Terminology and definition of software vulnerabilities

Software Dependability Course

Content:

- Introduction
- Error faults and failures
- Computer Policy
- Software vulnerability
- Taxonomy and Classification
- System Objects, Attributes and Constraints
- Conclusion
- Q&A

Introduction

terminology and the algorithmic conventions

Software Vulnerability

Errors made during software development lead to vulnerability

Problem Statement

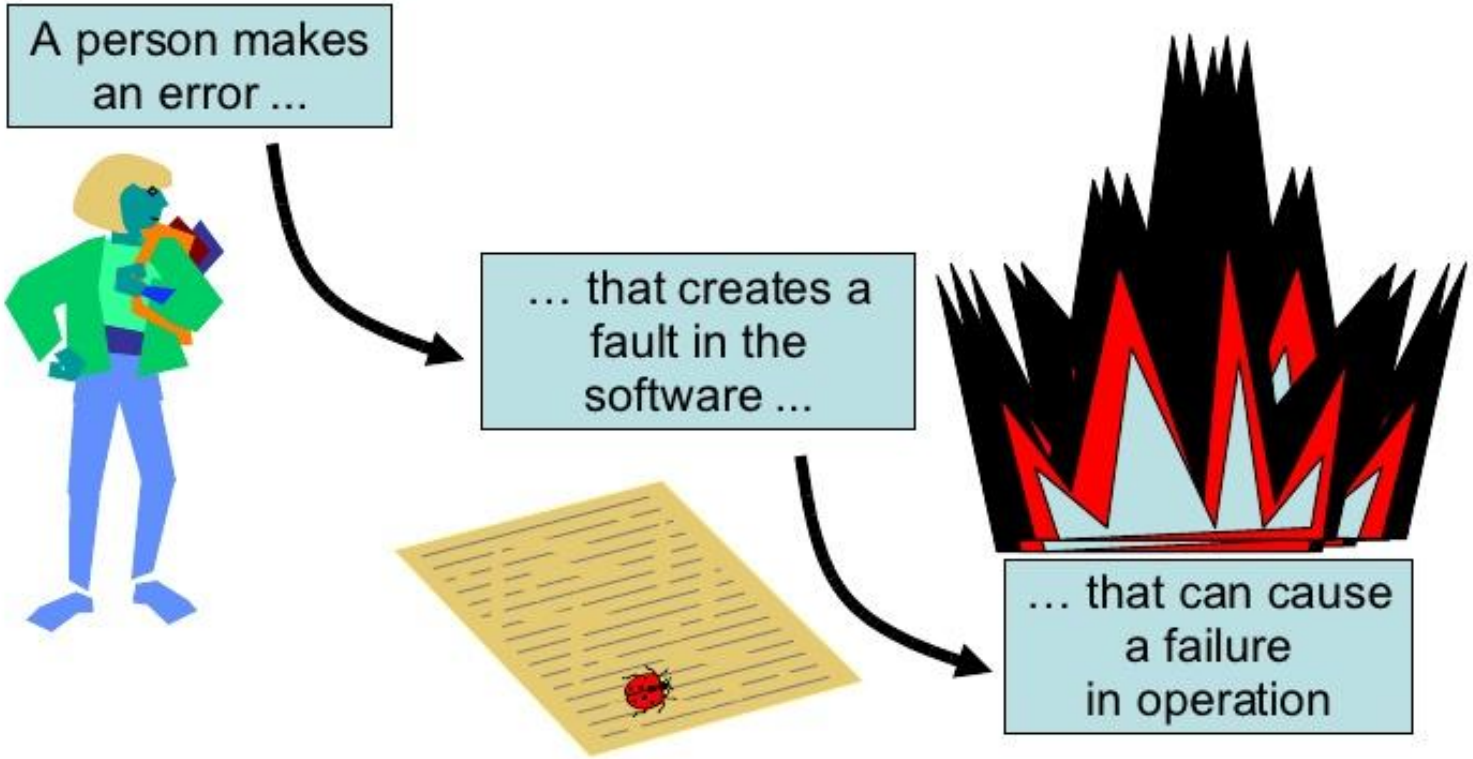
Error - Fault - Failure

Er

typogra

a misre
specifi

misunc
what th
does



e

Example

Fault example - is a programming mistake

Add function that work fine,except 5+3=7(8).

