what users are allowed to do
    - Ambiguity leads to state not classified allowed or disallowed
  - In practice, policies are rarely so precise;

- A security mechanism is a method, tool, or procedure for enforcing a security policy.
  - often require some procedural mechanisms that technology cannot enforce
  - Mechanisms can be nontechnical, such as requiring proof of identity before changing a password

Policy and Mechanism - Example

Suppose a university’s computer science laboratory has a policy that prohibits any student from copying another student’s homework files.

- The computer system provides mechanisms for preventing others from reading a user’s files. Anna fails to use these mechanisms to protect her homework files, and Bill copies them.
  - A breach of security has occurred, because Bill has violated the security policy.
  - Anna’s failure to protect her files does not authorize Bill to copy them.

Information Security – Mechanisms

- Confidentiality is enforced by the access control mechanism
- Integrity is enforced by the access control mechanism and by the semantic integrity constraints
- Availability is enforced by the recovery mechanism and by detection techniques for DoS attacks – an example of which is query flood

Information Security – How? Additional mechanisms

- *User authentication* - to verify the identity of subjects wishing to access the information
- *Information authentication* - to ensure information authenticity - it is supported by signature mechanisms
- *Encryption* - to protect information when being transmitted across systems and when being stored on secondary storage
- *Intrusion detection* – to protect against impersonation of legitimate users and also against insider threats

Goals of Security Mechanisms

- Prevention
  - Prevents attackers from violating security policy
  - Preventative mechanisms often are very cumbersome and interfere with system use to the point that they hinder normal use of the system
  - Involves implementation of mechanisms that users cannot override and that are trusted to be implemented in a correct, unalterable way
    - The resource protected by the mechanism need not be monitored for security problems, at least in theory.
  - Passwords
  - For example, if one attempts to break into a host over the Internet and that host is not connected to the Internet, the attack has been prevented.

Goals of Security Mechanisms

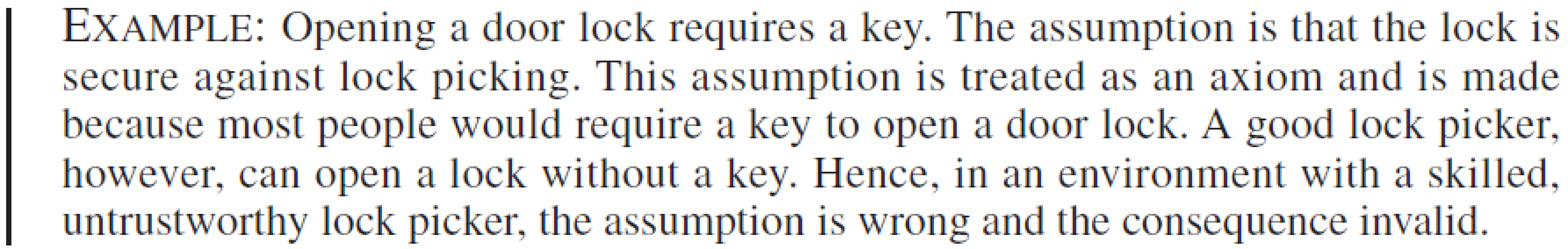
- Detection
  - Useful when an attack cannot be prevented
  - Indicate the effectiveness of preventative measures
  - Detect attackers' violation of security policy
  - Typical detection mechanisms monitor various aspects of the system, looking for actions or information indicating an attack
  - The attack may be monitored, however, to provide data about its nature, severity, and result
  - The resource protected by the detection mechanism is continuously or periodically monitored for security problems.

Goals of Security Mechanisms

- Recovery has two forms
  - Stop attack, assess and repair damage
    - Example: if the attacker deletes a file, one recovery mechanism would be to restore the file from backup tapes
    - In practice, recovery is far more complex, because the nature of each attack is unique
      - the type and extent of any damage can be difficult to characterize completely
    - Recovery involves identification and fixing of the vulnerabilities used by the attacker to enter the system because attackers may return
    - The system’s functioning is inhibited by the attack, recovery requires resumption of correct operation
  - Continue to function correctly even if attack succeeds
    - This type of recovery is quite difficult to implement
    - It draws on techniques of fault tolerance as well as techniques of security and is typically used in safety-critical systems
    - The system does not function incorrectly. However, the system may disable nonessential functionality

Assumptions and Trust

- How do we determine if the policy correctly describes the required level and type of security for the site?



