CS3235
Fifth set of lecture slides

Hugh Anderson

National University of Singapore
School of Computing

September, 2006
I've had a mental breakthrough on that problem I've been working on...

---

—not a complete solution ... more a fundamental shift in emphasis...

---

I've realised that someone, somewhere, is bound to know the answer to my problem!

---

My new problem is where to locate that someone...?
Outline

1. Administration

2. Security models
   - Access control matrix
   - Confinement

3. Formal models
   - BLP model - confidentiality
   - Biba model - integrity
   - Clark-Wilson
Assignment!!

If you are one of the (3) students who have not got organised:

- Please do so ASAP!

Perhaps you could meet outside in the foyer after this lecture.

Mid-semester test!

Will be on … 5th October

- MCQ, about 30 questions, 45 minutes, closed book
The sciences do not try to explain, they hardly even try to interpret, they mainly make models. [J. von Neumann]

**Definition:** *a range of formal policies for specifying the security of a system in terms of a (mathematical) model.*

- access control matrix
- Bell-LaPadula
- Biba
- Clark-Wilson
Security model approach

A three step approach

- Have a model
- Determine properties
- Verify implementations
## Access control matrix

Access control matrices

**Rows** of the matrix are subjects, **columns** are objects:

<table>
<thead>
<tr>
<th>Subjects</th>
<th>Objects</th>
<th>▪</th>
<th>▪</th>
<th>▪</th>
</tr>
</thead>
<tbody>
<tr>
<td>s₁</td>
<td>f₁</td>
<td>read</td>
<td>exec</td>
<td></td>
</tr>
<tr>
<td>s₂</td>
<td>f₂</td>
<td>write</td>
<td></td>
<td>read</td>
</tr>
<tr>
<td>s₃</td>
<td>f₃</td>
<td>read</td>
<td>write</td>
<td></td>
</tr>
<tr>
<td>s₄</td>
<td>f₄</td>
<td>read</td>
<td>write</td>
<td></td>
</tr>
</tbody>
</table>

s₄ cannot read f₁. But subjects may collude... (Note transitivity and so on)...
Example Access Control systems

Two categories

1. **DAC - Discretionary Access Control**
   - 1. Owner decides policy
   - 2. Unix file system - rwx permissions *(chmod)*

2. **MAC - Mandatory Access Control**
   - 1. Programs work within constraints
   - 2. Access rules provided by security policy, control by system administrator - *selinux* on FC4.
Confinement and covert channels

Secret channels for leaking information

- The confinement problem is one of preventing a system from leaking (possibly partial) information.

Sometimes a system can have an unexpected path of transmission of data, termed a covert channel, and through the use of this covert channel information may be leaked either by a malicious program, or by accident.
Example - modifying permissions on a file

- Consider the set of permissions on a file. [view]

An unscrupulous program could modify these permissions cyclically to transmit a very-low data-rate message to another unscrupulous program.
Confinement and covert channels

Classification of covert channels
We categorize covert channels into two:

1. **Storage channels:** using the presence or absence of objects
2. **Timing channels:** the speed of events

We can attempt to identify covert channels by building a **shared resource matrix**, determining which processes can read and write which resources.
Attacks on databases

Example of releasing unexpected information from databases
Governing body may keep secret *individual* information, but release *cumulative* information
For example: Today’s average temperature of SOC staff by nationality:

<table>
<thead>
<tr>
<th>Singaporean</th>
<th>PRC</th>
<th>Poland</th>
<th>German</th>
<th>Australian</th>
<th>New Zealand</th>
<th>....</th>
</tr>
</thead>
<tbody>
<tr>
<td>36.8</td>
<td>36.9</td>
<td>37.1</td>
<td>36.5</td>
<td>38.2</td>
<td>38.1</td>
<td>....</td>
</tr>
</tbody>
</table>
Attacks on databases

No sensitive information ... but ...

- what if another part of the database released the numbers of SOC staff by nationality...

<table>
<thead>
<tr>
<th>Singaporean</th>
<th>PRC</th>
<th>Poland</th>
<th>German</th>
<th>Australian</th>
<th>New Zealand</th>
<th>....</th>
</tr>
</thead>
<tbody>
<tr>
<td>23</td>
<td>14</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>1</td>
<td>....</td>
</tr>
</tbody>
</table>

By inference you can deduce that Hugh’s temperature is too high!
Confinement, covert channels

Specifying properties formally

- By tabulating the types of data in a system, and the properties of the operations (read, write, execute, transitive), it may be possible to specify that the system cannot leak information or be used to transfer information.

This is the goal of MAC systems.
Models

Two categories of most interest

1. Confidentiality - documents classified, and only the correct people (processes) can access them.
2. Integrity - documents protected, and the trustworthiness of the documents ensured.
Bell-LaPadula, confidentiality

BLP from the names of the two authors of [BL75]

- Military style to assure confidentiality services.
- Security levels in a (total) ordering formalizing a policy which restricts information flow from a higher security level to a lower security level.
- Lower-level subjects from accessing higher-level objects.
Bell-LaPadula, levels

BLP has four levels of security

1. Top secret ($T$)
2. Secret ($S$)
3. Confidential ($C$)
4. Unclassified ($U$)

where $T > S > C > U$. Access operations visualized using an access control matrix, and are drawn from \{read, write\}.
Import of the properties

We can view them as the activities in a secure building

More confidential

Less confidential
BLP security property

Mathematical treatment

The clearance classification for a subject $s \in S$ or object $o \in O$ is denoted $L(s) = l_s$ or $L(o) = l_o$. We might then assume we can use this to construct a first simple security property:

- **No read-up-1**: $s$ can read $o$ if and only if $l_o \leq l_s$, and $s$ has read access in the access control matrix.

