**CS 370 Introduction to Security** **Week 6: Problem Set 6**
Instructor Name: Rakesh Bobba

# Introduction / Purpose

The purpose of this assignment is to help you gain a better understanding and insight into role-based and mandatory access control models covered in Week 6.

Before beginning make sure you have watched the lecture videos on the following and completed the associated practice quizzes.

- Introduction to Role-based Access Control
- Role-Based Access Control Models
- Role Engineering
- Introduction to Mandatory Access Control, Bell-LaPadula (BLP) Model
- Biba Integrity Model
- Chinese Wall Model

Please make sure you read Chapter 8 and Chapter 9 till 9.2.2 from text book

# Instructions/Questions

Please answer the questions below.

## Access Control Concepts

Q1 [2 pts]: What is the difference between a “role” in RBAC and a “group” commonly used in UNIX?

- A role is a collection of permissions. This is contrasted with a group, which is a collection of users.

Q2 [3 pts]: What is separation-of-duty? And what is the difference between static separation-of-duty (SSD) and dynamic separation-of-duty (DSD)

- Separation of Duty is the act of breaking down critical operations such that they require two or more people to complete.
- SSD is the process of using $RBAC_2$ constraints to ensure that two role memberships end up mutually exclusive. This is done by setting a maximal cardinality for a certain user and role set, so that they cannot assume too many (thus conflicting) roles at the same time.
- DSD is another way of resolving conflicts of interest by only letting a user assume n roles of their available role set in a session.

## Role-Based Access Control

Q3 [8 pts]: Consider the following scenario. An organization employs product managers, programmers and testers. The organization operates with the following kinds of files:

development code and executables, testing code and executables, test reports, and production code and executables.

> Product Managers can view and execute the development executables and production executables to verify correctness. Programmers can create, edit, delete, and execute development code and executables.
>
> Programmers can also promote development code to the test level.
>
> Testers can edit, delete, and execute test code and executables. The testers write test reports that can be read by everyone. The testers can promote test code to production level or demote it back to development.
>
> Everyone can view and execute production code and executables.
>
> Eve is the product manager, Alice and Bob are programmers. Carol and Dave are testers

Would the access control for the scenario above benefit from being implemented in a RBAC system? If yes, explain why and create access matrices that define an RBAC that would enforce this scenario? If not, describe why not and present another scenario that would be better defined as an RBAC system rather than a straight DAC.

- Yes, the above would slightly benefit from RBAC, as there are multiple subjects who are preforming the same jobs or roles (i.e. Alice and Bob are both programmers, so they would both be assigned to the programmer role).

Role Assignments:

| | Prod. Mgr | Programmer | Tester |
|---|---|---|---|
| Eve | X | | |
| Alice | | X | |
| Bob | | X | |
| Carol | | | X |
| Dave | | | X |

Development code = DC
Development executables = DX
Testing code = TC
Testing executables = TX
Test reports = TR
Production code = PC
Production executables = PX

Permission to Role Mapping

| Roles | | | | Files | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | PM | Program. | Tester | DC | DX | TC | TX | TR | PC | PX |
| PM | control | | | | [1] Read execute | | | [5] Read | [7] read execute | [1], [7] Read execute |
| Program | | control | | [2] Write exec. Delete read | [2] Write exec. Delete read | [3] write | | [5] Read | [7] read execute | [7] read execute |
| Tester | | | control | [6] write | | [4] Write exec. Delete read | [4] Write exec. Delete read | [5] Write Delete read | [6] write [7] read execute | [7] read execute |

[1] Product Managers can view and execute the development executables and production executables to verify correctness.
[2] Programmers can create, edit, delete, and execute development code and executables.
[3] Programmers can also promote development code to the test level.
[4] Testers can edit, delete, and execute test code and executables.
[5] The testers write test reports that can be read by everyone.
[6] The testers can promote test code to production level or demote it back to development.
[7] Everyone can view and execute production code and executables.

Q4 [7 pts]: A company has 20 job functions. On average there are 200 employees in each job function. Similarly, on average an employee in each job function needs 1500 permissions to properly execute their task. Compare the number of assignments that need to be managed i) when using a DAC model vs. ii) when using RBAC model. Generalize the comparison to when the number of job functions is N, number of employees per job function is $U_i$, where i indexes the job-function, and the number of permissions required per job function is $P_i$.

