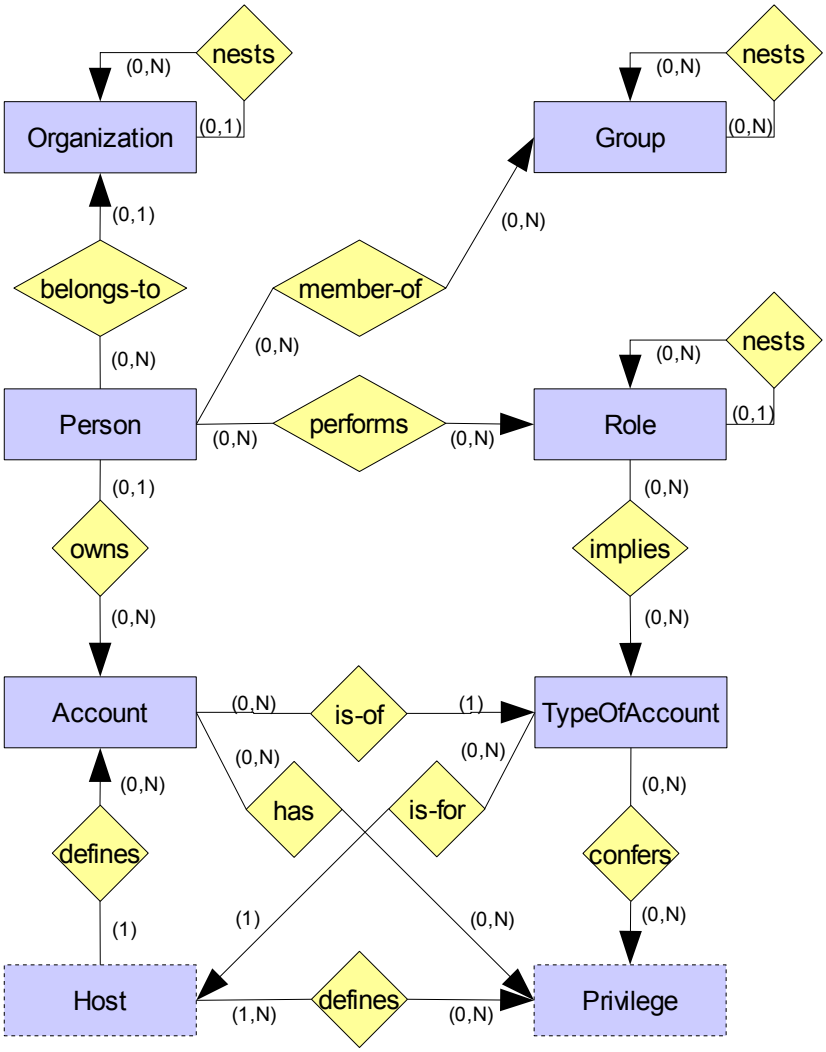
1. Introduction

2. Overview

2.1. Domain Model for Identity Management

This section introduces entities and relationships common to the domain of identity management.

