A MODULAR FRAMEWORK FOR ADMINISTERING SPATIAL CONSTRAINTS IN CONTEXT-AWARE RBAC

by Rafae Bhatti, Maria Damiani, David W. Bettis, Elisa Bertino, Arif Ghafoor

Center for Education and Research in
Information Assurance and Security,
Purdue University, West Lafayette, IN 47907-2086

A Modular Framework for Administering Spatial Constraints in Context-Aware RBAC

Rafae Bhatti, Maria Damiani, David W. Bettis, Elisa Bertino, Arif Ghafoor

The design of context-aware access control models with spatial constraints is still far from satisfactory in a very important respect, vis-à-vis flexibility of policy administration. We motivate in this paper the requirement for providing a uniform and modular constraint specification interface for a context-aware access control model based on RBAC, with specific focus on administration of spatial constraints. Although models for spatial access control exist, the lack of a sophisticated spatial vocabulary and constraint modularity reduces the applicability of these models across a broad range of location-based applications in pervasive computing environments. In this paper, we present a location-based profile of the X-GTRBAC access control framework which allows the modular administration of context-aware access control policies. The profile is based on GEO-RBAC, a spatially aware access control model which is expressive enough to model sophisticated spatial constraints. We discuss the design challenges in developing the profile, and address the requirements for modularity and flexibility in spatial constraint administration by providing support for late-binding of constraint processing routines, and for parametric evaluation of constraint predicates. We describe the use of our profile in a location-based application. The profile has also been implemented within the existing X-GTRBAC system. The profile, although developed for the X-GTRBAC framework, can also be used by other access control frameworks through the invocation of an API for X-GTRBAC based on the standardized SAML protocol.

Introduction

Many context-aware access control models with support for spatial and location-based capabilities (or constraints) have recently been proposed [6, 8, 9]. Models with location-based access control are becoming increasingly necessary in pervasive computing applications, and are adequately receiving considerable popularity in the research and industrial community alike.

However, the design of context-aware access control models with spatial constraints is still far from satisfactory in a very important respect, vis-à-vis flexibility of policy administration. Many of the existing models have dealt with issues related to the introduction of spatial constraints in access control models without, however, taking into account requirements for modularity of constraint specification and flexibility of policy administration. In our opinion, flexible policy administration demands a clear separation between the processing of constraints and the processing of the policy; the latter uses only the result of the former (such as whether a predicate is true or not), and does not need to depend on the constraint processing details (such as how the predicate is evaluated). With this key abstraction in mind, policy administration can be simplified by providing a uniform and modular constraint specification interface that is used to encode both spatial and non-spatial constraints, with modular and pluggable domain-specific constraint processing mechanisms to support constraint evaluation. This modularity facilitates policy administration, since the administration of spatial constraints introduces no more a burden on the system that the existing constraint processing mechanisms. We believe that the wide-spread adoption of spatially aware context-aware access control models will depend on these factors as much as the expressive power of the proposed spatially aware model.

While we mention it second, expressive power of a spatially aware access control model is also an issue we consider alongside the requirements for flexible policy administration. Although many models capture location-based capabilities, there are shortcomings. The simplest of these models use low-level mechanisms to deal with location parameters, such as IP addresses [6, 8], which are not sufficient to capture high-level location abstractions, such as "near a building". More advanced models are specialized to handle only a specific location abstraction, such as relying on GPS coordinates to identify a location [9]; however most such models do not provide any means for expressing a sophisticated set of constraints on the location space (such as "contained in", etc.). These models also do not provide a modular mechanism to import external constraint specification defined by respective domain experts. The lack of a sophisticated spatial vocabulary and constraint modularity reduces the applicability of these models across a broad range of location-based services, since there is little opportunity to adapt the constraints according to the domain-specific requirements. This issue is especially important when dealing with spatial constraints, since various kinds of spatial vocabularies are likely to exist for various domains.

Keeping these goals of flexible policy administration and modularized spatial constraint specification in mind, in this paper we provide support for location-based access control in an access control framework that addresses these goals. The policy specification language underlying our framework is based on the Role Based Access Control (RBAC) model which is widely recognized for simplified administration [12]. The policy language is part of the recently proposed X-GTRBAC access control framework [3], and supports evaluation of contextual constraints. In particular, the use of temporal constraints in X-GTRBAC has already been reported in earlier work [3]. In accordance with our stated goals, our current work does not focus on the development of a new spatial extension to the model. Rather, we take a software engineering-oriented approach, and design a *location-based profile* of our existing context-aware policy specification language. The location-based profile we develop is based on an existing spatially-aware RBAC model called GEO-RBAC which has recently been proposed [1]. In particular, through the definition of a location-based profile, we are able to integrate the spatial constraint model of GEO-RBAC with the X-GTRBAC framework. It is important to note that the integration of GEO-RBAC in X-GTRBAC is not straightforward because of the fact that GEO-RBAC includes the notion of *role schema*, a new construct specifically developed to support spatial constraints. X-GTRBAC, as all the other RBAC models, on the other hand does not support such construct. To address such mismatch, we have exploited the credential construct that is part of X-GTRBAC. Credentials had been initially introduced in X-GTRBAC to support interoperability across multiple access control domains, as well as with other security protocols (such as SAML [16]). It has turned out that credentials, suitably extended, can also be an important mechanism for the support of profiles for X-GTRBAC. The required extensions to the X-GTRBAC credential model are described in this paper.