- DAC Model: each user's permissions are managed separately, so you must assign 1500 permissions to 20*200 users. This means 1500*20*200 = 6,000,000 assignments. The formula for this is: sum(i, 0, N, $U_i$ * $P_i$), where I is iterated over the range 0..N.
- RBAC Model: permissions are managed by group, reducing the number of permission assignments needed drastically. The number of assignments is 20*1500 = 30,000 as the 1500 permissions need to be only applied to the 20 roles. The formula for this is: sum(i, 0, N, $P_i$), where I is iterated over the range 0..N.

## Mandatory Access Control Models

Q5 [4 pts]: What is *-property in BLP confidentiality model and why is it needed?

- The Star Policy of BLP is the policy that states that a user may only write to documents above or equal to their current clearance. This, when combined with the SSP results in the user only being able to read and write documents that are of an equal security level to theirs.

Q6 [4 pts]: Compare and contrast BLP and Biba models.

- BLP Model of MAC emphasizes confidentiality, and protects reads and writes on it's objects. This is contrasted with BIBA model of MAC, where the primary (and only) concern is about integrity of object data or execution. Despite these differences, they are both subgenres of the MAC. They also differ as BLP prevents read up and write down, whereas BIBA restricts the user to no write up and no read down.

Q7 [2 pts]: What is the difference between a security level and an integrity level?

- An integrity level concerns how much confidence a user has in the program or file's accuracy, reliability or (in the case of a program) proper execution. A security level is a tier of access, and is also the simplest security class. These tiers are arranged in a completely ordered list, and represent the hierarchy of information 'sensitivity'.

Q8 [3 pts]: How is Chinese Wall model different from BLP and Biba?

- Chinese wall, as opposed to BIBA and BLP, is focused on preventing conflicts of interest from occurring. This differs from BLP, where the goal is to preserve information's secrecy, and also differs BIBA, where the only goal is to preserve integrity. Chinese wall also differs in how it breaks down the information it has (i.e. objects).

Q9 [6 pts]: When using DAC under MAC in BLP:

a) Does a user get access to an object if MAC policy doesn't permit it? Explain why or why not.
   i) No – the MAC checks (specifically star policy and simple security policy) must pass before the DAC is even considered, and in this case, the check would not reach the DAC.

b) Does a user get access to an object if DAC policy doesn't permit it? Why or why not.
   i) No – there's no reason to add DAC to the MAC system if this were the case. If a user could get access if only the MAC passes, and DAC fails, then there is no use having the additional check, as the user would only need to fulfill the MAC requiremnents, and DAC could be ignored.

Q10 [8 pts]: The table below lists subjects, objects, and their associated security levels. The relationship between the levels is as follows: purple > green > orange

| Subject | Subject Clearance | Object | Object Classification |
|---|---|---|---|
| Alice | Green | Yoyo | Purple |
| Bob | Purple | XRay | Green |
| Carol | Orange | Zebra | Green |

a) Compute whether the specified subject has read or append (i.e., write but not necessarily read) access to the specified object (see table below) following the Bell LaPadula model.

BLP: No read up, no write down

| Subject | Object | Rights |
|---|---|---|
| Alice | XRay | Read, Write |
| Bob | Zebra | Read |
| Carol | Yoyo | Append |
| Carol | Zebra | Append |

b) The security labels are updated to include project categories, p1, p2, and p3. The updated labels are shown in the table below. Re-evaluate the rights (read or append) associated with each subject and object pair following the Bell LaPadula model.

| Subject | Subject Clearance | Object | Object Classification |
|---|---|---|---|
| Alice | Green:{p1,p2} | Yoyo | Purple:{p1} |
| Bob | Purple:{p2} | XRay | Green:{p1, p2} |
| Carol | Orange: {p1, p3} | Zebra | Green: {p3} |

| Subject | Object | Rights |
|---|---|---|
| Alice<br>Green:{p1,p2} | XRay<br>Green:{p1, p2} | Read, Write |
| Bob<br>Purple:{p2} | Zebra<br>Green: {p3} | No Read, No Write<br>Read: p3 !subset p2<br>Write p2 !subset p3 |
| Carol<br>Orange: {p1, p3} | Yoyo<br>Purple:{p1} | No Read, No Write<br>Read: Purple !<= Orange<br>Write:{p1, p3} !sub {p1} |
| Carol<br>Orange: {p1, p3} | Zebra<br>Green: {p3} | No Read, No Write<br>Read: Purple !<= Orange<br>Write:{p1, p3} !sub {p3} |

