A Multi-Tenant Usage Access Model for Cloud Computing

Zhengtao Liu[1, *], Yun Yang[1], Wen Gu[1] and Jinyue Xia[2]

Abstract: Most cloud services are built with multi-tenancy which enables data and configuration segregation upon shared infrastructures. It offers tremendous advantages for enterprises and service providers. It is anticipated that this situation will evolve to foster cross-tenant collaboration supported by Authorization as a service. To realize access control in a multi-tenant cloud computing environment, this study proposes a multi-tenant cloud computing access control model based on the traditional usage access control model by building trust relations among tenants. The model consists of three sub-models, which achieve trust relationships between tenants with different granularities and satisfy the requirements of different application scenarios. With an established trust relation in MT-UCON (Multi-tenant Usage Access Control), the trustee can precisely authorize cross-tenant accesses to the trustor's resources consistent with constraints over the trust relation and other components designated by the trustor. In addition, the security of the model is analyzed by an information flow method. The model adapts to the characteristics of a dynamic and open multi-tenant cloud computing environment and achieves fine-grained access control within and between tenants.

Keywords: Multi-tenant, usage access control model, cloud computing.

1 Introduction

Cloud computing is a pay-as-you-go model that allows users to interact with providers with minimal administrative costs and service. It also allows users to access IT resources and services, including network bandwidth, servers, storage, application programs, and computing services, from configurable computing resource pools shared by the network based on their actual needs [Mell and Grance (2011)]. Because of its unique advantages of broadband interconnection, resource pool sharing, flexible configuration, on-demand services, and charging by service, cloud computing significantly reduces the maintenance cost of computing and storage for users as well as many constraints imposed on users with limited storage and computing resources.

In recent years, the multi-tenant technology has been rapidly developed and widely applied in the field of cloud computing. The technology isolates resources on the cloud

[1] School of Computer Science and Engineering, Sanjiang University, Nanjing, 210012, China.

[2] International Business Machines Corporation (IBM), New York, USA.

[*] Corresponding Author: Zhengtao Liu. Email: lzt_jh_cn@163.com.

Received: 01 April 2020; Accepted: 21 April 2020.

by introducing in the system multiple tenants that share the same physical resource, while they are logically isolated from each other and do not affect each other. A single tenant monopolizes the entire physical resource, and from the perspective of its actual deployment, each tenant has its own independent configuration, such that the single-application multi-configuration architecture enables cloud computing resources to be rapidly expanded and deployed. In a multi-tenant environment, a user generally belongs to a tenant and shares and uses the tenant's resources with other users in the same tenant. The multi-tenant model has the characteristics of resource sharing, configurability, and certain independence. Application of the multi-tenant technology in cloud computing can bring enormous advantages to enterprises and service providers. In particular, this superior technology enables providers to focus on software development and maintenance and enterprises to focus on their core business alone, as the need for them to implement and manage an application is eliminated. Sharing software and hardware can greatly reduce the cost of operation and maintenance for enterprises. Flexible on-demand payment greatly reduces the threshold of enterprise information and brings new development opportunities for small and medium enterprises [Feng, Qin, Yuan et al. (2015); Gu, Wu, Yin et al. (2020); Wang, Yang, Xu et al. (2015)].

The new technology also brings new security issues, and one of the most important issues a multi-tenant environment is how to ensure that resources are not illegally accessed. Access control technology ensures that digital resources are not illegally accessed and used by restricting users' access to resources. In the traditional computing mode, the current access control models can better constrain the access process and protect resource security. However, in a multi-tenant environment, the computing model changes, posing new challenges for access control. According to the relationship between a subject and an object in the access, the access in a multi-tenant environment can be divided into intra- and cross-tenant access. Intra-tenant access refers to the access of a user in a tenant to the resources in the same tenant. Although the access control demand of this part is similar to that of traditional access control, it still needs to be able to adapt to the dynamically changing cloud environment. By contrast, cross-tenant access refers to the access of a user in a tenant to the resources of another tenant, which is a new access mode. With the development of cloud computing, an increasing number of services are migrated to the cloud, and services are isolated from each other in the form of tenants. However, there is also a potential partnership between tenants, which requires the use of cross-tenant access control technology to constrain them.