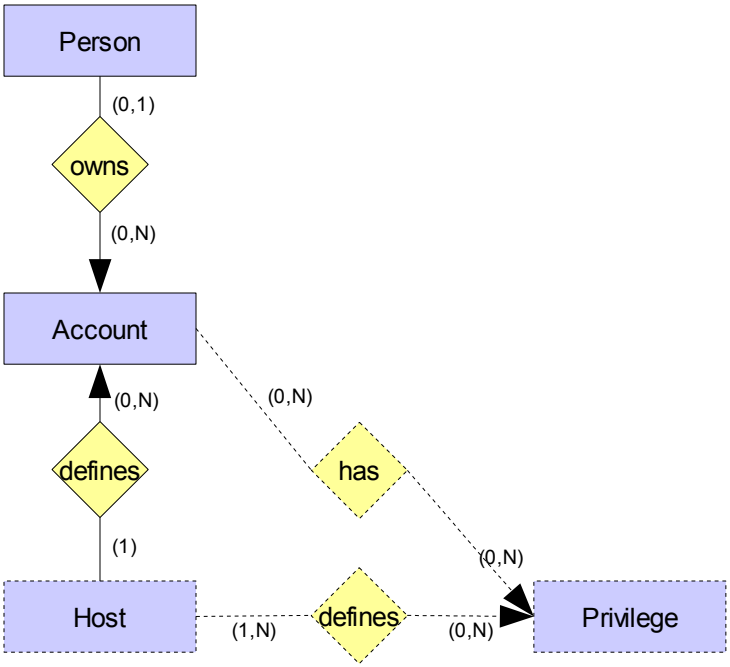


Each of the following subsections presents a subset of the domain model, beginning with the most familiar:

- The first subsection below presents Person, Account and Privilege.
- The next subsection below presents Organization, Group and Role.
- A third subsection below presents Role and Type-of-Account.

A final subsection entitled "SIMPLEST Relationships" discusses how the SIMPLEST Schema uses object classes and attributes to represent these entities and relationships. The section entitled "SIMPLEST Schema" presents these object classes and attributes more formally.

2.1.1. Person, Account and Privilege



The Person and Account schema entities are fundamental to Identity Management. An instance of Person normally represents a human being. An instance of Account normally represents a person *within the scope of a particular computer system or application*. **Each person may *own* (that is, may be responsible for) any number of accounts. At most one person may own each account.**

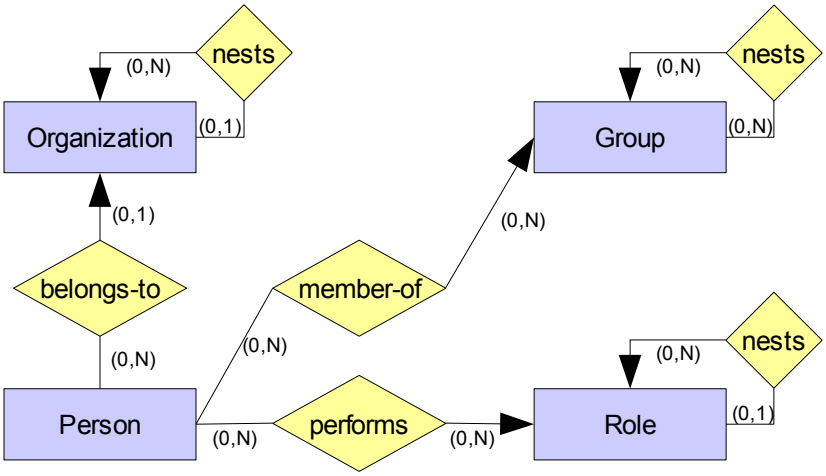
A Host is a computer system or application that defines accounts. **A host may define any number of accounts. Exactly one host defines each account.**

The concept of a Host is closely related to SPML's concept of a Target. A Host is a *physical endpoint* for provisioning, whereas a Target is a *logical endpoint* for provisioning that a provider exposes to requesters. An SPML provider may expose a host as a target. On the other hand, rather than expose an actual host, an SPML provider may expose as a target an abstract collection of hosts or (may expose as a target) a functional description that is more like a role. In short: **A host may be a target, but a target is not necessarily a host**.

Each Host determines the set of privileges that it supports and determines what each privilege means. **Each host may define any number of privileges. At least one host must define each privilege.**

A Privilege (represents a characteristic of the account that) allows an account to perform a specific action on a host. **Each account may have any number of privileges. Any number of accounts may have each privilege.**