Another major issue is to be able to provide location-awareness support in a seamless manner and agnostic to the particular target environment. This requires mechanisms for associating location constraints with roles without having to change the constraint evaluation mechanism in X-GTRBAC, and for defining a spatial constraint vocabulary that can be widely-adopted. We address these issues, respectively, through the use of reflective constraint evaluation mechanism which allows us to associate spatial constraints with role enabling similarly as any other kind of constraint, and through the introduction of a spatial constraint vocabulary defined using the concepts from the Open GIS community model and represented using the Geographical Markup Language (GML) [15]. The design of the GML vocabulary to support constraint specification and that of the reflective evaluation mechanism are also described in the paper. To the best of our knowledge, the resulting integrated access control system is the first to support modular specification of both temporal and spatial constraints.

1.1. Contributions and Organization

Our primary contribution in this paper is two-fold. We highlight the issue of flexible policy administration and modular constraint specification, and we specifically focus on spatial constraints for location-based applications in pervasive computing environments. In response to the outlined challenges, we provide a location-based profile of the X-GTRBAC access control framework. The profile is based on GEO-RBAC, a spatially aware access control model. The use of the GEO-RBAC profile within the X-GTRBAC framework makes it possible to integrate powerful spatial constraint processing capabilities within an access control framework. This modular approach makes use of the uniform constraint specification mechanism provided by X-GTRBAC, and demonstrates how domain-specific constraint processing capabilities can be integrated while retaining flexibility of policy administration. The paper not only addresses the relevant design issues in developing the GEO-RBAC profile, but also presents a proof-of-concept prototype that demonstrates the applicability of the basic principles underlying our approach. We also emphasize that the GEO-RBAC profile we have developed can be used with other context-aware access control models in addition to X-GTRBAC. The authors have already proposed a SAML profile for X-GTRBAC [2], and therefore, any entity can communicate with the X-GTRBAC API through the SAML standard, and can effectively utilize it as a policy decision point for context-aware constraint evaluation, including spatial constraints as per the GEO-RBAC profile.

The remainder of this paper is organized as follows. The next section discuses related work in context-aware access control in general, and location-based access control in particular, and identifies the shortcomings of the existing approaches with respect to the challenges outlined above. Section 3 presents a summary of the GEO-RBAC spatial constraint specification model. Section 4 presents the details of the proposed GEO-RBAC profile for X-GTRBAC. It first presents an overview of the policy specification language in X-GTRBAC, and then presents a GEO-RBAC profile of X-GTRBAC that is used in our system to express spatial constraints. This section also describes an example of the use of the profile in a location-based application. Section 5 discusses the implementation architecture for implementing the GEO-RBAC profile within the X-GTRBAC system. Section 6 concludes the paper and provides directions for future work.

2. Related Work

Several access control models with spatial and non-spatial context-aware extensions have recently been proposed [6, 8, 9]. As we have already indicated, we do not intend to compare our work in terms of expressivity with current models; rather our focus is to highlight and address issues related to flexible policy administration and modularized spatial constraint specification. In this regard, none of the previously reported approaches provides significant contributions, since all of them treat an isolated system that does not provide support for modular constraint specification, and hence none of them addresses the requirement of flexible policy administration within a context-aware access control system.

One notable work in this regard is reported in [11] which proposes a condition framework for evaluation of dynamic context conditions supplied as part of a policy specification. The Antigone Condition Framework (ACF) [11] is along the lines of our goal of achieving flexible policy administration through the use of modularized constraint specification which can be dynamically evaluated. The work in [11] takes the approach of defining conditions as more than mere predicates, in that conditions can also be general purpose programs (which hints at the possibility of domain-specific extensions). That work is also motivated by the separation between policy evaluation and constraint evaluation, and the authors expect ACF to be usable with multiple policy languages. While being interesting as a generic condition specification framework, ACF has not subsequently been applied in any practical context-based application environment involving the evaluation of domain-specific condition expressions. We take a step further and actually illustrate the concept of modularized constraint specification and the resulting flexibility in policy administration it offers by developing a profile of a constraint specification model for an existing context-aware policy specification language and demonstrating its integration within the access control framework.

An approach (seemingly) very similar to ours in terms of specification of spatial constraints for the RBAC model has been recently proposed [5]. Such approach extends a temporal RBAC model with spatial extensions. While the resulting access control model is highly expressive, it does not address issues related to flexible policy administration. Instead, such approach follows a principle which is quite the opposite of the principle underlying our model. Instead of supporting a modular specification of constraints, the approach in [5] lumps together temporal and spatial constraint definition in one constraint model. Furthermore, the result model does not provide any policy specification language that can include modular constraint specification and help the system designers and administrators with specification and administration of contextual constraints, which is the approach we take.

With respect to policy administration, various administrative models for RBAC have been proposed [7, 10, 13]. While none of these models support policy administration in the presence of contextual constraints, such a comparison is not the focus of this paper. In this paper, we do not provide a new administrative model; the discussion of the administrative model for X-GTRBAC framework appears elsewhere [4]. Our goal here is to illustrate, through the use of GEO-RBAC profile, the applicability of the policy administration features of X-GTRBAC for spatially-aware access control in location-based applications.

3. GEO-RBAC

The conceptual framework of the GEO-RBAC model is centered on three major concepts: the *spatial role*, which specifies a spatially bounded organizational function assigned to mobile users, the *position model* which describes the position that users occupy in space, and the *role schema* which describes the invariant properties of homogeneous spatial roles. In this section, we provide an overview of these concepts. For a complete description of the model, we refer the reader to [1].

3.1. The position model

We first introduce the notion of position since it is fundamental to characterize the mobile user. Unlike most proposals in pervasive computing which describe the position uniquely in geometric terms, e.g. a point, the model introduces the distinction between the *real* and the *logical* position. The real position corresponds to the position of the user on Earth acquired through some positioning technology. Real positions can thus be represented as geometric elements of different types since, depending on the chosen technology and accuracy requirements, they may correspond to points or polygons. Conversely, the *logical position* is defined at a higher level of abstraction to represent positions in a way that is almost independent from the underlying positioning technology. Further, besides a geometry, the logical position has a semantics. For example, logical positions can be a house, an address number, or a road. Moreover, logical positions are represented in terms of spatial objects, specifically *spatial features*. The logical position is then computed from real positions by using a *location mapping function*. For example, a location mapping function can be defined to map a position acquired through GPS onto the object representing the closer road segment.