To achieve access control between cross-tenants, Tang et al. [Tang, Li and Sandhu (2013)] proposed a multi-tenant role-based access control (MT-RBAC) model by extending the role-based access control (RBAC) model. In this model, two built-in components of the issuer and the tenant are integrated to the traditional model in order to achieve collaboration between different tenants by establishing trust between tenants. The traditional RBAC model can better solve the authorization problem in a static environment, but an access policy once defined in the MT-RBAC model will not change during the entire access process. A multi-tenant cloud computing environment is open and dynamic. After access is established, the attributes of subjects and objects change continuously, which results in the previously granted authority no longer meeting the requirements of the access decision. Therefore, the authorization requires adjustments, which in turn may further change the attributes of the subjects and objects, thereby changing the state of the system.

Based on the usage access control model (UCON) [Park and Sandhu (2004)], a multi-tenant usage access control model (MT-UCON) is proposed in this paper. In this model, the components of the tenant and issuer are added. An issuer establishes a trust relationship between tenants with different granularities. The security of the model is analyzed by the information flow method, and the model realizes dynamic and fine-grained access control in a multi-tenant cloud computing environment.

The remainder of this paper is organized as follows: Section 2 reviews the related works. Section 3 presents the proposed MT-UCON model. The security of the model is analyzed by an information flow method in Section 4. Section 5 draws the conclusion.

2 State-of-the-art

The following two aspects are analyzed for the UCON model in a cloud computing environment. The first aspect is to design a UCON access control mechanism and system that is more appropriate for cloud computing. The other aspect, considering the integration of obligations and conditions in the model, is how to build constraints on location and time such that the model achieves more efficient control on users' access to data. However, effective update of attributes in an actual application and ensuring their accuracy as well as updated attributes that affect further control causes great risks to the authorization problem in use. Krautsevich et al. [Krautsevich, Lazousk, Martinelli et al. (2010)] introduced risk assessment in the authorization mechanism of UCON, and thus improved the flexibility and security of the UCON model, which mainly focused on the access control requirements of a distributed computing environment. Chu et al. [Chu and Qin (2010)] proposed an implementation scheme for usage control in a distributed computing environment. Although the UCON model has obvious advantages in distributed and cross-tenant environments, the authorization management of this model is complicated. Tavizi et al. [Tavizi, Shajari and Dodangeh (2012)] constructed a new UCON model to solve attribute mutability and obligation handling of the subject in the cloud computing environment; however, their work lacked in-depth theoretical research. Lazouski et al. [Lazouski, Mancini, Martinelli et al. (2012)] proposed an authorization framework based on the UCON and XACML standard specifications. User access control was implemented using the framework integrated with the OpenNebula toolbox, which provided uninterrupted access control during the authorization process. Msahli et al. [Msahli, Abdeljaoued and Serhrouchni (2013)] introduced a new entity proof manager (PM) in a cloud computing environment to manage the proof between data providers and users, and used the UCON model for access control management in cloud computing. The exchange of evidence, authorization, and other issues between data providers and users was managed through the model to ensure the normal release of the cloud platform security policy.