2.1.2. Organization, Group and Role



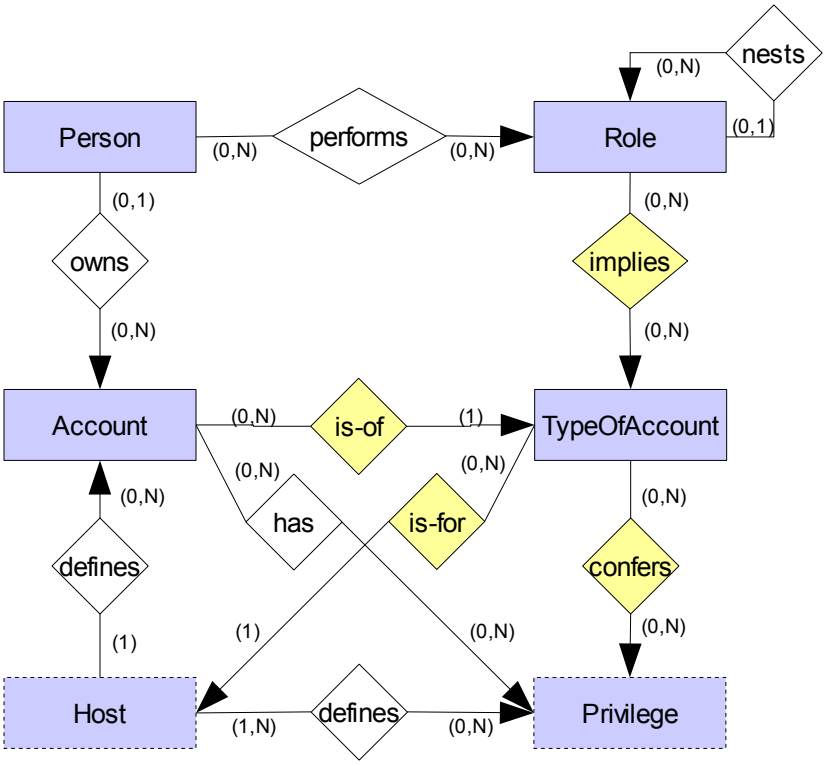
The Organization schema entity is ubiquitous in directory services (and therefore is common in identity management systems). An instance of Organization usually represents the management structure of a corporate entity—that is, an entity that consists of more than one person. The most common management structure is a hierarchy: **Each organization may nest any number of organizations. Exactly one organization nests each organization (except the topmost, which none nests).**

Persons are "leaf" nodes in an organizational hierarchy. **Each person may belong to at most one organization. Any number of persons may belong to each organization.**

The Group schema entity usually represents an arbitrary collection of persons. (A group need not contain persons, but typically does.) **Each person may be a member of any number of groups. Any number of persons may be a member of each group.** Classically (as derived from Unix groups) a group cannot contain other groups, but many modern systems and applications allow this. Many modern groups may form hierarchies—or may form structures more flexible than hierarchies. **Each group may contain any number of groups. Any number of groups may contain each group.** Whoever contains groups is responsible for preventing cycles—that is, a group must not contain itself directly or indirectly. The most important difference between Group and Organization or Role is *semantic*: Group membership is assumed to be orthogonal to (that is, a dimension independent of) both organizational hierarchy and job function.

The Role schema entity represents a job function that each person may perform. Like group membership, role membership is not exclusive. **Each person may perform any number of roles. Any number of persons may perform each role.** Like organizations, roles may be nested to form a hierarchy. **Each role may nest any number of roles. At most one role may nest each role.** However, role is assumed to be *orthogonal to organization*. That is, a role hierarchy represents (a taxonomy of job function that is) a dimension independent of management hierarchy. The semantic difference between Group and Role is that group membership is generally "shallow"--that is, group membership entails little or no data beyond the fact of membership. Role membership is usually "deeper": a role may confer privileges that govern access to specific targets. The next section entitled "Role and Type-of-Account" discusses this further.

