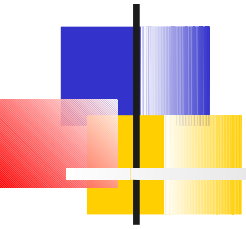
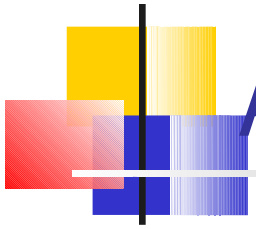


Asset Analysis

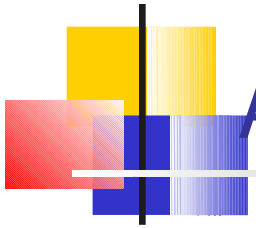
Asset Analysis -I

- It discover the assets that result in an impact (a loss for the organization) if successfully attacked
- It should discover which ICT resources an organization needs to work in an efficient way
- Which processes are fundamental for the organization
- Which ICT resources are fundamental for these processes
  - Evaluate the impact for the organization if
    - The process is stopped (integrity or availability of the resource)
    - The resource has to be rebuilt ex novo (integrity)
    - The information in the resource are known to the attacker (confidentiality)



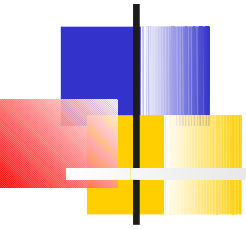
Asset Analysis -II

- Physical and Logical Resources
  - Database
  - Applications to access the database and compute the outputs of interest (may be even more important than the database)
  - Computational power
  - Communication bandwidth

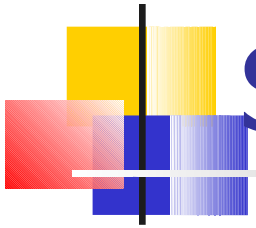


Asset Analysis -III

- In general it is rather complex to approximate the value of a resource
- A possible heuristics consider the cost of rebuilding the resource if it disappears
- This analysis is useful non only for security reasons but also to be aware of which resources do exist and how they are used (catalogue of resources)



Security Policy



Security Policy

A set of rules that an organization introduces both to minimize the risk and to define the goals of security

- Defining the goal of security = which resources are to be protected
- Defining the correct behavior of all the users
- Forbidding dangerous behaviors and components
- It implies the definition of
  - System architecture
  - Catalogue of components and of application
  - Users (rights and constrains)
  - Administrators (rights and constrains)
  - Legal use of the resources
  - Who has to verify that the policy is applied
  - What happens if the policy is violated

Security Policy

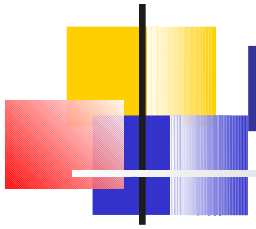
- It is critical because it defines
  - The goals of an organization
  - Legal behaviour for each class of users
  - Whether components can still have some vulnerabilities and how they should be used
  - Rules to manage both human and ICT resources
  - Roles and responsibility
- The security policy cannot violate the legislation that concerns ICT systems

Subject and object

- A more abstract definition of a policy can represents user and resources in a more abstract way where an object defines some operations that can be invoked
- A subject is any entity that can invoke the operations defined by an object
- An object that invokes some operations defined by other objects is both a subject and an object
- The implementation of subjects and objects depends upon the implementation level (e.g. the VM) of interest

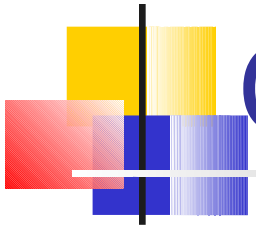
Subject = user, application, program, process, thread, instruction …

Object = instance of an abstract data type, procedure or function, variable, logical or physical resources



Rights

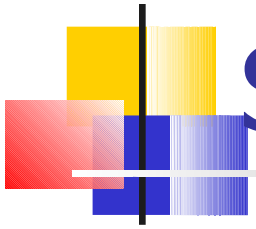
- A right implies that a subject is entitled to invoke an operation of an object
- Rights are directly or indirectly deduced from the security policy
  - Direct = S can read the file F
  - Indirect = since S can read F then a program P executed by S can read the memory segment MS that stores a record of F
    = the right of P on MS is deduced from the one of S on F