In order to achieve multi-tenant collaboration, Calero et al. [Calero, Edwards, Kirschnick et al. (2010)] informally presented a multi-tenancy authorization system (MTAS) which extends the well-known role-based access control (RBAC) model by building trust relations among collaborating tenants. This model supported hierarchical role-based access control, path-based object hierarchies and federation. It also described Security, privacy and trust management aspects for the authorization system. To support collaboration between cloud services, Tang et al. [Tang, Sandhu and Li (2015)] presented an MTAS model based on an informally specified MTAS, which extends the RBAC model by building trust relations among collaborating service. Based on the MT-RBAC, a cross-tenant RBAC (CT-RBAC) model is proposed [Liu and Xia (2019)]. Compared with the MT-RBAC model, the CT-RBAC model not only considers different types of authorization modes among different tenants, the exposure of users and role information in authorization, and management of role inheritance but also extends the RBAC model in the multi-tenant cloud computing mode. Zuo et al. [Zuo, Xie, Qi et al. (2017)] provided a formal definition of a new tenant-based access control model based on administrative role-based access control (ARBAC) for multitenancy architecture and sub-tenancy architecture in service-oriented SaaS called TMS-ARBAC. Autonomous areas (AA) and AA-tree were proposed to describe the autonomy of tenants, including their isolation and sharing relationships. Authorization operations on AA and different resource-sharing strategies were defined to create and deploy the access control scheme in single-threaded apartment (STA) models. Alam et al. [Alam, Malik, Akhunzada et al. (2017)] proposed a cloud resource mediation service offered by cloud service providers, which played the role of trusted third-party among its different tenants. It formally specified the resource-sharing mechanism between two different tenants in the presence of their proposed cloud resource mediation service. The correctness of permission activation and delegation mechanism among different tenants using four distinct algorithms (activation, delegation, forward revocation, and backward revocation) was also demonstrated using formal verification. Bendiab et al. [Bendiab, Shiaeles, Boucherkha et al. (2019)] introduced a novel dynamic trust model for Federated Identity Management. The proposed model relies on fuzzy cognitive maps for modeling and evaluating trust relationships between the involved entities in federated identity management systems. This trust mechanism facilitates the creation of trust relationships between prior unknown entities in a secure and dynamic way and makes Federated Identity Management systems more scalable and flexible to deploy and maintain in cloud computing environments.

3 MT-UCON model

To achieve fine-grained access control for multi-tenant collaboration in a cloud computing environment, an MT-UCON model is designed on the basis of trust relationships between tenants.

3.1 Basic architecture of the MT-UCON

The MT-UCON model consists of the following eight components: issuers, tenants, subjects, objects, authorization, duties, conditions, and usage decisions. Its architecture is presented in Fig. 1.

Issuers: An issuer is a customer in the cloud who uses cloud computing services. The customer can be an organization or an administrator who manages his own tenants.

Tenants: In a cloud computing environment, a cloud computing service provider uses a tenant as a logical unit to provide storage, computing, networking, applications, and other services to users. A tenant can be an organization or a work unit. For example, an IaaS cloud computing service provider provides a 100 G memory space for a tenant. Tenants can allocate the usage rights of the memory space to internal users according to their own needs. The tenant set is denoted as T.

Subjects: Subjects are active entities that have certain usage rights to objects, and the subject set is denoted as S. A subject can be a user, a group, a role, or a program. A subject belongs to a tenant and its attribute is used to identify its ability and characteristics; the subject attribute, such as identity, role, credit, membership, or security level, is an important parameter in the process of decision-making process, and is denoted as ATT(S).

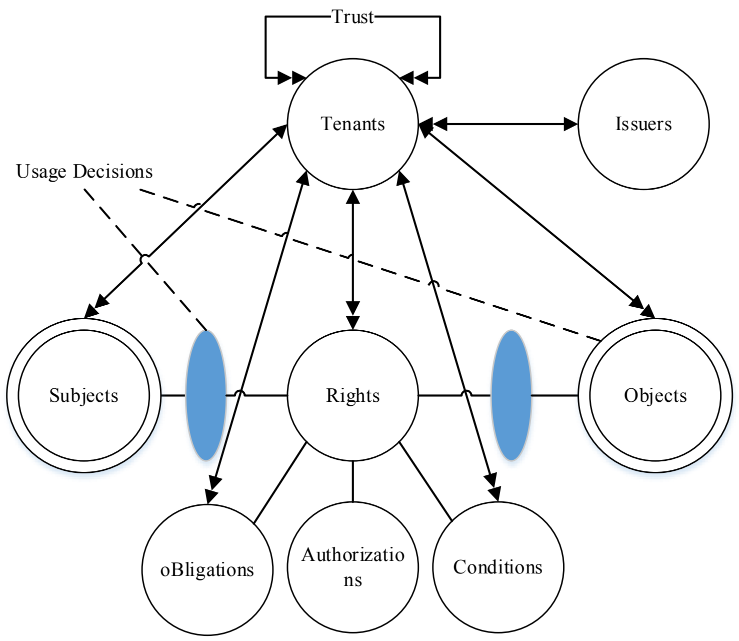

Figure 1: Architecture of the MT-UCON model

Objects: Objects are passive entities that accept subject access as specified by the permission set and they belong to tenants. The object attribute, such as documents, data, and various devices, is denoted as O.

