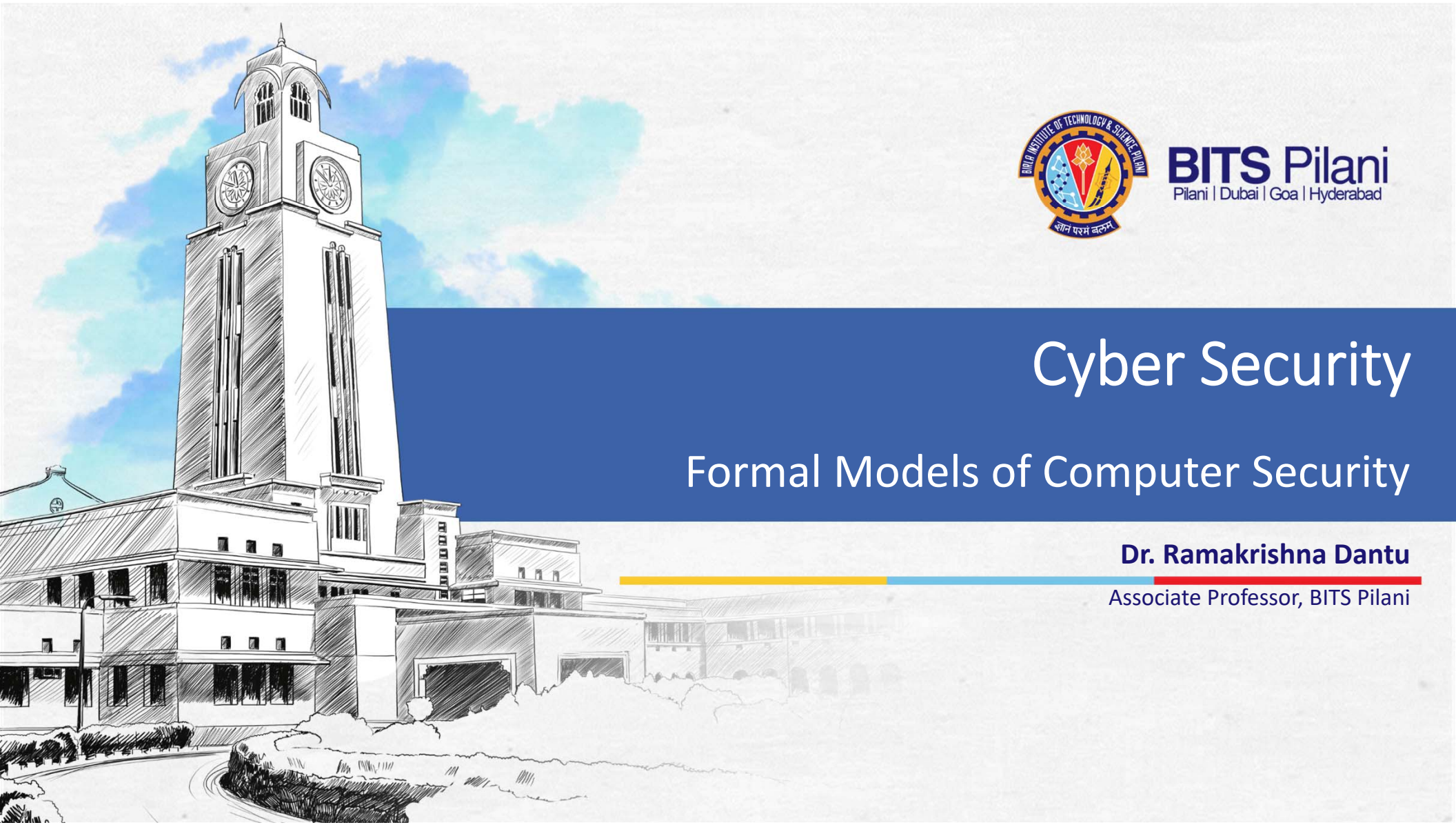


BITS Pilani
Pilani | Dubai | Goa | Hyderabad

# Cyber Security

## Formal Models of Computer Security

**Dr. Ramakrishna Dantu**
Associate Professor, BITS Pilani

## Disclaimer and Acknowledgement

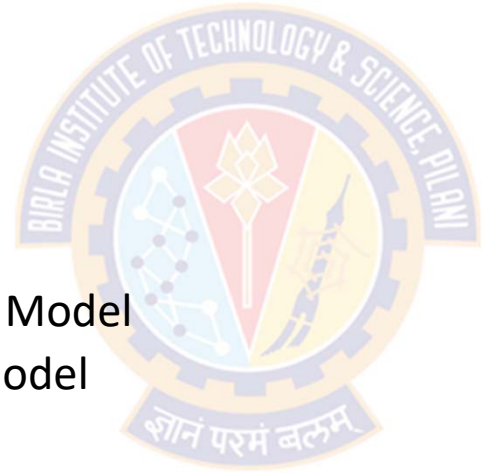

- The content for these slides has been obtained from books and various other source on the Internet
- I here by acknowledge all the contributors for their material and inputs.
- I have provided source information wherever necessary
- I have added and modified the content to suit the requirements of the course

# Formal Models of Computer Security



## Agenda

- The CIA Classification:
  - Confidentiality Policies:
    - Bell-LaPadula Model
  - Integrity Policies:
    - The Biba Model
    - Lipner's Integrity Matrix Model
    - Clark-Wilson Integrity Model
    - Trust Models
  - Availability Policies:
    - Deadlock
    - Denial of Service Models



# Lipner's Integrity Matrix



## Subjects/Objects and Clearance/Classifications - Revised

| Subjects | Clearance | Integrity Level | Objects | Classification | Integrity Level |
|---|---|---|---|---|---|
| Ordinary users | (SL, { SP }) | (ISL, { IP }) | Development code/test data | (SL, { SD }) | (ISL, { IP} ) |