Q11 [8 pts]: The table below lists subjects, objects, and their associated ***integrity*** levels. The relationship between the levels is as follows: purple > green > orange

| Subject | Subject Level | Object | Object Level |
|---|---|---|---|
| Alice | Green | Yoyo | Purple |
| Bob | Purple | XRay | Green |
| Carol | Orange | Zebra | Green |

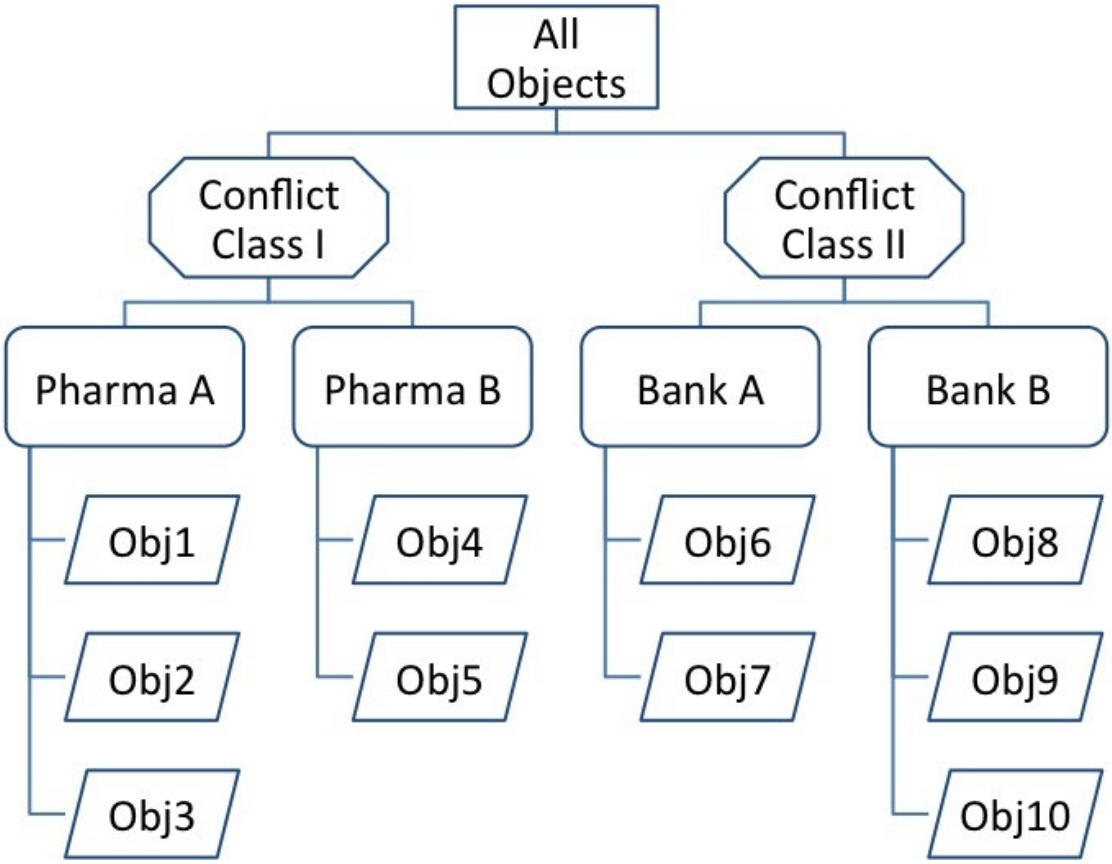
b) Compute whether the specified subject has ***observe (read)*** or ***modify (append or update)*** access to the specified object (see table below) following the ***Biba Strict Integrity Policy***.

