



Access control and security policy models

T-110.402

Jan.Hlinovsky@hut.fi

What to protect and how?

- Confidentiality: only authorized entities can read
- Integrity: only authorized entities can change
- Availability: authorized entities can use

⇒ **Access control**



Access controls at different levels

- Access control can be implemented at many levels:
- Upper levels use the mechanisms of the level below
- Higher levels tend to have richer and more complex mechanisms, while lower level mechanisms tend to be more basic and reliable
- We concentrate now on the OS level

Applications
Services/Middleware
Operating system
Hardware



Subjects and Objects

- In computer security passive resources are called *objects* and active entities that utilize the resources are called *subjects*
- Typical objects include: files, directories, memory, printers, ...
- Typical subjects include: users, processes, ...
- The roles depend on situation: For example, a process can request access to some resource (act as a subject) and later be a target of access request (act as an object)
- Common practice in diagrams: subjects are depicted as circles and objects as squares



Access operations

- What subjects can do to objects?
- In Unix, typical file access types are *read*, *write*, *execute*
- The file owner can control the permissions to these operations
- Windows NT has, in addition to these, permissions for *delete*, *change ownership* and *change permissions*
- Some systems define *write* so that it includes reading rights. These systems often have one operation more, *append* (blind write)

Access control matrix

- Access rights can be defined in form of access control matrix, where rows correspond to subjects and columns to objects. The matrix entry (subject, object) then has the allowed access rights

- Example:

	doc_1	passwd	progr_1
Alice	rw	r	x
Bob	r	r	-
Ronald	rw	rw	rwX

- In real systems, however, access control matrices are not very practical :
 - the matrix is usually sparse and there is a lot of redundancy
 - new subjects and objects can be added or removed easily, but the centralized matrix could become a bottleneck

Access control lists (ACL)

- We take objects as the starting point
- ACL is a column of the access control matrix
- Typically stored with the object itself
- Example:
ACL for doc_1: {Alice: r,w; Bob: r; Ronald: r,w}
- Simple to implement, not so simple to manage
 - e.g. when a user is removed from the system, have to go through every ACL
- Unix file permissions can be seen as a simplified version of ACL

Capabilities

- Take subjects as the starting point
- Rows of access control matrix
- Example:
Alice's capability: {doc1: r,w; passwd: r; progr_1: x}
- Strengths:
 - Delegation of rights is easy
 - Runtime checking is more efficient compared to ACLs
- Weaknesses:
 - Difficult to find out which subjects have access rights to a specific object
 - Revoking delegated rights can be difficult
- Attribute certificates are capabilities for distributed systems ("holder of this key has the right to do x")



Other ways to grant access

- A *group* is a list of subjects with similar access rights. It is used as an intermediate access control layer
- A *role* is a collection of access permissions that a user or a set of users have in some context. Many systems implement roles as groups
- Some systems may use additional information for making access control decisions, e.g. what program the subject is using to access an object

Discretionary and mandatory access control

- Who decides the access rights?
- Discretionary access control (DAC): the owner of a resource decides
- Mandatory access control (MAC): the administration decides (based on the policy in use)
- Most widely-used operating systems have discretionary access control
- Note that the Orange Book uses the terminology differently, there DAC and MAC are specific policies



Multilevel security (MLS)

- Security levels
 - a hierarchy of sensitivity attributes (ordering of levels)
 - typical military-style hierarchy:
- An object's sensitivity attribute is called *classification*
- Subjects have *clearances* to access objects in the hierarchy
- *dominates*-relation: we say that x *dominates* y iff $\text{level}(x) \geq \text{level}(y)$
- *allow*-function:
allow: subjects \times objects \times access type \rightarrow boolean

Top secret
Secret
Confidential
Open/Unclassified

Security policy models

- Here *security policy* refers to a computer system's *internal* security policy, **not** organization-level security policy
- You can think of these as "access control policy models"
- Idea of security models is to leave irrelevant details out and concentrate on some interesting property
- Formal models can be used to prove that a system preserves a wished property (e.g. confidentiality)
- Typically the models enforce mandatory access control

Central concepts

- *Reference monitor* is an abstract system part that mediates and controls access requests (kind of an "access request filter")
- The implementation (HW & SW) of reference monitor is called *security kernel*
- *Trusted computing base* (TCB) is the set of components (HW, SW etc) that is responsible for enforcing the security policy. This includes the security kernel and other protection mechanisms.

