Role-Centric Circle-of-Trust in Multi-Tenant Cloud IaaS

Navid Pustchi and Ravi Sandhu

Institute for Cyber Security and Department of Computer Science
University of Texas at San Antonio, San Antonio, Texas
tam498@my.utsa.edu,ravi.sandhu@utsa.edu

Abstract. Currently, collaboration is a major challenge in adopting cloud Infrastructure-as-a-Service (IaaS). Enterprise work-flow intrinsically mandates collaboration across its tenant boundaries as well as with associated organizations' tenants in the cloud. In this paper, we investigate a Circle-of-Trust approach where tenants establish trust within a circle of tenants for the purpose of collaboration. We present a novel extension of role-centric access control models to provide collaboration in the context of homogeneous and heterogeneous circles. In a homogeneous circle, our approach allows tenants to equally assert cross-tenant user assignments to enable access to shared resources. In a circle with non-uniform tenants, attributes are added to distinguish user-assignments where tenants are differentiated by type in the heterogeneous circle. Particularly, tenant-trust relation is established within a group of tenants authorizing user-role assignments across tenants.

Keywords: Attribute-Based Access Control, Collaboration, Federation, Circle-of-Trust, Multi-Tenant, Authorization, Security.

1 Introduction

Cloud IaaS is firmly accepted by enterprises for its cost benefits, reliability, and dynamicity at scale [12]. Its benefits are well documented and well practiced in the industry, but still organizations resist to fully migrate to cloud IaaS which arises from security, performance, and vendor lock-in concerns. Enabling collaboration mitigates such concerns regarding vendor lock-in and different security levels required, and improves performance by utilizing distinct cloud providers.

In multi-tenant platforms which utilize shared physical infrastructure, users' data are isolated into tenants to protect privacy and integrity. A tenant could be an organization, a department of an organization, or an individual cloud consumer, which is represented by an account in AWS [1] or a domain in OpenStack [2]. Furthermore, current cloud service providers offer federation APIs to enable collaboration between tenants such as AWS and OpenStack platforms. Besides federation between two tenants, collaboration can also be established between a set of organizations where tenants adhere to a common set of policies, trust relations and collaboration interfaces within a circle. We denote this

collaboration model as a *Circle-of-Trust*. Scenarios such as a large enterprise with multiple tenants collaborating in a public cloud, an organization with tenants across public and private clouds, or tenants from multiple organizations performing collaborative tasks are motivating use cases for Circle-of-Trust.

In this paper we present novel role-based and role-centric attribute-based access control models to enable federation in a multi-tenant cloud IaaS Circle-of-Trust. Our scope of contribution is homogeneous and heterogeneous multi-tenant circles in cloud IaaS.

To better clarify the concept, consider the example in Figure 1 where ACME, a multinational technology corporation, aims to implement its enterprise requirements with cloud services. ACME migrates its IT infrastructure to a public cloud service provider where each tenant represents a department. ACME utilizes multiple tenants to satisfy distinct security levels required for each department. For example, Finance Dept. resources should not co-locate in the same tenant with Research & Development Dept., as Finance Dept. retains sensitive data. Furthermore, ACME organizational structure demands collaboration between its departments which is thereby required in its cloud adoption. To this end, ACME establishes a Circle-of-Trust among its tenants in the cloud and starts adding its tenants to the circle. For instance a new tenant created as Sales tenant in ACME, requests to join the circle. Adding additional tenants requires all ACME circle members to agree on trusting the new Sales tenant. When Sales tenant joins the circle, it trusts members assertions and its assertions are likewise trusted by other ACME circle members. In particular, Circle-of-Trust offers an association of ACME principals to collaborate in the circle.

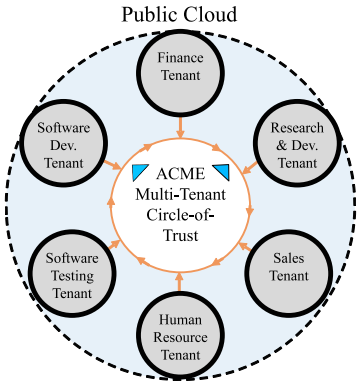


Fig. 1: ACME Corporation Multi-Tenant Circle-of-Trust.

Role-based access control (RBAC) [5, 16] and its variations has been successfully applied to cloud IaaS providing collaboration within single-cloud [17, 18] and multi-cloud systems [13]. In RBAC access permissions are assigned to roles and roles are assigned to users. Roles are central to RBAC for formulating policy and its commercial success, where it abstracts permissions into roles and role relations. With its dominance for the past two decades, RBAC limitations have been recognized leading to a push towards using attributes [6, 7, 15] with roles [9]. One method, is to add attributes to roles as role-centric attributes which takes advantage of roles' simplicity and attributes flexibility [8]. Attributes are defined as name:value pairs representing entities' properties. We anticipate cloud service providers will incorporate ABAC features to their current RBAC models such as role-centric to adopt convenience of RBAC with flexibility of ABAC models.

Our contribution in this paper is to design multi-tenant role-centric models with cross-tenant user-assignments. To our knowledge this is the first work considering role-centric models in Circle-of-Trust context.

The remainder of this paper is organized as follows. Section 2 overviews trust properties applicable in Circle-of-Trust and corresponding trust relations between tenants. In section 3, our multi-tenant role-based access control in circle denoted MT-RBAC$_c$ is proposed and specified. Section 4 introduces our multi-tenant role-centric attribute-based model in circle denoted MT-RABAC$_c$. Related work and conclusion is presented in sections 5 and 6 respectively.

2 Concept of Trust in Circle

In a Circle-of-Trust, trust relationships are defined between all circle entities. We use terms entities and principals interchangeably. Principals make assertions in the circle, assigning users to roles.

