Policy, Models, and Trust
Security Policy

• A **security policy** is a well-defined set of rules that include the following:
  • **Subjects**: the agents who interact with the system,
  • **Objects**: the informational and computational resources that a security policy is designed to protect and manage.
    • Examples include critical documents, files, and databases, and computational resources include servers, workstations, and software.
  • **Actions**: the things that subjects may or may not do with respect to the objects.
    • Examples include the reading and writing of documents, updating software on a web server, and accessing the contents of a database.
  • **Permissions**: mappings between subjects, actions, and objects, which clearly state what kinds of actions are allowed or disallowed.
  • **Protections**: the specific security features or rules that are included in the policy to help achieve particular security goals, such as confidentiality, integrity, availability, or anonymity.
Security Models

• A **security model** is an abstraction that provides a conceptual language for administrators to specify security policies.

• **Security models define hierarchies of access or modification rights** that members of an organization can have, so that subjects in an organization can easily be granted specific rights based on the position of these rights in the hierarchy.

• Examples include military classifications of access rights for documents based on concepts like
  • Unclassified
  • Confidential
  • Secret
  • Top secret
Discretionary Access Control

• **Discretionary access control**, or **DAC**, refers to a scheme where users are given the ability to determine the permissions governing access to their own files.
  
  - **DAC** typically features the concept of both users and groups, and allows users to set access-control measures in terms of these categories.
  - In addition, DAC schemes allow users to grant privileges on resources to other users on the same system.
Mandatory Access Control

• **Mandatory access control** is a more restrictive scheme that does not allow users to define permissions on files, regardless of ownership. *Instead, security decisions are made by a central policy administrator.*

  • Each security rule consists of a **subject**, which represents the party attempting to gain access, an **object**, referring to the resource being accessed, and a series of permissions that define the extent to which that resource can be accessed.

• **Security-Enhanced Linux (SELinux)** incorporates mandatory access control.
Trust Management

• A trust management system is a formal framework for specifying security policy in a precise language, which is usually a type of logic or programming language, together with a mechanism for ensuring that the specified policy is enforced.

• A trust management system consists of two main components:
  • Policy language
  • Compliance checker

• Policy rules are specified in the policy language and are enforced by the compliance checker.
Trust Management Systems

• A trust management system typically has rules describing:

• **Actions**: operations with security-related consequences on the system

• **Principals**: users, processes, or other entities that can perform actions on the system

• **Policies**: precisely written rules that govern which principals are authorized to perform which actions

• **Credentials**: digitally signed documents that bind principal identities to allowable actions, including the authority to allow principals to delegate authority to other principals.
Policy Languages-KeyNote

- KeyNote define an application or a program that uses KeyNote.
- A policy compliance value (PCV) is the answer issued by KeyNote in response to a request by a principle to perform some action.
  - Application queries the KeyNote system when a principle requests action.
  - Action is described by a set of attribute-value pairs known as **action-attribute set**.
- KeyNote replies with a PCV indicating whether the action is allowed or not.
Policy Languages - XACML

- The Extensible Access control Markup language (XACML) was released in 2009
- XACML leverages XML to define security policies and described how these policies should be enforced
- XACML components:
  - Policy Administration Point (PAP)
  - Policy Decision Point (PDP)
  - Policy Enforcement Point (PEP)
  - Policy Information Point (PIP)
Access Control Models

• Various models have been developed to formalize mechanisms to protect the **confidentiality** and **integrity** of documents stored in a computer system.
  • The Bell-La Padula (BLP) model
  • The Biba model
  • The Low-Watermark model
  • The Clark-Wilson model
  • The Chinese Wall model (The Brewer and Nash model)
The Bell-La Padula Model

• The Bell-La Padula (BLP) model is a classic mandatory access-control model for protecting confidentiality.

• The BLP model is derived from the military multilevel security paradigm, which has been traditionally used in military organizations for document classification and personnel clearance.
The Bell-La Padula Model

• The BLP model has a *strict, linear ordering on the security of levels* of documents, so that each document has a specific security level in this ordering and each user is assigned a strict level of access that allows them to view all documents with the corresponding level of security or below.
Total Orders and Partial Orders

• A linear ordering for documents can be defined in terms of a comparison rule, . We say that such a rule defines a total order on a universe set, U, if it satisfies the following properties:
  1. **Reflexivity:** If x is in U, then x ≤ x.
  2. **Antisymmetry:** If x ≤ y and y ≤ x, then x = y.
  3. **Transitivity:** If x ≤ y and y ≤ z, then x ≤ z.
  4. **Totality:** If x and y are in U, then x ≤ y or y ≤ x.

• All of the usual definitions of “less than or equal to” for numbers, such as integers and real numbers, are total orders.

