EasyChair Preprint

№ 1380

A authentication and access authorization mechanism on the PaaS platform

Xu Shuangshuang and Zhu Hongliang

August 9, 2019

A authentication and access authorization mechanism on the PaaS platform

Shuangshuang Xu
School of Beijing University of Posts and Telecommunications
Beijing, China
xiaoxingxingno_1@163.com

Hongliang Zhu
School of Beijing University of Posts and Telecommunications
Beijing, China
zhuhongliang@bupt.edu.cn

***Abstract*— With the development of cloud computing and Docker technology, the continuous delivery technology has matured. The PaaS platform provides a new software architecture development architecture that is more suitable for business expansion and functions. The PaaS platform system provides agile development and good scalability for multi-tenancy. At the same time, the security of the system becomes a key factor for the sustainable development of the system. PaaS can customize different identity authentication and access control for different tenant that uses different services. By comparing the research of identity authentication in the traditional environment, this paper analyzes the limitations and shortcomings of its use under the PaaS platform to multi-tenant and focuses on the characteristics of multi-tenant sharing service on PaaS platform. Firstly, the identity authentication is realized through the ticket authentication method. Then, based on the cloud computing environment and the resource dynamics under multi-tenancy, the timeliness of cloud resources and other factors, from the perspective of user service session access control, based on RABC and UCON model ,the user, authority, resources and control are proposed. The access control method described by the metadata is used to ensure the security of the user's access to the cloud resources in the PaaS environment. The paper elaborates on the security and usability of the key generation, distribution, update, and metadata access control processes. Practice shows that the PaaS environment based on the proposed unified authentication and metadata access control can effectively protect the dynamic access control and security isolation of different services for different tenants. At the same time, according to the built cloud resource access control model, cloud resource access control systems with permission separation, user attribute and cloud resource attribute constraints, lease time constraints, usage rate control can be flexibly constructed. And the related constraint elements can be expanded as needed according to the business requirements, so as to better meet the cloud resource access control requirements with multi-tenant sharing and dynamic characteristics in the cloud environment.**

Keywords—authorization; authentication; PaaS; Metadata driven;

I. INTRODUCTION

PaaS transforms the development environment or various middleware in the cloud into a service that is delivered to users. The PaaS platform can be divided into two categories, one is the application deployment and operation platform APaaS (application platform as a service), and the other is the integrated development platform IPaaS (Integration platform as a service) [1]. PaaS provides multi-tenant services. It provides multi-tenant with full or partial application development, deployment and testing platforms or development interfaces that can be accessed[2]. Developers use these tools or interfaces to develop applications and deploy developed applications to the PaaS vendor's cloud infrastructure[3]. Tenants do not have to maintain cloud infrastructure such as servers and operating systems, but can manage their deployed applications. You can manage the environment parameters that run your application. PaaS's multi-tenancy feature enables maximum level of application and database resource sharing, allowing developers to focus on application development[4]. The PaaS platform enables resource sharing, which inevitably brings security issues. Access control technology is an important tool. .Access control can be considered from two aspects: authentication and authority[5-6]. Authentication refers to determining the identity of the requester and setting it at the boundary of the system. It is the first step to enter the system. Authority refers to a kind of judgment and control of whether the subject of access allows access to a specific resource [7]. The authentication method can be based on a username password, hardware credentials ,biometric identification. In the PaaS environment, it consists of a large number of services, frequent calls between services, large differences in service operating environments, mutual service impact, high system openness[8]. PaaS design ideas for multi-tenancy, scalable, customizable, and fault-tolerant. The Characteristics of PaaS makes the tenant's authority management more complicated. The traditional method cannot be fully adapted. However, there is no uniform standard in the industry to provide reference for the access control implementation of PaaS for multi-tenancy.

II. RELATED WORK

The cloud platform generally adopts two methods to complete the authentication:

- access layer c and service layer authorization. If the internal security of the PaaS system is guaranteed, access between services is strictly controlled and encrypted, which can greatly simplify system implementation

- service authentication and authorization. In addition to access layer services, each service must be authenticated.