| Application developers | (SL, { SD }) | (ISL, { ID }) | Production code | (SL, { SP }) | (IO, { IP }) |
| System programmers | (SL, { SSD }) | (ISL, { ID }) | Production data | (SL, { SP }) | (ISL, { IP }) |
| System managers and auditors | (AM, { SP, SD, SSD }) | (ISL, { IP, ID}) | Software tools | (SL, {∅} ) | (IO, { ID }) |
| System controllers | (SL, { SP, SD }) and downgrade privilege | (ISP, { IP, ID}) | System programs | (SL, {∅} ) | (ISP, { IP, ID }) |
| Repair | (SL, { SP }) | (ISL, { IP }) | System programs in modification | (SL, { SSD }) | (ISL, { ID }) |
| | | | System and application logs | (AM, { *appropriate* }) | (ISL, {∅} ) |
| | | | Repair | (SL, {SP}) | (ISL, { IP }) |

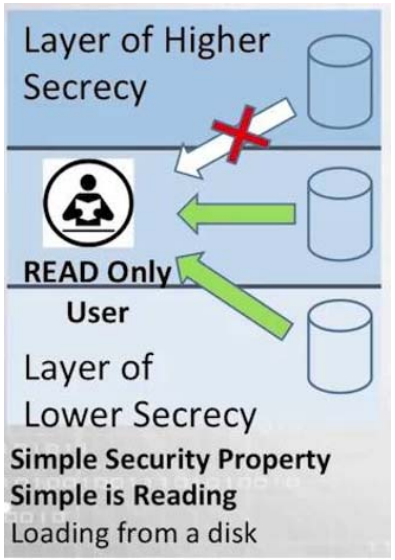
ISP > IO > ISL

# Lipner's Integrity Matrix



## Repair Class of Users

- Has the same integrity and security clearance as that of production data
  - so can read and write that data
- It can also
  - read production code
    - same security classification and (IO, {IP}) *dom* (ISL, {IP})
  - read system programs
    - (SL, {SP}) *dom* (SL, { ϕ }) and (ISP, { IP, ID }) *dom* (ISL, { IP })
  - repair objects
    - same security classes and same integrity classes