Error - activated fault

We call Add with 5 and 3 , get 7 and put it in some variable

Failure - duration in system behavior

Schedule a meeting for 7 a.m instead of 8 a.m

Based on these steps, a **vulnerability** may appear.

Computer Policy

Policy helps to define what is considered valuable, and species what steps should be taken to safeguard those assets

Policy is defined as the set of laws, rules, practices, norms, and fashions that regulate how an organization manages.

Access to a system may be granted only if the appropriate clearances are presented. **Policy** defines the clearance levels that are needed by system subjects to access objects.

In an access control model, **policy** species the access rules for an access control framework.

POLICY AND EXPECTED POLICY

Policy

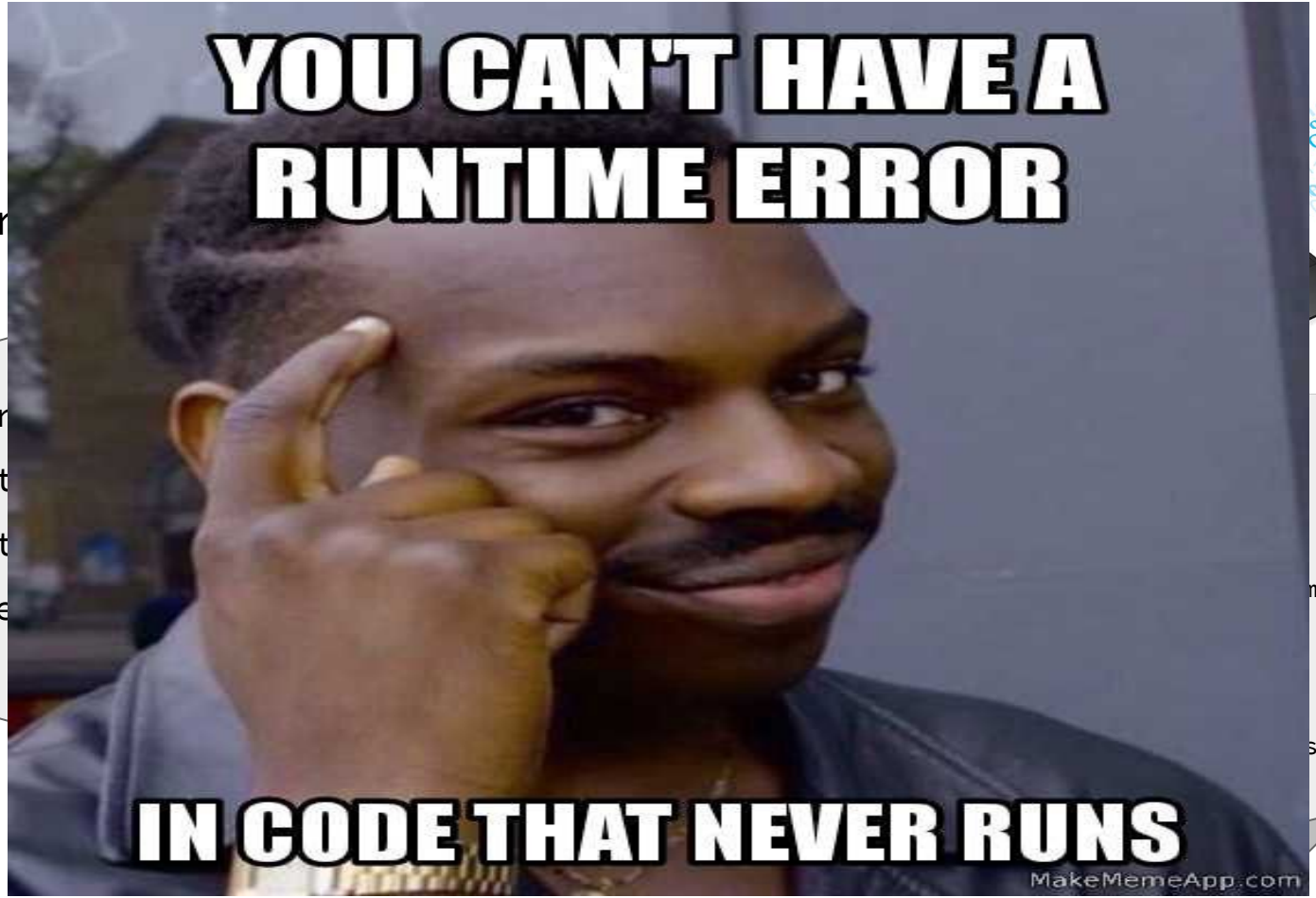
is a set of rules that determine the acceptable value of a system and how its state changes over time.