3.2. Spatial roles

The concept of *spatial role* is the most distinguishing aspect of the GEO-RBAC model. A *spatial role* has besides a name, a *role extent* which defines the boundaries of the region in space, in which users are enabled to play the given role and thus obtain the associated permissions. For example Student(Purdue) is a spatial role: *Student* is the role name and *Purdue* is the role extent, that is, the identifier of a spatial object describing a region in the reference space. The set of available spatial roles to a user is called the *session roles*. Roles in this set may have a status which is *enabled* or *disabled*. For a session role to be *enabled*, the user should be logically located within the space of the

Figure 1: Core GEO-RBAC model

corresponding role extent. As the user moves, the status of its roles changes; therefore, depending on the position of the user, only a subset of the session roles is enabled and permissions granted. As a result the set of services which the user can access at a given point in time depends on both the session and the position of the user.

3.3. Role schema

The third characterizing aspect of the model is the distinction between role schema and role instance. A *role instance* is a spatial role defined over a specific extent, in compliance with the role schema. A *role schema* defines some common properties of roles with a similar meaning. Specifically, a role schema defines: a) a common name for a set of roles; b) the type of role extent; c) the type of logical location; d) the mapping function relating the real position with the logical position. For example: Student(Campus, CampusSector, m_*sector*) is the schema for student roles defined for the role extent represented by a campus (the spatial role Student(Purdue) is an instance of this role schema where the campus is Purdue). The logical position of the users is identified by the sector of the campus, assuming that the campus is subdivided in zones, which is computed by applying function m_*sector* to the real position. Further, permissions are assigned to both role schemas and role instances. The permissions which are assigned to a schema are then inherited and shared by all the instances of the schema.

The general structure of the GEO-RBAC model is illustrated in Figure 1. Note that the major difference with respect to the RBAC model is the use of role instances (R_i) and role schemas (R_s) as opposed to only roles, and the specialization of the constraints applicable on user sessions (SES) as being spatial in nature.

4. Policy Specification Language

The specification language we have designed to express spatial constraints is developed as a GEO-RBAC profile of X-GTRBAC [3].We first provide a brief overview of X-GTRBAC, and then present the GEO-RBAC profile of X-GTRBAC.

4.1 X-GTRBAC

X-GTRBAC is an XML-based access control framework. It is an extension of the role-based access control (RBAC) model [12], which is known for its support for privilege management in large scale systems. RBAC uses the notion of roles to embody a collection of permissions; permissions are associated with roles through a permission-to-role assignment, and the users are granted access to resources through a user-to-role assignment. X-GTRBAC extends RBAC to provide a generalized mechanism to express a diverse set of constraints on user-to-role and permission-to-role assignments, role enabling, and role activation. It is this constraint specification mechanism which is of specific relevance to us for this paper, and we shall discuss that below.

X-GTRBAC supports a declarative predicate-based constraint specification mechanism to define temporal and non-temporal contextual constraints. An assignment, enabling, or activation constraint in X-GTRBAC comprises of a set of conditions, where each condition has associated with it an optional temporal constraint expression and an optional set of non-temporal logical expressions. The syntax of assignment constraint is described in Table 1; syntax of enabling and activation constraint is similar, except for the difference in the tag name.

Table 1: X-GTRBAC assignment constraint expression

`<AssignConstraint op="AND|OR">` `[<AssignCondition [cred_type=""]` `[pt_expr_id=""]>` `[<!--<Logical Expression>-->]*` `</AssignCondition>]+` `</AssignConstraint>`	`AssignConstraint`: represents a set of constraints to apply to the assignment. The attribute `op` defines the evaluation mode of the included conditions. `AssignConstraint/AssignCondition`: represents a contextual condition. It may specify a credential type and a periodic time expression. The former indicates that the subject of the constraint (user or role) must present a credential, whose attributes must satisfy the rules defined in this condition. The latter represents a temporal constraint expression (See [3]).
`<!--<Logical Expression>-->::=` `<LogicalExpression op="AND|OR">` `[<!--<Predicate Block>-->]+` `<LogicalExpression>`	`AssignConstraint/AssignCondition/LogicalExpression`: represents a logical expression. It contains one or more predicates. The attribute `op` defines the evaluation mode of the predicates.
`<!--<Predicate Block>--> ::=` `<!--<Logical Expression>--> |` `<!--<Predicate>-->`	`AssignConstraint/AssignCondition/LogicalExpression/`*`{PredicateBlock}`*: represents either another logical expression or a simple predicate.
`<!--<Predicate>--> ::=` `<Predicate>` `<Operator />` `<FuncName />` `<ParamName />` `<RetValue />` `</Predicate>`	`AssignConstraint/AssignCondition/LogicalExpression/Predicate`: A simple predicate defines rules on credential attributes of the constraint subject (user or role). It includes a comparison using an (`Operator`) between the value of the credential attribute computed using a function (`FuncName`) having one or more arguments (`ParamName`) and the expected (`RetValue`).

