

Assignment 2:

Access Control

Information and Communications
Security (SS 2011)

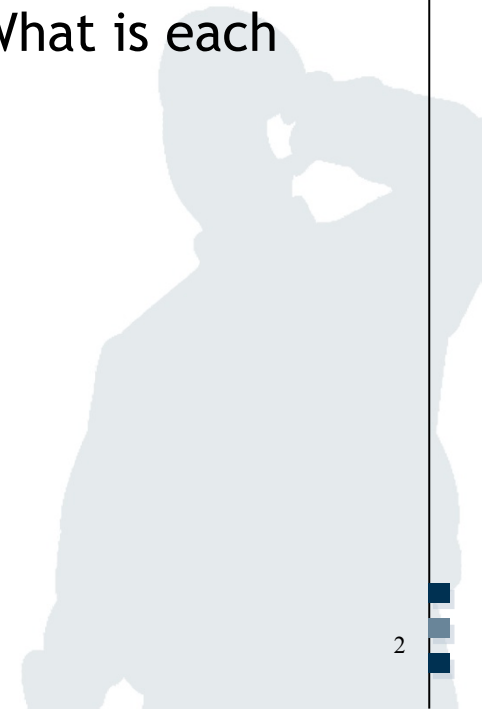
Prof. Dr. Kai Rannenber

M.Sc. Ahmad Sabouri

T-Mobile Chair for
Mobile Business & Multilateral Security
Johann Wolfgang Goethe University Frankfurt a. M.
www.m-chair.net



- Alice can read and write to the file x, can read the file y, and can execute the file z. Bob can read x, can read and write to y, and cannot access z.
 - Write a set of access control lists for this situation. Which list is associated with which file?
 - Write a set of capability lists for this situation. What is each list associated with?



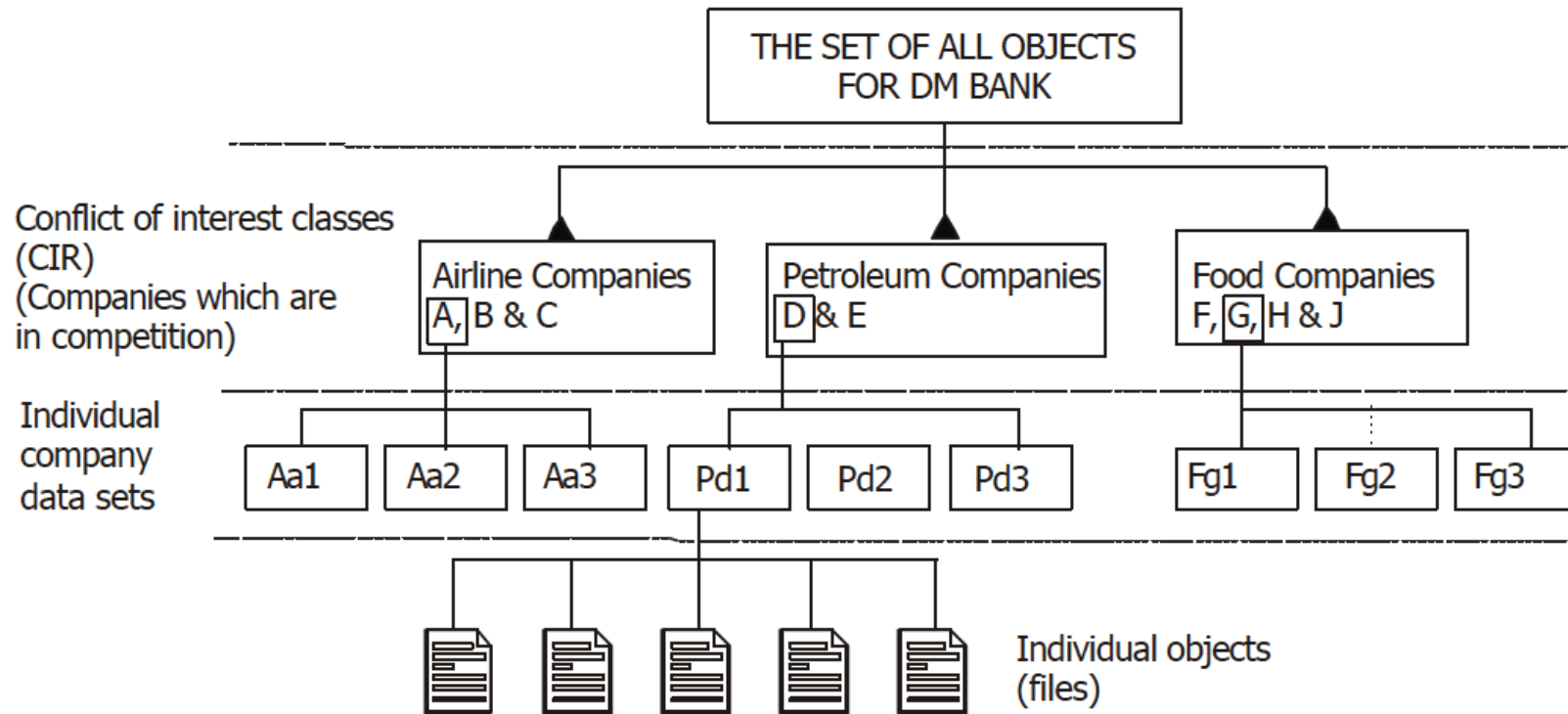
- x Alice: read, write; Bob: read
- y Alice: read; Bob: read, write
- z Alice: execute
- Each list is associated with the object