2.1.3. Role and Type-Of-Account



A role usually implies some type of account. (That is, each job function that is modeled as a role usually requires that the person be granted some level of access to a host.) **Each role may imply any number of types of account. Any number of roles may imply each type of account.**

In the simplest case, a role simply implies that a person should have at least a basic access to a target. That unqualified assignment of a host—the "default" type of account—implies a normal or standard account for that target. In some cases, however, a role may also imply a specific type of account—for example, an "administrator" account. A specific type of account (for example, an "administrator" account) has specific set of privileges on the target. **Each type of account may confer any number of privileges. Any number of types of account may confer each privilege.**

NOTE: Identity management systems differ in the extent to which each supports Role-Based Access Control and in the manner in which each supports it. However, the fact that a role implies a specific type of account for a target (rather than conferring privileges onto whatever accounts for that target that person owns) becomes clear when a role (or when the set of roles that a particular person performs) implies more than one type of account for the same host. This is especially clear when a person must use each type of account for a distinct purpose.

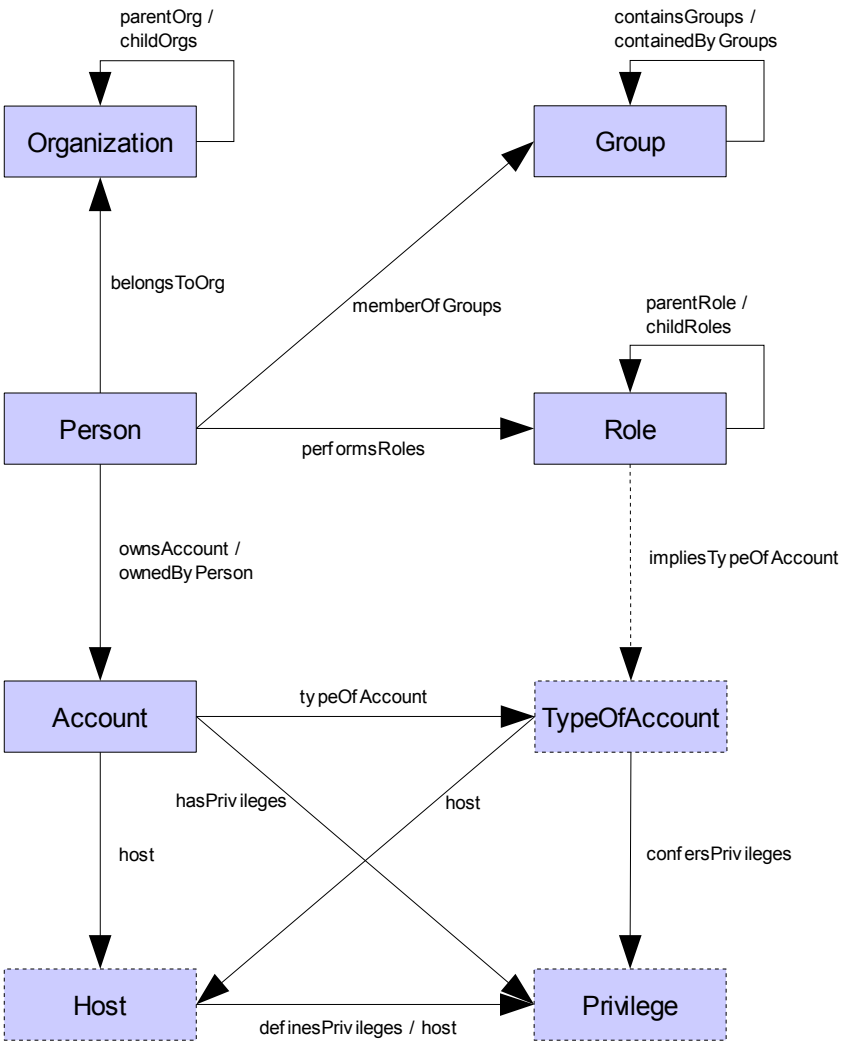
Imagine the following situation:

- An "HRUser" role implies a normal "user" account on the "HR" target.
- An "HRAdministrator" role implies a special "administrator" account on the "HR" target.
- A person who has both roles—and who is therefore both an administrator and a user—must use the special "administrator" account to perform all administrative functions and must user the normal "user" account to perform all "end-user" functions. This enables the company to keep a clean audit log of who did what when—and in what capacity.