Right: Right is a subject's operation on an object, such as reading and writing; it belongs to tenant T and is denoted as R. Different from the concept of right in traditional access control, the right here is not defined in the access control matrix before accessing and is not independent of subject attributes. Subjects, objects, and right often form an authority (s, o, r).

Authorization: Authorization is based on the attribute values of subjects and objects and the required power. Granting of a certain right to a subject may be followed by an update of the attribute of the subject and object. Then, after the attributes of the subject and object are updated, the system state is transferred to the next state. At this time, it is necessary to judge whether the authorization predicate is satisfied, and if it is not, the authorization is revoked.

oBligations: oBligations are certain operations that a subject must perform before accessing an object or in the process of accessing an object. For example, the "accept" button must be clicked to access certain website content, and some advertisements must be clicked to continue the subject's access to a website. The attributes of subjects and objects can be used as the basis for determining whether or not the duties are performed. Once a duty is performed, an attribute value may be updated, which may affect the current or future usage decision.

Conditions: Conditions are environmental or systemic factors such as time, place, and the system state, including safety state, high-risk state, and attacked state. A condition is not directly related to subjects or objects and is not individually controlled by subjects. Checking the condition factors does not result in a change in the attributes of subjects and objects.

Usage decision: Usage decision is based on authorization, duties, and conditions to provide a subject decision about certain right to an object. When a subject requests access to an object, the decision-making component relies on authorization, duties, and conditions to determine whether to grant or revoke the right requested by the subject to the object.

The MT-UCON model integrates the support of decision continuity and attribute mutability to traditional access control models. Decision continuity means that decisions take effect before or during the access process and are not just pre-defined.

3.2 MT-UCON0

The MT-UCON0 model is the basic model of the MT-UCON model family, which achieves intra-tenant and cross-tenant access control under full trust between tenants.

Definition 1: Tenant trust relationship. T is a set of all tenants, and tenant trust relationship $TT \subseteq T \times T$ is a many-to-many relationship. For $\forall t_i, t_j \in T$, t_i is a trustor, and t_j is the trustee, and the relation is recorded as $t_i \triangleleft t_j$. If t_i and t_j represent the same tenant, then $t_i \equiv t_j$.

The trust relationship between tenants is one-way and independent; that is, tenant t_i trusting t_j does not mean that t_j must trust t_i, and the trust relationship between any two tenants does not affect the trust relationship between either of the tenants and other tenants. t_i trusting t_j means that a subject in tenant t_j can access an object in t_i. The trust relationship between tenants is established and revoked by issuers.

Definition 2: The formal definitions of MT-UCON0 are as follows.

● S represents a finite set of subjects. Subjects are the entities that initiate access and they have attributes.

● O represents a finite set of objects. Objects are entities accessed by subjects and they have attributes.

● R represents a finite set of authorities. Authorities are the operations performed by a subject when accessing an object.

● A represents a finite set of authorization assertions.

● T represents a finite set of tenants. Tenants are isolated from each other before a trust relationship is established between them.

● C represents a finite set of conditional assertions. Conditions are divided into two parts: intra-tenant conditions and cross-tenant conditions.

● B represents a finite set of obligation behaviors. Obligations are divided into two parts: tenant obligations and trust obligations. Tenant obligations are subordinate to tenants and are related to a subject-object pair. Trust obligations are subordinate to the entire system and are related to a pair of trustor-trustee.

● ATT represents a finite set of attributes. Subjects, objects, tenants, and the system, all have attributes. The attribute of an entity e is represented as e.attr.

● subjectOwner: (s:S) → T is the function mapping from subject s to its tenant.

● objectOwner: (o:O) →T is the function mapping from object o to its tenant.

● rightOwner: (r:R) →T is the function mapping from a right to its tenant.

● sattrOwner: (sattr:SATTR) →T is the function mapping from a subject attribute to its tenant.

● oattrOwner: (oattr:OATTR) →T is the function mapping from an object attribute to its tenant.

● conditionOwner: (c:C) →T is the function mapping from a condition to its tenant, which is a partial function that only maps from a tenant condition to its tenant.

