Security Engineering
Fall 2015

Lecture 10 –Access Control Models

Fabio Massacci

Recaps: Types of Access Control

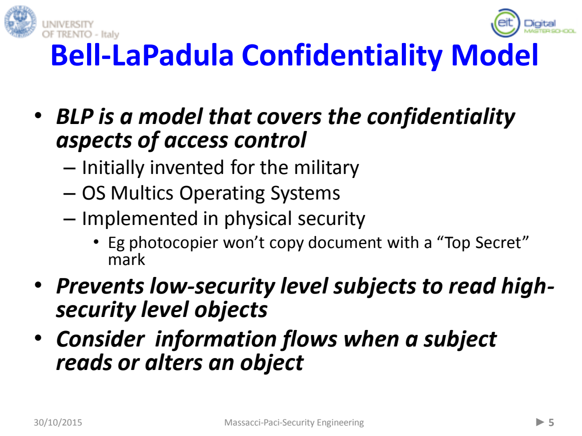
- ***Discretionary Access Control***
  - Policy decided by individual subjects
  - Access based on identity of subjects
- ***Role based Access Control***
  - Policy decided by system
  - Subjects assigned to Roles,
  - (Action,Objects) assigned to Roles
  - Access based on roles activated by subjects
- ***Mandatory Access Control***
  - Policy decided by system
  - Subject assigned to security levels (clearance),
  - Object assigned to security labels
  - Access based on matching objects' labels to subjects' clearances
- ***Credential based Access Control***
  - Access based on attributes qualifying a subject
    - Essentially "self-service" PIP signed by accredited PAPs

Mandatory Access Control

- ***Organization Access Policy is always MAC***
  - I do not decide who can read the grades of my course
- ***Implements***
  - Legislation
  - Commercial Confidentiality – Integrity requirements
  - Paranoia of Board of Directors
  - Pet projects of the above (security holes)
- ***Any policy can be specified → enough to have gigantic tables***
  - Objects → Labels
  - Subject → Labels
  - Match: Action x Object x Subject → {True/False}
- ***Example on RedHat Security Enhanced Linux***
  - "TE uses a matrix of domains and object types derived from the policy. "
  - allow httpd_t net_conf_t:file { read getattr lock ioctl }; gives the domain associated with httpd [=subject] the permissions to read data out of specific network configuration files [=object] such as /etc/resolv.conf.
- ***Example on TSA for flying armed [=object]***
  - Subject [=subject] must be Federal Law Enforcement Officer AND ….
  - Be commissioned to enforce criminal statutes or immigration statutes AND
  - Be authorized by the employing agency to have the weapon in connection with assigned duties:
  - provision of protective duties… OR control of a prisoner… OR …

Security Models

- ***MAC is complicated…***
  - "For Red Hat Enterprise Linux 4 the policy has been designed to restrict only a specific list of daemons. All other processes run in an unconfined state. This policy is designed to help integrate SELinux into your development and production environment. It is possible to have a much more strict policy, which comes with an increase in maintenance complexity."
- ***Security Model = MAC with specific focus***
  - Policy encodes some "default" action in the match function
- ***Security Models allows***
  - Simplification of matching process (essential for humans, less for computers)
  - Simplification of administration
  - Formal verification of security