Traditional access control models, including Discretionary Access Control (DAC), Mandatory Access Control (MAC), and Role-based access control (RBAC). Due to the emergence of virtual technology, the access control technology in the cloud computing environment has expanded from user authorization to virtual resource access and secure access to cloud storage data. The scope of use and control methods of access control technology have increased significantly; the various services in the cloud computing environment are in different security management domain. When users access resources across domains, you need to consider issues such as unified policies, mutual authorization, and resource sharing. Traditional access control models are no longer able to meet the new cloud computing architecture requirements. Cloud computing has a multi-tenant-centric, big data-based service model, so the access control in cloud computing has to redefine the concepts of subject and object, which leads to the optimization of traditional access control models. The update makes it more suitable for the cloud computing environment; the problem of complex role permissions, frequent user changes, numerous administrator roles, complex levels, and the distribution of permissions is quite different from the traditional computing model.

At present, the research on the access control model in cloud computing has different research contents and methods according to the different functions of the access control model. Literature [9-11] adopts cloud computing access control based on task model. The research focuses on the task angle modeling in the workflow, and dynamically manages the permissions according to the different tasks and task states. Every user-level access can be modeled and analyzed according to task constraints, which greatly enhances the dynamics of access control in the cloud. Literature [12-14] uses attribute-based access control in the field of cloud security. The main focus is on combining RBAC and ABAC to ensure user privacy and access control. In addition, time constraints are more important attribute constraints in the cloud computing environment. Time factors are everywhere. Users only have specific roles in specific time periods, and the working mode of cloud computing is on-time billing. Therefore, it is necessary to constrain the access control of data in the cloud through time. Therefore, it is urgent to support the RBAC model to support complex time-constrained modeling. The literature [15-17] uses the control model (use control, UCON for short). UCON includes two elements of obligations and conditions in addition to the basic elements of the authorization process. Literature [18-19] studies cloud computing access control based on BLP model. The BLP model is a mandatory access control model, which is mainly used to compare systems with emphasis on confidentiality or cloud environments, such as military and financial industries. Currently, research on BLP models in cloud computing focuses on modifying traditional BLP models to make them more Suitable for cloud computing environments

These models cannot be fully copied into the PaaS platform [20-21].

At present, research related to security domains and tenant domains in the cloud environment focuses on a certain local point [22-25], and it lacks systematic analysis and research work combined with cloud computing dynamics and multi-tenancy. Lack of relevant reference indicators that can be used to guide specific practices. At present, the access control under the PaaS environment has the following three problems to be solved for multi-tenant access control:

- There is no effective integration between the identity authentication method and the tenant access control policy;
- The PaaS platform cannot dynamically and conveniently handle the multi-tenant customization of platform service access policy changes, and the platform cannot dynamically limit tenant usage.
- Handling multi-tenant authentication and access control services is not efficient.

In view of the above problems, this paper proposes a multi-tenant-based customizable access control method in PaaS. Configure access control policies through metadata definitions, combining access control with tenant session mechanisms,

Firstly, identity authentication is implemented by means of tickets, which solves the problem of complicated storage of traditional ticket and low access efficiency. In view of the above problems, this paper proposes a multi-tenant-based customizable access control method in PaaS. Configure access control policies through metadata definitions, combining access control with tenant session mechanisms, Firstly, identity authentication is implemented by means of bills, which solves the problem of complicated storage of traditional bills and low access efficiency. Second,
The PaaS platform streamlines access control. The variable points in the process are described by metadata in the RABC and UCON models, enabling the platform to dynamically control tenant access to the platform. Finally, each service in the PaaS platform uses the agent to implement security calls and permission access control within the service. The internal services implement access control and secure calls to ensure seamless security of the entire platform. The agent's metadata access control policy by resolving the identity authentication does not require additional calls to the remote interface to achieve tenant access, which can achieve better efficiency. It can reduce the time for the tenant to request the service on the PaaS to a certain extent. At the end of the paper, it is verified by experiments that the efficiency of the ticket customized by JCE is higher than that of the JWT ticket [26-27], and the access control efficiency is compared and the metadata is proved under the two typical application scenarios of using

metadata customization and not using metadata customization. Access control can dynamically implement access control, while access efficiency is better than without metadata for access control. Therefore, in the cloud computing environment, the metadata-based access control policy satisfies the dynamic implementation of the multi-tenant access control of the PaaS platform, and effectively controls and manages computing resources, storage resources, and network resources in the cloud computing, thereby ensuring user friendliness. And with good performance, it is a good solution.

