Information Assurance: Homework 5 – Answers

Due October 25, 2010

1. The following policy is enforced in a business:
    - Tellers can update customer balance data.
    - Auditors can review all customer data
    - Manager can approve the end of day balance review
    - Manager can create and close customer accounts

Consider a specific case with the following entities:

- Alice and Bob are tellers
- Carol is an auditor
- Dave and Ellen are managers
- Fred and Ginger are customers

There are of course multiple ways of accurately answering this question. Here one option is listed below. Many people chose a "grouped" option. Instead of listing Alice, etc as subjects, and Alice's account, Bob's account as objects, they listed groups (Tellers, Managers, Auditors, customers) as subjects, and sets of objects like "Customer Balance", "Other customer data". The problem with the grouping approach comes with limited granularity. In particular for part b, you need to distinguish Alice's account from Bob's account, from everyone else's account.

Some people created "context" dependent objects like "Own account" and "everyone else's account". This solution didn't fly. The ACM doesn't support the idea of objects where the meaning of the object changes depending on the associated subject.

a. Define the rights involved and create an Access Control Matrix to encode the protection state for this scenario.

Considering the following rights:
U – Update right. Holder can update the balance data
R – Review right. Holder can review the associated customer data
A – Approve right. Holder can approve the balance of the associated customer data
C – Close/Create right. Holder can create and delete customer accounts.

	Fred's Data	***Ginger's Data***	***Alice's Data***	***Bob's Data***	***Carol's Data***	***Dave's Data***	***Ellen's Data***	***Global Rights***
Alice	*U*	*U*	*U*	*U*	*U*	*U*	*U*	
Bob	*U*	*U*	*U*	*U*	*U*	*U*	*U*	
Carol	*R*	*R*	*R*	*R*	*R*	*R*	*R*	
Dave	*A*	*A*	*A*	*A*	*A*	*A*	*A*	*C*

	Fred's Data	***Ginger's Data***	***Alice's Data***	***Bob's Data***	***Carol's Data***	***Dave's Data***	***Ellen's Data***	***Global Rights***
Ellen	*A*	*A*	*A*	*A*	*A*	*A*	*A*	*C*

Adding a column that corresponds to no customer data to store the fact that the managers have global rights to create and delete customers. The close right could be associated with existing accounts/data. But the create right only makes sense before an object exists.
Do not necessarily need to have all employees be customers too. Though for part b), it would be more interesting if at least Alice and Bob are customers of the bank.

b. Another rule is added to the policy. A teller cannot update their own balance data. Update the ACM to reflect the protection state with this new rule.

	Fred's Data	***Ginger's Data***	***Alice's Data***	***Bob's Data***	***Carol's Data***	***Dave's Data***	***Ellen's Data***	***Global Rights***
Alice	*U*	*U*		*U*	*U*	*U*	*U*	
Bob	*U*	*U*	*U*		*U*	*U*	*U*	
Carol	*R*	*R*	*R*	*R*	*R*	*R*	*R*	
Dave	*A*	*A*	*A*	*A*	*A*	*A*	*A*	*C*
Ellen	*A*	*A*	*A*	*A*	*A*	*A*	*A*	*C*

c. Express the ACM from part b as a set of access control lists.

For customers Fred, Ginger, Carol, Dave, and Ellen
ACL(Customer Data) = (Alice, {U}), (Bob, {U}), (Carol, {R}), (Dave, {A}), (Ellen, {A})
ACL(Alice's Data) = (Bob, {U}), (Carol, {R}), (Dave, {A}), (Ellen, {A})
ACL(Bob's Data) = (Alice, {U}), (Carol, {R}), (Dave, {A}), (Ellen, {A})
ACL(Global Rights) = (Dave, {C}), (Ellen, {C})

d. Express the ACM from part b as a set of capabilities.

Cap(Alice) = (Fred's Data, {U}), {Ginger's Data, {U}), (Bob's Data, {U}), (Carol's Data, {U}), (Dave's Data, {U}), (Ellen's Data, {U})
Cap(Bob) = (Fred's Data, {U}), {Ginger's Data, {U}), (Alice's Data, {U}), (Carol's Data, {U}), (Dave's Data, {U}), (Ellen's Data, {U})
Cap(Carol) = (Fred's Data, {R}), (Ginger's Data, {R}), (Alice's Data, {R}), (Bob's Data, {R}), (Carol's Data, {R}), (Dave's Data, {R}), (Ellen's Data, {R})

Cap(Dave) = Cap(Ellen) = (Fred's Data, {A}), (Ginger's Data, {A}), (Alice's Data, {A}), (Bob's Data, {A}), (Carol's Data, {A}), (Dave's Data, {A}), (Ellen's Data, {A}), (Global Rights, {C})

2. In this question you will work through evaluating labeled access following the Bell-LaPadula confidentiality model. For the first sections consider the following labeled entities:

Subject	**Object**	**Label**
Alice	Plan1	L5
Bob	Plan2	L4
Carol	Plan3	L3
Dave	Plan4	L2
Ellen	Plan9	L1

The labels follow a complete ordering L1 > L2 > L3 > L4 > L5.

a. Interpret the labels as security labels in the simplified Bell-LaPadula model. Fill the the access column with the access that BLP would give each subject to the corresponding object: read, append (also mentioned in lecture as a pure write).