# Lipner's Integrity Matrix



## Repair Class of Users (Contd…)

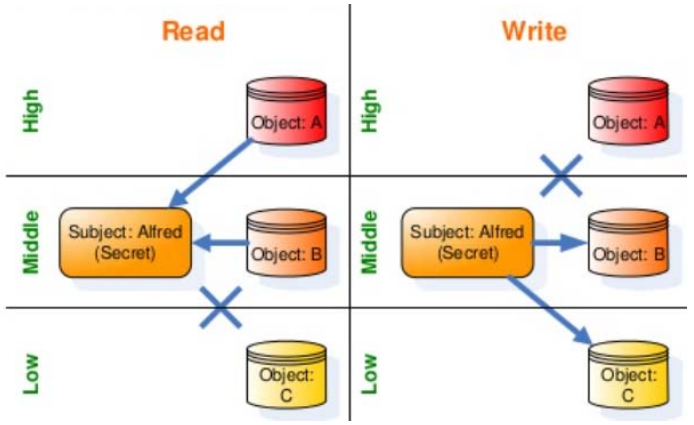
- It can write, but not read
  - the system and application logs, because
    - (AM, { SP }) *dom* (SL, { SP }) and (ISL, { IP }) *dom* (ISL, { ϕ })
- It cannot access
  - development code/test data
    - since the security categories are disjoint
  - system programs in modification
    - since the integrity categories are disjoint
  - software tools
    - since the integrity categories are disjoint
- Thus, the repair function works as needed

# Lipner's Integrity Matrix



## What can an ordinary user do?

- Ordinary users can : (SL, { SP }) (ISL, { IP })
  - Production data (same security integrity levels)
    - Read and write
  - Production code
    - Can Read – same classification
    - Cannot Write – (IO, IP) *dom* (ISL, {IP})
  - System program
    - (SL, {SP}) *dom* (SL, ∅) &
    - (ISP, {IP,ID}) *dom* {ISL, {IP})
  - Repair objects (same levels)
  - Write (not read) the system and application log
    - (AM, {SP}) *dom* (SL, {SP}) &
    - (ISL, {IP}) *dom* {ISL, ∅})




Biba model


# Clark-Wilson Integrity Model

# Clark-Wilson Integrity Model



## Overview

- Clark-Wilson (CW) model distills a security policy out of the centuries-old practice of double-entry bookkeeping
- Its main goal is to ensure the integrity of a bank's accounting system and to improve its robustness against insider fraud
- In double-entry bookkeeping, each transaction is posted to two separate books, as a credit in one and a debit in the other
- For example:
  - When a firm is paid $100 by a payer, the amount is entered as a debit in the accounts receivable (the firm is now owed $100 less) and as a credit in the cash account (the firm now has $100 more cash)
  - At the end of the day, the books should balance, that is, add up to zero.

# Clark-Wilson Integrity Model



## Overview

- The assets and the liabilities should be equal
- If the firm has made some profit, then this is a liability the firm has to the shareholders
- In larger firms the books will be kept by different clerks, and have to balance at the end of every month (at banks, every day)
- By suitable design of the ledger system, we can see to it that each shop, or branch, can be balanced separately
- Thus most frauds will need the collusion of two or more members of staff
- This principle of split responsibility is complemented by audit

# Clark-Wilson Integrity Model



## Overview

- Clark and Wilson proposed a more elaborate and practical integrity model in 1987
- The Clark-Wilson integrity model (CWM) is specifically designed for commercial operations
- The CWM defines each data item and allows modifications through only a small set of programs
- The CWM does not use a lattice structure used to define the levels of security that an object may have and that a subject may have access to
- Instead, it uses a three part relationship of {subject/program (or transaction)/object} known as a triple or an access control triple

# Clark-Wilson Integrity Model



## Overview

- Integrity defined by a set of constraints
  - Data is said to be in a *consistent* state (or consistent) if it satisfies given properties
- Example:
  - Consider a Bank scenario, and let
    - *D* = today's deposits
    - *W* = withdrawals
    - *YB* = yesterday's balance, and
    - *TB* = today's balance
  - Integrity constraint: $YB + D - W = TB$
- Before and after each action, the consistency conditions must hold.