Objects, operations and types

- The definition of an object with the operations it defines (implement) implies a data type definition
- A type system can be used to allow only those invocations of an operation on an object that are entitled by the policy
- However dynamic controls cannot be avoided due to vulnerabilities in the compiler or in the run time support that may result in run time behavior that differs from the expected one defined by the specification

A Classification of Security Policies

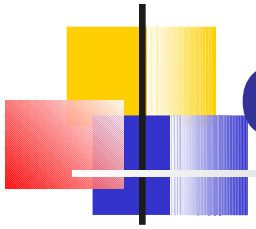


- Default allow = it defines forbidden behaviours and allows anything that is not forbidden = enumerating badness
- Default deny = it defines legal behaviors and forbids anything that is not defined eg anything else
- Default allow is very dangerous = enumerating badness but we can forget to enumerate some bad behavior

An analogy

- Default allow = defines a set S by describing the elements that do not belong to S

- Default deny = defines a set by describing the elements that belong to S



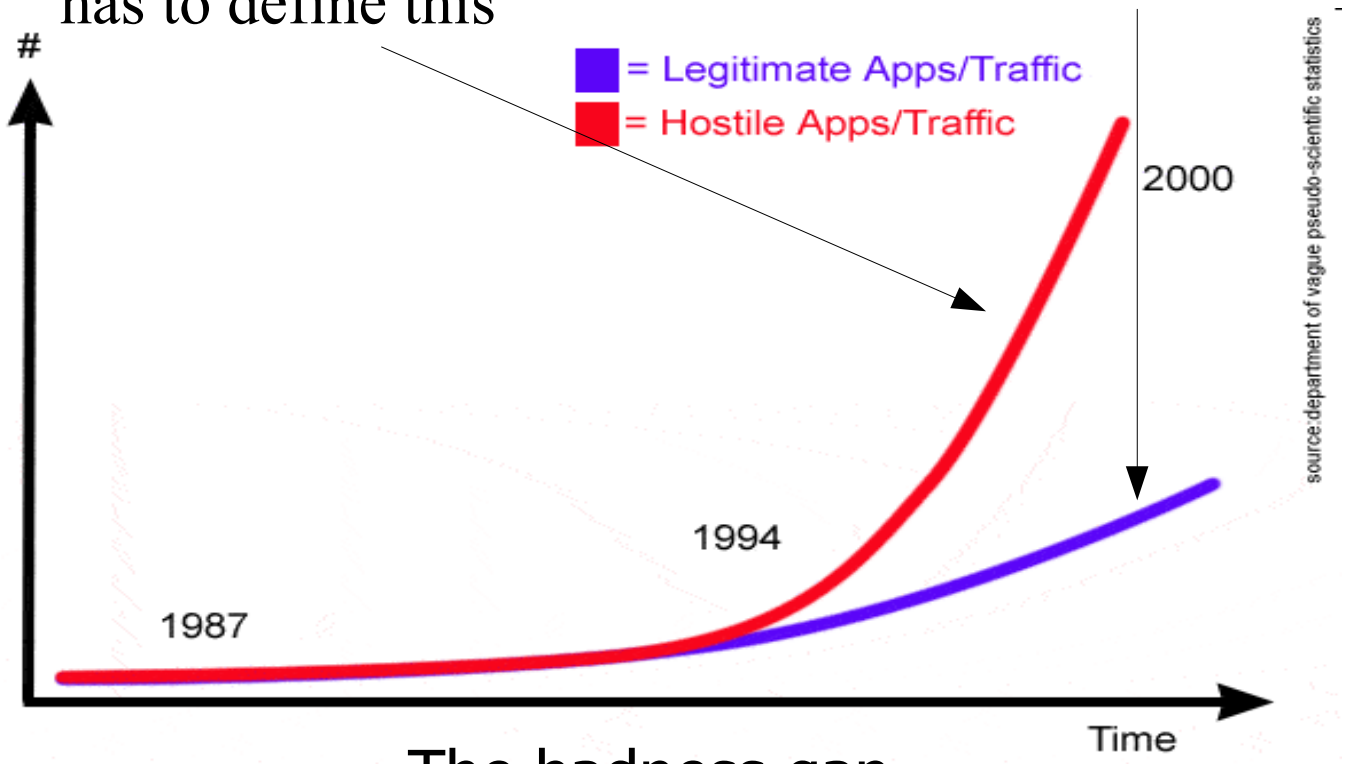
The Six Dumbest Ideas in Computer Security (M.Ranum)