III. Design of Identity Authentication and Access Control

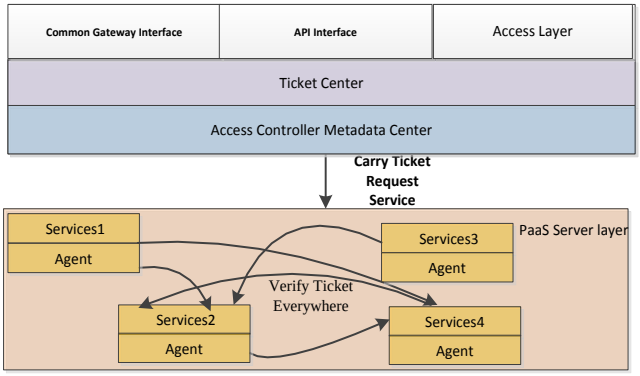

Fig. 1. The overall architecture of the PaaS platform authorization and Authorization

In view of the usage scenario of PaaS platform, this paper defines the complex and diverse authentication means of the whole system as a unified authentication method: access layer service promulgate tickets, other services verify ticket. Access layer authenticates key information such as user identity. After authentication, a string containing user information and signature information is generated. This process is called " promulgate tickets ". "Ticket" represents the identity of this request. The precondition for generating ticket is to prove identity. The generation of ticket is guaranteed by authentication, so the ticket themselves can be used as authentication. After the ticket is generated, as the request is transmitted backwards, each service receiving the request and the ticket can authenticate by verifying the validity and identity consistency of the ticket signature. This process is called " Verify tickets " . Ticket use RSA public-private key system. The service that generates the ticket holds the private key, and the service that verifies the ticket holds the public key. Authentication and ticketing must be performed in the same service. Once the two are separated, the generated ticket are not credible. Metadata customized access control

A. Data Structure Of Tickets

The ticket is designed based on the idea of asymmetric encryption, and the plaintext of the ticket is designed by serializing the data of the JCE structure .The content of ticket is mainly authentication information, that is, all the fields participating in authentication. “mac_data” performs a Hash signature calculation on all fields below mac_data (excluding mac_data itself) to ensure that the ticket is not tampered . At present, the signature calculation uses the key calculation method hamc-sha256 (generating a 32-byte signature), and other people cannot generate the signature even if they know the plaintext. Request Unique ID: In theory, a ticket in one request link should not appear in another request link, and the ID should remain unchanged throughout the request chain. Create Time: it represents the ticket generation time. It is mainly used to judge whether the ticket has expired. Tenant ID: The tenant id information associates the ticket information with the tenant information. You can obtain the tenant's permission information through the tenant id in the ticket for access control.

B. Metadata customized access control

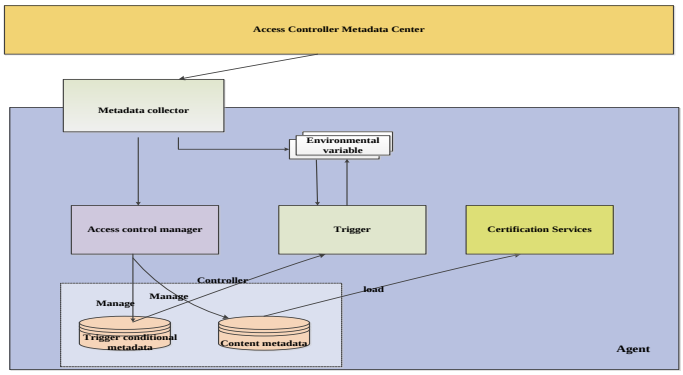

Fig. 2. Metadata-driven access control details

Combining the dynamics of resources in a cloud computing environment, the timeliness of cloud resources, and the diversity of multi-tenant needs, From the perspective of user service session access control, this paper dynamically builds a customizable PaaS access control model in PaaS platform based on **RBAC** and **UCON** models.This article defines the basic metadata for access control as:

Tenant: Tenants are the specific operators of the application system. Tenants can customize their own services or the service level. The tenant set is expressed as **U**

Role: Has the ability to use specific resources and access specific services. The character set is represented as **R.**

User attribute (ATT(U)): Identify tenant capabilities and characteristics.

Operation: The specific access behavior of the tenant to the target service in the PaaS platform or the use of the target resource by the tenant. The Operation set is represented as **O.**

Object： Target objects that can be accessed and used in the PaaS platform. The resource set is represented as **B**.

Object attribute （ ATT(B) ） : Identify important information about resources

Permission: it refers to the right of a subject to access an object in a specific way (such as reading or writing). Traditional access control treats rights as static elements independent of the subject's activity. The UCON model is in the subject's attempt. When accessing the object, the user's operation authority is dynamically determined according to the object attributes, permissions, obligations, and conditions.