EXAMPLE: Opening a door lock requires a key. The assumption is that the lock is secure against lock picking. This assumption is treated as an axiom and is made because most people would require a key to open a door lock. A good lock picker, however, can open a lock without a key. Hence, in an environment with a skilled, untrustworthy lock picker, the assumption is wrong and the consequence invalid.

- A policy consists of a set of axioms that the policy makers believe can be enforced.
  - Designers of policies always make two assumptions.
    - the policy correctly and unambiguously partitions the set of system states into “secure” and “nonsecure” states.
    - the security mechanisms prevent the system from entering a “nonsecure” state.
  - If either assumption is erroneous, the system will be nonsecure.

Assumptions and Trust

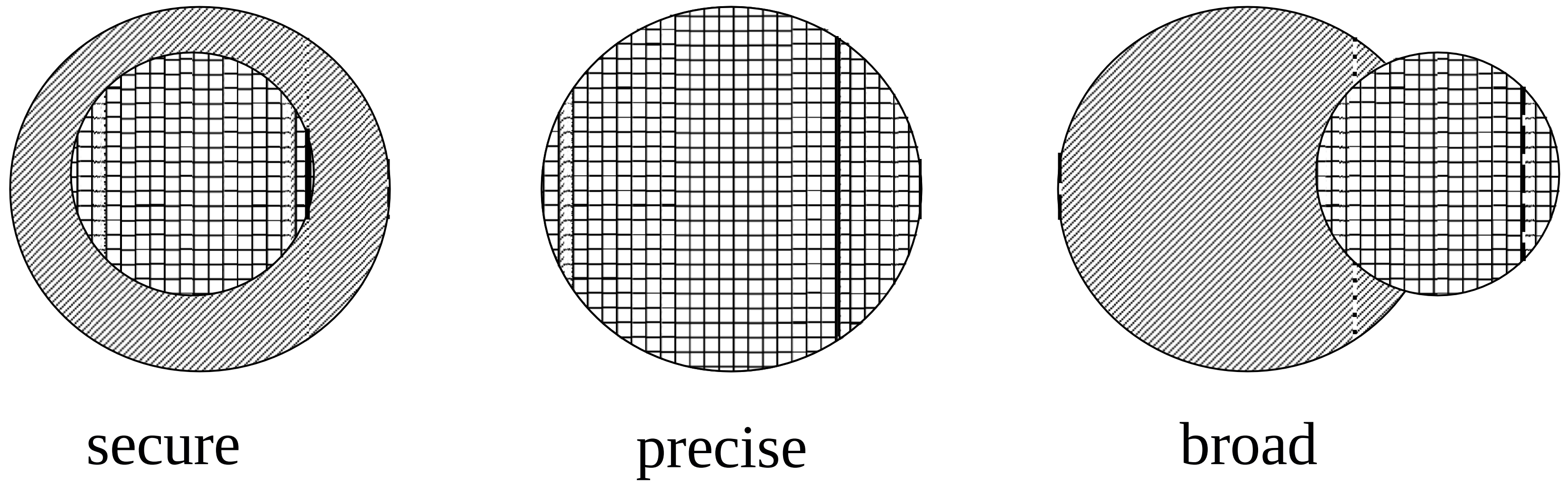
Let P be the set of all possible states. Let Q be the set of secure states (as specified by the security policy). Let the security mechanisms restrict the system to some set of states R (thus, $R \subseteq P$). Then we have the following definition.

A security mechanism is secure if $R \subseteq Q$; it is precise if $R = Q$; and it is broad if there are states r such that $r \in R$ and $r \notin Q$.

Trusting that mechanisms work requires several assumptions.

- Each mechanism is designed to implement one or more parts of the security policy
- The union of the mechanisms implements all aspects of the security policy.
- The mechanisms are implemented correctly
- The mechanisms are installed and administered correctly

Types of Mechanisms



▦ set of reachable states

▨ set of secure states

Information Security: A Complete Solution

- It consists of:
  - first defining a *security policy*
  - then choosing some *mechanism* to enforce the policy
  - finally providing *assurance* that both the mechanism and the policy are sound