A constraint is satisfied if all included conditions are satisfied, according to the supplied operator which defaults to `AND` if none is supplied. A condition is satisfied if (i) the associated temporal constraint expression, if any, is satisfied; a temporal constraint expression checks for time-based conditions, such as periodicity, interval or duration, and (ii) the associated set of logical expressions, if any, is satisfied; a logical expression is satisfied if all included predicates are satisfied (according to the supplied operator). A logical expression defines rules on the credential attribute of the constraint subject (user or role). As an example, an assignment constraint can state that "role `r` can access resource `o` if (a) the access occurs between `9AM and 5PM` during the month of `January` in year `2006`, and (b) the `location` is "London" and the `system load` is "low". Here, (a) is an example of a temporal constraint, represented as a temporal constraint expression in X-GTRBAC, and (b) is an example of a non-temporal constraint represented as a set of logical expressions. To evaluate this logical expression, the role `r` must supply a credential having the attributes `location` and `system load`.

For our current work, we are particularly interested in expressing location-based contextual constraints. A naïve approach would be to encode a location-based constraint by defining a predicate on a location attribute. However, this approach is insufficient to capture many useful location abstractions that are needed in modern day pervasive computing applications, as we have discussed in Section 2. Instead of redesigning yet another spatial extension, we incorporate the rich spatial constraint specification already provided by the GEO-RBAC model, and design a profile that will help us integrate GEO-RBAC constraints with the constraint specification mechanism provided by X-GTRBAC. In the following section, we describe the results of this effort- the GEO-RBAC profile of X-GTRBAC.

4.2. GEO-RBAC profile of X-GTRBAC

As described in Section 3, GEO-RBAC extends RBAC with spatial constraints introduced using the concepts of role schema and role instances. The main idea is essentially that a spatial role (i.e. role instance) is associated with a position in space called role extent. All spatial roles adhere to a role schema having the same name as the spatial role. The schema defines the type of role extent, the type of logical position that can be assumed by the role, and the mapping function used to map physical positions of users onto a logical position for comparison with the role extent. A spatial role corresponding to a role schema can only be enabled when the logical position of the role is "contained in" the role extent. GEO-RBAC is primarily concerned with the mechanism for specifying the various feature types and geometries used to define logical locations, and how such a decision process is integrated into the RBAC model. Our goal now is to express these concepts using the policy specification language of the X-GTRBAC framework to allow flexible and modular administration of spatial constraints within the X-GTRBAC framework. In other words, we will develop a GEO-RBAC profile of X-GTRBAC.

To provide a clear understanding of the development of the profile, we use a concrete example involving the GEO-RBAC concepts of role schema and role instance. Consider a PurdueECEStudentRole. Suppose that we wish, as a matter of policy, to define the role extent of this role as the ECE campus sector in Purdue University. We wish to develop a methodology to encode this policy as a spatial constraint within the X-GTRBAC framework. As a side note, neither GEO-RBAC nor our current work is concerned with the details of determining the location of a user, or making any claims about the purported location of a user. Mechanisms are being researched to determine the authenticity of purported contextual information. For example, in the context of sensor networks, a subject (prover) can propose a location and undergo a simple protocol with an authentication system (verifier), the result of which allows the verifier to infer bounds on the subject's location. This is an interesting topic of research, and results from these researches can be easily integrated in the implementation of the spatial profile of X-GTRBAC.

Returning to the example of the PurdueECEStudentRole, and the associated spatial constraint, we are faced with a key challenge in attempting to find a straightforward translation of this constraint into X-GTRBAC (or any other RBAC-based framework). This challenge stems from the fact that GEO-RBAC uses the notion of role schemas as distinct from role instance, whereas the standard RBAC does not make this distinction (i.e., all roles are treated as instances, and there is no notion of a schema.) Since X-GTRBAC follows the RBAC standard, there is no direct way of incorporating this concept. However, it is also important at the same time to capture this behavior because the design of spatial constraints in GEO-RBAC relies on the use of this concept. A second, more obvious issue is the representation of GEO-RBAC spatial constraints and the notion of role extent using the X-GTRBAC language specification.

The essential idea used in our profile is to map the GEO-RBAC spatial constraints to enabling constraints of roles in X-GTRBAC. These constraints are encoded, for the most part, using the existing X-GTRBAC credential type system, as described in Table 1. The credential type system forms the basis of constraint evaluation in X-GTRBAC and constraints are defined and evaluated based on an associated credential type. As we have indicated, both users and roles can have credentials. Credentials may have attributes that characterize the entity holding the credential. User credentials can be used in assignment constraints to decide on user-to-role assignment, whereas role credentials can be used in enabling and/or activation constraints to make enabling and/or activation decisions. Credential types therefore act as a layer of abstraction between the policy and the constraint definition, since credential types are defined by the system designer, and are used by the administrators to define constraints. It is this abstraction that we leverage to not only introduce the concept of a role schema (as per GEO-RBAC) in X-GTRBAC, but also associate with it a set of spatial constraints that are inherited by the respective spatial roles (i.e. role instances).

4.2.1 Schema credential types

To incorporate the notion of a GEO-RBAC role schema in X-GTRBAC, we extend the credential type system with the introduction of a schema credential type, with the following semantics:

- Credential types in the X-GTRBAC framework are either a schema credential type or a non-schema credential type.
- A non-schema credential type explicitly refers to the associated schema credential type (if any).
- Roles that include an enabling constraint[1] defined on a credential type are said to “instantiate” that credential type.
- Roles that instantiate a schema credential type are equivalents of a role schema in GEO-RBAC.
- Roles that instantiate a non-schema credential type are instances of the roles that instantiate the corresponding schema credential type (if any), and inherit all constraints defined on the latter.