Authorization: An important part of the UCON model, abbreviated as **A**. It is used to make decisions whether the subject can operate on the object. the attributes of the subject and object/the requested permissions (such as read and write rights) /set of permissions rules which determines authorization . The authorization in UCON includes both traditional pre-authorization and authorization based on different control rules during the access process. In addition, the update the value of the subject and object is caused the access, which in turn affects this or other access decisions.For example, if the tenant accesses the KAFKA service provided by the PaaS platform, the platform will monitor the use process. The monitoring result will be an important basis for the tenant's authority to use the platform this time or later.

Obligation refers to the action that the subject must perform before or during the visit, referred to as **B**.pre-obligation means that the subject must satisfy certain conditions before the access request is executed. On-obligation refers to the condition that the authority must be continuously satisfied or periodically met during the exercise. The subject fulfillment obligation is not statically set in advance, but is dynamically determined according to the attributes of the subject and object. The performance of the obligation may update the variable attributes of the subject and object, and these updates affect the current or future usage decisions.

Condition refers to the decision factor that is oriented to the environment or system, abbreviated as **C**. The condition evaluates the current hardware attributes or system-related restrictions to determine whether the user request is satisfied. For example, the user must use the service at a specified terminal or a specified time period; or limit the network traffic to a certain extent.

The PaaS data table is composed of a basic table, a metadata table, and an extended table. The basic table is a shared data column, and the RecordID column is used as a foreign key to associate with the customized data in the extended table; all tenant's customized data is stored in the public extended table. The data in the extended table is increased vertically in terms of scale. Information about custom fields stored in the metadata table, such as FieldName, DataType, etc.

PaaS uses the metadata described above to control PaaS access control through metadata-driven customization techniques. Throughout the process, the dynamic access control information in the platform is interpreted as reasonable metadata, and the process of extracting, converting, manipulating, and controlling the metadata, and finally loading the tenant customization information and the access control information into the application.

The access control metadata information is stored in the metadata table. The PaaS platform normalizes the access control into a process, which divides the entire process into many basic points that are customized through conditional parameters. In the access control process, the PaaS platform is customized through operations. The corresponding configuration conditions are associated to customize these parts in the base point, so that the variable base points in the access control flow can be controlled according to the configuration conditions of the access control metadata, and the access control is customized, in order to improve performance efficiency.

IV. TECHNICAL REALIZATION

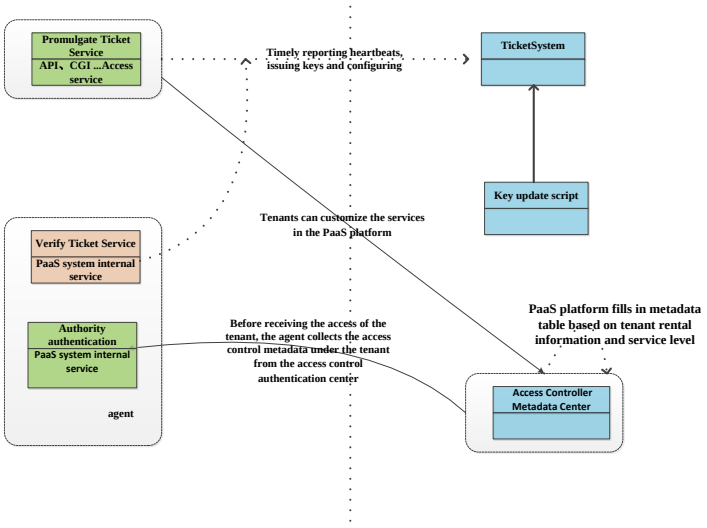

Fig. 3. Identity and access control details

A. *Realized Of Ticket*

1) Methods Of Promulgating And Checking Tickets.