Expected policy

are the rules that the user expects the system and applications to enforce so as to maintain the value of the system as it changes through time

Softwar

Ther
soft
of t
cate



SV: EXAMPLE OF ACCESS MATRIX

O

S

	File 1	File 2	File 3	Program 1
Ann	own read write	read write		execute
Bob	read		read write	
Carl		read		execute write

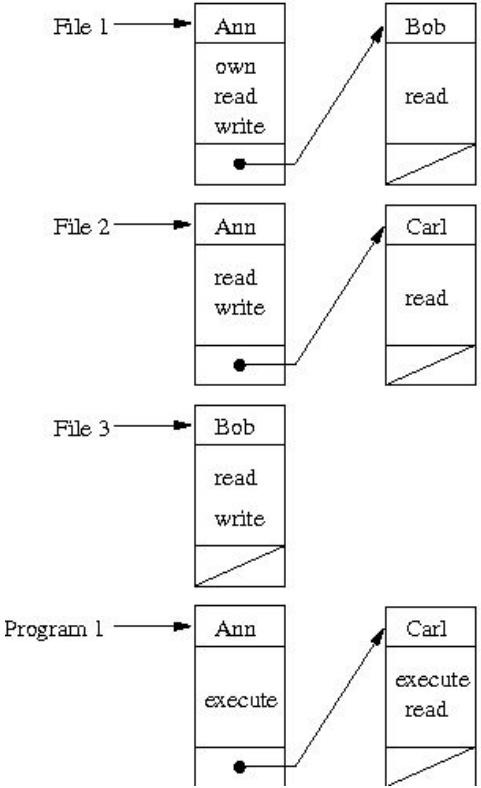
SV: There are three approaches to implementing the access matrix in practical way

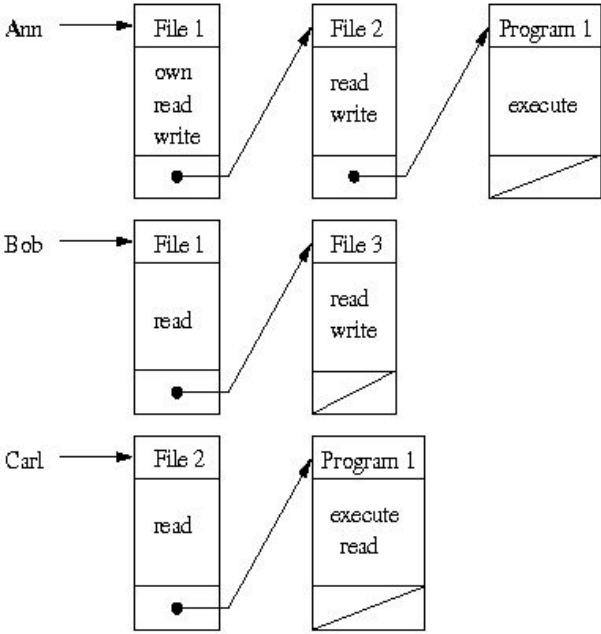
Authorization Table Non empty entries of the matrix are reported in a table with three columns, corresponding to subjects, actions, and objects, respectively. Each tuple in the table corresponds to an authorization.

Access Control List (ACL) The matrix is stored by column. Each object is associated with a list indicating, for each subject, the actions that the subject can exercise on the object.