2.1 Trust Properties in Circle

Trust in the circle has the following properties, *entity coupling*, *initiation*, *direction*, and *transitivity*. Figure 2 gives a logical hierarchy of these trust properties discussed below. Vertical placement of characteristics is selected to better illustrate trust relations in our scope of contribution.

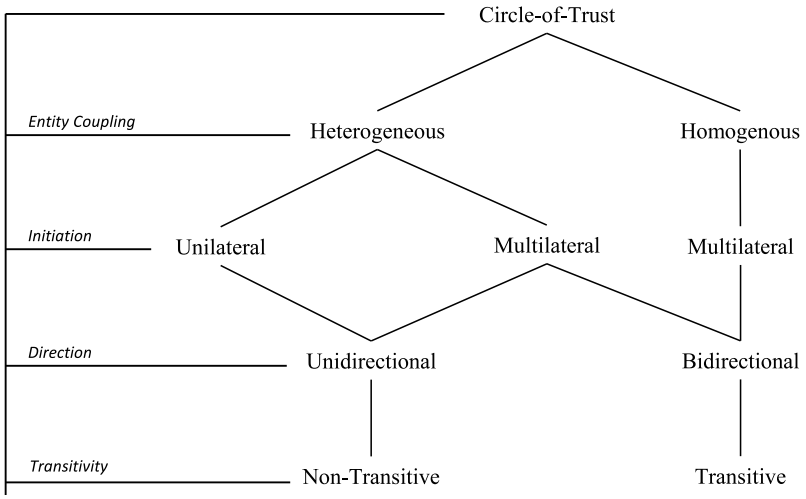


Fig. 2: Circle-of-Trust Characterization.

Entity Coupling (Homogeneous vs. Heterogeneous). In a circle-of trust, type of entities engaging in interactions determines homogeneity or heterogeneity of the circle, shaping its authorized interactions between tenants. Moreover,

with each circle type a set of trust properties are applicable. By *homogeneous circle* we denote the case where entities are uniform. For instance a circle of universities forms a homogeneous circle. In a homogeneous circle, collaborating principals are equally authorized to make cross-tenant authorization assertions. A *heterogeneous circle*, is an association of non-uniform entities where each type of entity is authorized specifically to make certain assertions. For instance, a circle consisting of universities, insurance companies, and banks establishes a heterogeneous circle. In this scenario, universities can assign users to discounted insurance plans in insurance companies while insurance companies cannot assign their users to resources in the universities. In this paper, we use type and domain interchangeably denoting the type of entities in a heterogeneous circle.

Initiation (Multilateral vs. Unilateral). If trust initiation to join a circle is required to be confirmed by all circle members, trust is considered *multilateral*. In special situations when joining members are not authorized to make assertions (in heterogeneous circles) trust initiation is not required to be confirmed by all circle members denoted as *unilateral* trust. For instance a domain of insurance companies joins a heterogeneous, unilateral circle of institutions. Insurance entities in the circle are not authorized to make assertions whilst institution entities are authorized to assert their users to discounted plans available to universities.

Direction (Bidirectional vs. Unidirectional). In a circle, direction of trust determines whether both participating circle members have equal authorizations or only one side is authorized to make assertions. If partners are authorized equally to make assertions, trust relation is *bidirectional*, otherwise it is *unidirectional* trust. Homogeneous circles' relations are bidirectional while heterogeneous circles support both trust directions. Unilateral heterogeneous circles such as given example above are only unidirectional in trust relations. Circle of universities is an example of bidirectional trust in a homogeneous circle. Sharing files in Dropbox is an example of a unidirectional trust where a user can share files with a group of users unidirectionally.

Transitivity (Transitive vs. Non-transitive). In a trust relation when principal "A trusts B" and "B trusts C" result in implication that "A trusts C", trust relation is denoted as transitive. In a homogeneous circle, bidirectional trusts are essentially transitive where all members trust and likewise trusted by other circle members. In heterogeneous unidirectional circles, trust relations cannot be transitive. For example, in the heterogeneous unidirectional circle of institutions, banks, and insurance companies, an institution can assign students to bank specific account types in banks whilst banks can assign employees to health insurances in insurance companies. Considering heterogeneous domains in the circle, a university trusting a bank and a bank trusting an insurance entity does not necessarily imply that the university can assign students to insurance resources.

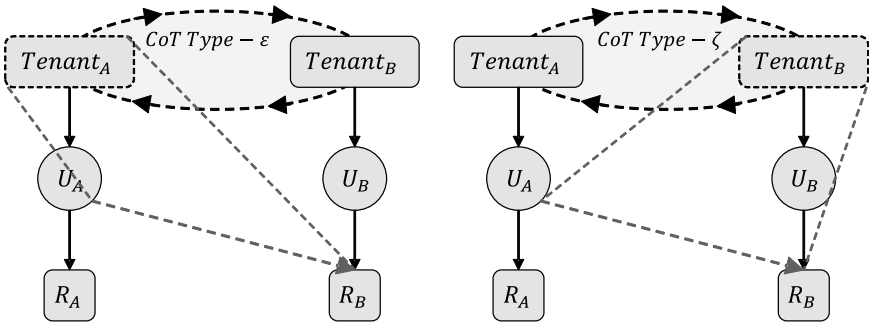
In this paper, we consider multilateral, bidirectional, and transitive trust relationships for homogeneous circles. Trust relations between tenants in heterogeneous circles are considered multilateral, unidirectional, and non-transitive. In

the following we identify how trust relations authorize cross-tenant assignments in a Circle-of-Trust federation model.