Bell-LaPadula model (BLP)

- David Bell and Leonard LaPadula 1973
- Concentrates on confidentiality
- Two rules: NRU and NWD
- No read up (NRU) or the simple security property:
 $\forall s \in \text{subjects}, o \in \text{objects}: \text{allow}(s, o, \text{read})$ iff s dominates o
(a subject can not read objects whose classification is higher than the subject's clearance)
- No write down (NWD) or the *-property:
 $\forall s \in \text{subjects}, o \in \text{objects}: \text{allow}(s, o, \text{write})$ iff o dominates s
(a subject can not write to objects whose classification is lower than the subject's clearance)

BLP continued

- The NWD-rule is the big innovation here
 - this way, if a Trojan is executed at high level it can not write to lower levels
- Sounds secure all right, but...
isn't it terribly inflexible if the high level subjects can not write anything to lower levels?
- Yes. There are two ways to get round this



BLP continued

- One possibility: temporarily lower the "working classification" of subjects
 - to avoid problems, we define *tranquility property*:
 - *weak tranquility*: "security labels of subjects and objects never change in such a way as to violate a defined security policy"
 - *strong tranquility*: "labels never change during system operation"
- Another solution: introduce *trusted subjects*, i.e. subjects that are allowed to break the NWD-rule



High water mark principle

- If two documents are combined, the result is classified (at least) at the level of the higher of the two
- If a user at level *secret* modifies an *unclassified* document the result is (at least) *secret*
- As a result, classifications tend to rise up.

BLP problems and criticism

- BLP is only concerned about confidentiality (but this is a design decision)
- E.g. blind write-ups are a threat to integrity (which is why many practical implementations allow writing only to objects at same level)
- Not very well suited for distributed systems
- Management is outside the system (e.g. creating and deleting files)



Covert channels

- Mechanisms that are not intended for communication but can be used to leak information from High to Low.
- For example the status (busy or free) of some shared device at some specific moment can leak one bit of information (if the High user can affect the device)
- Covert channels can be reduced but it's very difficult to avoid them completely
- Related concept: *Subliminal channels* are similar information leak mechanisms in cryptographic protocols



Biba model

- Ken Biba 1975
- concentrates on integrity
- "BLP upside down"
- Two rules: NRD and NWU
- No read down (NRD):
 $\forall s \in \text{subjects}, o \in \text{objects}: \text{allow}(s, o, \text{read}) \text{ iff } o \text{ dominates } s$
- No write up (NWU):
 $\forall s \in \text{subjects}, o \in \text{objects}: \text{allow}(s, o, \text{write}) \text{ iff } s \text{ dominates } o$
- Same logic is used in many practical systems

Low water mark models

- Dynamic versions of Biba model
- Subject low water mark: subjects may read down, but the subject's integrity level is also downgraded to the level of the object as a result
- Object low water mark: subjects may write up, but as a result the object's integrity level is downgraded to the level of the subject



Biba model problems and criticism

- Tendency of integrity classifications to go down, never up
- Many of the problems in BLP apply to Biba, too. E.g. management is outside the model
- What does it mean to have different levels of integrity? (Can information be "just a little incorrect"?)



Clark-Wilson model (CW)

- David Clark and David Wilson 1987
- Integrity model based on well known, time-tested accounting practices
- Data is categorized to two levels:
 - Constrained data items (CDI)
 - Unconstrained data items (UDI)
- CDIs can only be accessed through special *transformation procedures* (TP)
- The integrity of CDIs is checked with *integrity verification procedures* (IVP)

Clark-Wilson continued

- Not a formal model, but a collection of rules such as:
 - subjects have to be authenticated
 - TPs have to preserve integrity
 - there must be a separation of duty policy
- Not a MLS model, but can be combined e.g. with Biba
- CW drew attention to different kinds of security models, not just military-style MLS
- Problems:
 - implementing IVPs and TPs for complex systems is difficult
 - internal integrity of a transaction does not guarantee that the transaction was correct (but there is auditing)

Multilateral security

- Subjects and objects are classified into categories instead of levels
- Also known as compartmentation
- Example: Customers of an Internet bank can not see each others' data nor can they make their data visible to others (not even accidentally)
- More examples: medical patients, company departments, ...

client a	client b	client c	client d
-------------	-------------	-------------	-------------

Chinese Wall model

- David Brewer and Michael Nash 1989
- Rules to prevent conflict of interest
- Basically a formal description of normal commercial non-disclosure practice
- Idea is that a professional worker should not deal with different clients who are competitors with each other so that he can not be accused of using "inside knowledge"