Capability The matrix is stored by row. Each user has associated a list, called capability list, indicating, for each object, the accesses that the user is allowed to exercise on the object.

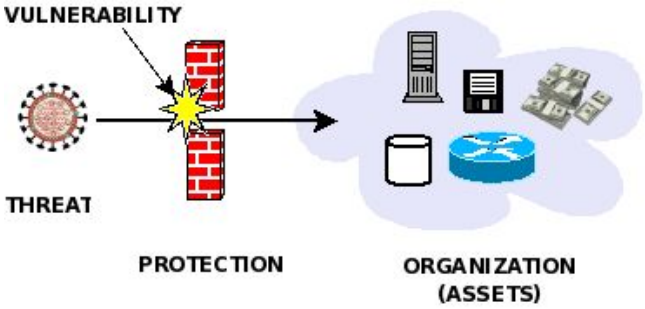
User	Access mode	Object
Ann	own	File 1
Ann	read	File 1
Ann	write	File 1
Ann	read	File 2
Ann	write	File 2
Ann	execute	Program 1
Bob	read	File 1
Bob	read	File 3
Bob	write	File 3
Carl	read	File 2
Carl	execute	Program 1
Carl	read	Program 1





SV: Software Vulnerability has a three forms:

- Access Control
- State Space
- Fuzzy



It is a policies that define what is allowed or desired in the system, dependency of “notion of computer vulnerability” to “notion of a policy”.

A computer system is composed of **STATES** describing the current configuration of the entities that make up the computer system.

The system computes through the application of **state transitions** that change the **state of the system.**

All the states reachable from a given initial state using a set of **state transitions** fall into the class of **authorised** or **unauthorised.**

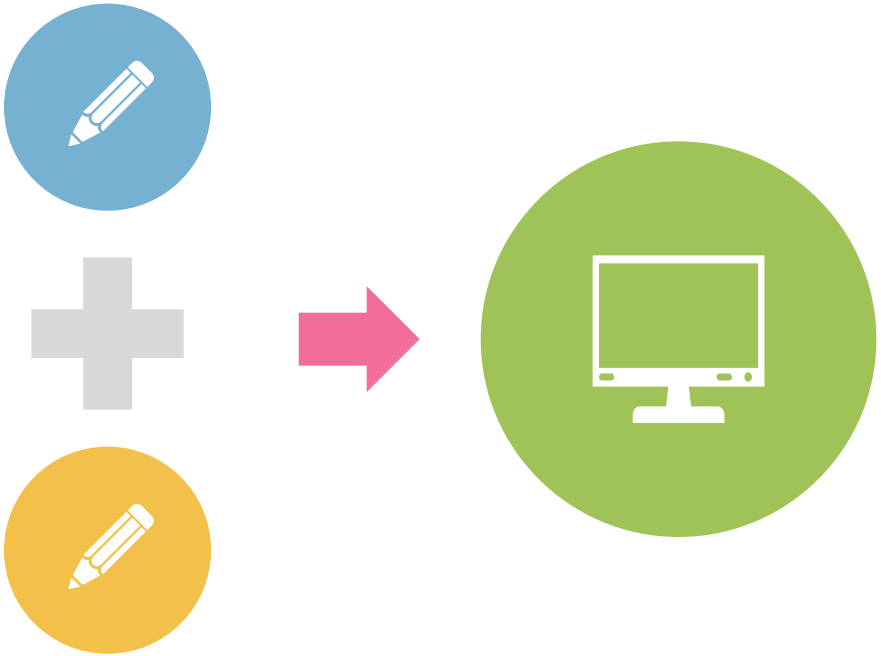
A **vulnerable state** is an authorized state from which an **unauthorized state** can be reached using **authorized state transitions.**