2.2 Tenant-Trust in Circle

In a circle, trust is defined between tenants as *tenant-trust* relationship. In a unidirectional trust relationship, common in peer-to-peer, trust is initiated and established between two tenants denoted as trustor and trustee. In a trust relation, trustor tenant is willing to trust another tenant denoted as trustee tenant. In our scope, trust is initiated multilaterally between principals in a circle. In the context of circle, trustor and trustee are not distinguished in trust relations between tenants. We identify tenants involve in a cross-tenant assignment as user-owner and resource-owner tenants. User-owner tenant owns the users in the cross-tenant assignment and resource-owner tenant owns the roles to which users are assigned. Central to tenant-trust defined in this paper, is authorizing user-owner or resource-owner tenants to assert cross-tenant user-role assignments.

We use "$\triangleleft$" to represent tenant-trust where $T_A \triangleleft T_B$ signifies that tenant A trusts tenant B. In this relation, T_A is user-owner tenant and T_B is resource-owner tenant. Regardless of circle entity coupling, we define two types of tenant trust relations denoted as type-ϵ and type-ζ. Each tenant-trust relation type is applied to all tenants in the circle. In type-ϵ circle, user-owner tenants are authorized to assign users to roles in the circle. The following defines type-ϵ tenant-trust illustrated in Figure 3a.



(a) A trusts B in Circle Type-ϵ. (b) A trusts B in Circle Type-ζ.

Fig. 3: User-Role Assignment in Circle-of-Trust Tenant-Trust.

Definition 1. *If $T_A \triangleleft_\epsilon T_B$, then tenant T_A is authorized to assign its users to T_B's roles. Tenant T_A controls user assignments.*

In type-ζ circle, resource-owner tenants are authorized to assign users in the circle to their roles. Type-ζ is defined as follows and is depicted in Figure 3b.

Definition 2. *If $T_A \triangleleft_\zeta T_B$, then tenant T_B is authorized to assign T_A's users to its roles. Tenant T_B controls user assignments.*

In homogeneous circles, all peers trust each other and trust is transitive, therefore $T_A \triangleleft T_B$ if and only if $T_B \triangleleft T_A$. However, in heterogeneous circles trust relations are unidirectional and non-transitive as a result $T_A \triangleleft T_B$ may not imply $T_B \triangleleft T_A$ or vise versa. Each tenant-trust type caters to a different security concern and objective in the Circle-of-Trust collaboration. Type-ϵ enable tenants in the circle to assign their users to roles of other tenants in the circle. The advantage is its simplicity to administer and implement as long as tenants' resources shared are not sensitive within the circle and tenants are willing to delegate user-role assignments to trusted tenants in the circle. For instance, an academic Circle-of-Trust is a motivation of this type of circle where academic tenants establish a Circle-of-Trust to share computing resources. Any academic tenant can assign its users to resources across tenants in the circle.

Type-ζ on the other hand, follows a different purpose to protect shared resources where user-role assignments are administered by resource-owner tenants. Tenants do not want to delegate trusted tenants permission to make assertions to their shared resources. A circle of financial institutions is a motivating example of type-ζ tenant-trust. Financial institutes do not want to expose their resources for collaboration in the circle since their resources are highly sensitive even with respect to trusted tenants in the circle. In this scenario, a resource-owner tenant administrator assigns users in the circle to its roles, authorizing access to its shared resources.

3 Homogeneous Role-Based Circle-of-Trust

This section introduces a multi-tenant role-based access control model to enable federation in a homogeneous Circle-of-Trust which we refer to as MT-RBAC$_c$. In a homogeneous circle, tenants are equally authorized to make assertions. Collaboration in MT-RBAC$_c$ is issued through cross-tenant user-role assignments with respect to circle types ϵ and ζ. MT-RBAC$_c$ model component sets and relations are depicted in Figure 4. We use a circle of institutions called Cyber Security Research (CSR) shown in Figure 5 as a running example to exemplify the concepts throughout this section. The formal definition of MT-RBAC$_c$ is given in Table 1. We discuss MT-RBAC$_c$ in parts through the following subsections, in context of these figures and table.

3.1 MT-RBAC$_c$ Basic Sets and Functions

The basic sets of MT-RBAC$_c$ are as follows: tenants (T), users (U), private roles (R_{prv}), public roles (R_{pub}), roles (R), operations (OPS), objects (OBS), and permissions ($PRMS$). Many of these are familiar from the traditional RBAC models [5, 16] and will not be further discussed here. The new sets in MT-RBAC$_c$ are tenants (T) and private and public roles (R_{prv} and R_{pub} respectively).

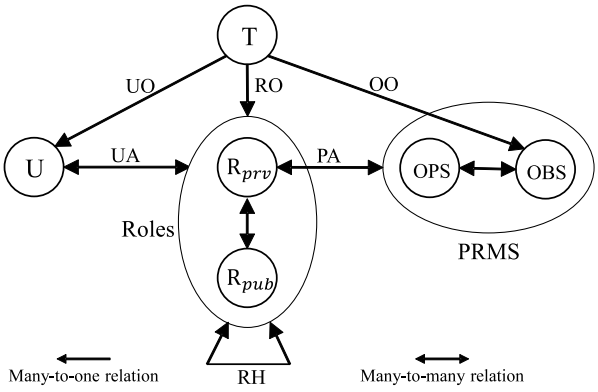


Fig. 4: Multi-Tenant RBAC Circle-of-Trust.

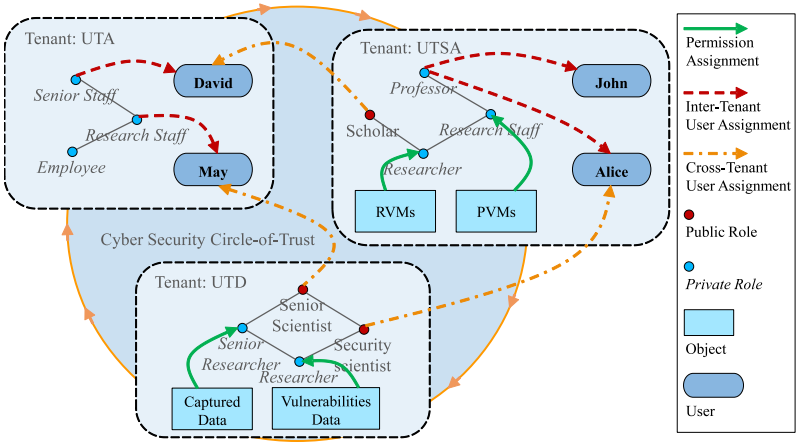


Fig. 5: Example of a Multi-Tenant RBAC$_c$ Homogeneous Circle-of-Trust.