1. **Default Permit**
2. **Enumerating Badness**
3. **Penetrate and Patch**
4. **Hacking is Cool**
5. **Educating Users**
6. **Action is Better Than Inaction**

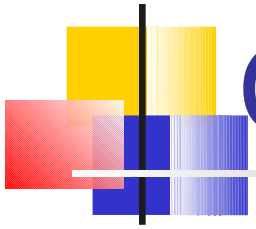
Enumerating Badness

A default allow policy has to define this

A default deny policy defines this

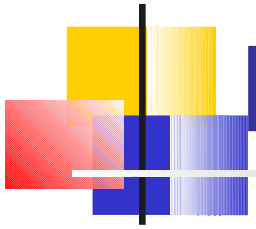


The badness gap



Classes of security policy

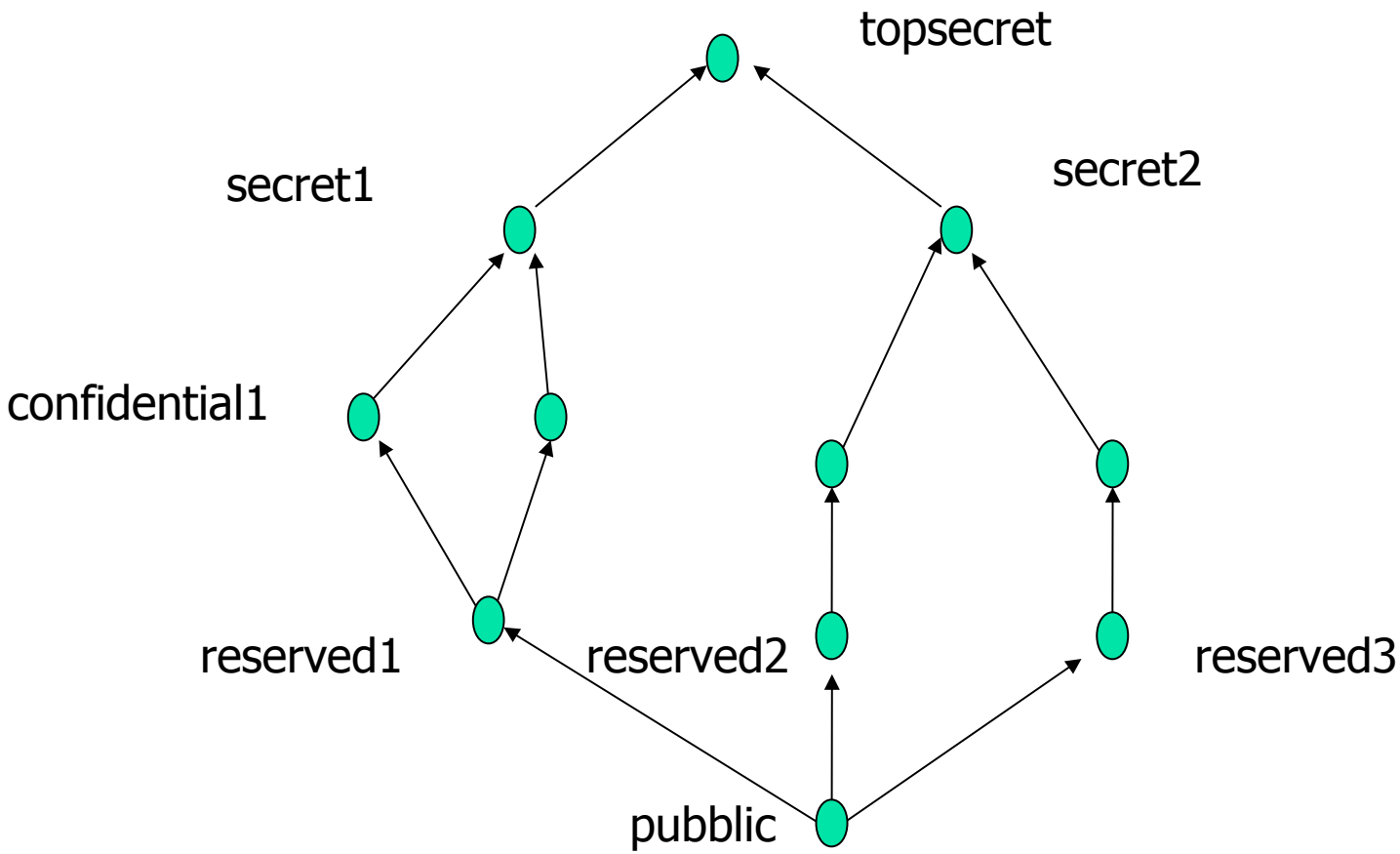
- Discretionary access control
  - An owner exists for each object
  - The owner defines
    - Which subjects can operate on the object (need to access)
    - The rights for each subject
- Mandatory access control
  - There are some system wide rules that the owner has to satisfy