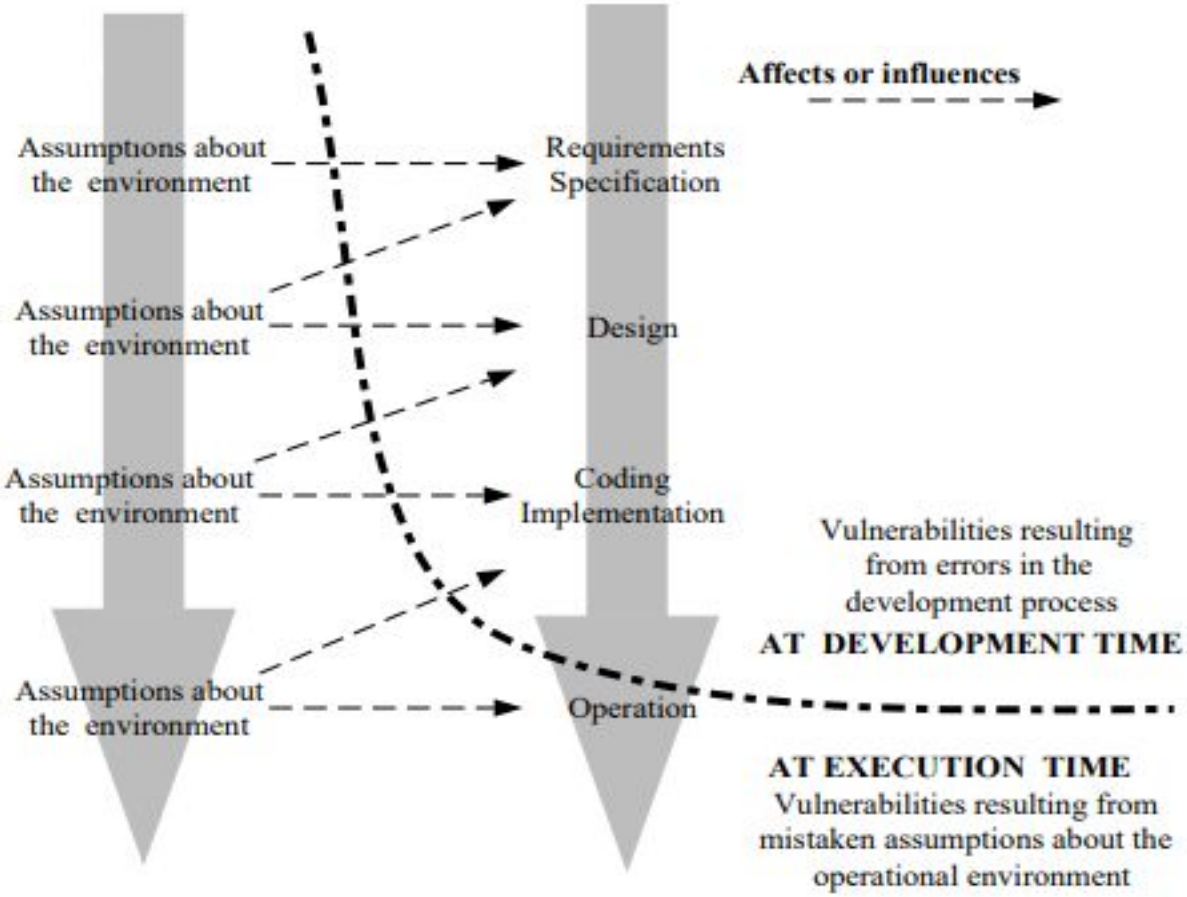
A **compromised state** is the state so reached. An **attack** is a sequence of **authorised state transitions** which end in a **compromised state,** an attack begins in a **vulnerable state...**

Arguments for the development of a unifying and practical definition for software vulnerabilities