A *tenant* is considered as a virtual container with tenant-specific environment for cloud services leased to cloud consumers. Practically, a tenant hosts a project, a department, or an organization. Each tenant is represented as $t \in T$ where T is the global set of tenants in the cloud. In Figure 5, each tenant represents an institution, UTSA, UTA, and UTD respectively. Each user, role, and object is identified with a single owner tenant, shown within the dashed tenant boundary in Figure 5. UTSA and UTD have similar Researcher roles, however in the cloud they are distinguished as Researcher#UTSA and Researcher#UTA. Similarly for objects and users.

Within each tenant the roles are partitioned into disjoint sets of public roles and private roles, R_{pub} and R_{prv} respectively, as depicted in Figure 4 and expressed in Table 1 by the *owner* functions. In Figure 5 private roles are shown

as blue circles named in italics while public roles are shown as red circles named in regular script, e.g., the UTSA tenant has private roles Professor, Researcher and Research Staff and a public role Scholar.

A central principle of MT-RBAC$_c$ is that permission to role assignment can only occur within a tenant boundary, and only to private roles. This is formalized in the definition of the permission assignment (PA) relation in Table 1. It is our first departure from traditional RBAC which in general allows any permission to be assigned to any role.

3.2 MT-RBAC$_c$ Tenant-Trust and User-Role Assignment

A Circle-of-Trust CoT is a subset of tenants T who mutually trust each other within the scope of MT-RBAC$_c$. In Figure 5, UTSA, UTA, and UTD form a homogeneous CoT of institutes. In general, multiple and possibly overlapping CoTs can be established among different subsets of tenants. For our purpose in this paper it suffices to focus on a single CoT. Tenant-trust between two members of a circle is indicated by the $\triangleleft$ symbol, which is a reflexive, transitive and symmetric relation.

MT-RBAC$_c$ distinguishes two kinds of trust, named type-ϵ and type-ζ and distinguished by a subscript applied to the symbols CoT and $\triangleleft$. In type-ϵ trust each tenant in the CoT_ϵ can assign users from another tenant in the circle to its own public roles. In type-ζ trust each tenant in the CoT_ζ can assign it own users to public roles belonging to another tenant in the circle. In Figure 5, if CSR is a type-ϵ circle then the tenant administrator of UTA can assign its users, e.g., David and May, to roles in UTSA and UTD. If CSR is a type-ζ circle then the tenant administrator of UTA can assign users from UTSA, e.g., Alice and John, as well as users from UTD to roles in UTA. In both types of circles such cross-tenant user-role assignments is limited to public roles. These concepts are formalized in Table 1. These restrictions on user-role assignment constitute a second major departure from traditional RBAC.

3.3 Limited Role Hierarchy

A third significant departure from traditional RBAC is to limit the role hierarchy with respect to public and private roles. We use the symbol $\succeq$ to represent the role hierarchy where $r_1 \succeq r_2$ means that the permissions assigned to role r_2 are also available to users assigned to role r_1. MT-RBAC$_c$ imposes the following requirements on the role hierarchy.

- Private roles can inherit private roles only if both are owned by the same tenant, e.g., Senior Researcher $\succeq$ Researcher in the UTD tenant in Figure 5.
- Private roles cannot inherit public roles. The Researcher role in UTD tenant cannot be senior to the Scholar role in UTSA.
- Public roles can inherit private roles only if both owned by the same tenant. In the UTD tenant, Security Scientist role inherits the Researcher role.

Table 1: MT-RBAC$_c$ Component Sets and Functions.

Basic Sets and Functions

- T, U, R, OPS, and OBS (tenants, users, roles, operations, and objects, respectively). $t \in T$, $u \in U$, $r \in R$, $op \in OPS$, and $ob \in OBS$.
- R_{pub} is a set of public roles and R_{prv} is a set of private roles where $R_{pub} \subseteq R$, $R_{prv} \subseteq R$, $R_{pub} \cap R_{prv} = \emptyset$, and $R_{pub} \cup R_{prv} = R$.
- $PRMS = OPS \times OBS$, the set of permissions.
- $UO \subseteq U \times T$, a many-to-one user-to-tenant owner relation. Also written as $owner\_user : (u : U) \rightarrow T$, the mapping of user u into its owner tenant. Formally: $owner\_user(u) = t$ where $(u, t) \in UO$.
- $RO \subseteq R \times T$, a many-to-one role-to-tenant owner relation. Also written as $owner\_role : (r : R) \rightarrow T$, the mapping of role r into its owner tenant. Formally: $owner\_role(r) = t$ where $(r, t) \in RO$.
- $OO \subseteq OBS \times T$, a many-to-one object-to-tenant owner relation. Also written as $owner\_object : (ob : OBS) \rightarrow T$, the mapping of object ob into its owner tenant. Formally: $owner\_object(ob) = t$ where $(ob, t) \in OO$.
- $PA \subseteq PRMS \times R$, a many-to-many mapping permission-to-role assignment relation requiring that
$((op, ob), r) \in PA \Rightarrow (owner\_object(ob) = owner\_role(r) \wedge r \in R_{prv})$.
Also written as $assigned\_permissions : (r : R) \rightarrow 2^{PRMS}$, the mapping of role r into a set of permissions. Formally: $assigned\_permissions(r) = \{p \in PRMS \mid (p, r) \in PA\}$.

