A Semi-Formal Multi-Policy Secure Model for Semantic Spatial Trajectories

Xingang Wang  
Shandong Academy of Sciences  
P.R.China  
wangxingang2009@hotmail.com

Keywords: Semantic Spatial Trajectory; Role Based Access Control; Bell-LaPadula Model; Multi-Policy; Web Ontology Language

1. Introduction

With the proliferation of locating devices, more and more raw spatial trajectories are formed, and many works enrich these raw trajectories with semantics, and mine patterns from both raw and semantic trajectories, but access control of spatial trajectories is not considered yet. We present a multi-policy secure model for semantic spatial trajectories. In our model, Mandatory Access Control, Role Based Access Control and Discretionary Access control are all enforced, separately and combined, and we represent the model semi-formally in Ontology Web Language.

1.1 An illustrative case of our scheme is: a leader of a company is involved in spatial trajectories related with government issues, and some employees in the company will access the spatial trajectories, and some officers in the government need to access the spatial trajectories too, where the company employs a RBAC model, and the government organization employs a MAC model, specifically, BLP model.
Fig. 1. Ontology of Semantic Trajectory Episode (STEP) model. Diamond-shaped nodes are datatypes.

The access control of these spatial trajectories shall be considered.

Besides, we give the concept of future trajectory to motivate our work further, that is, the two organizations above may give a future trip plan to the leader in the company. This is a semantic trajectory, not with raw spatial trajectory data. Then in the real trip, these future trajectories shall be updated with the actual trajectories with semantics enriched. The access control of these future trajectories shall be given as the above spatial trajectories.

Next in section 2 we give the preliminaries, in section 3 we give our multi-policy secure model for semantic spatial trajectories, and draw a conclusion in section 4.

2. Preliminaries

We take the semantic trajectory model of Nogueira and Martin [12] as the base of our secure model, that is, the Semantic Trajectory Episode (STEP) model. They give the model in a relative later time and we think this is a reasonable and useful model. They use STEP to structure trajectories and context into spatial-temporal episodes. A graphical representation of STEP is shown in Fig 1. It is represented in OWL.

The Agent class denotes the moving object. A Trajectory instance may have a number of FeatureofInterest instances and one RawTrajectory instance. By the hasRawTrajectory property, the Trajectory instance has at most one Raw-Trajectory instance. A RawTrajectory instance may have a series of Fix instances and Fix is composed of time:Instant and sf:Point. Through the hasFeature property, Trajectory is related with FeatureofInterest, and FeatureofInterest is related to ContextualElement, and Trajectory and ContextualElement instances all may have one more FeatureofInterest instances. A FeatureofInterest instance has at least one Episode instance.

Episode is the smallest semantic entity, which encapsulates values that a feature of interest may assume. An Episode instance should have at least one SemanticDe-
cription instance, which can be a quantitative value or a qualitative description, where quantitativeValue can be associated with different data types by anySimpleType defined in the XML schema, and QualitativeValue is used to represent other kinds of values. Besides, we give another subclass Issue to SemanticDescription, that is, each episode is related with some semantic issues, for marking what to do in the locations of the episodes. The Extent class allows to specify spatial and/or temporal limits for an episode. Three sub class SpatialExtent, TemporalExtent, and SpatialtemporalExtent can represent the extent. A SpatialtemporalExtent instance has one or more Keypoint instance, and SpatialtemporalExtent is connected to Keypoint through three properties: hasStartingPoint, hasEndingPoint, and hasPoint. Keypoint describes the key location in the trajectory. Episode is related to ContextualElement.

3. Semi-Formal Multi-Policy Secure Model

We first give the RBAC, BLP and DAC enforcement in semantic spatial trajectories respectively and then give the enforcement of RBAC, BLP and DAC in a combined way.

3.1. RABC Enforcement

RBAC is always employed in organizations like companies, and the issues related with authorization is an organization action.

First, we give the permission definition. Because the permission and permission assignment are also used in DAC, not specific to RBAC. So we make the permission part as a part of STEP. We define Object, Operation and Permission as follows.

```
STEP:Object a owl:Class.
STEP:Operation a owl:Class.
STEP:Permission a owl:Class.
```

Then we define two properties of Permission, to associate an operation on an object with permission.

```
STEP:hasObject a rdfs:Property;
  rdfs:domain STEP:Permission;
  rdfs:range STEP:Object.

STEP:hasOperation a rdfs:Property;
  rdfs:domain STEP:Permission;
  rdfs:range STEP:Operation.
```

Then a permission can be assigned an operation on an object as follows.

```
<PermissionName> STEP:hasObject <ObjectName>
<PermissionName> STEP:hasOperation <OperationName>
```

Here objects mean the instances of the classes in STEP. And the objects and the operations on the objects are illustrated in Table 1. read, modify and remove are related to each object in the STEP model. annotate is only related with the
Episode class, by which a user can annotate an episode, such as creating an issue for an episode, and relatesto_Context is only with Episode too. insert is related to episodes, by which a user can create instances of each subclass of Extent and insert them to episodes. insert is also related to SpatialtemporalExtent, by which a user can create Keypoint instances. Other classes related with insert include Trajectory, RawTrajectory, and FeaturesOfInterest. Then we can define the operations in Table 1 as follows.