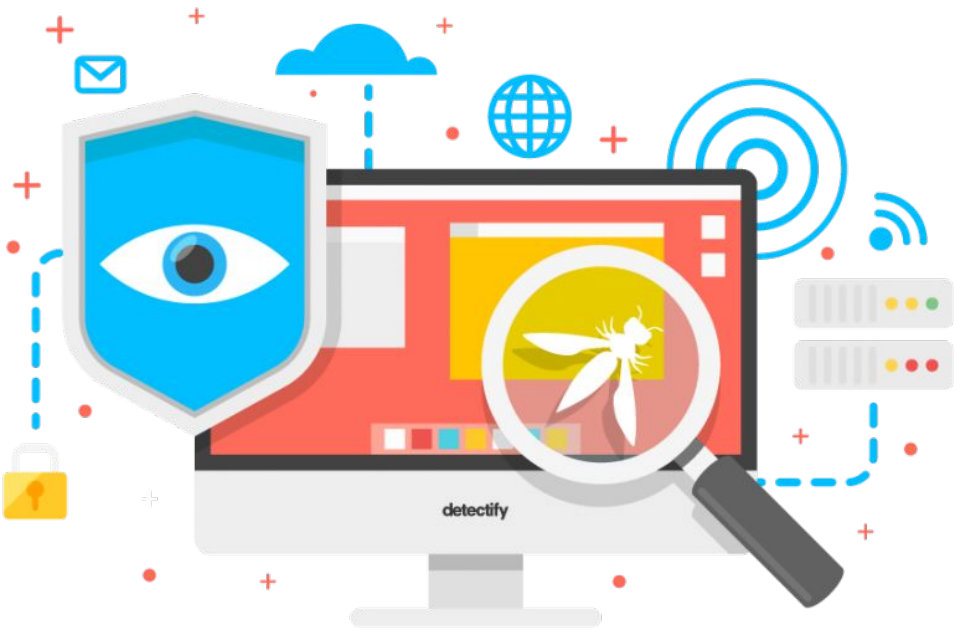
Improvement of Software Systems

Identification of Protection Domains




Affects or influences
Assumptions about the environment
Requirements Specification
Assumptions about the environment
Design
Assumptions about the environment
Coding Implementation
Assumptions about the environment
Operation
Vulnerabilities resulting from errors in the development process
AT DEVELOPMENT TIME
AT EXECUTION TIME
Vulnerabilities resulting from mistaken assumptions about the operational environment

Software vulnerability



Reasons of vulnerabilities:

- errors in:
  - specifications
  - design
  - coding
  - configuration
- the mismatch between the assumptions made during the development about the execution environment of the software and the environment in which the program executes