Tenant-Trust

- $CoT \subseteq T$, is a subset of T called Circle-of-Trust. For every two tenants that are member of CoT ($t_1, t_2 \in CoT$) trust relationship is written as $t_1 \triangleleft t_2$, which is symmetric so $t_1 \triangleleft t_2$ iff $t_2 \triangleleft t_1$, reflexive so $t_1 \triangleleft t_1$, and transitive.
- $HomogeneousCoT_\epsilon$, for all tenants t_1 where $t_1 \triangleleft_\epsilon t_2$, tenant t_1 is authorized to assign its users to public roles in t_2. Tenant t_1 controls t_1's users to t_2's roles assignments.
- $HomogeneousCoT_\zeta$, for all tenants t_1 where $t_1 \triangleleft_\zeta t_2$, tenant t_2 is authorized to assign users from t_1 to its public roles. Tenant t_2 controls t_1's users to t_2's roles assignments.

User Role Assignment

- $UA \subseteq U \times R$, a many-to-many mapping user-to-role assignment relation requiring that $(u, r) \in UA \Rightarrow (owner\_user(u) = owner\_role(r) \wedge r \in R_{prv}) \vee ((owner\_user(u) \triangleleft_\epsilon owner\_role(r) \vee owner\_role(r) \triangleleft_\zeta owner\_user(u)) \wedge r \in R_{pub})$.
Also written as $assigned\_user\_roles : (u : U) \rightarrow 2^R$, the mapping of user u into a set of roles. Formally: $assigned\_user\_roles(u) = \{r \in R \mid (u, r) \in UA\}$.

Limited Role Hierarchy

- $RH \subseteq R \times R$, is a partial order on R called hierarchy relation, written as $\succeq$, requiring that $(r_1, r_2) \in RH \Rightarrow ((owner\_role(r_1) = owner\_role(r_2)) \wedge \neg(r_1 \in R_{prv} \wedge r_2 \in R_{pub})) \vee ((owner\_role(r_1) \triangleleft_\epsilon owner\_role(r_2) \vee owner\_role(r_2) \triangleleft_\zeta owner\_role(r_1)) \wedge (r_1, r_2 \in R_{pub}))$.

Authorized User Permissions Derived Function

- $authorized\_user\_permissions : (u : U) \rightarrow 2^{PRMS}$, the mapping of user u into a set of permissions. Formally: $authorized\_user\_permissions(u) = \bigcup_{r \in assigned\_user\_roles(u)} \bigcup_{r' \preceq r} assigned\_permissions(r')$.

– Public roles can inherit public roles from trusted tenants in the circle. In UTD tenant, Senior Scientist $\succeq$ Security Scientist where both are UTD's public roles. It is also possible for a public role of one tenant to be senior to a public role in another tenant. We include this possibility for generality, although role to role assignment is outside the scope of MT-RBAC$_c$.

3.4 MT-RBAC$_c$ Trust Properties

In terms of the circle trust properties of Section 2.1 MT-RBAC$_c$ is homogeneous in entity coupling since all tenants in the circle are treated equivalently. In term of initiation in joining or leaving a circle of trust, MT-RBAC$_c$ does not explicitly formalize this aspect. As such MT-RBAC$_c$ is neutral on this issue. Different models of initiation such a multilateral or unilateral are compatible with MT-RBAC$_c$. Regarding direction and transitivity, a circle in MT-RBAC$_c$ is explicitly defined to be bidirectional and transitive.

4 Heterogeneous Role-and-Attribute Based Circle-of-Trust

This section, introduces a multi-tenant role-centric attribute-based access control model (MT-RABAC$_c$) enabling federation in a heterogeneous Circle-of-Trust. Our model is motivated by a previously defined role-centric model [8] for combining roles and attributes. In a heterogeneous circle, entities are from non-uniform types. In MT-RABAC$_c$, tenants are not equally authorized and cross-tenant user-role assignments are limited with respect to tenant's domain type attribute.

MT-RABAC$_c$ adds attributes to enforce cross-tenant user-role assignment separation. Attributes are used to denote tenant types where tenants are only authorized to assert cross-tenant user assignments on certains type of tenants. Figure 6 depicts elements in MT-RABAC$_c$, where *tenant attributes* ($TATT$), *user attributes* ($UATT$), and *object attributes* ($OATT$) are added to the tenant, user, and object components of Figure 4 respectively. We use a heterogeneous circle of institutions (UTA and UTSA) and a bank (BoA) in Figure 7 as a running example to exemplify the concepts throughout this section. The extensions and modifications to the MT-RBAC$_c$ model to obtain MT-RABAC$_c$ are formally given in Table 2. Similar to the description of MT-RBAC$_c$ in the previous section, we will describe MT-RABAC$_c$ systematically in the following subsections, in context of the afore-mentioned figures and table.