We note that while spatial roles and corresponding role schemas in GEO-RBAC have the same name, we do not follow this convention in our profile. The roles that instantiate schema credential types do not have the same name as those that instantiate non-schema credential types. This is again because there is no separate concept of a role schema in X-GTRBAC; all roles belong to the same class (i.e. Roles), and therefore no two roles in the system can have the same name. The inheritance association between roles is achieved, instead, through the referencing mechanism involving the credential types. This means that a role designed as a role schema will instantiate a given schema credential type, and a role designed as a spatial role that corresponds to this role schema will instantiate a non-schema credential type which refers to this schema credential type. At present, we require that only one role exists that instantiates a given schema credential type at one time. The reason for this restriction is to avoid the complexities of multiple inheritance in system design and implementation. This complexity can easily be avoided in practice by choosing the right granularity of roles that instantiate schema credential types. For example, in our case, we can define a PurdueStudentSchemaRole that instantiates a PurdueStudentSchema credential type, with an associated spatial enabling constraint which can then be inherited by all instances of this role. The role extents of each spatial role in the system (such as PurdueECEStudentRole or PurdueMEStudentRole) can be compared against the associated spatial constraint to determine if the spatial role can be enabled. Also, all constraints relevant to spatial student roles on Purdue campus can be included in the singly created PurdueStudentSchemaRole.

This notion of inheritance allows us to associate spatial constraints with multiple spatial roles by following the same convention as in GEO-RBAC: the role extent can be defined for each role designed as a spatial role, whereas the information regarding logical position and mapping functions can be defined in the role designed as the equivalent of role schema. This, however, requires support for representing the concepts used to model these elements in GEO-RBAC.

4.2.2 GML Support

The notion of role extent in GEO-RBAC uses the concepts developed by the GIS community which can be represented quite naturally in GML. GML is an XML-based grammar for modeling, communicating, and storing geographic information represented according to the conceptual modeling framework adopted as standard by the Open Geospatial Consortium [15]. The key concept for the modeling of real world phenomena is that of *geographical feature* (or *spatial feature*). A spatial feature represents an abstraction of a real world phenomenon associated with a location relative to the Earth. Spatial features are characterized by named and typed properties, which depend on *feature type* definition. A specific category of features are the *spatial features with geometry*, that is, features in which properties may be of geometric type, specifically point, line, polygon or collection of geometric elements. For example, a road can be represented by a feature in which a geometry-valued property describes the road as a line. GML defines the various entities of interest such as features and geometric elements through a hierarchy of *GML objects*. GML objects correspond to XML elements of a type derived directly or indirectly from the most general *AbstractGMLType*. Feature types are specific GML objects of type derived from *AbstractFeatureType*. Finally, a *GML Application Schema* is an XML Schema written according to the GML rules and defines a vocabulary of spatial objects for a particular application domain. To support the specification of role extents and spatial constraints in X-GTRBAC, we therefore represent them as GML elements.

To specify a role extent, we include in the credential type of the role an attribute representing a spatial feature type. The role extent is defined for each spatial role, whereas the information regarding logical position and mapping function is defined by the role that is designed as the equivalent of the role schema. Since the spatial constraint acts on the corresponding role instance, such an approach requires a parametric function evaluation based on the attributes of the role instance. For example, a role extent representing a Purdue ECE campus sector may be defined for a PurdueECEStudentRole, and a function “contained in” may be defined in PurdueStudentSchemaRole to determine if the logical position of the user enabling a spatial role is within the role extent defined for the role. Since the function must work with multiple role instances (i.e. not just PurdueECEStudentRole but also with, say, PurdueMEStudentRole), it is imperative that the role extent against which the comparison is to be made be supplied as a dynamic parameter based on the actual role instance.

4.2.3 Example

We provide an example of the use of GEO-RBAC profile within X-GTRBAC framework in Table 2. The top left side shows the schema credential type PurdueStudentSchema that is instantiated by a PurdueStudentSchemaRole (bottom left), while the top right side shows a credential type PurdueECEStudent that is instantiated by a

[1] While only enabling constraints are relevant to this profile, the semantics extend similarly to assignment and activation constraints.

PurdueECEStudentRole (bottom right). The link between the credential type PurdueStudent and the schema credential type PurdueStudentSchema is established by the ref attribute in the former which indicates that it links to the latter. Using this example, any user assigned to PurdueECEStudentRole will be constrained by the enabling constraint defined on PurdueStudentSchemaRole. This constraint states that any PurdueECEStudentRole can only be enabled when the logical position of the role instance is “contained in” the role extent defined for this role.

Table 2: Example of use of the GEO-RBAC profile for X-GTRBAC

```
<CredentialType cred_type_id="cPSS"
 type_name ="PurdueStudentSchema"/>
```

```
<CredentialType cred_type_id="cPECES"
 type_name="PurdueECEStudent"
 ref="PurdueStudentSchema">
 <AttributeList>
   <Attribute name="campus" type="Feature"
           usage="mand" />
 </AttributeList>
</CredentialType>
```

```
<Role role_id="rPSS"
 type_name ="PurdueStudentSchemaRole">
<CredType cred_type_id="cPSS" />
<EnabConstraint op="AND">
 <EnabCondition cred_type_id="cPSS">
  <LogicalExpression op=” AND”>
    <Predicate>
      <Operator>contained_in</Operator>
      <FuncName>getCampusSector</FuncName>
      <RetValue type="reference">
                 campus</RetValue>
    </Predicate>
  <LogicalExpression>
</EnabCondition>
</EnabConstraint>
</Role>
```

```
<Role role_id="rPECES"
 type_name ="PurdueECEStudentRole">
 <CredType cred_type_id="cPECES">
 <CredExpr>
   <Attribute name="campus">
    <Feature>
     <gml:extentOf>
      <gml:Envelope>...
        <!--definition of ECE campus sector-->
      </gml:Envelope>
     </gml:extentOf>
    </Feature>
   </Attribute>
 </CredExpr>
</CredType>
</Role>
```

The role extent for the role instance is given by the campus attribute in the PurdueECEStudent credential type. This attribute is of type Feature, which we use to define an element corresponding to a spatial feature type in GML. The value of this attribute for PurdueECEStudentRole is defined using GML extentOf element to be equal to the ECE campus sector in Purdue University. For sake of clarity and brevity, we have omitted the GML vocabulary from the figure, but it is shown in Figure A.1 in Appendix A. We note that the use of GML data types in our model allows us to represent high-level logical locations in addition to just a low-level location (for e.g. based only on IP address[2]), and we can therefore provide a more sophisticated form of access control. This is especially relevant in mobile environments where fixed IP addresses cannot be hard-coded into access control policies.