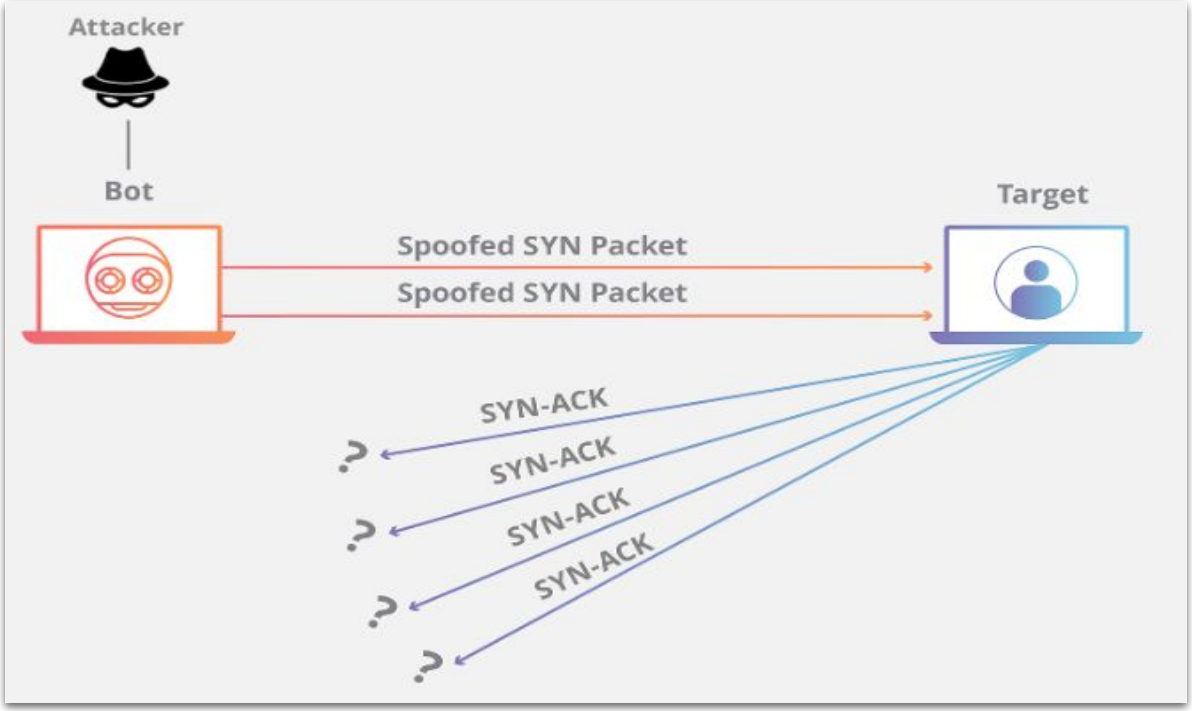
Examples of vulnerabilities

TCP LAND(Local Area Network Denial)

Example of IP-datagram in LAND attack			
4(IP version)	Length of header	0(Type of service)	Total length
1234(Identifier)		0(Flags)	0(Fragment offset)
29(Time to live)	06(Protocol)	Header checksum	
195.164.125.16(Source address)			
195.164.125.16(Destination address)			
0(Options)		0(Padding)	