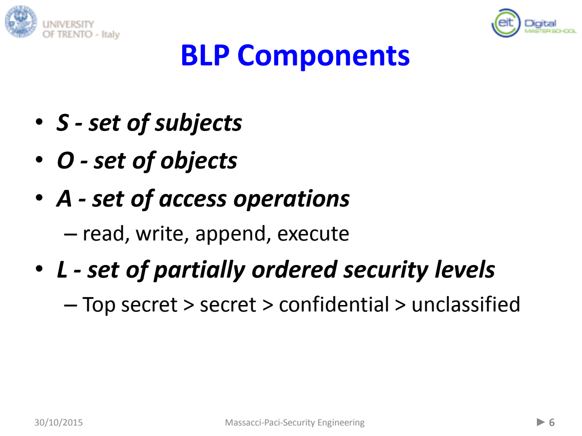


UNIVERSITY OF TRENTO - Italy

eit Digital MASTER SCHOOL

Bell-LaPadula Confidentiality Model

- ***BLP is a model that covers the confidentiality aspects of access control***
  - Initially invented for the military
  - OS Multics Operating Systems
  - Implemented in physical security
    - Eg photocopier won't copy document with a "Top Secret" mark
- ***Prevents low-security level subjects to read high-security level objects***
- ***Consider information flows when a subject reads or alters an object***

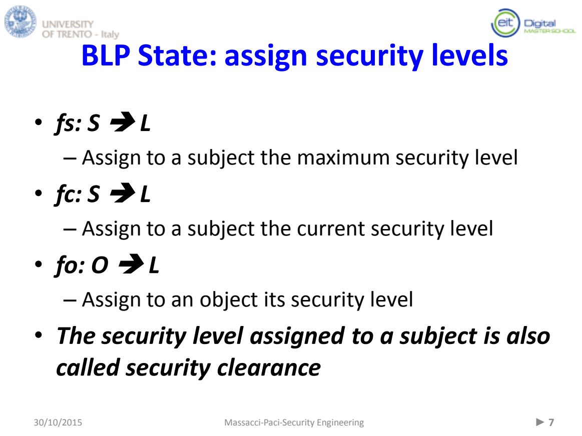


UNIVERSITY OF TRENTO - Italy

eit Digital MASTER SCHOOL

BLP Components

- ***S - set of subjects***
- ***O - set of objects***
- ***A - set of access operations***
  - read, write, append, execute
- ***L - set of partially ordered security levels***
  - Top secret > secret > confidential > unclassified

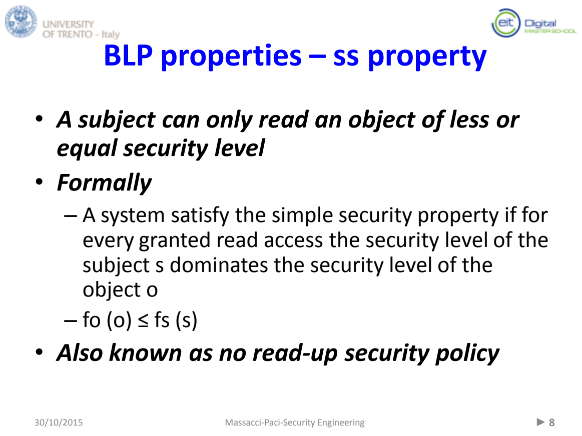


UNIVERSITY OF TRENTO - Italy

eit Digital MASTER SCHOOL

BLP State: assign security levels

- ***fs: S ➔ L***
  - Assign to a subject the maximum security level
- ***fc: S ➔ L***
  - Assign to a subject the current security level
- ***fo: O ➔ L***
  - Assign to an object its security level
- ***The security level assigned to a subject is also called security clearance***



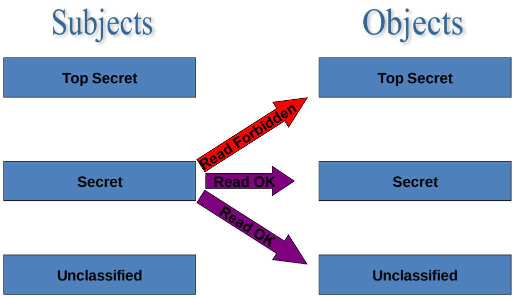
UNIVERSITY OF TRENTO - Italy

eit Digital MASTER SCHOOL

BLP properties – ss property

- ***A subject can only read an object of less or equal security level***
- ***Formally***
  - A system satisfy the simple security property if for every granted read access the security level of the subject s dominates the security level of the object o
  - fo (o) ≤ fs (s)
- ***Also known as no read-up security policy***