This single property is insufficient to ensure the restriction we need for the security policy.
BLP - no read up

What does no-read-up mean?

Fred with Secret clearance
l(fred)=S

Subjects (people, programs)

Objects (files)

Top secret
Secret
Confidential
Unclassified
BLP Trojan Horse property

Trojan horses could still happen, so need another rule. Consider the case when a low security subject creates a high security object (say a program) which then reads a high security file, copying it to a low security one. This behaviour is commonly called a Trojan Horse. A second property is needed:

- **No write-down-1**: $s$ can write $o$ if and only if $l_s \leq l_o$, and $s$ has write access in the access control matrix.

These two properties can be used to enforce our security policy, but with a severe restriction. For example, how does any subject write down without invalidating a security policy?
BLP - no write down

What does no-write-down mean?

Fred with Secret clearance
l(fred)=S
Extend model with categories

A security category \( c \in C \) is used to classify objects in the model, with any object belonging to a set of categories. Each pair \((l \times c)\) is termed a *security level*, and forms a *lattice*.
The domination relation

We define a relation between security levels:

- The security level \((l, c)\) dominates \((l', c')\) (written \((l, c) \text{ dom} (l', c')\)) iff \(l' \leq l\), and \(c' \subseteq c\).

A subject \(s\) and object \(o\) then belong to one of these security levels.
BLP extended

Properties for the new extended model

The new properties are:

- **No read-up-2**: \( s \) can read \( o \) if and only if \( s \ text{dom} o \), and \( s \) has read access in the access control matrix.

- **No write-down-2**: \( s \) can write \( o \) if and only if \( o \ text{dom} s \), and \( s \) has write access in the access control matrix.
BLP security

The security theorem

- A system is considered **secure** in the current state if all the current accesses are permitted by the two properties.
- A **transition** from one state to the next is considered secure if it goes from one secure state to another secure state.
- The basic **security theorem** states that if the initial state of a system is secure, and if all state transitions are secure, then the system will always be secure.
BLP limits

Note the limitations of this system

BLP is a static model, not providing techniques for changing access rights or security levels\(^a\).

However the model does demonstrate initial ideas into how to model, and how to build security systems that are provably secure.

\(^a\)You might want to explore the Harrison-Ruzo-Ullman model for this capability.
Biba model, integrity

A different kind of assurance

- **Trustworthiness** of data and programs - assurance for *integrity* services.
- Levels like clean or dirty (in reference to database entries).
- Biba model is a kind of *dual* for Bell-LaPadula. *integrity* vs *confidentiality*. 
A similar approach, only integrity instead of confidentiality

- The integrity levels $\mathcal{I}$ are ordered as for the security levels
- Function $i : O \rightarrow \mathcal{I}$ ($i : S \rightarrow \mathcal{I}$) which returns the integrity level of an object (subject).
Biba properties

Mathematical rules

The properties/rules for the *main* (static) Biba model are:

- **No read-down**: $s$ can read $o$ iff $i(s) \leq i(o)$.
- **No write-up**: $s$ can write $o$ iff $i(o) \leq i(s)$.
- **No invoke-up**: $s_1$ can execute $s_2$ iff $i(s_2) \leq i(s_1)$.
Biba - dynamic

Dynamic rules

Biba models can also handle dynamic integrity levels, where the level of a subject reduces if it accesses an object at a lower level (in other words it has got dirty). The low-watermark policies are:

- **No write-up**: \( s \) can write \( o \) iff \( i(o) \leq i(s) \).
- **Subject lowers**: if \( s \) reads \( o \) then \( i'(s) = \min(i(s), i(o)) \).
- **No invoke-up**: \( s_1 \) can execute \( s_2 \) iff \( i(s_2) \leq i(s_1) \).
Biba - ring

Further rules

Finally, we have a ring policy,

- **All read**: $s$ can read $o$ regardless.
- **No write-up**: $s$ can write $o$ if and only if $i(o) \leq i(s)$.
- **No invoke-up**: $s_1$ can execute $s_2$ if and only if $i(s_2) \leq i(s_1)$.

Each of these policies have an application in some area.
Clark-Wilson, integrity

The Clark-Wilson model has the following terminology:

**Transactions** defined through **certification** rules.

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDI</td>
<td>Constrained Data Item (data subject to control)</td>
</tr>
<tr>
<td>UDI</td>
<td>Unconstrained Data Item (data not subject to control)</td>
</tr>
<tr>
<td>IVP</td>
<td>Integrity Verification Procedures (for testing correct CDIs)</td>
</tr>
<tr>
<td>TP</td>
<td>Transformation Procedures (for transforming the system)</td>
</tr>
</tbody>
</table>