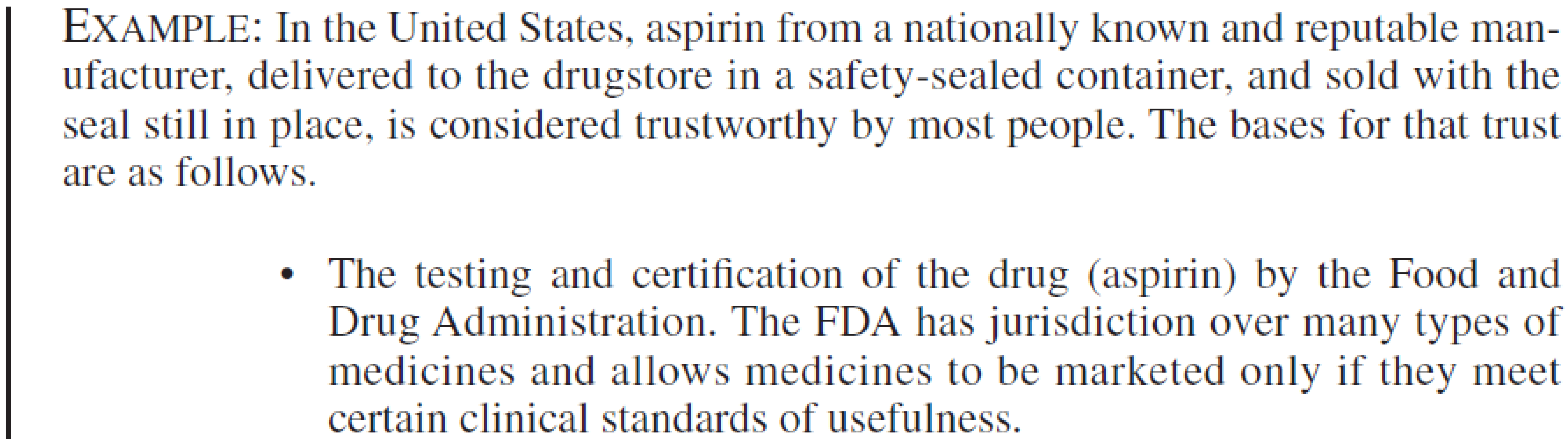
$$\Downarrow$$

SECURITY LIFE-CYCLE

Assurance

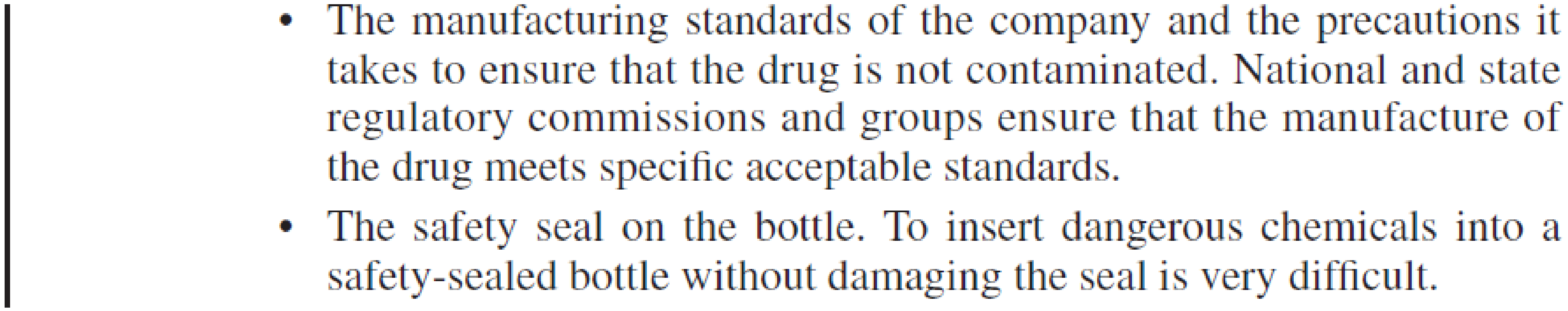
- Specification
  - Requirements analysis
  - Statement of desired functionality
- Design
  - How system will meet specification
- Implementation
  - Programs/systems that carry out design

Assurance



EXAMPLE: In the United States, aspirin from a nationally known and reputable manufacturer, delivered to the drugstore in a safety-sealed container, and sold with the seal still in place, is considered trustworthy by most people. The bases for that trust are as follows.

- The testing and certification of the drug (aspirin) by the Food and Drug Administration. The FDA has jurisdiction over many types of medicines and allows medicines to be marketed only if they meet certain clinical standards of usefulness.