# Clark-Wilson Integrity Model



## Four Basic Constraints

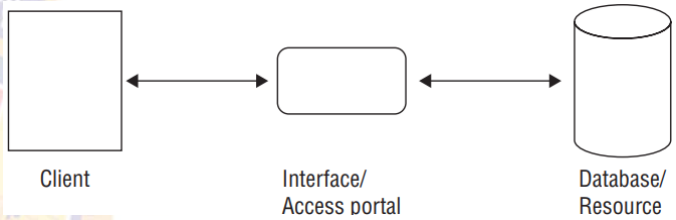
- Clark and Wilson claimed that the following are four fundamental constraints of any reasonable commercial integrity model:
- Authentication:
  - identity of all users must be properly authenticated.
- Audit:
  - modifications should be logged to record every program executed and by whom, in a way that cannot be subverted.
- Well-formed transactions:
  - Users manipulate data only in constrained ways. Only legitimate accesses are allowed.
  - Are operations that transition the system from one consistent state to another consistent state
- Separation of duty:
  - Who examines and certifies that the transactions are performed correctly?
  - The system associates with each user a valid set of programs they can run and prevents unauthorized modifications, thus preserving integrity and consistency with the real world.

# Clark-Wilson Integrity Model



## Two Principles

- In CWM, subjects do not have direct access to objects
  - Objects can be accessed only through programs




- CWM provides an effective means of protecting integrity through the use of two principles:
  - Well-formed transactions:
    - A user should not manipulate data arbitrarily, but only in constrained ways that preserve or ensure the integrity of the data.
  - Separation of duty among users:
    - Any person permitted to create or certify a well-formed transaction may not be permitted to execute it (at least against production data)

# Clark-Wilson Integrity Model

innovate achieve lead

## Entities

- The CWM defines data as:
  - Constrained Data Items (CDIs)
    - Is any data item whose integrity is protected by the security model
  - Unconstrained Data Items (UDIs)
    - Any data item whose integrity is not protected by the security model
    - Any data that is to be input and hasn't been validated, or any output. E.g., a simple text file
- The CWM also defines two sets of procedures:
  - Integrity verification procedures (IVPs)
    - Procedures that ensure CDIs conform to the integrity constraints at the time the IVPs are run
  - Transformation procedures (TPs)
    - Are the only procedures that are allowed to modify a CDI
    - Procedures that change the state of the data in the system from one valid state to another
    - TPs implement well-formed transactions

# Clark-Wilson Integrity Model

innovate achieve lead

## Certification and Enforcement Rules

- The CWM enforces integrity by means of certification rules and enforcement rules on TPs
- Certification rules
  - are security policy restrictions on the behavior of Integrity verification procedure (IVPs) and Transformation procedures (TPs)
- Enforcement rules
  - are built-in system security mechanisms that achieve the objectives of the certification rules

# Clark-Wilson Integrity Model



## Certification and Enforcement Rules

- CR1: All IVPs must ensure that CDIs are in a valid state when the IVP is run
- CR2: All TPs must be certified as integrity-preserving
- CR3: Assignment of TPs to users must satisfy separation of duty
- CR4: The operation of TPs must be logged
- CR5: TPs executing on UDIs must result in valid CDIs
- ER1: Only certified TPs can manipulate CDIs
- ER2: Users must only access CDIs by means of TPs for which they are authorized
- ER3: The identity of each user attempting to execute a TP must be authenticated
- ER4: Only the certifier of a TP may change the list of entities associated with that TP. Neither the certifier of a TP, nor an entity associated with that TP, may ever have execute permission with respect to that entity

Transformation Procedures (TPs)
Integrity verification procedures (IVPs)
Constrained Data Items (CDIs)

# Clark-Wilson Integrity Model

innovate achieve lead

## Certification and Enforcement Rules

- Certification Rules 1 & 2
  - CR1:
    - Rule: All IVPs must properly ensure that all CDIs are in a valid state at the time the IVP is run.
    - That is, when any IVP is run, it must ensure all CDIs are in a valid state
  - CR2:
    - Rule: All TPs must be certified to be valid and integrity-preserving
    - That is, they must take a CDI to a valid final state, given that it is in a valid state to begin with

