Assignment 2: Take-Grant Model

Subramanian Vasudevan

Question 1 (a): Two properties we can analyse using an Access Control Matrix but not using the Take-Grant Model:

The strength of the Access Control Matrix Model is its flexibility. Unlike the Take-Grant model, it imposes no conditions on the types of rights or their flow between subjects and objects, and hence can be used to model a wider variety of protection systems.

- The consequences of destruction of entire objects can be studied only using the Access Control Matrix model, since the Take-Grant Model does not include this in its list of permitted operations.
- The Take-Grant model is not expressive enough to be used to study the change in protection state of a system over parameters such as time of the day and role of the subject, for example. The Access Control Matrix, however, is powerful enough to be able to specify the rights that subject 's' possesses over an object 'o', as a function of many such parameters.

Question 1 (b): Two properties that we can analyse using the Take-Grant protection model but not using an Access Control Matrix:

- The primary strength (and motivation behind) the Take-Grant model is the ability to analyse the safety of a protection system in time linear with the number of subjects and objects in the system. This is provably impossible in the case of protection systems modelled using a general Access Control Matrix with no restrictions specified.
- In case the safety of a system has been compromised with respect to a right r, the Take-Grant model allows us to analyse all the possible subjects that needed to be involved in order to leak that right. It is even possible to analyse whether a theft of that right is possible by a subject without needing the cooperation of any other subject. This characterization of the possible flow of rights across subjects and objects would enable us to quantify the amount of trust that the system would be placing on different subjects.

Question 2 (a):

can_share (read, x, w, G) = False.

This is a trivial case because there exists no subject (or object) that possesses a read right over w. Hence it would not be possible to 'share' that right to subject x.

Question 2 (b):

can_share (delete, z, u, G) = False.

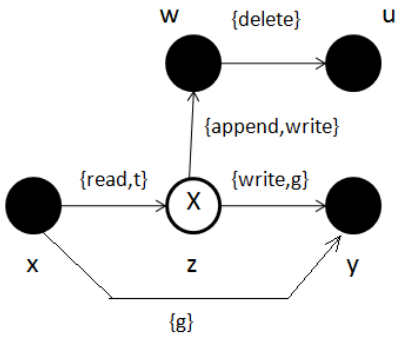
In this case, there exists a subject 'w' that possesses the 'delete' right over object u, but there exists no take/grant-path between 'w' and 'z', which is necessary for that right to be shared to vertex 'z'.

Question 2 (c):

can_share(g, x, y, G) = True.

There exists a node 'z' that possesses the right 'g' over subject 'y' AND there exists a 'take' right from subject 'x' to vertex 'z'. This allows 'x' to take the right 'g' from vertex 'z', thus the right 'g' has been effectively shared. (Contd.)

The Graph G' for which the can_share property holds is as below:



Question 3: Application of take, grant, create, and remove rules over G_0 where can_share(alpha, x, y, G_0) = False to transform to G_k where can_share(alpha, x, y, G_k) = True.

This is not possible in the Take-Grant protection model, as it is essentially monotone in the sense that, the create primitive in the Take-Grant model does not allow creation of new vertices with ARBITRARY rights over other existing vertices in the graph. The new vertex can at most obtain (take or be granted) all the rights of its creator over other vertices.

Suppose can_share(alpha, x, y, G_0) = False. This means one of the following is true:

- There does NOT exist any vertex s with alpha rights over y. To create such a vertex, we need ATLEAST one other subject with alpha rights over y to be its creator, which is a contradiction. (OR)
- There does NOT exist any subject s' which terminally spans to s, which can 'take' alpha rights from s. Using the same argument, the creation of such a subject would require ANOTHER subject as creator, which already has the series of take rights to terminally span to s, which is a contradiction. (OR)
- There does not exist any subject x' which initially spans to x, which can 'give' alpha rights to x. If this is the case, then creation of such a vertex is not possible using the same argument as above. (OR)
- There does NOT exist a take/grant-connected path of bridges and islands enabling transfer of rights from s' to x'. Suppose this path CAN be completed by creating a vertex z with a take or grant right. This right must have to be taken from or granted by it's creator, say z', in which case the path would ALREADY be tg-connected through z', which is a contradiction.

The above proof shows that there exist no series of the operations take, grant, create and remove that can transform a graphs G_0 where can_share(alpha,x,y, G_0) = False to a graph G_k where can_share(alpha,x,y, G_k) = True.

Question 4: Application of take, grant, create, and remove rules over G_0 where can_share(alpha, x, y, G_0) = True to transform to G_k where can_share(alpha, x, y, G_k) = False.