- read a STEP:Operation
- annotate a STEP:Operation
- insert a STEP:Operation
- modify a STEP:Operation
- remove a STEP:Operation
- relatesto_Context a STEP:Operation

Next, we define roles and role hierarchy in the way of roles being values like the work in [6]. we define roles as follows.

- rbac:Role a owl:Class.

We define role hierarchy by introducing the property of subRole.

- rbac:subRole a owl:TransitiveProperty;
- rdfs:domain rbac:Role;
- rdfs:range rbac:Role.

Then role hierarchy can be constructed as follows.

\[
<\text{RoleName}> \text{rbac:subRole} <\text{SuperRoleName}>.
\]

We give permission-role assignment as follows. Fist, we define a property of Role.

- rbac:permitted a rdfs:Property;
- rdfs:domain rbac:Role;
- rdfs:range STEP:Permission.

Then the assignment of permission to role can be represented as follows:

\[
<\text{RoleName}> \text{rbac:permitted} <\text{PermissionName}>
\]

Another way that a role is with permission is through role hierarchy. We have a
rule as follows.

\[ \text{role1 : permitted ?permission; } \]
\[ \text{role1 : subRole role2; } \]
\[ \Rightarrow \text{role2 : permitted ?permission} \]

We define User as an class of STEP, and this class is used in RBAC,MAC,DAC
and the combined case.

\[ \text{STEP:User a owl:Class.} \]

Then we define a property of the class User.

\[ \text{rbac:hasRole a rdfs:Property; } \]
\[ \text{rdfs:domain STEP:User; } \]
\[ \text{rdfs:range rbac:Role.} \]

Then users can be assigned roles as follows.

\[ <\text{UserName}> \text{rbac:hasRole } <\text{RoleName}> \]

We define a subclass of the class Operation.

\[ \text{rbac:PermittedOperation subClassOf STEP:Operation.} \]

And we present that an operation in the access decision is permitted as follows.

\[ <\text{OperationName}> \text{a rbac:PermittedOperation} \]

Then we have the following access decision rule.

\[ \text{object1 a STEP:Object; } \]
\[ \text{operation1 a STEP:Operation; } \]
\[ \text{user : hasRole role; } \]
\[ \text{role : permitted ?permission; } \]
\[ \text{permission : hasObject object2; } \]
\[ \text{permission : hasOperation operation2; } \]
\[ \text{object1 = object2; } \]
\[ \text{operation1 = operation2} \]
\[ \Rightarrow \text{operation1 a rbac:PermittedOperation} \]

3.2. BLP Model Enforcement

The BLP model is always employed in organizations like government or military
organizations. In the BLP model, each object is assigned a security label, and the
user to access these objects shall be with a security label too. Each label is composed
of two elements: secure level and category. Then we define secure level, category and
security label as follows:

\[ \text{mac:Level a owl:Class.} \]
\[ \text{mac:Category a owl:Class.} \]
\[ \text{mac:Label a owl:Class.} \]

And we define two properties of Label to associate it with its security levels and
category.

\[ \text{mac:hasLevel a rdfs:Property; } \]
\[ \text{rdfs:domain mac:Label; } \]
\[ \text{rdfs:range mac:Level.} \]
Then we define the partial order relation of security labels by the following property.

A security level is higher or greater than another security level:

\[
\text{mac:greater} \ a \text{ rdfs:Property;}
\text{rdfs:domain} \text{ mac:level;}
\text{rdfs:range} \text{ mac:level.}
\]

A security level is equal to another security level:

\[
\text{mac:equal} \ a \text{ rdfs:Property;}
\text{rdfs:domain} \text{ mac:level;}
\text{rdfs:range} \text{ mac:level.}
\]

A security level is greater than or equal to another security level:

\[
\text{mac:greater-equal} \ a \text{ rdfs:Property;}
\text{rdfs:domain} \text{ mac:level;}
\text{rdfs:range} \text{ mac:level.}
\]

A security category is greater than another security category:

\[
\text{mac:greater} \ a \text{ rdfs:Property;}
\text{rdfs:domain} \text{ mac:category;}
\text{rdfs:range} \text{ mac:category.}
\]