If the person gains a “SuperUser” role that also implies a special “administrator” account on the “HR” target, then there should be no net change (even if that person subsequently loses the “HRAdministrator” role). If the person loses both the “HRAdministrator” role and the “SuperUser” role, that person should lose the special “administrator” account on the “HR” target but that person should keep the normal “user” account.

2.1.4. SIMPLEST Relationships

SIMPLEST defines an *object class* to represent each of the schema entities in the domain model for identity management. SIMPLEST defines (for each of these object classes) *attributes* that represent relationships between (instances of) these object classes. Reworking the domain model to show relationships in terms of attributes yields the following diagram.



Person, Account and Privilege.

SIMPLEST defines Person and Account as object classes. SIMPLEST uses attributes of these object classes to represent relationships between (instances of) Person and Account. An instance of Person may expose an “owns-Account” attribute. The “owns-Account” attribute may have multiple values. Each value of the “owns-Account” attribute identifies an instance of Account for which the person is responsible. SIMPLEST also represents the inverse relationship: an instance of Account may expose an “owned-By-Person” attribute. The “owned-by-Person” attribute may have at most one value. Any value of the “owned-by-Person” attribute identifies the (instance of Person that represents the) person who is responsible for the account.

SIMPLEST currently does not define Privilege as an object class. Instead, SIMPLEST represents each privilege as the value of an attribute. An instance of Account may expose a “has-Privileges” attribute. Each value of the “has-Privileges” attribute identifies a privilege that the account has.

SIMPLEST currently does not define Host as an object class. Instead, SIMPLEST represents each host as the value of an attribute. An instance of Account has a “host” attribute that identifies the host that defines the account.

NOTE: Many identity management systems conflate (that is, do not distinguish between) Person and Account. Therefore SIMPLEST defines many of the same attributes for both of these two schema entities (so that an instance of either schema entity will be appropriate for many purposes). Nonetheless, an SPML requester or provider that uses the SIMPLEST profile SHOULD clearly distinguish clearly between Person and Account.

Organization, Group and Role.

SIMPLEST represents the hierarchical nesting of organizations using the “parentOrg” and “childOrgs” attributes of Organization. SIMPLEST allows an instance of Person to refer to an instance of Organization using the “ou” attribute (A.K.A. “member-Of-Organization”).

NOTE: SIMPLEST Organization could expose a “has-Members” attribute that would allow an organization to refer to each person that the organization contains. However, this approach tends to scale poorly because a “has-Members” attribute may have a large number of values. This approach also introduces a requirement to synchronize the “has-Members” attribute with any inverse attribute such as “member-Of-Organization”. It is usually better simply to have each instance of Person refer to an instance of Organization.

SIMPLEST allows group nesting using the “contains-Groups” and “contained-By-Groups” attributes of Group. SIMPLEST allows a person to refer to any number of groups by means of the “member-of-Groups” attribute of Person. This approach scales better than having a Group refer to each of its members—see the discussion of “hasMember” above in this section.

SIMPLEST allows a role nesting using the “parentRole” and “childRoles” attributes of Role. The “roles” attribute of Person allows a person to refer to any number of roles. This approach scales better than having a Group refer to each of its members—see the discussion of “hasMember” above in this section.

NOTE: Group and Role are sometimes conflated--much as Person and Account are sometimes conflated. SIMPLEST therefore defines the Group and Role schema entities with many of the same attributes. Nonetheless, an SPML requester or provider that uses the SIMPLEST profile SHOULD clearly distinguish clearly between Group and Role.

Role and Type-of-Account.

SIMPLEST does not currently define a “Type-of-Account” as an object class. Instead, SIMPLEST represents each type of account as the value of an attribute. An instance of Account has a “type-of-account” attribute that identifies the type of the account.

The section entitled “SIMPLEST Schema” presents more formally these same schema entities and relationships.