4.1 MT-RABAC$_c$ User and Object Attributes and Meta-Attributes

An *attribute* is considered as a function which takes a tenant, user or object as input and return a value from the attribute's range. For example, an atomic-valued user attribute function such as $employeeType$ returns employee status of a user $john$ where $employeeType \in UATT$, $john \in U$ and $employeeType(john) = full\_time$. Range or scope of an attribute is a finite set of atomic values specifying

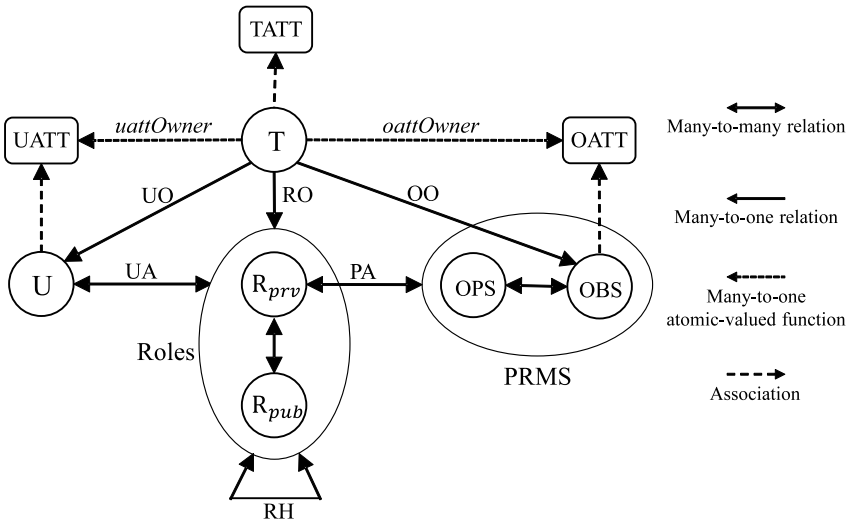


Fig. 6: Multi-Tenant Role-Centric ABAC Circle-of-Trust.

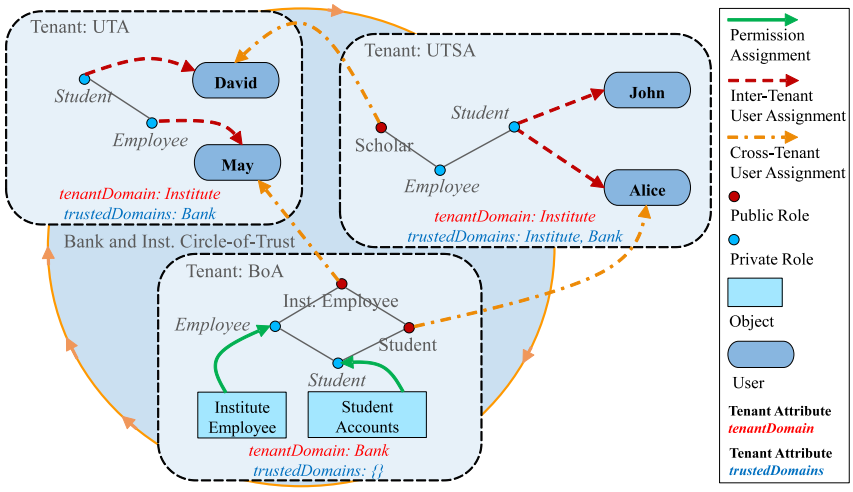


Fig. 7: Example of a Multi-Tenant RABAC$_c$ Heterogeneous Circle-of-Trust.

the valid range of attribute functions. Attribute functions either return a single value or set of values, which are respectively called *atomic-valued* and *set-valued* attribute types. In MT-RABAC$_c$, users and objects are respectively associated with attributes in the sets $UATT$ and $OATT$. Each user attribute $uatt \in UATT$ is a partial function since not every attribute is defined for every user. Similarly, each object attribute $oatt \in OATT$ is a partial function.

Each user and object attribute is owned by a single tenant. This is realized by means of meta-attributes *uattOwner* and *oattOwner*. Users and objects can only be assigned attribute values for attributes owned by the same tenant as the user or object. User and object attributes do not impact user-role assignment and are included in the model for generality and uniformity.

4.2 MT-RABAC$_c$ Tenant Attributes

Tenant attributes are fundamental to MT-RABAC$_c$ to enforce constraints on cross-tenant user-role assignments. Each tenant administrator can only assign values to its set of tenant attributes. In a heterogeneous circle, tenants are from different types which needs to be recognized in cross-tenant user-role assignments. To that end, we define a *domain* as a set of tenants grouped together with respect to their type in the system. Each domain is a subset of T defined in Table 2. For instance in Figure 7, circle includes two types of tenants, Institute and Bank domains. Particularly, a tenant is related to a domain with an atomic-valued required attribute function, *tenantDomain*. It is defined as an atomic attribute to signify that each tenant only belongs to one domain. In Figure 7, UTSA and UTA have Institute and BoA has Bank tenantDomain attribute values. Moreover, to separate user-role assignments in MT-RABAC$_c$, *trustedDomains* is defined as a required set-valued tenant attribute. In MT-RABAC$_c$, each tenant administrator specifies the group of domains it trusts with their assertions, including its own domain. For instance in Figure 7, UTSA trusts assertions from Institute and Bank domain while UTA only trusts Bank domain assertions meaning UTA does not trust assertions from its own domain. In the heterogeneous circle in Figure 7, BoA does not allow any assertions from tenants in the circle.

4.3 MT-RABAC$_c$ Tenant-Trust

In MT-RABAC$_c$, tenant-trust is limited with trustedDomains attributes. In type-ϵ circle, user-owner tenant can assign its users to roles from tenants which it is a member of their trustedDomains attribute set. In type-ζ, user assignment is modified to satisfy the condition where role-owner tenant can assign users from tenants in the circle, if it is a member of their trustedDomains attribute set. Type ϵ and ζ is defined in Table 2. In Figure 7, if circle is a type-ϵ, then the tenant administrator of UTA can assign its users, e.g., David and May, to roles in UTSA since UTSA trusts assertions from its domain. If circle is a type-ζ, then the tenant administrator of BoA can assign users from UTSA, e.g., Alice

Table 2: MT-RABAC$_c$ Component Sets and Functions.

Basic Sets and Functions

- $TATT$, $UATT$, and $OATT$ represent finite set of tenant, user, and object attribute functions respectively.
- For each att in $TATT \cup UATT \cup OATT$, $Scope(att)$ represents the attribute's scope, a finite set of atomic values.
- $attType : TATT \cup UATT \cup OATT \rightarrow \{set, atomic\}$, specifies attributes as set or atomic valued.