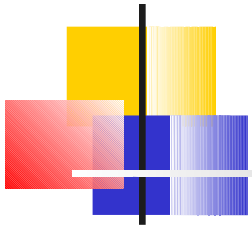


Mandatory Access Control

- All the objects are partitioned into classes
- All the subjects are partitioned into the same classes (not strictly required but it simplifies everything)
- All the classes are partially ordered
- A subject may be enabled to invoke an operation only if the classes of the subject and of the object satisfy a predefined condition

Partial Order

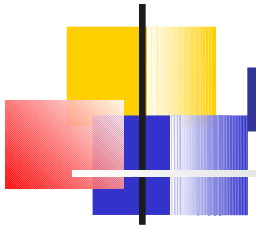




MAC information flow - I

- Object = file
- Operations = read/write/append
- A subject in a class C may be enabled to
  - Read any file with a class lower than or equal to C
  - Write any file with a class equal to C
  - Append a record to a file with a class larger than C
  - The owner of the file can grant the rights provided that the previous rules are satisfied

This policy prevents lost (leaks) of information (No write down)

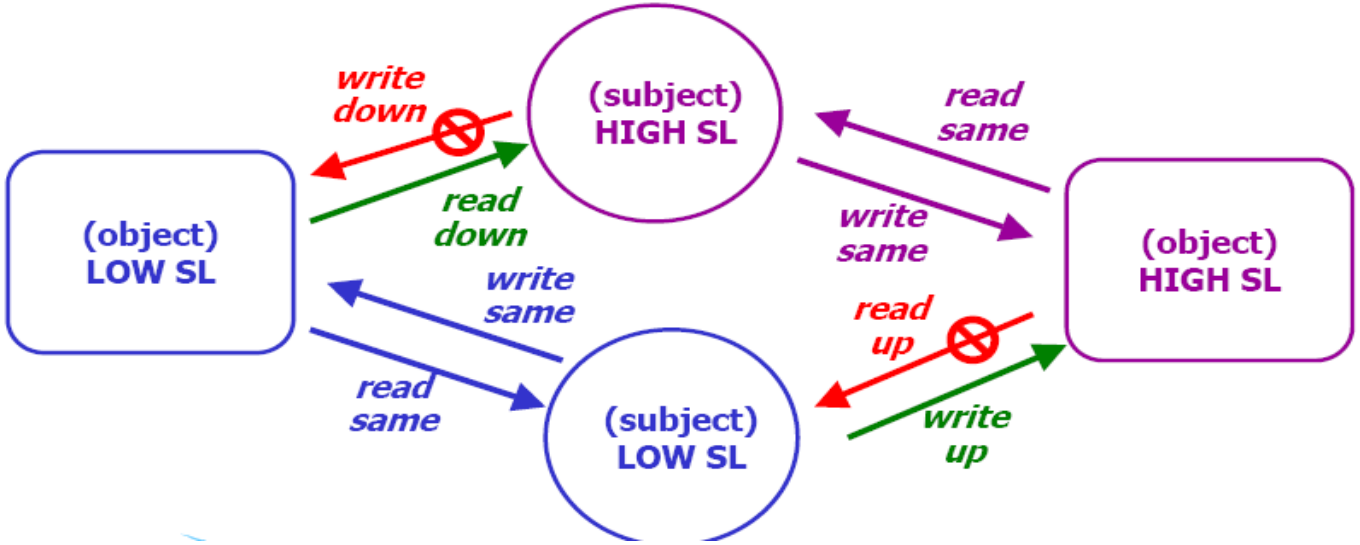


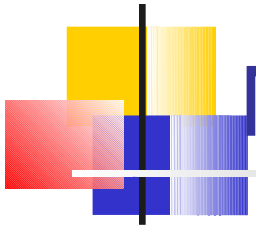
no write down

- Prevents an information flow from object with an high level to those with a lower one