We now focus on constraint specification which leverages the GML concepts. As already mentioned, the spatial constraint is encoded as an Enabling Constraint expression of X-GTRBAC. The constraint is a combination of Boolean predicates, where each predicate comprises of a function, a comparison operator, an input parameter, and an expected return value which is compared with the output of the function according to the comparison operator. The constraint in the example essentially evaluates the following predicate: contained_in(getCampusSector(·), <PurdueECECampus>).The contained_in is a binary Boolean operator, so the predicate evaluates to either true or false, depending on the result of the operation. The operator compares the logical position of the role with the role extent (ECE campus sector) as indicated in the role definition. The current campus sector of the role (relative to a set of pre-defined sectors of the spatial system defined according to the GML notations) is returned by the function getCampusSector. If the sector is indeed contained in the Purdue ECE campus, the contained_in operator returns true. A key capability required here is the evaluation of the getCampusSector function in a non-obtrusive manner to preserve the modularity of the process of constraint administration. For example, it is desirable to have the X-GTRBAC system being able to call the location-specific APIs implementing this and similar functions in a modular and flexible manner. It is also desirable for the system to support a parameterized evaluation of the predicate based on the location attribute associated with the given instance of PurdueECEStudentRole. We will comment on the modular invocation of location-based APIs in the next section. Concerning the second requirement, we note that the parameterized evaluation of this predicate is made possible through the use of a technique similar to the use of parameter passing by reference or by value in programming languages. We use a type attribute in the RetValue tag to indicate the mode of obtaining the parameter. If the type attribute is set to value (which is the default in the system), then the text contained in the RetValue element is the actual parameter. If the type is reference (as in the case of GEO-RBAC profile), it means that the return value actually refers to the location attribute defined for the given PurdueECEStudentRole, which is then extracted and used by the system in evaluation of the predicate.

[2] An IP address can always be used in our scheme using a non-GML data type attribute (such as string).

While the GEO-RBAC profile focuses on enabling constraints, we mention that users can issue activation requests for a role once they are assigned to a role. In our example, the assignment policies for PurdueECEStudentRole will be defined according to the assignment constraint expression in Table 1. No assignment policies are allowed (or needed) for roles that instantiate schema credential types (PurdueStudentSchemaRole in our example), because a role schema in GEO-RBAC is an abstract entity, and no user can be assigned to it.

5. Architecture and Implementation

This section discusses the architecture implementing the GEO-RBAC profile within the X-GTRBAC system, and then provides an overview of the prototype.

5.1 System Architecture

We base our architecture on the general architectural framework for location-based pervasive computing applications depicted in Figure 2. Such a framework consists of three fundamental components:

- A set of *mobile users* equipped with mobile terminals and connected to a wireless network. Users are assumed to be identified by their terminal ID.
- The *Application Server* providing a set of location-aware information services.
- The *Access Control System* (ACS). It is the main component of the system processing the users' requests. In our current framework, it is the X-GTRBAC system.
- To obtain the position of the user, the ACS accesses a *Location Server* which aggregates location data from different sources such as the network and mobile terminals equipped with GPS (connected by dotted lines in Figure 2) and responds to queries such as *retrieve the position of terminal ID*.

The following are the key components of X-GTRBAC which acts as the ACS in the proposed architecture:

- The *XML Policy Base* (XPB) is a database storing the XML-based security policies specifying, among other information, the spatial roles, the services available to each role, and the roles assigned to each user.
- The *Session Management Module* (SMM) records the *status* of sessions. The status at time t of session s is represented by the tuple: <s, t, SR, ER, ATTR> where: SR is the set of session roles, thus the roles activated by the user in s at time t, ER is the set of roles enabled in s at time t, and is the superset of all session roles, ATTR is the set of role attributes corresponding to ER (among which `campus` attribute is included).

Figure 2: Generic Architecture for Location-Based Pervasive Computing Applications

A request for a service is processed as follows: the user requests the service by sending a request message to the ACS. The ACS then determines whether the request can be accepted and if it is the case the request message is properly re-structured and sent to the Application Server for fulfilment. The ACS applies the access control policy and applicable spatial constraints to determine the set of enabled roles and thus the services which are accessible to the user at a given location. Since the enabling constraints are checked before a user can activate any role in X-GTRBAC, the approach currently adopted in our system is user-driven and the position of the user is computed on demand and thus the status of roles is determined only when required. SMM periodically retrieves from the Location Server the position of the users of current sessions and determines whether a state transition has occurred for the roles of each session. If this is the case, it updates the status of sessions and notifies the event to the corresponding terminal.

5.2. Implementation

In our current prototype, we have implemented the ACS component of the architecture and simulated the functionality of location-based services through software routines. The ACS component is the one that implements the GEO-RBAC profile for X-GTRBAC. Because of our emphasis on modularity and flexibility of policy administration, we have not adopted what would be the most straightforward way to implement the profile: provide a hard-coded call interface to invoke the location-based API (currently comprising the single function getCampusSector). Instead, we opted for a more sophisticated technique that would allow us to invoke *any* external API to preserve the modularity in policy administration, while also allowing parametric evaluation of spatial constraints. Preserving modularity requires late-binding of constraint processing routines and it has been achieved through the use of Java Reflection API.