Meta Attributes of User and Object Attributes

- $MATT = \{uattOwner, oattOwner\}$, required meta-attribute functions.

$uattOwner : (uatt : UATT) \rightarrow T$, required atomic user meta-attribute function, mapping user attribute $uatt$ to attribute owner tenant t.
$oattOwner : (oatt : OATT) \rightarrow T$, required atomic object meta-attribute function, mapping object attribute $oatt$ to attribute owner tenant t.

User Attribute

- Each user attribute function $uatt \in UATT$ is defined as a partial function mapping elements in U to atomic or set values.

$$\forall uatt \in UATT.uatt : U \hookrightarrow \begin{cases} Scope(uatt) \; if \; attType(uatt) = atomic \\ 2^{Scope(uatt)} \quad if \; attType(uatt) = set \end{cases}$$

$uatt(u : U)$ is defined only if $uattOwner(uatt) = owner\_user(u)$.

Object Attribute

- Each object attribute function $oatt \in OATT$ is defined a partial function mapping elements in O to atomic or set values.

$$\forall oatt \in OATT.oatt : O \hookrightarrow \begin{cases} Scope(oatt) \; if \; attType(oatt) = atomic \\ 2^{Scope(oatt)} \quad if \; attType(oatt) = set \end{cases}$$

$oatt(o : O)$ is defined only if $oattOwner(oatt) = owner\_object(o)$.

Tenant Attribute

- Each tenant attribute function $tatt \in TATT$ maps elements in T to atomic or set values.

$$\forall tatt \in TATT.tatt : T \rightarrow \begin{cases} Scope(tatt) \; if \; attType(tatt) = atomic \\ 2^{Scope(tatt)} \quad if \; attType(tatt) = set \end{cases}$$

- D is a finite set of domains.
- $tenantDomain : (t : T) \rightarrow D$, required tenant atomic attribute function mapping tenant t to tenant domain d where $tenantDomain \in TATT$.
- $trustedDomains : (t : T) \rightarrow 2^D$, required tenant set attribute function mapping tenant t to the powerset of trusted domains D where $trustedDomains \in TATT$.

Tenant Trust

- $HeterogeneousCoT_\epsilon$, for all tenants t_1 where $t_1 \lhd_\epsilon t_2$, if $tenantDomain(t_1) \in trustedDomain(t_2)$, then tenant t_1 is authorized to assign its users to public roles in t_2. Tenant t_1 controls t_1's users to t_2's roles assignments.
- $HeterogeneousCoT_\zeta$, for all tenants t_1 where $t_1 \lhd_\zeta t_2$, if $tenantDomain(t_2) \in trustedDomain(t_1)$, then tenant t_2 is authorized to assign users from t_1 to its public roles. Tenant t_2 controls t_1's users to t_2's roles assignments.

User Role Assignment

- $UA \subseteq U \times R$, a many-to-many mapping user-to-role assignment relation requiring that $(u, r) \in UA \Rightarrow (owner\_user(u) = owner\_role(r) \wedge r \in R_{Prv}) \vee$
$(owner\_user(u) \lhd_\epsilon owner\_role(r) \wedge r \in R_{pub} \wedge$
$tenantDomain(owner\_user(u)) \in trustedDomains(owner\_role(r))) \vee$
$(owner\_user(u) \lhd_\zeta owner\_role(r) \wedge r \in R_{pub} \wedge$
$tenantDomain(owner\_role(r)) \in trustedDomains(owner\_user(u)))$.

and John, as well as users from UTA to roles in BoA since both UTSA an UTA trust assertions from Bank domain tenants.

In this context, user-assignment is modified with respect to trustedDomains attributes. A user is assigned to a role only if

$$(owner\_user(u) = owner\_role(r) \wedge r \in R) \vee$$
$$(owner\_user(u) \lhd_{\epsilon} owner\_role(r) \wedge r \in R_{pub} \wedge$$
$$tenantDomain(owner\_user(u)) \in trustedDomains(owner\_role(r))) \vee$$
$$(owner\_user(u) \lhd_{\zeta} owner\_role(r) \wedge r \in R_{pub} \wedge$$
$$tenantDomain(owner\_role(r)) \in trustedDomains(owner\_user(u)))$$

In a Circle-of-Trust we allow only one trust type in the circle. We don't allow both type ϵ and ζ at once in a circle due to assignment conflict. Permission-assignment remains unchanged where a permission is assigned to a role only if

$$(owner\_role(r) = owner\_object(o) \wedge r \in R_{prv})$$

Authorized_user_permisisons denotes the set of permissions available to a user with respect to tenant types in the circle which is not changed from Table 1.

4.4 MT-RABAC$_c$ Trust Properties

In terms of the circle trust properties of Section 2.1 MT-RABAC$_c$ is heterogeneous in entity coupling since tenants in the circle are distinguished by their domain, and thereby not treated equivalently. In term of initiation in joining or leaving a circle of trust, MT-RABAC$_c$ does not explicitly formalize this aspect. As such MT-RABAC$_c$ is neutral on this issue. Different models of initiation such a multilateral or unilateral are compatible with MT-RBAC$_c$. Regarding direction and transitivity, a circle in MT-RBAC$_c$ is explicitly defined to be unidirectional and non-transitive.

5 Related Work

The Liberty Alliance Project [20] identified the conceptual framework and guidelines in a Circle-of-Trust as part of their federated identity vision. Considerable research on Circle-of-Trust has been devoted to identity federation such as [10] where trust requirements and patterns in a Circle-of-Trust identity federation are identified. In [3], Circle-of-Trust collaboration trust considerations in identity federation for assessment of entities' trust outside the circle are considered. Our work is focussed on authorization federation in a collaboration group of entities considered as tenants.