- The manufacturing standards of the company and the precautions it takes to ensure that the drug is not contaminated. National and state regulatory commissions and groups ensure that the manufacture of the drug meets specific acceptable standards.
- The safety seal on the bottle. To insert dangerous chemicals into a safety-sealed bottle without damaging the seal is very difficult.

A system is said to satisfy a specification if the specification correctly states how the system will function.

Assurance

Assurance in the computer world requires specific steps to ensure that the computer will function properly.
The sequence of steps includes:

- Detailed specifications of the desired (or undesirable) behavior
- An analysis of the design of the hardware, software, and other components to show that the system will not violate the specifications
- Arguments or proofs that the implementation, operating procedures, and maintenance procedures will produce the desired behavior

Specification

A statement of the desired functioning of the system

- Formal:
    - e.g., mathematical
- Informal
    - e.g., English to describe what the system should do under certain conditions

Specification can be low-level, combining program code with logical and temporal relationships to specify ordering of events.

The defining quality is a statement of what the system is allowed to do or what it is not allowed to do.

Design

- Translates the specifications into components that will implement them
- Design satisfy the specifications if under all relevant circumstances, the design will not permit the system to violate those specifications

Design depends on assumptions about what the specifications mean

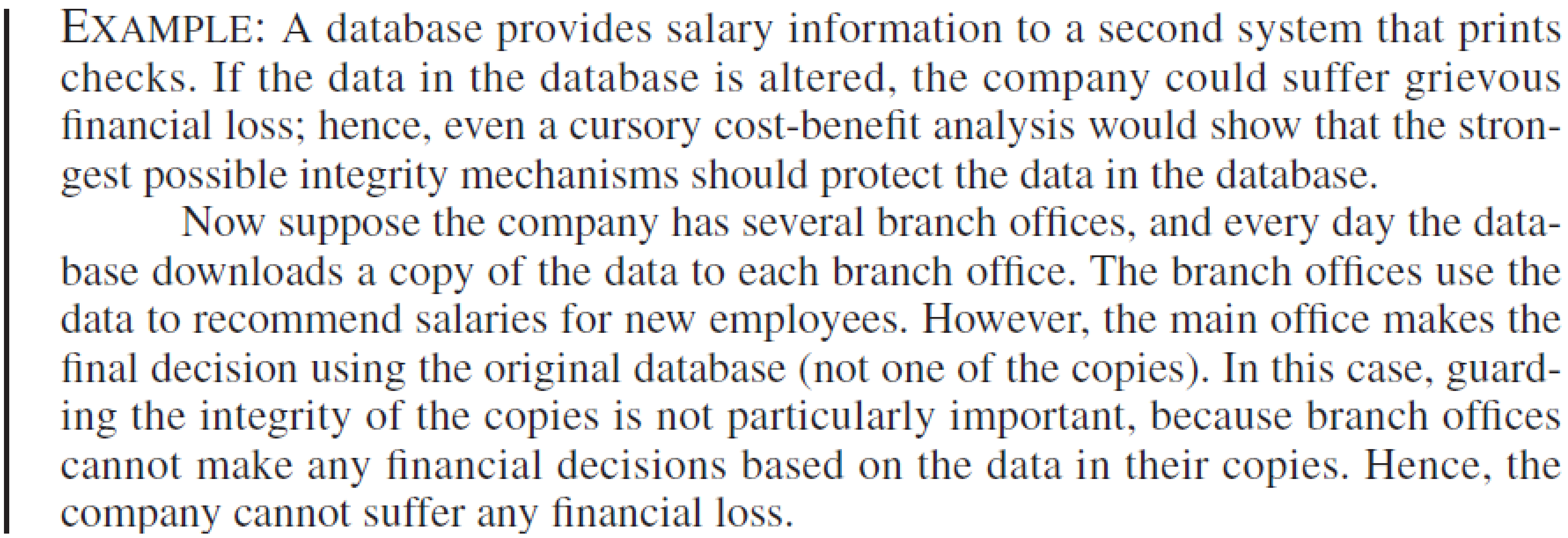
Implementation

- The implementation creates a system that satisfies that design. If the design also satisfies the specifications, then by transitivity the implementation will also satisfy the specifications.
- How to prove this?