- Guarantee confidentiality of information
- As a counterpart, information at the higher level increases because the level of information cannot decrease
- A further operation is required to periodically desecretate information to the lower levels

Mandatory Access Control - I

- Bell-LaPadula Policy (multilevel security)
  - access control attributes:
    - hierarchical security level
    - set of non hierarchical categories
  - fixed rules: "no read up, no write down"

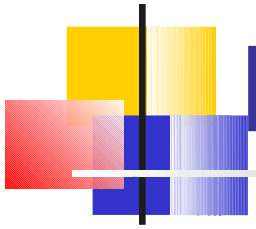




MAC information flow - II

- Object = file
- Operation= read/write
- A subject in class C may be enabled to
  - Write any file with a class lower than or equal to C
  - Read any file with a class larger than or equal to C

Integrity is privileged (No write up)

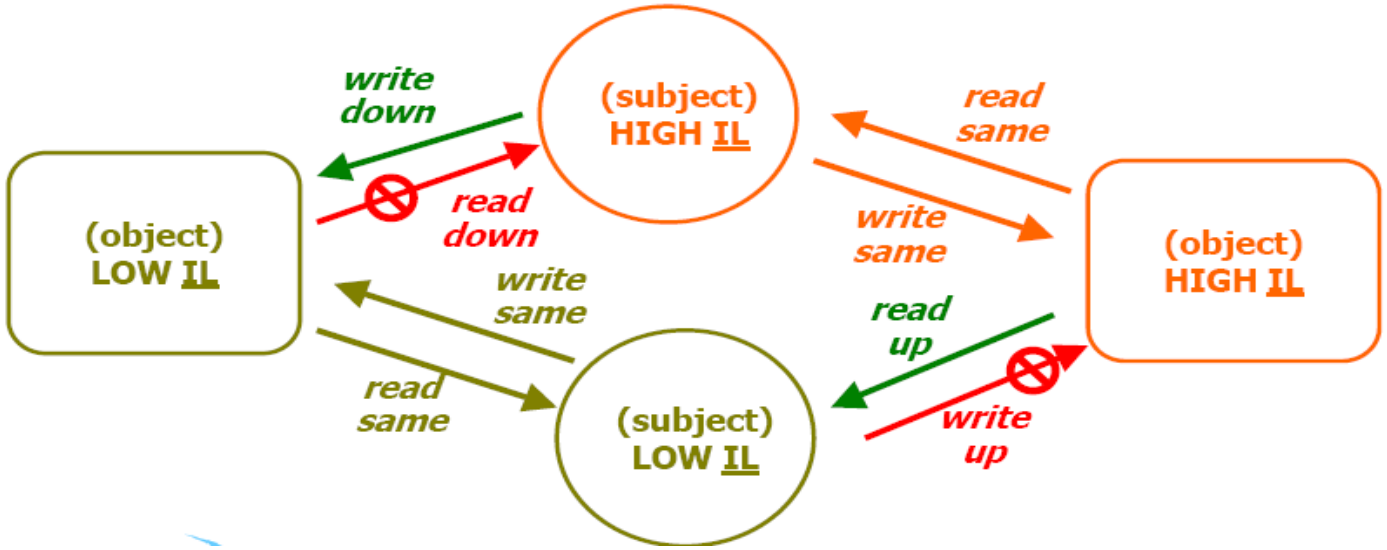


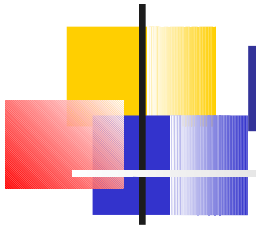
No write up

- A low integrity subject cannot update an highly integrity object
- Integrity is privileged at the expence of confidentiality

