

[0036] The actor-role scope key does not have to be a scalar value. The actor-role scope key can include a number of data values as previously illustrated. A actor-role scope key can resolve to one or more sets of people, but actor-role scope keys (and scope) can also resolve to entities that are not sets of people. For example, a role could “scope” an actor to a set of mutual funds. For this example, the actor-role scope key would resolve to a list of mutual funds that the actor could work on within a given application.

[0037] Actor-role scope keys can refer to a hierarchal scope structure. A scope structure can include hierarchical organizations, such as location, division, and/or department, or personnel hierarchies, such as executive, director, and/or employee, and/or any combination thereof. The ability of a actor-role scope key to resolve organizational scope could, for example, be facilitated by the following elements: a hierarchy actor-role scope, a navigation actor-role scope, and a base actor-role scope (referred to as a population group). Additional elements can also be defined as requirements dictate. Hierarchy and navigation can be used as starting points for a base actor-role scope. The hierarchy value can describe the type of hierarchy represented by the resolved scope. For example, possible values for “hierarchy” can include, “organization unit hierarchy” or “reporting hierarchy.” The navigation value can describe how an actor may interact with resolved scope based on an associated hierarchy value. For example, a navigation value could include “all-levels down” and/or “same level only.” Base actor-role scope values can describe one or more pointers into scope. The base actor-role scope can have values such as “work location” and/or “job grade code.” In addition, base actor-role scope values can contain attributes such as inclusion and exclusion rules and values. For example, exclusions allow the system to incorporate special cases where the resolved scope for a user, for example in human resources, can include all company employees, excluding certain top-level executives. Here, the actor-role scope can be a company, so all employees are included, with an exclusion of executives X, Y, and Z.

[0038] Scope is resolved (304d) based on, for example, the role of an actor, one or more pre-defined context parameters, a role scope, and a actor-role scope key. Actor-role scope key values allow for determination of the resolved scope, which includes, for example, the entities, system components, or resources that a user 110 is authorized to act on when he or she is working in a specified role. There may be one scope associated with a role, though a scope may be associated with many roles. Scope can define the entities (who and/or what) the actor is allowed to act on when operating in a particular role. Resolved scope can be dynamic such that the resolved scope would not, for example, be stored by the role-based authorization system.

[0039] One possible example of the “resolve role” process 304 can include a user 110 who has a designated role of “financial manager,” and a role scope that includes, for example, “department name” and “direct reports.” This means that during the assignment process of a user 110 to the “financial manager” role, a “department name” and “direct reports” are required to resolve the scope of the “financial manager” role. In this example, the actor-role scope key enables resolution of scope that includes, for example, the “purchasing department” for the department name and direct reports that include, for example, “John Smith and Edward Jones.” For this example, the financial manager can act on

John Smith and Edward Jones who report directly to this financial manager in the purchasing department.

[0040] FIG. 5 illustrates an exemplary process 500 for determining a policy instance based on a particular role and context parameters. A policy instance can be a unique instance of a policy type. Policy instances of a given policy type differ based upon the associated role and the one or more pre-defined context parameters. A policy instance for a user 110 can be derived from, for example, the identity of the user, the role of the user, the one or more pre-defined context parameters, and/or the policy type associated with the role of the user. For a given role and one or more pre-defined context parameters there can be one or more instances of a given policy type.

[0041] Since each policy instance is based, in part, upon certain pre-defined context parameters, customized policy instances can be associated with, for example, specific “practices,” “market segments,” “clients,” and/or other pre-defined context parameters. The use of role-related context parameters can provide a mechanism to support policy instance customization. Session-related context parameters can add an additional level of restriction on policy instances by allowing for the application of globally limiting factors at the role level. Session-based context can include, for example, the channel used to access the application (e.g., intranet, Internet, etc.), the authentication strength, and the like.

[0042] Context parameters can be hierarchical in nature. A context parameter hierarchy can consist of, for example, a number of context parameters listed in increasing order of precedence. Each context parameter hierarchy level identifies a potential policy instance override point, e.g., 501, 503, and/or 505. Context parameter hierarchies can have a root or default level.

[0043] Policy instances can be set at one or more levels based on hierarchical context parameters that include, for example, parameters “1” (the root or default), “2,” and/or “3” as shown in FIG. 5. The root level (e.g., determined by context parameter 1) can define 502 a default policy instance, (e.g., default policy instance 1) for a role. In the example illustrated in FIG. 5, default policy instance 1 can be overridden at one or more of the hierarchical levels 501, 503, and/or 505. Default policy instance 1 includes a complete definition of policy element name/value pairs. Default policy instance 1 implies the default context, that is, default policy instance 1 can match any context when no other context parameter match occurs. Policy instances at lower levels (e.g., 504, 506, and/or 507), such as policy instances 2, 3, and/or 4, can define those policy elements that override the default to a policy element value set at a higher level in the hierarchy. This allows for any changes to policy elements higher in the hierarchy to propagate downwards. A policy element for the policy type does not have to exist at each level below the root of the hierarchy. The absence of a policy element at a lower level implies the inheritance of the policy element value(s) from the next level up.

[0044] In one example, the system 100 can determine (501) whether context parameter 1 has a value of “X,” which can include a parameter such as the “practice” of the user. If context parameter 1 does not have a value of “X,” then the default policy instance 1 is associated (502) with the particular user. If the value of context parameter 1 is “X,” then the system 100 determines whether default policy instance 1 can be overridden, and if so, by which other policy instance.