BLP properties: ss property - II



Subjects

Objects

Top Secret

Top Secret

Secret

Read Forbidden

Read OK

Secret

Read OK

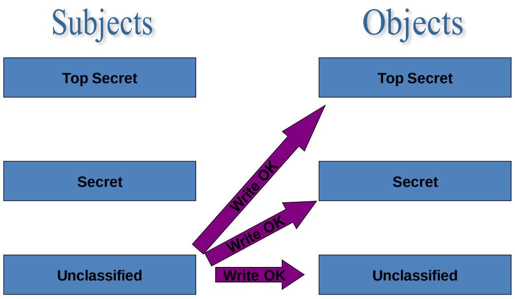
Unclassified

Unclassified

BLP properties - * property

- ***A subject can only write objects of greater or equal security level***
- ***Formally***
  - A system satisfies the * property if for every granted write/modify request the security level of the subject o dominates the security level of the object o
  - fs (s) ≤ fo (s)
- ***Also known as no write-down policy***

BLP properties - * property - II



Subjects

Objects

Top Secret

Top Secret

Write OK

Secret

Secret

Write OK

Unclassified

Write OK

Unclassified

The Basic Security Theorem

- ***A state is secure, if all current assignment of permissions to subjects satisfies the ss-property, ∗- property.***
- ***A state transition is secure if it goes from a secure state to a secure state***
- ***Basic Security Theorem***
  - If all the transitions are secure and the intial state is secure all the subsequent states will be secure regardaless the input

BLP properties - * property limitation

- ***The ∗- property implies that a high level subject is not able to send messages to a low level subject***
  - How can a general send an email to the secretary?
- ***There are several ways to escape from this restriction***
  - Allow a human to work at the same time on two systems
    - That was the original implementation.
  - Temporarily downgrade a high level subject. This is the reason for the current security level f_C.
  - Identify a set of trusted subjects, which are permitted to violate the ∗ - property.
  - Have a "declassification" function to downgrade some information

Tranquillity

- ***McLean: consider a system with an operation downgrade:***
  - downgrades all subjects to system low
  - downgrades all objects to system low
  - enters all access rights in all positions of the access control matrix
- ***The resulting state is secure according to BLP***
- ***Should such a system be regarded as secure?***
  - McLean: no, everybody is allowed to do everything
  - Bell: yes, if downgrade was part of the system specification
- ***Fact: BLP assumes tranquility, i.e. access control rules do not change "on-the-fly"***

Limitations of Bell-LaPadula

- ***Restricted to confidentiality***
- ***No policies for changing access rights***
  - A general and complete downgrade is secure
  - However, BLP is intended for systems with static security levels
- ***BLP contains covert channels***
  - Information flow that is not controlled by the model

Covert Channels

- ***Covert channels are information channels that are not controlled by the security mechanism of the system***
- ***Information can flow (leak) from a high security level to a low security level***
  - A subject assigned to a low-security level can detect the existence of an high-security level object when it is denied access
  - Sometimes, it is not sufficient to hide only the content of objects. Also their existence may have to be hidden.
- ***Telling a subject that a certain operation is not permitted constitutes information flow***

Bell-LaPadula Example

- ***ESSE3 Clearances***
  - Students' Secretariat > Professor > Assistant > Student
  - Not really true (ESSE3 is RBAC not BLP)
- ***Kate is a teacher for the Security Engineering course → clearance A***
  - She can login into the esse3 system as teacher and as student
- ***Andrea is student enrolled in the Security Engineering course → clearance S***
  - He can only login as student

Bell-LaPadula Example

- ***Kate***
  - creates file f1 with P security level
- ***Andrea***
  - creates file f2 with S security level
- ***Is Kate***
  - authorized to read f2?
  - authorized to write f2?
- ***Kate***
  - creates an exam file f3 with A security level
- ***Is Andrea***
  - authorized to read the f3?