Transformation Procedures (TPs)
Integrity verification procedures (IVPs)
Constrained Data Items (CDIs) = Data subject to integrity controls

# Clark-Wilson Integrity Model



## Certification and Enforcement Rules

- Enforcement Rules 1 & 2
  - ER1
    - Rule: Only certified TPs can manipulate CDIs
    - The system must maintain the list of certified TPs specified in CR2 and must ensure that only TPs certified to run on a CDI manipulate that CDI
  - ER2
    - Rule: Users must only access CDIs by means of TPs for which they are authorized
    - The system must associate a user with each TP and set of CDIs
    - The system must maintain a list of relations of the form (UserID, TPi, (CDIa, CDIb, CDIc, . . . )), which relates a user, a TP, and the data objects (CDIs)
    - The TP may access only those CDIs listed in the relation on behalf of the associated user
    - The TP cannot access a CDI on behalf of a user not associated with that TP and CDI

# Clark-Wilson Integrity Model



## Certification and Enforcement Rules

- Users and Rules
  - CR3
    - Rule: Assignment of TPs to users must satisfy separation of duty
    - The list of relations in ER2 must be certified to meet the separation of duty requirement.
  - ER3
    - Rule: The identity of each user attempting to execute a TP must be authenticated
      - Authentication not required before use of the system, but is required before manipulation of CDIs (requires using TPs)
- Logging
  - CR4
    - Rule: The operation of TPs must be logged
    - All TPs must be certified to write to an append-only CDI (the log)
    - All TPs must append enough information necessary to reconstruct the operation
      - Auditor needs to be able to determine what happened during reviews of transactions

CDI: Constrained Data Item
TP: Transaction Procedure

# Clark-Wilson Integrity Model



## Certification and Enforcement Rules

- Handling Untrusted Input
  - CR5
    - Rule: TPs executing on UDIs must result in valid CDIs
    - Any TP that takes a UDI as an input value must be certified to perform only valid transformations, or else no transformations, for any possible value of the UDI
    - Typically, this is an edit program.
    - The transformation should take the input from a UDI to a CDI, or the UDI is rejected.
      - In bank, numbers entered at keyboard are UDIs, so cannot be input to TPs
      - TPs must validate numbers (to make them a CDI) before using them; if validation fails, TP rejects UDI

UDI: Unconstrained Data Item
CDI: Constrained Data Item
TP: Transaction Procedure