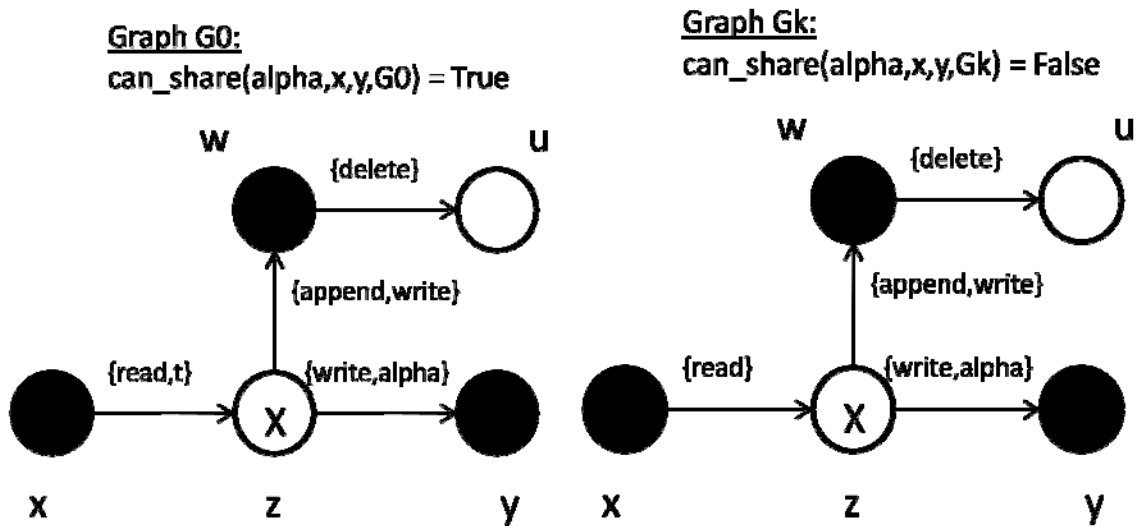
One or more applications of the remove rule should be enough to transform G_0 to G_k and make sure can_share does not hold.

Since can_share(alpha, x, y, G_0) = True, the following points must all be true:

a) There exists an object s which has alpha rights over y.
b) There exists a subject s' which terminally spans to s.
c) There exists a subject x' which initially spans to x.
d) There must exist a take/grant-path (of bridges and islands) between x' and s'.

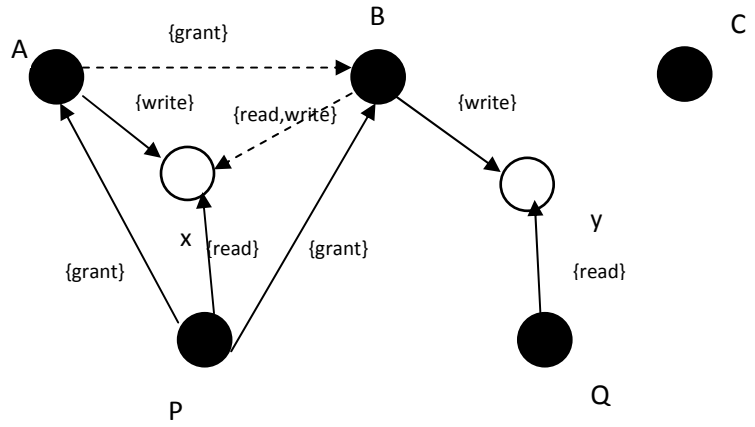
By using the remove rule successively to remove take/grant rights on the edges between s' and s, we can make s' NO LONGER terminally span to s. We may also choose to remove the initial span from x' to x in this manner, OR break the take/grant-path between subjects x' and s' in the same way. When this has been performed for all applicable vertices s, s' and x', we will find that we have transformed G_0 into G_k where can_share(alpha, x, y, G_k) does not hold.

For example consider the following transformation from a Graph G_0 to a Graph G_k. By removing the 't' right from subject x to vertex g, we are breaking the take/grant-path between them so that the alpha rights can no longer be shared.



Question 5: Modelling the prescribed health-care provider system using the Take-Grant protection model.

Consider the following Take-Grant protection system.



Let us assume that P and Q are patients and A,B and C are the healthcare providers. A has a write right over patient P's record 'x'. P is modelled as the owner of its record, which means it has grant rights over all healthcare providers.

- Since P has the grant right to B, it can grant its read right over x to B. This is akin to P allowing B to see its record. This operation involves the action of P alone.
- Since P has the grant right to A, it can grant A its grant right over B. Now, since A has grant right to B, it can grant its 'write' right over x to B. Hence the cooperation of both patient P and healthcare provider A is required for B to be able to write to the record 'x'.
- The same argument can be extended to show how B and Q need to cooperate so that a new healthcare provider 'C' can write to record y.

However, if we use the Take-Grant protection model to model this kind of a scenario, there WILL ALWAYS be an unauthorised sharing of rights. The conventional Take-Grant mechanism for exercising control over the movement of rights suffers from an unfortunate limitation: it cannot enforce strictly unidirectional channels for the flow of rights. That is, if rights can be moved directly or indirectly from the provider A in the graph to another provider B, then one cannot prevent rights from flowing in the opposite direction, from B to A.

It can be shown by construction that once a tg-path of arbitrary length has been formed between A and B (after A has a grant right to B), it becomes possible for A to take any right that B has over y (write), without the involvement/authorisation of Q.

Hence, the Take-Grant model cannot support this scenario, as disallowed sharing can happen.