• If we drop the requirement of totality, we get a **partial order**.
  • The classic example of a partial order is the set of courses taught at a college or university, where we say that, for two courses A and B, A ≤ B, if A is a prerequisite for B.
How the BLP Model Works

• The security levels in BLP form a partial order, $\leq$.
• Each object, $x$, is assigned to a security level, $L(x)$. Similarly, each user, $u$, is assigned to a security level, $L(u)$. Access to objects by users is controlled by the following two rules:
  • Simple security property. A user $u$ can read an object $x$ only if $L(x) \leq L(u)$.
  • *-property. A user $u$ can write (create, edit, or append to) an object $x$ only if $L(u) \leq L(x)$.
• The simple security property is also called the “no read up” rule, as it prevents users from viewing objects with security levels higher than their own.
• The *-property is also called the “no write down” rule. It is meant to prevent propagation of information to users with a lower security level.
Defining Security Levels Using Categories

- The BLP defines the partial order of security levels starting from a set **B** of **basic levels** that have a linear order and a collection of **S** of categories called (**compartments**)
  - Security level \( L(x) \) consists of a pair \((b(x), S(x))\), where \( b(x) \in B \) and \( S(x) \) is a subset of \( S \)
  - The level of \( x \) precedes the level of \( y \) if the basic level of \( x \) is less than basic level of \( y \) and the subset of categories of \( x \) is contained in the subset of categories of \( y \)
    \( (b(x), S(x)) \leq (b(y), S(y)) \iff b(x) \leq b(y) \) and \( S(x) \) is a subset of \( S(y) \)
The Biba Model

• The Biba model has a similar structure to the BLP model, but it addresses integrity rather than confidentiality.

• Objects and users are assigned integrity levels that form a partial order, similar to the BLP model.

• The integrity levels in the Biba model indicate degrees of trustworthiness, or accuracy, for objects and users, rather than levels for determining confidentiality.
  • For example, a file stored on a machine in a closely monitored data center would be assigned a higher integrity level than a file stored on a laptop.
  • In general, a data-center computer is less likely to be compromised than a random laptop computer. Likewise, when it comes to users, a senior employee with years of experience would have a higher integrity level than an intern.
The Biba Model Rules

• The access-control rules for Biba are the reverse of those for BLP. That is, Biba does not allow reading from lower levels and writing to upper levels.

• If we let \( I(u) \) denote the integrity level of a user \( u \) and \( I(x) \) denote the integrity level for an object, \( x \), we have the following rules in the Biba model:
  • A user \( u \) can read an object \( x \) only if \( I(u) \leq I(x) \).
  • A user \( u \) can write (create, edit or append to) an object \( x \) only if \( I(x) \leq I(u) \).

• Thus, the Biba rules express the principle that information can only flow down, going from higher integrity levels to lower integrity levels.
Role-Based Access Control

• The role-based access control (RBAC) model can be viewed as an evolution of the notion of group-based permissions in file systems.

• An RBAC system is defined with respect to an organization, such as company, a set of resources, such as documents, print services, and network services, and a set of users, such as employees, suppliers, and customers.
RBAC Components

• A **user** is an entity that wishes to access resources of the organization to perform a task. Usually, users are actual human users, but a user can also be a machine or application.

• A **role** is defined as a collection of users with similar functions and responsibilities in the organization. Examples of roles in a university may include “student,” “alum,” “faculty,” “dean,” “staff,” and “contractor.” In general, a user may have multiple roles.
  - Roles and their functions are often specified in the written documents of the organization.
  - The assignment of users to roles follows resolutions by the organization, such as employment actions (e.g., hiring and resignation) and academic actions (e.g., admission and graduation).

• A **permission** describes an allowed method of access to a resource.
  - More specifically, a permission consists of an operation performed on an object, such as “read a file” or “open a network connection.” Each role has an associated set of permissions.

• A **session** consists of the activation of a subset of the roles of a user for the purpose of performing a certain task.
  - For example, a laptop user may create a session with the administrator role to install a new program.
  - Sessions support the principle of least privilege.
Hierarchical RBAC

• In the role-based access control model, roles can be structured in a hierarchy similar to an organization chart.

• More formally, we define a partial order among roles by saying that a role R1 inherits role R2, which is denoted

\[ R1 > R2, \]

if R1 includes all permissions of R2 and R2 includes all users of R1.

• When \( R1 \geq R2 \), we also say that role R1 is senior to role R2 and that role R2 is junior to role R1.
  • For example, in a company, the role “manager” inherits the role “employee” and
  • Also, in a university, the roles “undergraduate student” and “graduate student” inherit the role “student.”
Visualizing Role Hierarchy

- Role hierarchies can be graphically represented with a diagram where each role is connected to its immediate predecessor and successors in the hierarchy.

\[ R_1 \geq R_2 \geq R_3 \]
Constrained RBAC

- RBAC allows us to define constraints that prevent users from having incompatible roles that create conflicts of interest
  - A Constraint with a pair of roles (R1, R2)
  - **Separation of duty** A more general constraint is defined by a pair (S, k), where S is a subset of roles and K≥ 2 is an integer
    - **Static:** The constraint holds for the assignment of users to roles. No user can assigned to k or more roles in S
    - **Dynamic:** The constraints holds for the activation of roles of users in sessions. That is no user can have k or more roles in S activated in a session.
  - Example:
    
    (\{supervisor, traveler\}, 2)