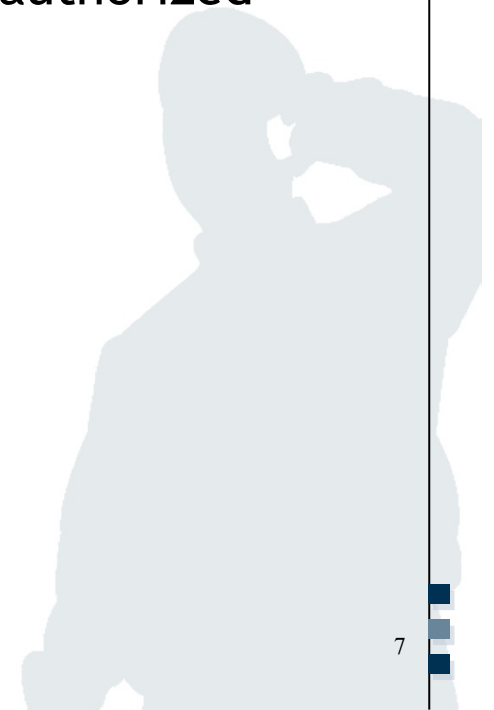
- Alice's capabilities:
 - x: read, write;
 - y: read;
 - z: execute
- Bob's capabilities:
 - x: read;
 - y: read, write;



- Name a fundamental difference between the security model of Bell-LaPadula and the Chinese Wall Model. What are the additional security assets that the Chinese Wall Model is supposed to refer to?
 - The critical difference from Bell-LaPadula is that the Chinese Wall Model needs to retain state in order to keep track of the objects that someone accessed. Initially someone is free to access all objects in the Chinese Wall Model, but then, the more he accesses, the more constrained he becomes. The BLP Model constrains from the beginning the set of objects that someone can access.
 - Bell-LaPadula Model refers only to confidentiality: who can read a message
 - The Chinese Wall (CW) model is a model of a security policy that refers equally to confidentiality: who can read a message
 - and *integrity*: who gets permission to modify a data set.



- The concept of roles has been introduced to you during the last lectures. Please describe this concept briefly.
 - A role is a collection of job functions. Each role r is authorized to perform one or more transactions. The set of authorized transactions for r is written $trans(r)$.



- Please assume that you are working for the University of Frankfurt as a research assistant after having received your diploma. Please identify four different roles that you might be authorised to assume. Please note for each identified role one or two problems that might be encountered by a role-based access control model.