A security category is equal to another security category:

\[
\text{mac:equal} \ a \text{ rdfs:Property;}
\text{rdfs:domain} \text{ mac:category;}
\text{rdfs:range} \text{ mac:category.}
\]

A security category is greater than or equal to another security category:

\[
\text{mac:greater-equal} \ a \text{ rdfs:Property;}
\text{rdfs:domain} \text{ mac:category;}
\text{rdfs:range} \text{ mac:category.}
\]

Objects in the STEP is assigned a security label by the following defined property.

\[
\text{mac:hasLabel} \ a \text{ rdfs:Property;}
\text{rdfs:domain} \text{ STEP:Object;}
\text{rdfs:range} \text{ mac:Label.}
\]

The operations we consider for BLP includes read, modify, annotate, insert, and relates_to_Context. We make the insert, annotate, relates_to_Context as write in the BLP model, and make the read operation as read in the BLP model, and make modify as both the read and write operation in the BLP model. Then a user \( u \) can perform annotation or insert or relates_Context operation on an object \( o \), only if the level and category of \( o \) are both greater-equal the level and category of \( u \), and a user \( u \) can perform read on an object \( o \), only if the level and category of \( u \) greater-equal the level and category of \( o \), and a user \( u \) can perform modify on an
object o only if the level and category of u equal the level and category of o.

We define a subclass of the class Operation.

mac:PermittedOperation subClassOf STEP:Operation

And we present that an operation in the access decision is permitted as follows.

<OperationName> a mac:PermittedOperation

Then the access decision rule are given as follows.

?user mac:hasLabel ?uLabel;
?object mac:hasLabel ?oLabel;
?uLabel mac:hasLevel ?uLevel;
?uLabel mac:hasCategory ?uCategory;
?oLabel mac:hasLevel ?oLevel;
?oLabel mac:hasCategory ?oCategory;
?uLevel mac:greater-equal ?oLevel;
?uCategory mac:greater-equal ?oCategory
⇒ read a mac:PermittedOperation

?user mac:hasLabel ?uLabel;
?object mac:hasLabel ?oLabel;
?uLabel mac:hasLevel ?uLevel;
?uLabel mac:hasCategory ?uCategory;
?oLabel mac:hasLevel ?oLevel;
?oLabel mac:hasCategory ?oCategory;
?oLevel mac:greater-equal ?uLevel;
?oCategory mac:greater-equal ?uCategory
⇒ insert a mac:PermittedOperation;
annotate a mac:PermittedOperation;
relatesto_Context a mac:PermittedOperation

?user mac:hasLabel ?uLabel;
?object mac:hasLabel ?oLabel;
?uLabel mac:hasLevel ?uLevel;
?uLabel mac:hasCategory ?uCategory;
?oLabel mac:hasLevel ?oLevel;
?oLabel mac:hasCategory ?oCategory;
?oLevel mac:equal ?uLevel;
?oCategory mac:equal ?uCategory
⇒ modify a mac:PermittedOperation;
read a mac:PermittedOperation;
insert a mac:PermittedOperation;
annotate a mac:PermittedOperation;
relatesto_Context a mac:PermittedOperation
Our scheme supports polyinstantiation[8], for example, if one object is annotated by a user with a security level, then if another user with higher security level modify his annotation, then two corresponding annotations exist, and the former user can only see his annotation and cannot see the modified part.

### 3.3. DAC Enforcement

For DAC, we take the operations: `read`, `annotate`, `insert`, `modify`, `relatesto_Context`. Then we represent issuing permission to users as follows.

\[
\text{dac:hasPermissionDac} \ a \ \text{rdfs:Property};
\text{rdfs:domain} \ \text{STEP:User};
\text{rdfs:range} \ \text{STEP:Permission}.
\]

We define a subclass of the class Operation.

\[
\text{dac:PermittedOperation} \ \text{subClassOf} \ \text{STEP:Operation}
\]

And we present that an operation in the access decision is permitted as follows.

\[
<\text{OperationName}> \ a \ \text{dac:PermittedOperation}
\]

If a user is issued a permission, then the user can perform the corresponding operation. The access decision rule is given as follows.

\[
\text{?object1} \ a \ \text{STEP:Object};
\text{?Operation1} \ a \ \text{STEP:Operation};
\text{?user} \ \text{dac:hasPermissionDac} \ ?\text{permission};
\text{?permission} \ \text{STEP:hasObject} \ ?\text{object2};
\text{?permission} \ \text{STEP:hasOperation} \ ?\text{operation2};
\text{?object1} = \ ?\text{object2};
\text{?operation1} = ?\text{operation2}
\Rightarrow \ ?\text{operation1} \ a \ \text{dac:PermittedOperation}
\]