Sharing resources among organizations has been investigated in multiple aspects. ROBAC [21] extended RBAC to consider authorization in multiple organizations, but collaboration within organizations is not considered. GB-RBAC [11] extends RBAC with groups to support collaboration. In GB-RBAC, administrator cannot manage users in the groups. In our model, each tenant administers its collaboration policy by controlling users or roles in user-role assignments across the tenants in the circle. In [19], a dynamic coalition-based access control (DCBAC) model is proposed that allows automatic access to resources of

one coalition entity by users from another coalition entity. O2O [4] defined an approach to deal with access control in interoperability context based on virtual private organizations (VPO) and role single-sign on (RSSO). Our contribution is differentiated based on the collaboration framework to enable collaboration between tenants. In O2O, each organization is responsible to define its security policy for roles whereas in our federation framework, each tenant defines its collaboration policy through public roles for a group of tenants in the circle.

Further in cloud IaaS, models such as CTTM [17] extended RBAC to enable collaboration in multi-tenant cloud systems. In [13], cross-tenant collaboration models discussed enabling federation in multi-cloud environments. In this paper, we focus on Circle-of-Trust federation compared to [17] and[13] where collaboration is enabled in a Peer-to-Peer federation. In ABAC collaboration in cloud, MT-ABAC [14] proposed collaboration between tenants by cross-tenant attribute assignment in cloud IaaS. Such attribute-based federation provides Peer-to-Peer collaboration, however our role-centric model provides federation in a circle.

6 Conclusion

This paper elaborated a fine-grained collaboration model in a Circle-of-Trust. We introduced the MT-RBAC$_c$ model in a homogeneous circle, in which collaboration is enabled through user to public role assignments. We identified, private and public roles with limited role hierarchy to control access on tenants' resources. Trust is defined on tenants with circle types ϵ and ζ authorizing user-owner and resource-owner tenants' assertions respectively. Moreover, tenant attributes in MT-RABAC$_c$ classifies tenants into domains in heterogeneous circles, where tenant-trust is defined conditionally with *trustedDomain* attributes. Using roles and attributes to enable cross-tenant user-role assignments is general and dynamic enough to address current issues while it is applicable to current platforms. For future work, we plan to extend this work with attribute-based models into further generalization in multi-cloud environments and implement proposed models in the current cloud platforms such as OpenStack.

Acknowledgement

This research is partially supported by NSF Grant CNS-1111925 and CNS-1423481.

References

1. Amazon AWS. `https://aws.amazon.com/`.
2. OpenStack. `http://www.openstack.org/`.
3. L. Boursas and V. A. Danciu. Dynamic inter-organizational cooperation setup in circle-of-trust environments. In *Network Operations and Management Symposium, 2008. NOMS 2008. IEEE*, pages 113–120. IEEE, 2008.

4. F. Cuppens, N. Cuppens-Boulahia, and C. Coma. O2O: Virtual private organizations to manage security policy interoperability. In *Information Systems Security*, pages 101–115. Springer, 2006.
5. D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli. Proposed NIST standard for role-based access control. *TISSEC*, 4(3):224–274, 2001.
6. V. C. Hu, D. Ferraiolo, et al. Guide to attribute based access control (ABAC) definition and considerations. *NIST Special Publication*, 800:162, 2014.
7. V. C. Hu, D. R. Kuhn, and D. F. Ferraiolo. Attribute-based access control. *Computer*, (2):85–88, 2015.
8. X. Jin, R. Sandhu, and R. Krishnan. RABAC: role-centric attribute-based access control. In *Computer Network Security*, pages 84–96. Springer, 2012.
9. D. R. Kuhn, E. J. Coyne, and T. R. Weil. Adding attributes to role-based access control. *Computer*, (6):79–81, 2010.
10. U. Kylau, I. Thomas, M. Menzel, and C. Meinel. Trust requirements in identity federation topologies. In *Advanced Information Networking and Applications, 2009. AINA'09. Int. Conf. on*, pages 137–145. IEEE, 2009.
11. Q. Li, X. Zhang, M. Xu, and J. Wu. Towards secure dynamic collaborations with group-based RBAC model. *Computers & Security*, 28(5):260–275, 2009.
12. P. Mell and T. Grance. The NIST definition of cloud computing. 2011.
13. N. Pustchi, R. Krishnan, and R. Sandhu. Authorization federation in IaaS multi cloud. In *Proc. of Security in Cloud Computing*, pages 63–71. ACM, 2015.
14. N. Pustchi and R. Sandhu. MT-ABAC: A multi-tenant attribute-based access control model with tenant trust. In *Network and System Security*, pages 206–220. Springer, 2015.
15. R. Sandhu. The authorization leap from rights to attributes: maturation or chaos? In *Proc. of SACMAT*, pages 69–70. ACM, 2012.
16. R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *Computer*, 29(2):38–47, 1996.
17. B. Tang and R. Sandhu. Cross-tenant trust models in cloud computing. In *Proc. of Int. Conf. IRI*, pages 129–136. IEEE, 2013.
18. B. Tang, R. Sandhu, and Q. Li. Multi-tenancy authorization models for collaborative cloud services. In *Proc. of CTS*, pages 132–138. IEEE, 2013.
19. J. Warner, V. Atluri, and R. Mukkamala. A credential-based approach for facilitating automatic resource sharing among ad-hoc dynamic coalitions. In *Data and Applications Security XIX*, pages 252–266. Springer, 2005.
20. T. Wason, S. Cantor, J. Hodges, J. Kemp, and P. Thompson. Liberty id-ff architecture overview. *Liberty Alliance*, 2004.
21. Z. Zhang, X. Zhang, and R. Sandhu. ROBAC: Scalable role and organization based access control models. In *Proc. of CollaborateCom*, pages 1–9. IEEE, 2006.