# Clark-Wilson Integrity Model



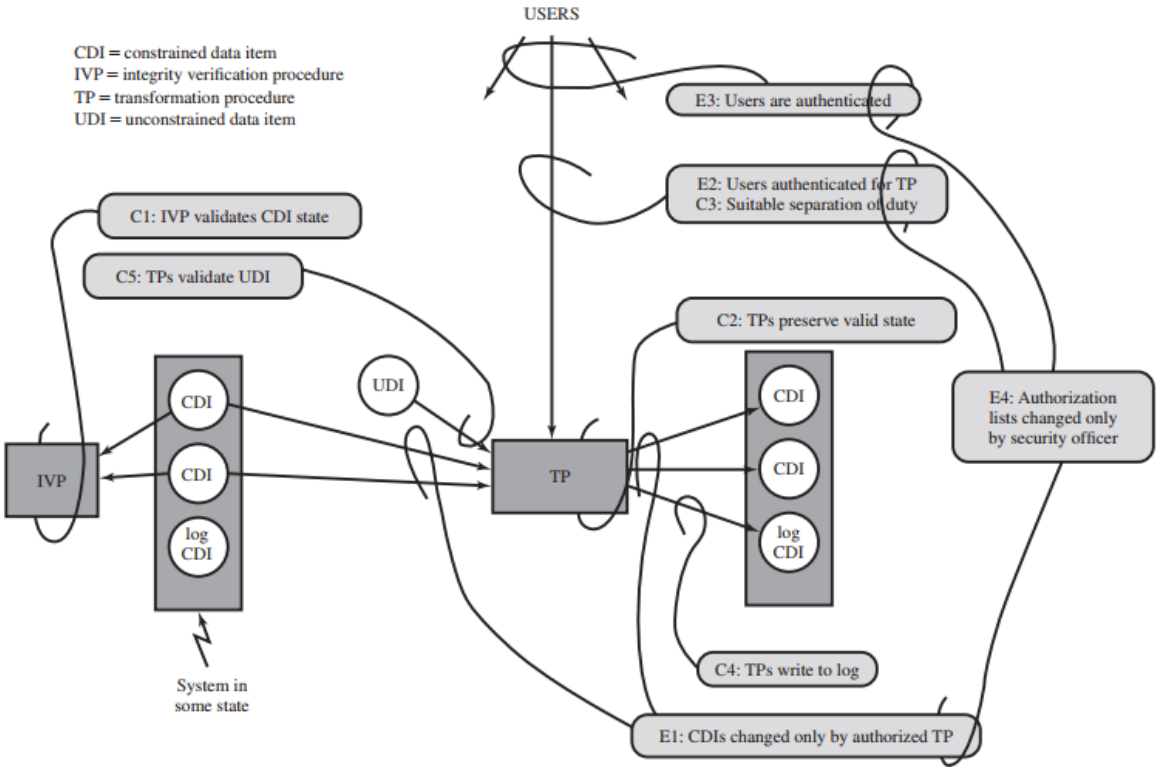
## Certification and Enforcement Rules

- Separation of Duty Model
  - ER4
    - Rule: Only the certifier of a TP may change the list of entities associated with that TP. Neither the certifier of a TP, nor an entity associated with that TP, may ever have execute permission with respect to that entity
    - Only the agent permitted to certify entities may change the list of such entities associated with other entities:
      - Specifically, the list of TPs associated with a CDI and the list of users associated with a TP
    - An agent that can certify a TP or an entity associated with that TP may not have any execute rights with respect to that entity.
      - Enforces separation of duty with respect to certified and allowed relations

# Clark-Wilson Integrity Model

innovate achieve lead

## Certification and Enforcement Rules



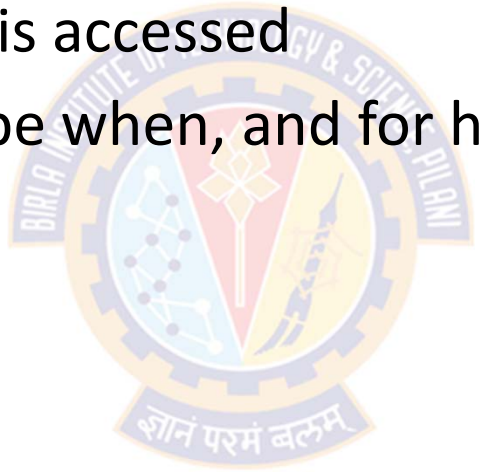

# Availability Policies

# Availability Policies

innovate achieve lead

## Overview

- Confidentiality and integrity policies describe what can be done once a resource or information is accessed
- Availability policies describe when, and for how long, the resource can be accessed




# Goals of Availability Policies

# Goals of Availability Policies



## Goals

- An availability policy ensures that a resource can be accessed in some way in a timely fashion
  - Availability is often expressed in terms of "quality of service"
- An availability policy defines the "type of access" and what a "timely fashion" means
  - "Timely fashion" depends on the nature of resource, the goals of subject using it

# Goals of Availability Policies



## Examples

- Example_1:
  - A commercial website selling merchandise will need to display details of items for customer requests in a matter of seconds or, at worst, a minute
  - The goal of the customer is to see what the website is selling, and the goal of the site is to make information available to the customer
  - However, the site does not want customers to alter prices displayed on the website, so there is no availability for altering information
- Example_2:
  - Consider a website that enables students to upload homework assignments
  - The website must allow some alterations
    - students must be able to upload their homework, possibly multiple times per assignment
  - There should be no access for the students to read other students' assignments