● obligationOwner: (b:B) →T is the function mapping from an obligation to its tenant, which is a partial function that only maps from a tenant obligation to its tenant.

● tryaccess, permitaccess, denyaccess, revokeaccess, endaccess, preupdate, onupdate, and postupdate are system operations that result in access allowed, access denied, access revoked, access stopped, pre-update, synchronization update, and post-update, respectively.

● R (s, o, r) is a Boolean function that represents whether subject s has the authority to operate right r on object o.

● canAccess (o:O) →2^T, a function mapping an object to a set of tenants who can access the object; formally, canAccess(o)=$\{t\}\cup\{t\in T|\text{objectOwner}(o)\triangleleft t\}$;

The system state transition of the MT-UCON model follows the following process, including the initial, requesting, denied, accessing, revoked, and end states. The usage session starts with the initial state. Subjects (s) apply for rights (r) to objects (o) by executing the tryaccess (s, o, r) operation, where objects (o) are accessible objects, including those within tenants and trustees. If the objects in the tenants are accessed, the r system goes to the requesting state. If the system denies the access request, denyaccess (s, o, r) is executed, and if there is an attribute update, the preupdate operation is executed. If authorization is permitted to this operation and if an attribute needs to be updated in advance, the preupdate operation as well as the permitaccess (s, o, r) operation is executed, and the system state is transferred to the accessing state. During the access execution process, if the system checks that the access no longer satisfies the condition, it executes the revokeaccess (s, o, r) operation and enters the revoked state, and if there is an attribute update, the preupdate operation is executed. If there is no revoking operation and there is an attribute update, the onupdate operation is executed. At the end of the access, the subjects (s) execute the endaccess (s, o, r) operation, and the system enters the end state, and if there is an attribute update, the postupdate operation is executed. Thus ends the state conversion.

In the MT-UCON model, authorization contains some rules that are based on subject and object attributes. Therefore, there are two types of authorization. One is local authorization, which is the same as that in the traditional UCON model and whose rules are established on intra-tenant local attributes. The other type is cross-tenant authorization, whose rules are established on cross-tenant attributes. Similarly, obligations and conditions are divided into local and cross-tenant obligations as well as local and cross-tenant conditions, respectively. Local obligations and conditions have the same meaning as those in the traditional UCON model. Cross-tenant obligations fulfill the obligations within the tenants and those of the tenants of the object when cross-tenant accessing. Cross-tenant conditions must satisfy the conditions that the tenants of a subject need to meet as well as the conditions of the tenants of an object.

The trust relationship between two tenants is revoked by issuers. When the trust relationship between two tenants is released, the cross-tenant authorization will be terminated, the rules established in the cross-tenant access will be automatically revoked, and the cross-tenant objects that the subject can access are automatically cleared.

MT-UCON0 provides the basic methods of cross-tenant access. It is a coarse-grained access model that requires trustees to expose all their objects to the trustor tenants. For a high trust level situation, the model satisfies the security requirements of access control. Therefore, the model is more suitable for access control requirements between different departments within an organization.

3.3 MT-UCON1

For higher security requirements, the objects in tenants can be classified into two categories: public and private objects. Public objects are visible to subjects in trustees, and issuers can establish access control rules between the subjects in trusted tenants and objects. Private objects are only visible within tenants, and issuers are not allowed to establish access rules for cross-tenant subjects on private objects.

Definition 3: MT-UCON1 inherits all the components of MT-UCON0 and simultaneously satisfies the following condition.

Pub (t: T)$\rightarrow 2^{O}$ is a mapping function from a tenant to a set of public objects. The objects in the set are visible to the subjects in all trusted tenants.

3.4 MT-UCON2

MT-UCON1 is more secure than MT-UCON0 and meets the security requirements of certain occasions; however, the public object set in the model is exposed to the subjects of all trustees, which is not secure for certain occasions. Therefore, a more fine-grained access control model needs to be established to meet higher security requirements.

Definition 4: MT-UCON2 inherits all the components of MT-UCON1 and simultaneously satisfies the following condition.

Pub (t: T)$\rightarrow 2^O$ is a mapping function from a tenant to a set of public objects. The objects in the set are visible to the subjects of all trustees.

