Vincent Yasi

CS 370 -Intro to Security-

Problem Set 5

Access Control Concepts

Q1[6 pts]: State and define the three most important components in access control, all starting with the letter 'A'?

Authentication – connecting a user to a particular principle and account in the system

Authorization – determining the access rights of a principle and regulating these rights regarding what they access in the system

Auditing- being able to monitor and assess the actions taken during access control

Q2 [4 pts]: What is the primary difference between DAC and MAC access model?

In DAC, the users get to determine the access control properties of the objects they create and control, setting the access controls how they want.

In MAC, however, access control is instead regulated by the labels and properties associated to the file, independent of what the particular user wants to set them to.

Q3 [4pts]: In access control, what does an "open policy" and "closed policy" mean?

Open and closed policy relates to the default access control status of the system. In an open policy, a user has access to objects unless they are specifically prohibited from doing so (blacklisting). With a closed policy, instead a user does not have access to an object unless explicitly granted the access (whitelisting).

Q4 [4 pts]: Explain the difference between Role-Based Access Control (RBAC) and Attribute-Based Access Control (ABAC)

Role-Based Access Control delegates access based on the role(s) a user has. Permissions are tied to the role itself, and the user then has the rights granted by that role.

With Attribute-Based, permissions are delegated based on the characteristics/attributes of the user, and possibility other things. The permissions are not tied to a particular role, but rather the individual attributes of the user itself, and are assessed on a case by case basis at the time of access.

Access Control Matrix

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

Q5 [3 pts]: Define the rights the access control system would need to enforce the requirements for this scenario. Associate an abbreviation that you can use in the following questions.

R – read/view a file

X – execute a file

C – create a file (presumably, a user who creates a file also owns it, but that is not explicitly said)

E – edit a file

D – delete a file

M(xxx) – promote/demote a file to level xxx (ex: M(test) would move a file to test level)

Q6 [7 pts]: Design an access control matrix for the scenario above for the users mentioned.

	Development Code	**Development Executable**	**Testing Code**	**Testing Executable**	**Test Report**	**Production Code**	**Production Executable**
Eve	-	R, X	-	-	R	R, X	R, X
Alice	C, E, D, X, M(test)	C, E, D, X	-	-	R	R, X	R, X
Bob	C, E, D, X, M(test)	C, E, D, X	-	-	R	R, X	R, X
Carol	-	-	E, D, X, M(product/develop)	E, D, X	C, R	R, X	R, X
Dave	-	-	E, D, X, M(product/develop)	E, D, X	C, R	R, X	R, X

Q7 [3 pts]: Assume the Access Matrix is being implemented by a system using Access Control Lists. Write the Access Control List for the Development Executables.

	Eve	**Alice**	**Bob**	**Carol**	**Dave**
Development Executable	R, X	C, E, D, X	C, E, D, X	-	-

Carol and Dave could be dropped off this list to make it more concise, but I left them in so as to make it explicit that they have no Development Executables permissions.

Q8 [3 pts]: Assume the Access Matrix is being implemented by a Capability system. Write the Capability list for Alice.

	Development Code	**Development Executable**	**Testing Code**	**Testing Executable**	**Test Report**	**Production Code**	**Production Executable**
Alice	C, E, D, X, M(test)	C, E, D, X	-	-	R	R, X	R, X

Once again, Testing Code and Executables could be removed, but I left them in to show Alice explicitly does not have permissions for those files.

Changing Access Control Policy/Matrix

	File 1	File 2	File 3	File 4	Subject A	Subject B	Subject C
Subject A	Own R W		Own R W		Control		Own
Subject B	R	Own R W	W	R*		Control	
Subject C	R W	R		Own R W			Control

Q9 [4 pts]: Keeping in mind the rules governing access control matrix change covered in class, and the access matrix shown above, answer whether or not the following changes to access matrix are allowed.

Explain in one sentence why or why not.

a) (**not allowed**) Subject C wants to Transfer R on File 2 to Subject A

   This is not allowed, as Subject C does not own File 2, and does not have transferable R rights for that file (which would be denoted by R*)

b) (**allowed**) Subject A wants to Delete R on File 2 from Subject C

   This is allowed, as Subject A has ownership of Subject C. Therefore, it can delete any permissions it wants from Subject C.

UNIX Permissions

Q10 [5 pts]: When a file in Unix is protected with mode "644" and is inside a directory with mode "730" can you describe a way in which the file can be compromised?

Mode 644 denotes read and write permissions for the user, as well as read permissions for their group and the world.

Mode 730 denotes read, write, and execute permissions for the user, write and execute permissions for the group, and no permissions for the world.

A directory with write permissions means you can modify its contents. With such a permission, a group of the directory can delete or overwrite (by saving a new file with the same name as the original file to the directory) the file in the directory, even though the permissions on the file itself do not want to allow anyone from the group to write to it.

Q11 [2 pts]: Suppose you are working as the security administrator at xyz.com. You set permissions on a file object in a network operating system which uses DAC (Discretionary Access Control). The Extended ACL (Access Control List) of the file is as follows:

Owner: Read, Write, Execute

User C: Read, Write, -

User B: Read, Write, Execute

Sales: Read, -, -

Marketing: -, Write, -

Mask: Read, Write, -

Other: Read, Write, -

User "A" is the owner of the file. User "B" is a member of the Sales group. What effective permissions does User "B" have on the file?

User B only has the Read permission. While initially they have Read, Write, and Execute, this is first run through the mask, and only permissions present in both the mask and user are kept (Read and Write in this case). Then, this is further run through the Sales permissions, once again only keeping those present in both. As Sales only has Read permissions, then User B ultimately only has Read permissions, as well.