Subject	**Object**	**Access?**
Alice	Plan4	*Append*
Bob	Plan2	*Read and Append*
Ellen	Plan3	*Read*
Dave	Plan9	*Append*

b. Now consider the case where the labels have categories in addition to the completely ordered levels. We add categories proj1 and proj2. The new label assignments are:

Subject	Subject Label	Object	Object Label
Alice	L1:{proj1}	Plan1	L1:{proj1}
Bob	L2:{proj1,proj2}	Plan2	L2:{proj2}
Carol	L3:{proj2}	Plan3	L3:{proj1, proj2}
Dave	L4:{proj2}	Plan4	L4:{proj1}
Ellen	L5:{proj1}	Plan9	L5:{proj2}

Interpret these labels according to the Bell-LaPadula Model. Fill the the access column with the access that BLP would give each subject to the corresponding object: read, append (also mentioned in lecture as a pure write).

It appears that some folks interpreted this part of the question using the max-level and current-level extensions that some trusted operating systems use. The question did not indicate that this extension was to be used. The standard BLP model does not address changes from maximum to current levels.

Subject	Object	Access?
Alice	Plan2	*No access*
Bob	Plan2	*Read*
Ellen	Plan4	*Append*
Dave	Plan9	*Read*

3. Biba proposed three different integrity models.

a. In the Low water mark policy, the subject level potentially drops to eliminate indirect integrity problems through the information transfer path. Give an example of how the low water mark rules prevents low integrity data from propagating into high integrity data indirectly via the information transfer path.

Consider integrity levels, High > Medium > Low.

Say the user reads data at integrity level Low. Then the user turns around and writes that information into a High integrity file.

Low(file1) → (User) → High(file2)

This is impossible to do in the low water mark policy. Regardless of the user's original integrity level, after reading file1 his integrity level will be set to Low. The second right to file2 will be denied because lower integrity subjects cannot write higher integrity data.

There was a lot of variance in the answers. Here are some common errors for part a).

- *Many overlooked the fact that any subject can read any object; there is no restriction on who can read what. The only difference with ring model is L(S) = min (L(S), L(O)) in low water mark policy.*
- *Some of you gave real life examples like OS processes, students reading/updating their grades, and so on, but did not mention who is at high level, who is at low level, how low water mark policy is used to lower the level, and how it prevents information flow.*
- *Some gave examples that does not utilize the fact that L(S) is lowered to min (L(S), L(O)).*
- *Some gave wrong explanation.*

b. In the Ring model, give an example of how low integrity data could be indirectly propagated via the information transfer path.

Consider the same example of User, file1 and file2.

In the ring model, the user's integrity level is not changed. An the user can read data at any integrity level. So the following sequence would be allowed, conceivably enabling an injection of low integrity data from file1 indirectly into file2

Low(file1) → High(User) → High(file2)

Here too there were many errors. Some common issues were:

- *Many overlooked the fact that subjects can read any object.*
- *The same example used in 3a could be used here, the only difference is since L(S) is not changed, information flow is possible.*

c. Does the strict Biba model solve the information transfer path integrity failure problem? Why or why not?

Strict Biba also solves the indirect information transfer path integrity problem because it enforces restrictions on both reads and writes. Our example that was allowed in the Ring model will not be allowed in strict Biba.

Low(file1) → High(User) → High(file2)

In this case the High integrity user could write to the high integrity file, but he could not read from the low integrity data.

Low(file1) → Low(User) → High(file2)

Similarly in this case, the Low integrity user could read the low integrity file but could not write the high integrity file. So if the subject cannot change integrity levels, the strict Biba model also controls the information transfer path problem

Half marks were given if you said yes, it can solve . Full marks for answers with correct explanation.

4. Suppose a database for a hospital contains an 'patient' table listing all patients' names, SSNs, diagnosis, assigned doctor, and notes. The patient table rows for three patients is shown below

Name	SSN	Diagnosis	Doctor	notes
Alice	xxx-xx-xxxx	pneumonia	Dr. Jones	
Bob	yyy-yy-yyyy	Broken leg	Dr. Jones	Change to walking cast
Carol	zzz-zz-zzzz	delivery	Dr. Smith	Induct by 7pm

In addition, there are employees.

- Nurses can read patient name, diagnosis and doctor. They should also be able to update notes.
- Doctors can read patient name. They can update diagnosis, doctor, and notes.
- Accountants can read name, ssn, and doctor.

Wendy is a nurse, Xavier is a doctor, and Zander is an accountant.

a. Suppose you are the database administrator responsible for enforcing the policies listed above. Show the SQL statements for these three employees to enforce this policy.

grant select(Name, Diagnosis, Doctor) on patients to 'Wendy';
grant update(notes) on patients to 'Wendy';

grant select(Name) on patients to 'Xavier';
grant update(Diagnosis, Notes, Doctor) on patients to 'Xavier'

grant select(Name, Doctor, SSN) on patients to 'Zander';

You could also express this as a view and then a grant. The exact syntax is not important as long as the key parts are there.

b. The company policy states that every patient should be able to view all fields about themselves in the 'patient' table. Show the SQL statements you would use to enforce this policy.

*create view alice_private as select * from patients where name = 'Alice';*

```
grant select on alice_private to Alice;

create view bob_private as select * from patients where name = 'Bob';
grant select on bob_private to Bob;

create view carol_private as select * from patients where name = 'Carol';
grant select on carol_private to Carol;
```