In MT-UCON2, for each trusted tenant, more fine-grained access control requirement than by MT-UCON1 is achieved. The model exposes the set of objects that can be accessed according to different trust relationships between tenants.

4 Information flow analysis of the MT-UCON model

Based on the information flow method used by Osborn [Osborn (2002)] to analyze the security of the RBAC model, Nauman et al. [Nauman, Ali, Khan et al. (2010)] analyzed the information flow of the $UCON_{ABC}$ model. For MT-UCON, the information flow may be within the same tenant or between tenants. This paper redefines the information flow in MT-UCON by adopting the analysis method used by Nauman.

To better define the information flow in MT-UCON, the following four assumptions are proposed.

(1) Only read and write operations cause information flow in the system, while other operations can be mapped to these two operations.

(2) The information flow is transitive.

(3) The subject set S is a subset of the object set O.

(4) For simplicity, only two usage sessions are assumed to be active. If there are more usage sessions, the information flow between them will be analyzed using assumption (2).

4.1 Basic rules of information flow

This paper uses the temporal logic to describe the information flow in the MT-UCON model [Lamport (1994)]. The related temporal operators are as follows.

● Once (◆): Returns true if the operand has been true in at least one past state.

● Eventually (◊): Returns true if the operand is true in at least one future state.

Rule 1: If states (s, o_i, read) and (s, o_j, write) exist simultaneously, then subject s will cause information flow from o_i to o_j, which is denoted as $o_i \dashrightarrow o_j$, and is formally described as follows.

State (s, o_i, read)=state (s, o_j, write)=accessing$\rightarrow o_i \dashrightarrow o_j$.

Rule 1 describes that in a certain access process, subject s performs a read operation on object o_i and a write operation on object o_j, which causes the information to flow from o_i to o_j. The two states must be tenable at the same time in the same usage session. Unlike the traditional static system, the state mentioned here may change any time. In other states, the information flow is not necessarily tenable. If o_i and o_j belong to two tenants, cross-tenant information flow is achieved, and the change of tenant trust relationship during the access process leads to the termination of information flow.

Rule 2: If there are two update operations, the attribute of the first subject s is updated using the attribute of object o_i, and the second update uses the attribute of the same subject to update the attribute of object o_j; in this case, information flow occurs from o_i to o_j.

If $update_1$ (s.attrx)=f (o_i.attry) and $update_2$ (o_j.attrz)=f (s.attrx) then $update_1$ (s.attrx)$\wedge\Diamond update_2$ (o_j.attrz)$\rightarrow o_i \dashrightarrow o_j$,

where $update_x$ can be either preupdate, onupdate or postupdate.

Rule 2 uses the transitivity of information flow, that is, $oi \dashrightarrow s$ and $s \dashrightarrow oj \rightarrow oi \dashrightarrow oj$. If o_i and o_j belong to two tenants, cross-tenant information flow is achieved, and the change of tenant trust relationship during the access process leads to the termination of information flow.

Rule 3: If there is an object attribute update related to a system attribute, the following information flow transitivity relation holds.

If $update_1$ (sys.attrx)=f (o_i.attry) and $update_2$ (o_j.attrz)=f (sys.attrx) then $update_1$ (sys.attrx) $\wedge\Diamond update_2$ (o_j.attrz)$\rightarrow o_i \dashrightarrow o_j$,

where $update_x$ can be either preupdate, onupdate or postupdate.

The basic principle of Rule 3 is the same as that of Rule 2, which just maps the environment into a subject. It is different from Rule 2 in that o_i and o_j must belong to the same tenant. The environment of one tenant cannot cause the update of an object attribute of another tenant; otherwise, the isolation of the tenant is destroyed.

4.2 Information flow of the MT-UCON model

Considering the persistence and attribute mutability of the MT-UCON model decision, the following four situations exist in the system. (1) Before access occurs and there is no attribute update operation; (2) before access occurs and there is an attribute update operation; (3) during the access process and there is no attribute update operation; and (4) during the access process and there is an attribute update operation. The information flow of these four situations is defined separately below.