Examples of vulnerabilities

TCP SYN Flood



Examples of vulnerabilities

Java vulnerability

Configuration errors

Wrong assumptions about environment

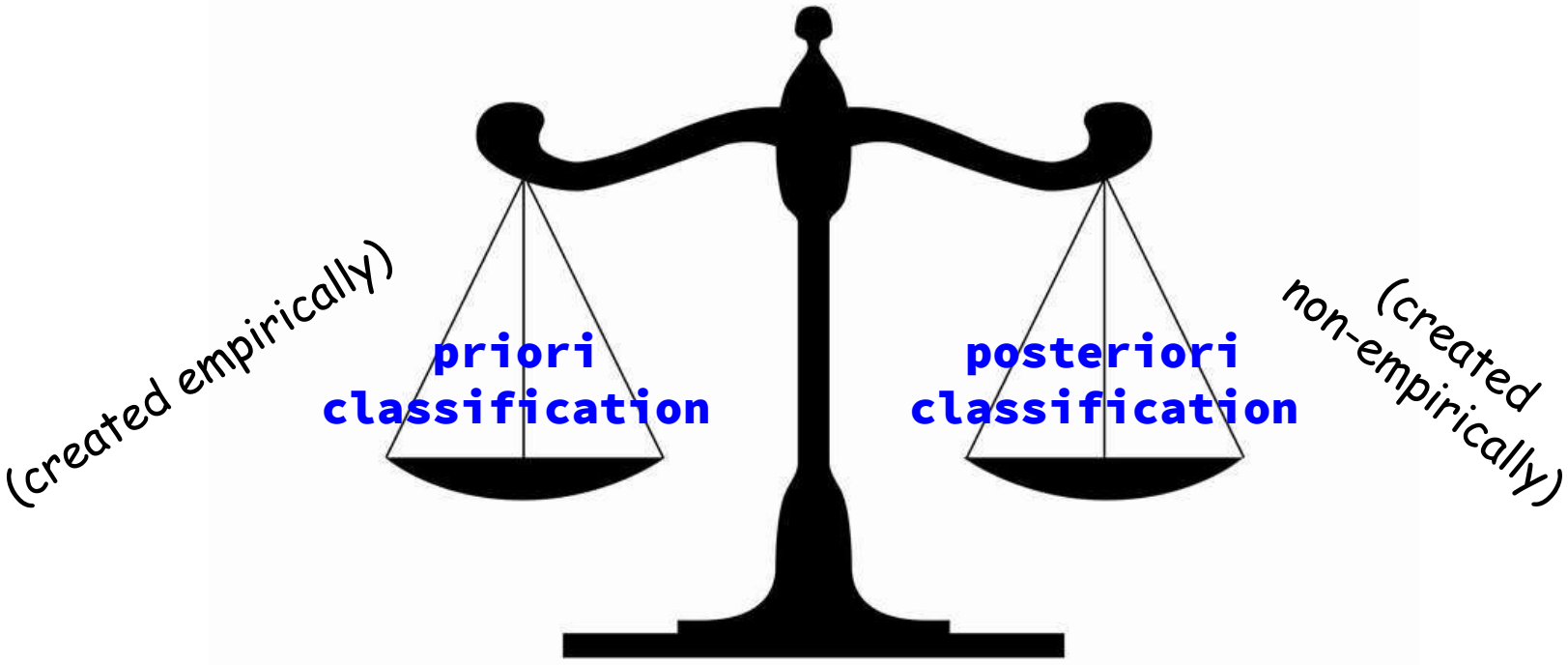
Outdated requirements

Taxonomy and Classification

Taxonomy is mainly a classification mechanism

- The *Cambridge* dictionary defines taxonomy as *“a system for naming and organizing things, especially plants and animals, into groups that share similar qualities”*.
- The *Merriam-Webster* dictionary defines taxonomy as *“Orderly classification of plants and animals according to their presumed natural relationships”*.
- The *Oxford* dictionaries define taxonomy as *“The classification of something, especially organisms”* or *“A scheme of classification”*.

Taxonomy and Classification



Classification

Taxonomy and Classification

Empirical

(experience)

Non-empirical

(theory)

Purely Empirical — **Research** — **Purely Non-empirical**

- facts
- experience
- observing
- nothing meaningful

- theory
- models
- non proofs

System Objects, Attributes and Constraints

System object is an entity that contains or receives information that has a unique name, and that has a set of operations that can be carried out on it.

An **attribute of an object** is a data component of an object.

A **property of an attribute** is a characteristic of the attribute that can be derived from the attribute by the application of a function to the attribute.

System Objects, Attributes and Constraints

```
--
-- Structure of the `delivery`
--
DROP TABLE IF EXISTS `delivery`;
CREATE TABLE IF NOT EXISTS `delivery` (
  `id_delivery` int(11) NOT NULL AUTO_INCREMENT,
  `date` date NOT NULL,
  `id_customer` int(11) NOT NULL,
  PRIMARY KEY (`id_delivery`),
  KEY `id_customer` (`id_customer`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

System Objects, Attributes and Constraints

An **attribute refinement** is a finite refinement of attributes within attributes, and results in the identification of the attributes about which assumptions are made. The attribute refinement cannot contain a property of an attribute.

System Objects, Attributes and Constraints

Running Programm

Environment

```
if [ -d /etc/profile.d ]; then
  for i in /etc/profile.d/*.sh; do
    if [ -r $i ]; then
      . $i
    fi
  done
  unset i
fi
JAVA_HOME=/usr/local/java/jdk1.6.0_45
PATH=$PATH:$HOME/bin:$JAVA_HOME/bin
JRE_HOME=/usr/local/java/jdk1.6.0_45
PATH=$PATH:$HOME/bin:$JRE_HOME/bin
export JAVA_HOME
export JRE_HOME
```

System Objects, Attributes and Constraints

The **Attribute Constraint** identifies the property or set of properties that are being assumed about that particular attribute.

```sql
--
-- Restricting foreign key of the table `delivery`
--
ALTER TABLE `delivery`
  ADD CONSTRAINT `delivery_ibfk_1` FOREIGN KEY (`id_customer`)
    REFERENCES `customer` (`id_customer`)
    ON DELETE CASCADE
    ON UPDATE CASCADE;
```

Notations

Operator	Meaning	Example	Explanation
$\wedge$	logical AND operator	$x > 0 \wedge x/y > 5$	
$\vee$	logical OR operator	$x > 0 \vee x < -100$	
$\in$	IN operator (tests for set membership)	$x \in S$	x is in S
$\forall$	FOR ALL operator	$\forall x > 10$	For all x greater then 10
$\triangleright$	string comparison and substring operator	$x \triangleright y$	
$\mid$	SUCH THAT operator	$\forall x \in S \mid x \in D$	
$\coloneqq$	Assignment operator		
$:\coloneqq$	The definition operator		Uses to define functions end terms


Q&A
wat


THANK YOU FOR
YOUR ATTENTION
memegenerator.net