# Goals of Availability Policies

innovate achieve lead

## Safety and Liveness

- When a resource or service is not available, a denial of service occurs
- This is related to two types of problems: safety and liveness
- Safety problem
  - A denial of service resulting from the service giving incorrect responses
  - That is, the service is not performing the functions that the client is expecting
- Liveness problem
  - A denial of service that prevents users from accessing the service itself
- But other problems can cause a denial of service, such as assignment of inadequate resources to a process

# Goals of Availability Policies



## Mechanisms to support availability

- Two types of mechanisms
  - Mechanisms used to support availability in general
  - Mechanisms used to support availability as a security requirement
- The difference between the two lies in the assumptions underlying the failures
  - That is, under what circumstances failures can occur

# Goals of Availability Policies



## Mechanisms to support availability

- Mechanisms to support availability in general
  - The failures occur naturally over time due to usage
  - Lack of accessibility can be modeled using an average case, following a statistical model
  - For example:
    - The failure rates of disk drives depends upon many factors such as the age, the manufacturer, and environment and can be statistically modeled, although the precise model to be used is unclear
- Mechanisms used to support availability as a security requirement
  - Lack of availability assumes worst-case
  - Here, an adversary deliberately tries to make the resource or information unavailable
  - Because attackers induce this condition, models used in computer security describe failures that are nonrandom, and indeed may well be non-statistical


# Deadlock

# Deadlock



## Overview

- A *deadlock* is a state in which some set of processes block each waiting for another process in the set to take some action.
- Deadlock can occur if four conditions hold simultaneously:
  - Mutual exclusion
  - Hold and wait
  - No preemption
  - Circular wait
- Usually not due to an attack

## Deadlock Conditions

- *Mutual exclusion:*
  - At least one resource must be held in a non-sharable mode
    - That is, if any other process requests this resource, then that process must wait for the resource to be released
- *Hold and wait:*
  - A process must be simultaneously holding at least one resource and waiting for at least one resource that is currently being held by some other process
- *No preemption:*
  - Once a process is holding a resource ( i.e. once its request has been granted ), then that resource cannot be taken away from that process until the process voluntarily releases it
- *Circular wait:*
  - A set of entities must be holding resources such that each entity is waiting for a resource held by another entity in the set

# Methods of Handling Deadlocks

- *Prevention*: prevent 1 of the 4 conditions from holding
  - Do not allow the system to get into a deadlocked state
  - Do not acquire resources until all needed ones are available
  - When needing a new resource, release all held
- *Avoidance*: ensure process stays in state where deadlock cannot occur
  - *Safe state*: deadlock can not occur
  - *Unsafe state*: may lead to state in which deadlock can occur
  - Abort a process or preempt some resources when deadlocks are detected
- *Detection*: allow deadlocks to occur, but detect and recover
  - If deadlocks only occur once a year or so, it may be better to simply let them happen and reboot as necessary than to incur the constant overhead and system performance penalties associated with deadlock prevention or detection
  - This is the approach that both Windows and UNIX take


# Denial of Service Models

# Denial of Service

innovate achieve lead

## Overview

- *"A denial of service occurs when a group of authorized users of a service makes that service unavailable to a (disjoint) group of authorized users for a period of time exceeding a defined maximum waiting time"*

# Denial of Service



## What do we mean by "authorized user"?

- If a user is not authorized, then in theory access control mechanisms that protect the server will block the unauthorized users from accessing the server
- But in practice, the access control mechanisms may be ineffective
  - E.g., An intruder may compromise a user's account to gain access to a server
- The policy controlling access to a network server may be unworkable
- For example:
  - A policy states that only customers interested in the products sold may access the server—but the access control mechanisms could not tell whether a remote user accessing the server was interested in the products, or trying to block access by others
- Hence the first "group of authorized users" is simply the group of users with access to the service, whether the security policy grants them access or not.