- The nature of roles is static, so RBAC lacks flexibility and responsiveness to the environment in which they are used.
- Secondly, RBAC does not encompass the overall context associated with any collaborative activity, so it is a passive security system that serves the function of maintaining permission assignments.
- Thirdly, RBAC lacks the ability to specify a fine-grained control on individual users in certain roles and on individual object instances, so it is not enough for collaborative environments.

- Revocation of an individual's access to a particular file is easy when an access control list is used. How hard is it to revoke a user's access to a particular set of files, but not all files? Compare and contrast this with the problem of revocation using capabilities (capability lists, c-lists).

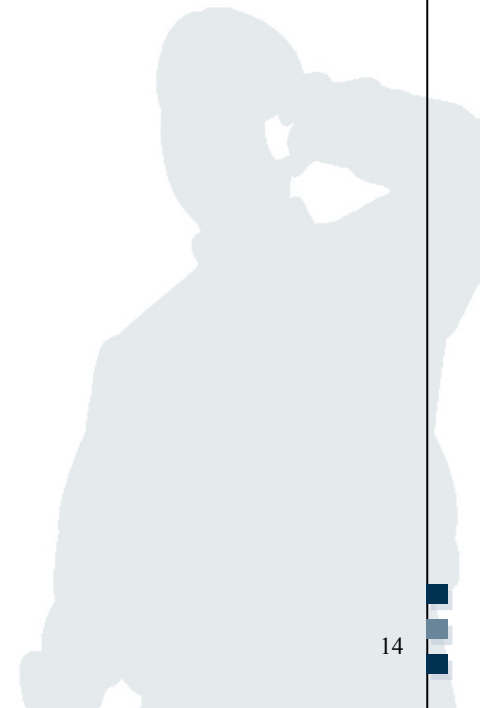


- One would have to enumerate through the ACL for each file in the set and delete the user from that file's ACL. To compare with C-lists, one would get the C-list for the user and delete all the files from the set from that C-list. ACLs require accessing and updating multiple ACLs (one per file) whereas C-lists require accessing and updating one C-list (the user's).

- Given the security levels TOP SECRET, SECRET, CONFIDENTIAL, and UNCLASSIFIED (ordered from highest to lowest), and the categories A, B, and C, specify what type of access (read, write, both, or neither) is allowed in each of the following situations. Assume that discretionary access controls allow anyone access unless otherwise specified.

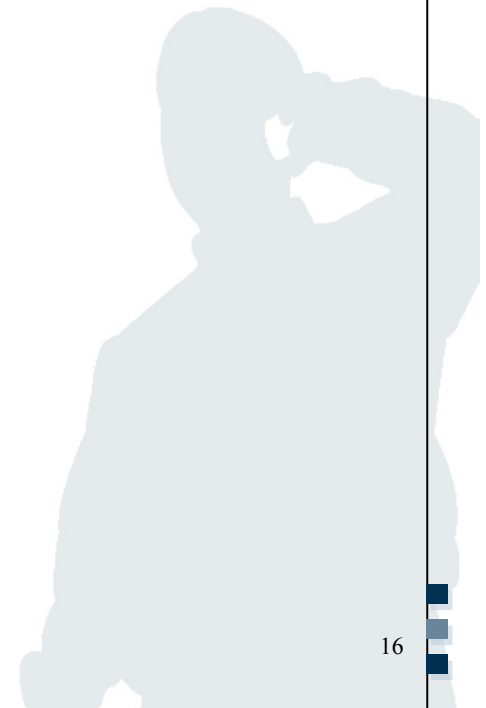
- Let A 's compartment be (L_A, C_A) and B 's be (L_B, C_B) . The simple security condition says that A can read B if and only if $L_A \geq L_B$ and $C_B \subseteq C_A$.
- The *-property says that A can write B if and only if $L_B \geq L_A$ and $C_A \subseteq C_B$.
- Remember that
TOPSECRET \geq SECRET \geq CONFIDENTIAL \geq UNCLASSIFIED.

- a) Paul, cleared for (TOP SECRET, {A, C}), wants to access a document classified (SECRET, {B, C}).
- $L_{\text{Paul}} = \text{TOPSECRET} \geq \text{SECRET} = L_{\text{doc}}$,
so Paul cannot write the document.
Paul cannot read the document either, because
 $C_{\text{doc}} = \{B, C\} \not\subseteq \{A, C\} = C_{\text{Paul}}$.