Biba Integrity Model

- ***State-machine model similar to BLP which focuses on integrity aspects of access control***
- ***Focus on preventing unauthorized modifications of data***
- ***Access permission based on***
  - Assignment of subjects and objects to integrity levels
- ***Prevents information flow from low-integrity levels to high-integrity levels***

Biba Integrity Model Components

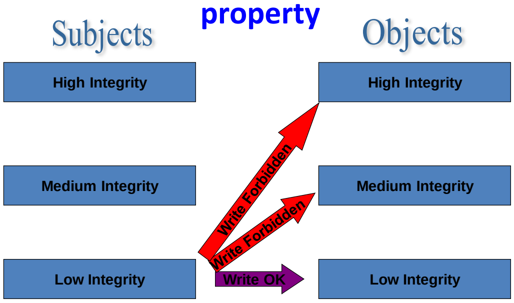
- ***S – set of subjects***
- ***O – set of objects***
- ***A – set of access operations***
  - modify, observe, execute, invoke
- ***fs: S → L***
  - Assign to a subject the integrity level
- ***fo: O → L***
  - Assign to an object its integrity level

•

Biba Integrity Model properties: si property

- ***A subject can modify an object only if the integrity level of the subject dominates the integrity level of the object***
- ***Formally***
  - – A subject s can modify (alter) an object o if fs (s) ≥ fo (s)
- ***Also known as no write-up policy***
  - –

Biba Integrity Model properties: si property



Subjects

Objects

High Integrity

Medium Integrity

Low Integrity

Write Forbidden

Write Forbidden

Write OK

High Integrity

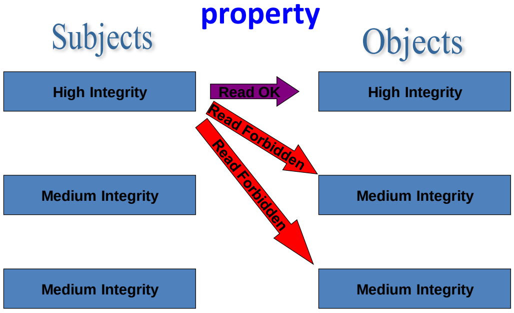
Medium Integrity

Low Integrity

Biba Integrity Model properties: property

- ***A subject can read an object only if the integrity level of the subject is dominated by the integrity level of the object***
- ***Formally***
  - – A subject s can read (observe) an object o if fs (s) ≤ fo (s)
- ***Also known as no read-down policy***

Biba Integrity Model properties: property



Subjects

Objects

High Integrity

Medium Integrity

Medium Integrity

Read OK

Read Forbidden

Read Forbidden

High Integrity

Medium Integrity

Medium Integrity

Biba Integrity Model: dynamic integrity properties

- ***Automatically adjust subjects and objects assigned integrity levels***
- ***Subject Low Watermark Security Policy***
  - A subject s can read (observe) an object o at any integrity level. The new integrity level of the subject s is the greatest lower bound of fs (s) and fo (o).
- ***Object Low Watermark Security Policy***
  - A subject s can modify (alter) an object o at any integrity level. The new integrity level of the subject s is the greatest lower bound of fs (s) and fo(o).

Biba Integrity Model properties: invoke and ring property

- ***Invoke Property***
  - A subject is only authorized to invoke subjects (tools) at lower integrity levels
  - Formally
    - A subject s1 can invoke a subject s2 if fs (s2) ≤ fs (s1)
- ***Ring property***
  - A subject s can read objects at any integrity level. It can only modify objects o with fo (o) ≤ fs (s); it can invoke a subject s’ only if fs (s) ≤ fs (s’)

Biba Implementation in Vista

- ***Vista marks files with an integrity level***
  - Low, Medium, High and System
  - Critical files are assigned System integrity level
  - Other objects are assigned Medium integrity level
  - Internet Explorer is assigned Low integrity level
- ***Vista implements the no write-up policy***
  - Files downloaded form IE can read most of the files in Vista file system but cannot write them
  - Limit the damage done by viruses and malwares