# Availability and Network Flooding



## Example: SYN Flood Attack

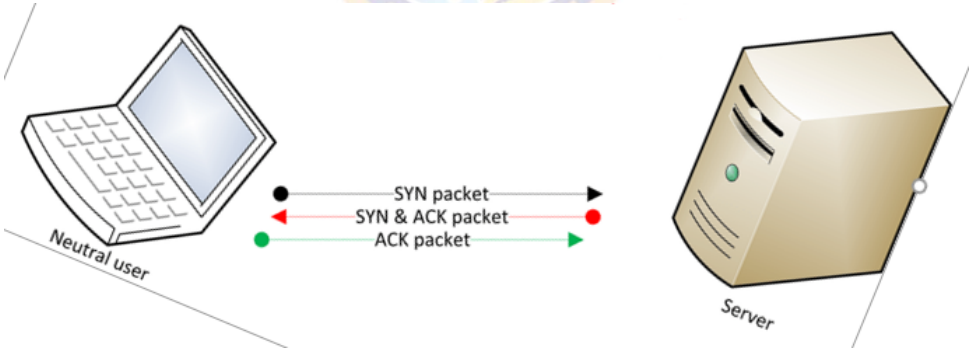
- Access over Internet must be unimpeded
- In flooding attacks attackers try to overwhelm system resources
- If many sources flood a target, it's called *distributed denial of service attack* (DDoS)
- The SYN (short for "synchronize") flood is the most common type of flooding attack
- It is based on the initiation of a connection using the TCP protocol
- A SYN flood sends a series of "SYN" messages to a computer (E.g., web server)

# Availability and Network Flooding

innovate achieve lead

## Example: SYN Flood Attack

- In a normal case, the user sends the SYN packet to the target
- When a server receives a SYN request, it responds with a SYN-ACK (synchronize acknowledge) message
- The source then responds with an ACK (acknowledge) message that establishes a connection between the two systems

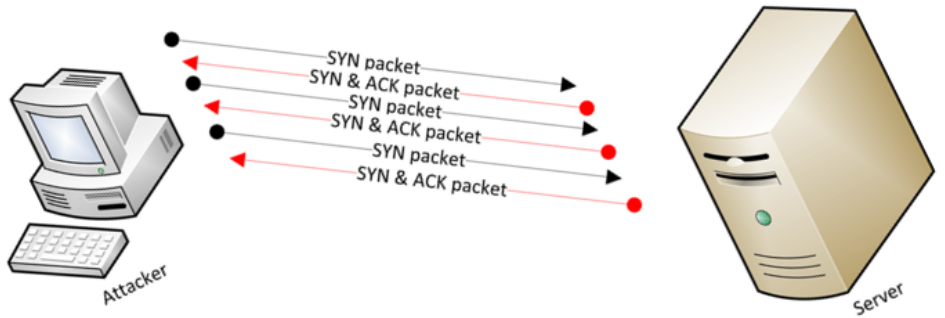


# Availability and Network Flooding

innovate achieve lead

## Example: SYN Flood Attack

- In a SYN flood attack, a computer sends a large number of SYN requests, but does not send back any ACK messages
- Therefore, the server ends up waiting for multiple responses, tying up system resources
- If the queue of response requests grows large enough, the server may not be able respond to legitimate requests
- This results in a unresponsive or slow server



# Denial of Service



## Components of Denial of Service Models

- Denial of service models have two essential components
  - waiting time policy
  - user agreement
- Wait time policy
  - Controls the time between a request for a resource and the allocation of that resource to the requesting process
  - A denial of service occurs when the bound set by this policy is exceeded
  - The environment in which the request is made influences the policy
  - Example:
    - The acceptable waiting time for a pacemaker to take action affecting a patient's heart beating is considerably different than the acceptable waiting time for a purchase from an Internet website to be acknowledged.

## Components of Denial of Service Models

- User agreement
  - Establishes constraints that a process ("user") must meet in order to ensure service
  - These are designed to ensure that a process will receive service within the waiting time
  - For example:
    - Consider parallel processes accessing a mutually exclusive resource
    - A user agreement for this situation would be that once a process acquires the resource, it must (eventually) release that resource
    - When released, there are enough unallocated resources to enable a process waiting for those resources to proceed


innovate
achieve
lead


BIRLA INSTITUTE OF TECHNOLOGY & SCIENCE, PILANI
ज्ञानं परमं बलम्

# Thank You!