- Programs complexity makes their mathematical verification difficult
- Verification assumes no hardware failure, buggy code, and failures in other tools
- Verification relies on conditions on the input
    - Any input that do not meet the conditions should be rejected by the program
- Proof of correctness is time-consuming
    - Testing
        - Simplicity vs. Assurance
        - Error in test procedure or documentation invalidate the testing results

Operational Issues

- Any useful policy and mechanism must balance the benefits of the protection against the cost of designing, implementing, and using the mechanism.
- Cost-Benefit Analysis
  - Is it more cost-effective to prevent or recover?



EXAMPLE: A database provides salary information to a second system that prints checks. If the data in the database is altered, the company could suffer grievous financial loss; hence, even a cursory cost-benefit analysis would show that the strongest possible integrity mechanisms should protect the data in the database.

Now suppose the company has several branch offices, and every day the database downloads a copy of the data to each branch office. The branch offices use the data to recommend salaries for new employees. However, the main office makes the final decision using the original database (not one of the copies). In this case, guarding the integrity of the copies is not particularly important, because branch offices cannot make any financial decisions based on the data in their copies. Hence, the company cannot suffer any financial loss.

Operational Issues

To determine whether an asset should be protected, and to what level, requires analysis of the potential threats against that asset and the likelihood that they will materialize.

Risk Analysis

- Should we protect some information?
- How much should we protect this information?

- Risk is a function of environment
- Risks change with time
- Many risks are quite remote but still exist
- Making risk analyses with no effort to act on those analyses

Operational Issues

- Laws and Customs
  - Are desired security measures illegal?
  - Will people adopt them?

Any policy and any selection of mechanisms must take into account legal considerations.

Complicating situations

- Involving the laws of multiple jurisdictions especially foreign ones.

Legal and acceptable practices

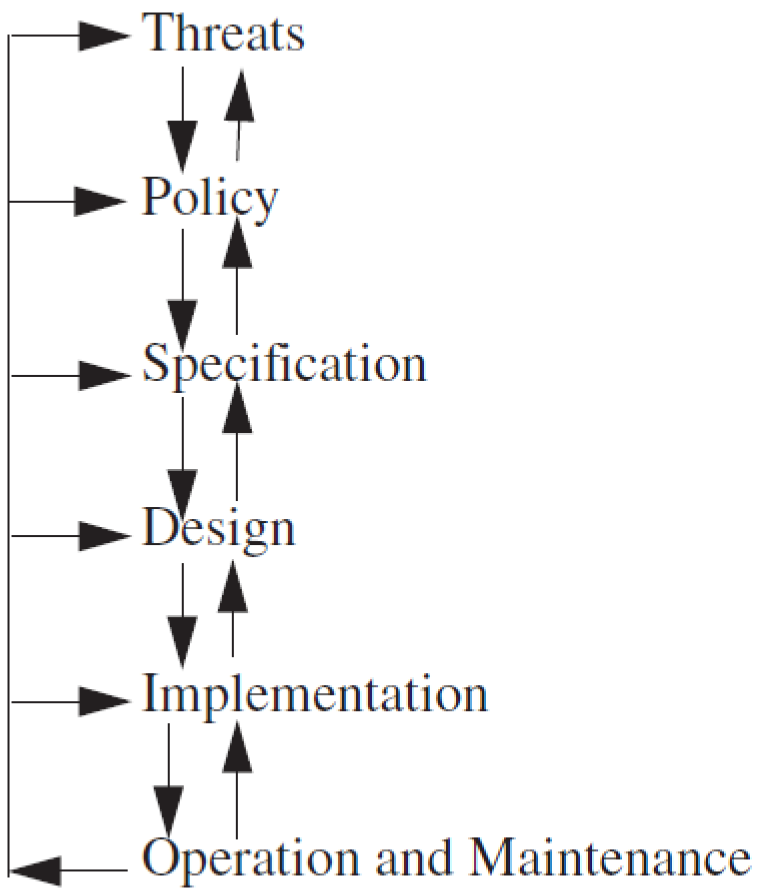
- Mechanisms with legal risks will be unused
  - Unused mechanisms is worse that nonexistent one
    - because it gives a false impression that a security service is available. Hence, users may rely on that service to protect their data, when in reality their data is unprotected.

Human Factor Issues

- Organizational Problems
  - Power and responsibility
  - Financial benefits

Human Factor Issues

- People problems
  - Outsiders and insiders
  - Social engineering


Threats
Policy
Specification
Design
Implementation
Operation and Maintenance