Rule 4: Pre models without updates

$o_i \dashrightarrow o_j \Leftrightarrow \blacklozenge$ ($\blacklozenge$ (tryaccess(s,o_i,read)$\wedge(p_1\wedge\ldots\wedge p_m)\wedge\Diamond$(tryaccess(s,$o_j$,write)$\wedge(q_1\wedge\ldots\wedge q_n)\wedge\Diamond$(end access(s,$o_i$,read))))$\vee\blacklozenge$ tryaccess(s,o_j,write)$\wedge(q_1\wedge\ldots\wedge q_n)\wedge\Diamond$(tryaccess(s,$o_i$,read)$\wedge(p_1\wedge\ldots\wedge p_m)$ $\wedge\Diamond$(endaccess (s,o_i,write))))

Rule 4 indicates that if the information flows from o_i to o_j, in a system state, o_i and o_j must be readable and writable, respectively, and subject s performs tryaccess(s,o_i,read) and tryaccess (s,o_j,write) operations on them, which would satisfy the access conditions.

Rule 5: Pre models with updates

$o_i \dashrightarrow o_j \Leftrightarrow \blacklozenge$ ($\blacklozenge$ (tryaccess(s,o_i,read)$\wedge(p_1\wedge\ldots\wedge p_m)\wedge\Diamond$(tryaccess(s,$o_j$,write)$\wedge(q_1\wedge\ldots\wedge q_n)\wedge\Diamond$(end access(s,$o_i$,read))))$\vee\blacklozenge$ (tryaccess(s,o_j,write)$\wedge(q_1\wedge\ldots\wedge q_n)\wedge\Diamond$(tryaccess(s,$o_i$,read)$\wedge(p_1\wedge\ldots\wedge p_m$)$\wedge\Diamond$(endaccess(s,o_i,write)))$\vee\blacklozenge$ (update(s.attr)$\wedge\Diamond$(update(o_j.attr))),

where update$\in${preupdate, onupdate, postupdate}.

The first part of the disjunctive form in this rule is the same as that of Rule 4. The second part describes that because of the attribute update of subject s, o_i and o_j are accessed simultaneously, and finally the attribute update of object o_j leads to information flow from o_i to o_j. According to Rule 2, if objects o_i and o_j belong to two tenants, cross-tenant information flow is achieved.

According to Rule 3, if subject s of the second part of the disjunctive form is replaced by the system (sys), it can also lead the information flow from o_i to o_j, but o_i and o_j must belong to the same tenant.

Rule 6: On models without updates

$o_i \dashrightarrow o_j \Leftrightarrow \blacklozenge$ (tryaccess(s,o_i,read)$\wedge$(tryaccess(s,o_j,write)$\wedge\lozenge$(endaccess(s,o_i,read)$\vee$revokeaccess(s,o_i,read)))

Since the MT-UCON model introduces a "continuous" attribute, it is not necessary to perform authorization judgment before the access operation as in the traditional access control model but only during the access process. During the access process, some access requests no longer comply with the access rules because of attribute changes, which will result in the interruption of the currently performed access operations. Rule 6 differs from Rule 4 in that it includes the removeaccess operation.

Rule 7: On models with updates

$o_i \dashrightarrow o_j \Leftrightarrow \blacklozenge$ (tryaccess(s,o_i,read)$\wedge$(tryaccess(s,o_j,write)$\wedge\lozenge$(endaccess(s,o_i,read)$\vee$revokeaccess(s,o_i,read)))$\vee\blacklozenge$ (update(s.attr)$\wedge\lozenge$(update(o_j.attr))),

where update$\in$\{preupdate, onupdate, postupdate\}.

The definition of the first part of the disjunctive form of Rule 7 is the same as that of Rule 6, and the second part is the same as the second part of Rule 5.

5 Conclusions

To solve the problem of access control within and between tenants in a cloud computing environment, this paper proposed a multi-tenant usage access control model (MT-UCON), based on the traditional UCON. The model inherits the characteristics of attribute mutability and decision continuity of the UCON. Further, by establishing trust relationships between tenants, collaborative access control between tenants is achieved. The MT-UCON model consists of three sub-models—MT-UCON0, MT-UCON1, and MT-UCON2—which realize access control with different granularities and satisfy various application scenarios. Finally, the security of the model was analyzed by information flow analysis.

Acknowledgement: The authors are grateful for the anonymous reviewers who made constructive comments and improvements.