Clark Wilson Integrity Model

- ***MAC Model + Emphasis on integrity***
  - internal consistency:
    - properties of the internal state of a system
  - external consistency:
    - relation of the internal state of a system to the outside world
- ***Access permission based on***
  - the assignment of subjects to trusted programs
  - Execution of trusted programs that mantains consistency
- ***May be applicable to you***
  - Instrumentd Flights Programs

CWI - Mechanisms

- ***Well-formed transactions***
  - A user should only access data through trusted programs
- ***Separation of duty***
  - Any person permitted to create or certify a well-formed transaction should not be permitted to perform it

CWI - Components

- ***Constrained Data Items (CDIs)***
  - Data items subject to strict integrity controls
- ***Unconstrained Data Items (UDIs)***
  - Unchecked data items
- ***Transformation Procedures (TPs)***
  - System transactions that transforms CDIs from a consistent state to another
- ***Integrity Verification Procedures (IVPs)***
  - Check integrity of data items

CWI - Certification Rules

- ***IVPs must ensure that all CDIs are in a valid state at the time the IVPs is run***
- ***TPs must be certified to be valid***
  - Valid CDIs must always be transformed in valid CDIs
  - TPs must be certified to access a specific set of CDIs
- ***Access rules must satisfy any separation of duty requirement***
- ***All TPs must write to an append-only log***
- ***Any TPs taking a UDI as input must either convert it to a CDI or reject the UDI***

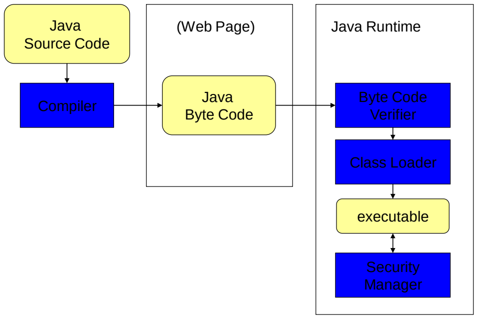
CWI - Enforcement Rules

- ***maintain and protect list of TPs and CDIs each TP is certified to access***
  - (TP1:CDIa1,CDIb1,...), (Tp2:CDIa2,CDIb2,...), (Tp3:CDIa3,CDIb3,...)
- ***system must maintain and protect the list of UserIDs and TPs each user can execute.***
  - (UId1TPa1,Tpa2,,Tpa3)
  - Maybe further refined by restricting also CDI on a per-user basis
- ***must authenticate each user wishing to execute a TP.***
- ***Only a subject that may certify an access rule for a TP may modify the respective entry in the list.***
  - This subject must not have execute rights on that TP

CWI - Credit Card Example

- ***Data (which is CDI, which is UDI?)***
  - Name, Surname – 18 UDI - rest CDI
  - Address All UDI - 4 CDI
  - Credit Card Number – All CDI
  - PIN Code - All CDI
  - Account Balance - All CDI
- ***Which is TP?***
  - Issue card (send card to customer’s address) – All TP
  - Issue PIN – All TP
  - Change Name -Only 2 TP
  - Change Address – No TP
  - Check credit history – Only 1 TP
  - Allow debit operation on CC number All TP
  - Load money on CC number All TP

The Java Execution Model



Java Source Code

Compiler

(Web Page)