The reflective evaluation works by allowing one (usually the domain experts) to create a Java class with a method that takes as its parameter an X-GTRBAC Predicate object and returns a Boolean type. Whenever the X-GTRBAC system needs to evaluate a constraint that involves the use of this function, this class is instantiated using the Reflection API and the appropriate method is called. The format for specifying a function f in a user-defined class c which is a member of some package p is as follows: (p.)?f.c. Here, ? is used to denote 0 or 1 of (p.). That is, the class might be in the default package, but if it is not, then it is necessary to specify the package path. The default X-GTRBAC package is called XGTRBAC.functions and includes three broad classes of functions: (i) Entity - entity-related functions (such as getCredentialAttributeValue); (ii) Sys - system-related functions (such as getSystemLoad); and (iii) Environment - environment-related functions, which include the location-based GEO-RBAC API. All function calls are pre-pended with the path XGTRBAC.functions. So, for instance, Environment.getCampusSector method of the GEO-RBAC API will be invoked as XGTRBAC.functions.Environment.getCampusSector. The use of reflective evaluation is illustrated in Figure 3.

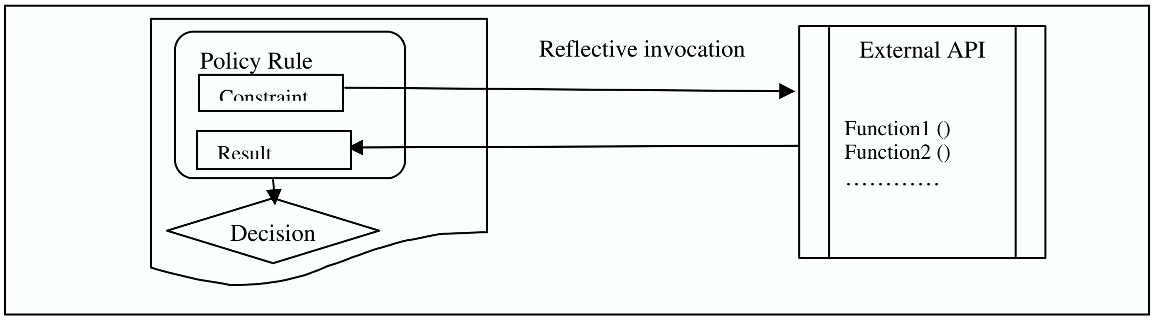

Figure 3: Mechanism for reflective constraint evaluation

In addition to the sophisticated and modular method call interface for constraint processing, the other significant modules of the architecture are those responsible for creating data structures to store GML data, creating the functions necessary to perform geometry calculations based on GML concepts, and extending the current parsers to take into account the GML data, especially handling complex data types, such as Feature. We also needed to extend the support for XML Schema in our system to the GML data types. The complete syntax of a campus attribute describing the campus sector in terms of GML vocabulary is given in Figure A.1 in Appendix A. The description involves the representation of the location coordinates describing the campus sector. The actual role location at the time the request is made is based on user location coordinates which are obtained from a software sensor in our prototype but can also be provided in practice by interfacing our prototype with a GPS receiver (which will be the next step in implementing the proposed architecture given in Figure 2). Using a software routine, the coordinates are updated whenever the role location changes, and the corresponding GML representation of a campus sector is then used for comparison in the spatial constraint. Figure A.2 in Appendix A shows a snapshot of dynamic evaluation of the spatial constraint in our example by comparing the campus sectors as GML-encoded feature types.

Although our profile is designed for X-GTRBAC, it can also be used by other access control frameworks through the invocation of an API for X-GTRBAC based on the standardized SAML protocol [16]. A SAML profile designed for X-GTRBAC [2] is used to issue access control requests to the system encoded as SAML Authorization Decision Query. This means that the profile we have designed acts as the Policy Decision Point (PDP) for heterogeneous context-aware applications in a pervasive computing environment, and the applications can serve as their own Policy Enforcement Point (PEP). Therefore, installation of the X-GTRBAC system is not required to avail the functionality of the profile we have developed in this work. We believe that this complements our goal of flexible administration since we have provided another level of modularity (vis-à-vis between PDP and PEP) in the policy administration process.

6. Conclusions and Future Work

In this paper, we have provided two-fold contributions. We highlighted the issue of flexible policy administration and modular constraint specification, while specifically focusing on spatial constraints for location-based applications in pervasive computing environments. In response to the outlined challenges, we have provided a location-based profile of the X-GTRBAC access control framework. The profile is based on GEO-RBAC, a spatially aware access control model. The use of the GEO-RBAC profile within the X-GTRBAC framework allows expressive spatial constraint processing capabilities to be integrated within an access control framework while also helps retain flexibility in policy administration. We have described the use of our profile in a location-based application. The profile has also been implemented within the existing X-GTRBAC system, and we have highlighted the key features of the system architecture. The profile can also be used by other access control frameworks through the invocation of an API for X-GTRBAC based on the standardized SAML protocol.

While our profile addresses the most relevant issues for spatial access control described in the GEO-RBAC model, there are certain aspects that have not been covered in the current profile. We have not considered the hierarchical model (GEO-HRAC), nor have we considered GEO-RBAC with SoD constraints. SoD constraints in GEO-RBAC are more fine-grained than those supported by X-GTRBAC (which are based on SoD constraints as defined in RBAC), and involve the notion of location-specific SoD. Also, GEO-RBAC allows objects to be defined using subsets of feature types, called feature type extensions, which may include spatial and non-spatial feature types. Our profile only deals with spatial feature types, and also treats objects as independent entities (i.e. without any spatial relevance). There are some extensions needed in the implementation of the current profile to support a wider range of feature types defined in GEO-RBAC; it defines three feature types, that is, points, lines, and polygons, whereas only points and envelopes (rectangles) are currently supported. As part of future work, we plan to extend the profile to provide a more comprehensive coverage of GEO-RBAC model.