Funding Statement: This work was sponsored by Qing Lan Project of Jiang Su Province.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

Alam, Q.; Malik, S.; Akhunzada, A.; Choo, K.; Tabbasum, S. et al. (2017): A cross tenant access control (CTAC) model for cloud computing: formal specification and verification. *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 6, pp. 1259-1268.

Bendiab, K.; Shiaeles, S.; Boucherkha, S.; Ghita, B. (2019): FCMDT: a novel fuzzy cognitive maps dynamic trust model for cloud federated identity management. *Computers & Security*, vol. 86, pp. 270-290.

Calero, J.; Edwards, N.; Kirschnick, J.; Wilcock, L.; Wray, M. (2010): Toward a multi-tenancy authorization system for cloud services. *IEEE Security & Privacy*, vol. 8, no. 6, pp. 48-55.

Chu, X.; Qin, Y. (2010): A distributed usage control system based on trusted computing. *Chinese Journal of Computers*, vol. 33, no. 1, pp. 93-102.

Feng, C.; Qin, Z.; Yuan, D.; Qing, Y. (2015): Key techniques of access control for cloud computing. *ACTA ELECTORNICA SINICA*, vol. 43, no. 2, pp. 312-319.

Gu, K.; Wu, N.; Yin, B.; Jia, W. (2020): Secure data query framework for cloud and fog computing. *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 332-345.

Krautsevich, L.; Lazouski, A.; Martinelli, F.; Yautsiukhin, A. (2010): Risk-aware usage decision making in highly dynamic systems. *Fifth International Conference on Internet Monitoring and Protection*, pp. 29-34.

Lamport, L. (1994): The temporal logic of actions. *ACM Transactions on Programming Languages and Systems*, vol. 16, no. 3, pp. 872-923.

Lazouski, A.; Mancini, G.; Martinelli, F.; Mori, P. (2012): Usage control in cloud systems. *International Conference for Internet Technology and Secured Transactions*, pp. 202-207.

Liu, Z.; Xia, J. (2019): A cross-tenant RBAC model for collaborative cloud services. *Computers, Materials & Continua*, vol. 60, no. 1, pp. 395-408.

Mell, P.; Grance, T. (2011): The NIST definition of cloud computing. *Communications of the ACM*, vol. 53, no. 6, pp. 50-50.

Msahli, M.; Abdeljaoued, R.; Serhrouchni, A. (2013): Access control in probative value cloud. *8th International Conference for Internet Technology and Secured Transactions*, pp. 607-611.

Nauman, M.; Ali, T.; Khan, M.; Alghathbar, K. (2010): Information flow analysis of UCON. *International Journal of Physical Sciences*, vol. 5, no. 6, pp. 865-875.

Osborn, L. (2002): Information flow analysis of an RBAC system. *Proceedings of the Seventh ACM Symposium on Access Control Models and Technologies*, pp. 163-168.

Park, J.; Sandhu R. (2004): The UCON ABC usage control model. *ACM Transactions on Information and System Security*, vol. 27, no. 1, pp. 128-174.

Tang, B.; Li, Q.; Sandhu, R. (2013): A multi-tenant RBAC model for collaborative cloud services. *Eleventh Annual Conference on Privacy, Security and Trust*, pp. 229-238.

Tang, B.; Sandhu, R.; Li, Q. (2015): Multi-tenancy authorization models for collaborative cloud services. *Concurrency and Computation: Practice and Experience*, vol. 27, no. 11, pp. 2851-2868.

Tavizi, T.; Shajari, M.; Dodangeh, P. (2012): A usage control-based architecture for cloud environments. *26th International Parallel and Distributed Processing Symposium Workshops & PhD Forum*, pp. 1534-1539.

Wang, Y. D.; Yang, J. H.; Xu, C.; Ling, X.; Yang, Y. (2015): Survey on access control technologies for cloud computing. *Journal of Software*, vol. 26, no. 5, pp. 1129-1150.

Zuo, Q.; Xie, M.; Qi, G.; Zhu, H. (2017): Tenant-based access control model for multi-tenancy and sub-tenancy architecture in software-as-a-service. *Frontiers of Computer Science*, vol. 11, no. 3, pp. 465-484.