Mandatory Access Control - II

- Biba Integrity model (multilevel security)
  - access control attributes:
    - hierarchical integrity level
    - set of non hierarchical integrity categories
  - fixed rules: "no write up, no read down"
    - exact opposite of BLP/multilevel security





No interference

- Each object and each subject is paired with a label that defines the corresponding level
- The label paired with an object is a function of the operations that are invoked and of the subject invoking these operation
- A system satisfies the no interference principle if the labels paired with an object do not change even if the subject with either a lower or an higher level are removed from the system (Bell-LaPadula/Biba)
- No information is leaked from the higher levels and no information from low level object can influence higher level ones

Clark -Wilson -1

- A policy in this class defines
  - A set of consistency contrains on a subset of the objects
  - Some operations (well formed transactions) on the objects that do preserve any consistency constrains
- The system evolution can navigate only in states that satisfies the consistency constrains

Clark -Wilson -2

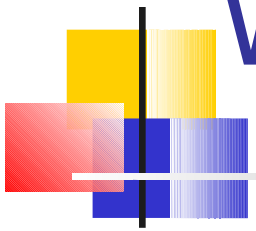
- Each well formed transaction is atomic, either is completed or does not begin
- Atomicity may be implemented by a backup copy of involved objects
- It is the user responsibility to prove that each transaction is well formed, e.g. it does not violate the consistency constrains

CW- Example

- Objects = Bank accounts
- Constrain
  - If money is moved betwen two accounts, their sum does not change
  - At the end of each day the sum of the accounts is equal to the sum of money that has been cashed plus the sum of the accounts at the beginnging of the day
- Any transaction must be atomic

Chinese Wall

- Objects are partioned into classes
- A subject that has invoked an operation on an object cannot invoke operations on objects in distinct classes
- Avoid conflict of interest
- Time dependent
- Can be integrated with a MAC policy



Watermark

- The level of a subject is not fixed but it is a function of the objects it has worked on
- To protect confidentiality, the level increases has the subject reads critical information
- Monotonic increase, after a given level has been reached no decrease is possible
- Time dependent policy

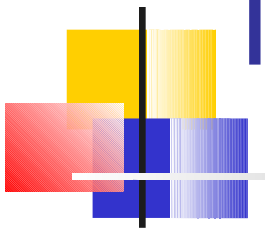
Overall Policy – I

- A real policy can merge several of the previous policy
- As an example
  - No write down
  - Chinese wall
- We have rules that define which objects can be read and other that forbid the access to some other objects

Overall Policy – II

- Distinct policies can be applied to the same object/subject
- There are two levels for a subject, one for confidentiality and one for integrity
  - Some objects consider the confidentiality level (no write down)
  - Some objects consider the integrity level (no write up)

Trusted Computing Base



- TCB includes any component that is involved in the implementation of the security policy
- These components are highly critical because any bug in a component in the TCB is, almost always, a vulnerability
- Any system needs to trust the components in its TCB.
- Assurance of these components is very important
- They should be carefully controlled

Size of the TCB

- The security level of a system and the trust in it increases as the size of the TCB decreases
- Correctness of a small TCB can be proved by applying formal method and this results in a high assurance level
- An important criteria to select among alternative implementation of the same policy

All together now ...

- We can define important resources by looking at process of the organization
- We can define subjects and objects in terms of these resources
- We can define rules on the resource usage and map them into rights
  - Default allow
  - Default deny
- Rights can be defined in one of two framework
  - Mac (system wide constrains)
    - Integrity
    - Confidentiality
    - Static or watermark
  - Dac (no global constrain)
- Data Types + Run time check = Trusted Computing Base
  - Size important for security