| Subject | Object | Rights |
|---|---|---|
| Alice (green) | Xray (green) | Observe, Update |
| Bob (purple) | Zebra (green) | Append |
| Carol (orange) | Yoyo (purple) | Observe |
| Carol (orange) | Zebra (green) | Observe |

c) The ***integrity*** labels are updated to include project categories, p1, p2, and p3. The updated labels are shown in the table below. Re-evaluate the rights (modify or observe) associated with each subject and object pair following the Biba model.

| Subject | Subject Class | Object | Object Class |
|---|---|---|---|
| Alice | Green:{p1,p2} | Yoyo | Purple:{p1} |
| Bob | Purple:{p2} | XRay | Green:{p1, p2} |
| Carol | Orange: {p1, p3} | Zebra | Green: {p3} |

| Subject | Object | Rights |
|---|---|---|
| Alice<br>Green:{p1,p2} | Xray<br>Green:{p1, p2} | Observe, Update |
| Bob<br>Purple:{p2} | Zebra<br>Green: {p3} | No Observe, No Modify<br>p3 !subset p2 && p2 !subset p3 |
| Carol<br>Orange: {p1, p3} | Yoyo<br>Purple:{p1} | No Observe, No Modify<br>Modify: Purple !<= Orange<br>Observe: {p1, p3} !sub {p1} |
| Carol<br>Orange: {p1, p3} | Zebra<br>Green: {p3} | No Observe, No Modify<br>Modify: Purple !<= Orange<br>Observe:{p1, p3} !sub {p3} |




Q12 [5 pts]: Figure above depicts organization of objects into datasets (e.g., Bank A) and conflict of interest classes (e.g., Conflict Class I) at consulting firm ConFirm X that uses Chinese Wall access model. Jane, Bob, Emily, Marcus, and Alice are consultants with the firm. Assume that the consultants currently have no other accesses than those explicitly stated. Please answer the following with respect to the above figure when using a Chinese Wall access model.

a) Can Bob be allowed to **read** Obj 6 and Obj2? Explain why or why not.

Subject S can read an object O only if
O is in same DS as object already accessed by S or
[OK] O belongs to CI from which S has not yet accessed any information

Yes, this access can be made, as both accesses are accesses to CI's that Bob has not already accessed.

b) Can Jane be allowed to **read** Obj7 and Obj10? Explain why or why not.

Subject S can read an object O only if
[NO] O is in same DS as object already accessed by S or
[NO] O belongs to CI from which S has not yet accessed any information

After accessing Obj7, the access to Obj10 is illegal because it is in the same CI and in a different DS than Obj7.

c) Can Emily be allowed to **read** Obj1 and **write** to Obj9? Explain why or why not.

Read: Obj1: Subject S can read an object O only if
O is in same DS as object already accessed by S or
[OK] O belongs to CI from which S has not yet accessed any information

Write: Obj9: A subject S can write to object O only if
[OK] S can read O according to the simple security rule AND
[NO] All objects that S can read are in the same DS as O

Not allowed to write to Obj9. This is because read access has happened to other DS's, and that fails the second part of the conditional for write access.

d) Can Marcus be given **read** and **write** access to Obj8 and **write** access to Obj10? Explain why or why not.

Write: Obj8: A subject S can write to object O only if
[OK] S can read O according to the simple security rule AND
[OK] All objects that S can read are in the same DS as O

Write: Obj10: A subject S can write to object O only if
[OK] S can read O according to the simple security rule AND
[OK] All objects that S can read are in the same DS as O

e) Can Alice be given **read** and **write** access to Obj6 and Obj 3? Explain why or why not.

Write: Obj3: A subject S can write to object O only if
[OK] S can read O according to the simple security rule AND
[OK] All objects that S can read are in the same DS as O

Write: Obj6: A subject S can write to object O only if
[OK]S can read O according to the simple security rule AND
Read: Obj6: Subject S can read an object O only if
O is in same DS as object already accessed by S or
[OK] O belongs to CI from which S has not yet accessed any information
[NO] All objects that S can read are in the same DS as O

Alice can read items from multiple DS's, so the second condition for write access to Obj6 is invalid, thus the operation is not allowed.

## Submission Details

Submit a PDF file with the questions and your corresponding answers
The assignment is worth 60 points. It is due Wednesday of Week 7 at Midnight.