### 3.4. Multi-Policy Enforcement in a Combined Way

Here we consider the case of RBAC, BLP and DAC all being enforced in a combined way. We know BLP is employed in military and government organizations, while RBAC is always employed in organization like Companies. Then there may be collaborations between the former organizations and the latter organizations. For example, some employees in a company and some officers in a government department collaborate to complete a task or a trip. Then the spatial objects are related to both of the two organizations, so both RBAC and BLP shall be enforced. The permission-role assignment related with these objects shall be completed by the security administrator of the company, and the label related to these objects shall be given by the security administrator of the government organization. And we make that if an officer has the security label not prohibiting him from perform the operation, then he can do the operation, but if an employee wants to perform the same operation, he must not only be issued the same label, but also be assigned a corresponding role.

We define a class representing the organizations in STEP.
STEP: Organization a owl:Class.

Then we define a property WorkIn of the class User.

STEP: WorkIn a rdfs:Property;
rdfs:domain STEP: User
rdfs:range STEP: Organization.

We define a property withOrganization of the class Label; this property is used to mark which organization a security label is issued by.

STEP: withOrganization a rdfs:Property;
rdfs:domain mac:Label
rdfs:range STEP: Organization.

And we define a subclass of the class Operation.

STEP: PermittedOperation subClassOf STEP: Operation

And we present that an operation in the access decision is permitted as follows.

<OperationName> a STEP: PermittedOperation

Then with respect to an object with a label, if a user is with a label allowing this user to perform some operation on this object, and this user belongs to the same organization as the security administrator issuing the label, then this user can perform the operation. This rule is represented as follows, with the read operation as an example.

?user mac:hasLabel ?uLabel;
?object mac:hasLabel ?oLabel;
?uLabel mac:hasLevel ?uLevel;
?uLabel mac:hasCategory ?uCategory;
?oLabel mac:hasLevel ?oLevel;
?oLabel mac:hasCategory ?oCategory;
?uLevel mac:greater-equal ?oLevel;
?uCategory mac:greater-equal ?oCategory;
?user STEP: WorkIn ?organization_u;
?oLabel STEP: withOrganization ?organization_o;
organization_u = organization_o
⇒ read a STEP: PermittedOperation

And with respect to an object with a label, if a user is with a label allowing this user to perform some operation on this object, and this user does not belong to the same organization as the security administrator issuing the label, then this user must have the corresponding role allowing him to do the operation. This rule is represented as follows, with the read operation as an example.

?user mac:hasLabel ?uLabel;
?object1 mac:hasLabel ?oLabel;
?uLabel mac:hasLevel ?uLevel;
?uLabel mac:hasCategory ?uCategory;
?oLabel mac:hasLevel ?oLevel;
?oLabel mac:hasCategory ?oCategory;
?uLevel mac: greater-equal ?oLevel;
We define a property of the class Object as follows.

\[
\text{STEP:AssignedToRole} \ a \ \text{rdfs:Property;}
\text{rdfs:domain} \ \text{STEP:Object;}
\text{rdfs:range} \ \text{rbac:Role.}
\]

We use this property \text{STEP:AssignedToRole} to see if an object is with RBAC. With respect to an object without label and with RBAC, if a user has a role which makes him do the operation, then the user can do the operation; this is just like in subsection 3.1. And if a user has not a role which makes him do the operation, then the user cannot do the operation, even if the user is allowed to do the operation by DAC.

With respect to the objects without label and without RBAC, then a user can perform operations on them only if DAC allows this as in subsection 3.3.

From above we can see that when RBAC or MAC is enforced, DAC is not considered for access decision, even if DAC permission assignment has been made.

As for our scheme, especially for the collaboration case, role assignment or label association can be completed in the way that the organizations issue credentials to users, and we consider the credential-based way is appropriate for semantic spatial trajectories.

4. Conclusion

We present a multi-policy secure semantic model for the access control of semantic trajectory data. In our model, RBAC, BLP and DAC can be enforced separately for semantic spatial trajectories access control, and can be all enforced in a combined way. Especially, the multi-policy enforcement model is appropriate for the case of collaboration between government or military organizations and other organizations such as companies, where involvers create spatial trajectories security-sensitive both for these organizations.

Acknowledgments

This research was financially supported by the Youth Science Funds of Shandong Academy of Sciences, Grant No. 2013QN040, by the Special Fund for Marine Pubb..
Public Welfare Scientific Research, Grant No. 201505031, and by the Qingdao Entrepreneurship and Innovation Leading Talent Project, Grant No. 13-CX-23 and 13-CX-24.

References


Authors’ Names


©2016 by the authors; licensee Preprints, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license http://creativecommons.org/licenses/by/4.0/.