Chinese Wall continued

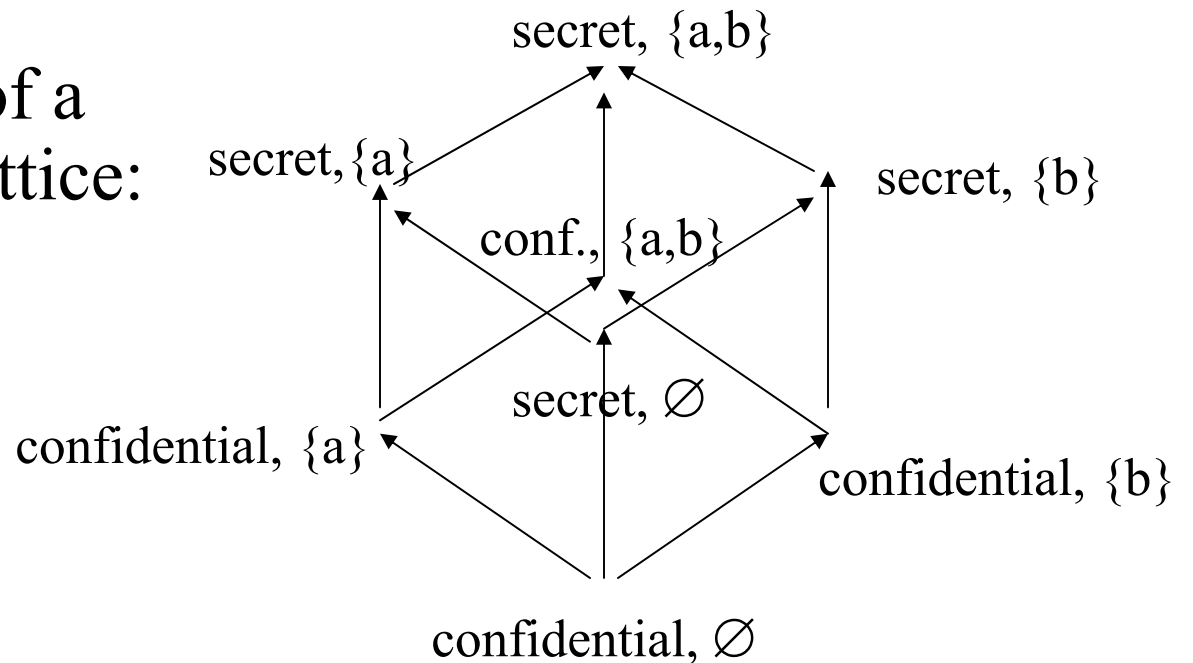
- Like BLP, Chinese Wall model has two access control rules that can be expressed formally
- Informally, the rules can be stated as:
 - the simple security property:
a subject s can access company c 's data only if
 - a) s has already accessed c 's data
 - or
 - b) s has not accessed any of c 's competitors' data
 - the *-property:
 s can write to c 's data only if s can not read any other company's sensitive data

Lattices

- A lattice is a mathematical construction with:
 - set of elements
 - a partial ordering relation
 - the property that any two elements must have unique least upper bound and greatest lower bound
- A security lattice model combines multilevel and multilateral security
- Lattice elements are security labels that consist of a security level and set of categories
- For the partial ordering relation we redefine the *dominates* relation a little bit:
 $\forall x, y \in \text{labels}: x \text{ dominates } y \text{ iff } \text{level}(x) \geq \text{level}(y) \text{ and } \text{categories}(x) \supseteq \text{categories}(y)$

Lattices continued

- Bell-LaPadula can be used on lattices, with the redefined *dominates*-relation
- Lattices are used to implement need-to-know principle
- Example of a security lattice:



Combining different models

- Combinations of models can be useful, but they must be made carefully
- For example when combining BLP and Biba there is a decision: use the same security labels for both model rules or not?
 - If yes, then special care has to be paid to avoid a system where access is allowed only to same level objects (unless you *really* know that you want that)



Summary of models

- There are different kinds of models, some are based on "military" thinking (such as BLP), others on commercial policies (like Clark-Wilson and Chinese Wall), and so on
- Multilevel security models concern information flow in hierarchical systems, multilateral security models concern information flow between departments
- Different models can be combined to get the properties needed
- Formal models are used in security evaluation (e.g. TCSEC level B)

References

- Following books have been used when making these slides:
 - Edward Amoroso: Fundamentals of Computer Security Technology. Prentice Hall, 1994.
 - Ross Anderson: Security Engineering. Wiley, 2001
 - Dieter Gollmann: Computer Security. Wiley, 1999