Java Byte Code

Java Runtime

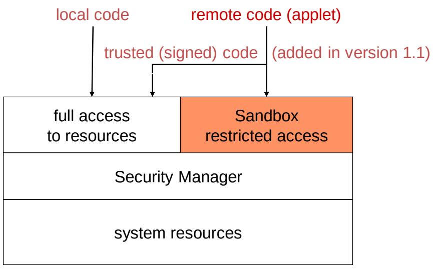
Byte Code Verifier

Class Loader

executable

Security Manager

JDK 1.1 Security Model



local code

remote code (applet)

trusted (signed) code

(added in version 1.1)

full access to resources

Sandbox restricted access

Security Manager

system resources

Discussion

- ***What kind of model is that?***

UNIVERSITY OF TRENTO - Italy

eit Digital MASTER SCHOOL

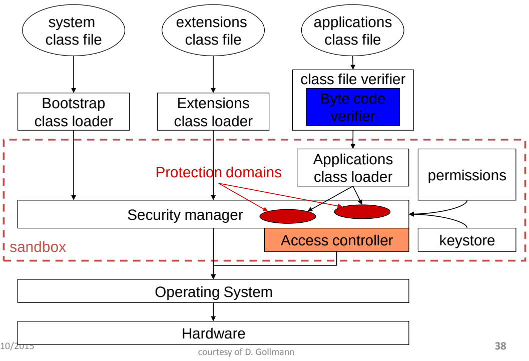
Limitations

- ***Local/remote is not a precise security indicator:***
  - Parts of the local file system could reside on other machines;
  - Downloaded software becomes "trusted" once it is cached or installed on the local system.
- ***Basic policy is quite inflexible:***
  - Local/signed code is unrestricted.
  - Applet/unsigned code is restricted to sandbox.
- ***No intermediate level:***
  - How to give some privileges to a home banking application?
- ***For more flexible security policies a customized security manager needed to be implemented.***
  - Requires security AND programming skills.

UNIVERSITY OF TRENTO - Italy

eit Digital MASTER SCHOOL

Java 2 Security Model



UNIVERSITY OF TRENTO - Italy

eit Digital MASTER SCHOOL

Terminology

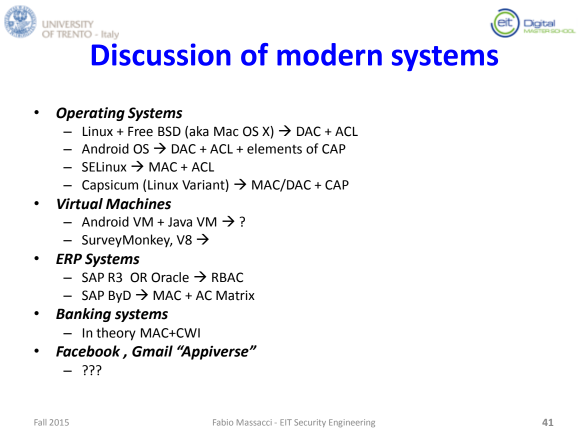
- ***Security Policy***
  - …mapping from a set of properties characterizing code, to a set of resource access permissions granted to the code…
- ***Protection Domain:***
  - …encapsulation of the code characteristics: location, signers and static permission granted to the code...

UNIVERSITY OF TRENTO - Italy

eit Digital MASTER SCHOOL

Code-based Access Control

- ***Security relevant parameters associated with code.***
  - Which parameters to use?
- ***Code source:***
  - URL (origin)
  - Digital certificates (code signers, if any)
- ***Principals: represent users or services***
- ***Protection domains: each class associated at load time with a protection domain.***
  - Contains: code source, principal, class loader reference, permission collection
- ***Question: is this really different from MAC+CAP?***



UNIVERSITY
OF TRENTO - Italy
eit Digital
MASTER SCHOOL

Discussion of modern systems

- ***Operating Systems***
  - Linux + Free BSD (aka Mac OS X) → DAC + ACL
  - Android OS → DAC + ACL + elements of CAP
  - SELinux → MAC + ACL
  - Capsicum (Linux Variant) → MAC/DAC + CAP
- ***Virtual Machines***
  - Android VM + Java VM → ?
  - SurveyMonkey, V8 →
- ***ERP Systems***
  - SAP R3 OR Oracle → RBAC
  - SAP ByD → MAC + AC Matrix
- ***Banking systems***
  - In theory MAC+CWI
- ***Facebook , Gmail "Appiverse"***
  - ???