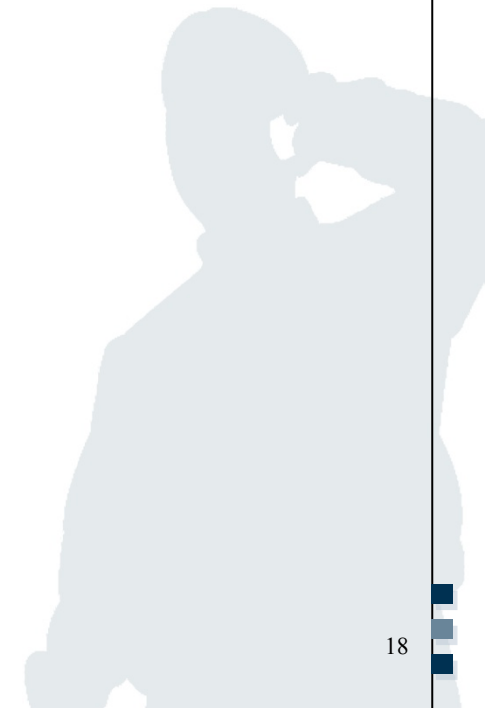
- b) Anna, cleared for (CONFIDENTIAL, {C}), wants to access a document classified (CONFIDENTIAL, {B}).
- $L_{\text{Anna}} = \text{CONFIDENTIAL} \geq \text{CONFIDENTIAL} = L_{\text{doc}}$, but
 $C_{\text{doc}} = \{B\} \not\subseteq \{C\} = C_{\text{anna}}$
so Anna cannot read the document, and
 $C_{\text{Anna}} = \{C\} \not\subseteq \{B\} = C_{\text{doc}}$,
so Anna cannot write the document.

- c) Jesse, cleared for (SECRET, {C}), wants to access a document classified (CONFIDENTIAL, {C}).
- $L_{\text{Jesse}} = \text{SECRET} \geq \text{CONFIDENTIAL} = L_{\text{doc}}$, and
 $C_{\text{doc}} = \{C\} \subseteq \{C\} = C_{\text{Jesse}}$,
so Jesse can read the document.
Since $L_{\text{Jesse}} > L_{\text{doc}}$,
Jesse cannot write the document.



- d) Sammi, cleared for (TOP SECRET, {A, C}), wants to access a document classified (CONFIDENTIAL, {A}).
- As $L_{\text{Sammi}} = \text{TOPSECRET} \geq \text{CONFIDENTIAL} = L_{\text{doc}}$ and $C_{\text{doc}} = \{A\} \subseteq \{A, C\} = C_{\text{Sammi}}$,
Sammi can read the document.
But the first inequality means Sammi cannot write the document.

- e) Robin, who has no clearances (and so works at the UNCLASSIFIED level), wants to access a document classified (CONFIDENTIAL, {B}).
- As $C_{\text{Robin}} = \Phi \subseteq \{B\} = C_{\text{doc}}$ and $L_{\text{doc}} = \text{CONFIDENTIAL} \geq \text{UNCLASSIFIED} = L_{\text{Robin}}$, Robin can write the document. However, because $L_{\text{doc}} \geq L_{\text{Robin}}$, she cannot read the document.



- Consider an access control method that wants to allow an object to have more than one owner. Explain how you would implement this with both ACLs and capabilities.



- The main goal here is to only allow access to the object if all owners of the object agree to give access.
- For capabilities, the access is defined by subject. Instead of the `{object, {read, write, execute}}` permissions for example, there will need to be some way to show which owner granted that permission. Say the object had two owners, Bob and Alice. Then the subject would need to have `{object, {read from Alice, read from Bob}}` to be able to read the object.
- For ACLs, the access is defined by object. Again, it will need to be expanded to track which owner has granted a permission. Instead of `{subject, {read}}`, it would need `{subject, {read from Alice, read from Bob}}` for the subject to be able to read the object.