We also plan to extend the implementation prototype with support for more features from the proposed architecture. We are also working on an extension that will allow an event-driven strategy for managing sessions, where the status of each role is determined asynchronously with respect to user requests. The advantage of this approach would be that user requests can be more efficiently processed because the current status of roles is available at the time of the request, and the user can only submit requests for roles which are enabled for the current position of the user. A consequence of adopting this approach, however, would be the requirement to adjust the access control when the user position changes from the one against which the access had been granted. This requires support for usage control mechanisms [14] which allow access control to persist after the access has been granted in order to let the system react to such an event. We will extend the existing PEP infrastructure of X-GTRBAC to allow support for usage control in our framework, which in conjunction with the GEO-RBAC profile would enable support for continuous access control in presence of user mobility in location-based applications. Finally, we plan to test the implementation prototype of our system by installing it in a realistic pervasive computing environment.

References

[1] E. Bertino, B. Catania, M. L. Damiani, P. Perlasca, "GEO-RBAC: A Spatially-Aware RBAC", Proceedings of the tenth ACM symposium on Access control models and technologies, Stockholm, Sweden, June, 2005.

[2] R. Bhatti, E. Bertino, A. Ghafoor, "A Policy Framework for Access Management in Federated Information Sharing", In proceedings of 2005 IFIP TC-11 WG 11.1 & WG 11.5 Joint Working Conference on Security Management, Integrity, and Internal Control in Information Systems, December 2005

[3] R. Bhatti, J. B. D. Joshi, E. Bertino, A. Ghafoor, "X-GTRBAC: An XML-based Policy Specification Framework and Architecture for Enterprise-Wide Access Control", ACM Transactions on Information and System Security (TISSEC), Vol. 8, No. 2.

[4] R. Bhatti, J. B. D. Joshi, E. Bertino, A. Ghafoor, "X GTRBAC Admin: A Decentralized Administration Model for Enterprise Wide Access Control", In proceedings of 9th ACM Symposium on Access Control Models and Technologies, New York, June 2-4, 2004.

[5] S. M. Chandran, J.B.D. Joshi, "LoT-RBAC: A Location and Time-Based RBAC Model", Proceedings of the sixth international conference on Web Information Systems Engineering (WSIE), New York, NY, 2005.

[6] M. Covington, W. Long, S. Srinivasan, A.K. Dey, M. Ahmed, G.D.. Abowd, "Securing Context-Aware Applications using Environment Roles", Proceedings of the sixth ACM symposium on Access control models and technologies, Chantilly, VA, June, 2001.

[7] J. Crampton, G. Loizou, "Administrative scope and role hierarchy operations," In proceedings of 7th ACM Symposium on Access Control Models and Technologies, June 2002

[8] J. Hu, A. C. Weaver, "Dynamic Context-Aware Access Control for Distributed Healthcare Applications", In proceedings of First Workshop on Pervasive Security, Privacy and Trust (PSPT), Boston MA, 2004.

[9] R. J. Hulsebosch, A.H. Salden, M.S. Bargh, P.W.G. Ebben, J. Reitsma, " Context-Sensitive Access Control", Proceedings of the tenth ACM symposium on Access control models and technologies, Sweden, June, 2005.

[10] A. Kern, A. Schaad, J. Moffett, "An administration concept for the enterprise role-based access control model", In proceedings of 8th ACM Symposium on Access Control Models and Technologies, June 2003

[11] P. McDaniel, "On Context in Authorization Policy", Proceedings of the eighth ACM symposium on Access control models and technologies, Como, Italy, June, 2003.

[12] R. S. Sandhu, E.J. Coyne, H.L. Feinstein, C.E. Youman, "Role-Based Access Control Models", IEEE Computer 29(2): 38-47, IEEE Press, 1996.

[13] R. Sandhu and Q. Munawer, "The ARBAC99 model for administration of roles", In Proceedings of the 15th Annual Computer Security Applications Conference, Dec 1999.

[14] Z. Zhang, J. Park, F. P. Presicce, R. Sandhu, "A Logical Specification for Usage Control", Proceedings of the ninth ACM symposium on Access control models and technologies, Yorktown Heights, NY, June, 2004.

[15] OpenGIS Consortium. OpenGIS Geography Markup Language (GML) Implementation Specification. Technical Report OGC 02-023r4, 2003.

[16] OASIS SAML. http://xml.coverpages.org/saml.html

Appendix A

```
<CredType cred_type_id="cS" type_name="Student">
   <CredExpr>
      <Attribute name="campus">
        <Feature>
          <gml:name>Purdue ECE Campus</gml:name>
          <gml:description>Purdue University ECE Campus Sector</gml:description>
          <gml:extentOf>
            <gml:Envelope>
               <gml:lowerCorner>0 0</gml:lowerCorner>
               <gml:upperCorner>100 100</gml:upperCorner>
             </gml:Envelope>
            </gml:extentOf>
        </Feature>
      </Attribute>
   </CredExpr>
</CredType>
```

Figure A.1: GML representation of a "campus" role extent defined for a role instantiating the credential type "PurdueECEStudent" (cPECES): it uses a set of coordinates to represent a bounded rectangular area as the ECE campus sector. (The coordinates used are for demonstration purpose only.)

(a)

(b)

(c)

(d)

(e)

(f)

Figure A.2: Snapshots of dynamic spatial constraint evaluation: (a)options for a user session (b) user "john" attempting to activate the role "PurdueECEStudentRole" (c) location coordinates of user outside ECE campus sector are supplied (d)system detects the coordinates to be outside the role extent, and role is not enabled (e) location coordinates of user within ECE campus sector are supplied (f) coordinates are detected to be within the role extent, and role is enabled and allowed to be activated by "john"