There are three main ways to promulgate ticket and verify ticket as shown in the following figure:

- Option 1: Deploy the local agent to pull the key, the key is stored in the Agent memory, and the service requests the Agent by means of a local call (such as Unix Socket) to promulgate ticket and verify ticket.
- Option 2: Deploy the local agent to pull the key. After the agent obtains the key, it writes it to the local file or shared memory, and the service directly reads it.
- Option 3: Business services directly request unified ticketing and ticketing services

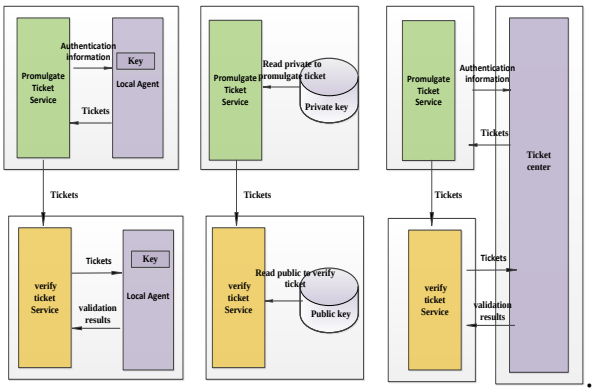

Fig. 4. Three ways for the business to issue tickets or check tickets

TABLE I. THREE SCHEME COMPARING

Three Scheme comparing			
Number	***Scheme***	***Advantage***	***Disadvantage***
1	local agent promulgate ticket and verify ticket	Key security is better, and public and private key permissions are much easier to control. There is no need to worry about theft after the key is sent to the local, or the private key is taken by the service without permission.	At present, the background service is mainly deployed on the physical machine. The private key is issued according to the machine. Multiple services in one machine must be checked tickets. If the first idea is used, resource isolation cannot be performed, which may cause a service to hang the agent and other services of this machine to not issue tickets and check tickets
2	Business direct read key to promulgate ticket and verify ticket	Usability is better. The Agent is only responsible for pulling the key, even if it is hanged, it does not affect promulgating and verifying ticket.	Need to rely on operation and maintenance means to control the private key: the promulgating ticket module can not be mixed with the ticket verifying module
3	Business visit unified promulgate and verify ticket service	No need to deploy Agent, simple architecture, simple logic	The ticketing and ticket checking services become a single point of the whole system. Once the problem occurs, the global service will be unavailable; authentication and ticketing cannot be completed together, and security cannot be guaranteed.

The hybrid deployment of Scenario 2 and Scenario 3 is adopted in the system. For scenario 2, when stored locally, Use the native IP as part of the key to do another symmetric encryption to prevent the private key from being copied to other machines. The ticket uses the global module "global agent" to deploy an agent (called "ticket agent") to synchronize the public and private keys of the ticket on the entire network. In addition, there is a module "ticket issue" for RPC to acquire tickets, and "ticket broker" module for RPC verification tickets.

The promulgating of the ticket, for the service that have the ability to authenticate (such as API check-in service, platform check-in service, etc.), as in scenario 2, the private key is directly deployed to these services and the ticket is directly generated. A scenario in which a ticket needs to be generated for holding other authentication rights, Referring to the scheme 3, a service of the ticket exchange service "ticket issue" is specially deployed, and the authentication basis is sent to the service. After the service is successfully verified, the ticket is generated and returned, and the ticket is set in the request protocol package on the client side.

Set in the request protocol package.

Ticket Checking. Most services, if the ticket agent can be deployed directly, such as option 2, the ticket can be completed directly. For some places where the ticket agent cannot be deployed, or interact with other ticket systems to complete the ticket verification, the ticket can be completed by RPC (Scheme 3).

2)Key Issuing

There are two ways to send a key: the agent actively pulls or the key service actively pushes.

TABLE II. TWO SCHEME COMPARING

Two Scheme comparing			
Number	***Scheme***	***Advantage***	***Disadvantage***
1	The Agent requests the	the Agent is only	The central service need

Two Scheme comparing			
Number	***Scheme***	***Advantage***	***Disadvantage***
	key by periodically polling the central service	responsible for pulling the key, and the central system is only responsible for giving the key. The central system can obtain the status of all agents according to the Agent polling request.	to judge whether agent have rignts to get the private key and the public key.
2	The central service actively pushes the key to the agent	The global push is triggered only when the key is updated, and When a new Agent is added, you only need to push to new agent. .	The agent needs to listen to the port. The central system needs to periodically check the machine changes, and actively resend the failed agent periodically, which is more complicated.

Option 1 is selected in this system. In order to control the permissions of different machines to obtain keys (can get the public or private key). The central system records a white list of the machines IP and the permissions of these machines which accessing the ticket system. When the agent comes to the request, the key storage service will find the authority of the corresponding machine according to the source IP and answer the corresponding key. Since the module deployment situation may change at any time, another scheduled task is responsible for querying all the machines that access the ticket system and adding them to the whitelist.

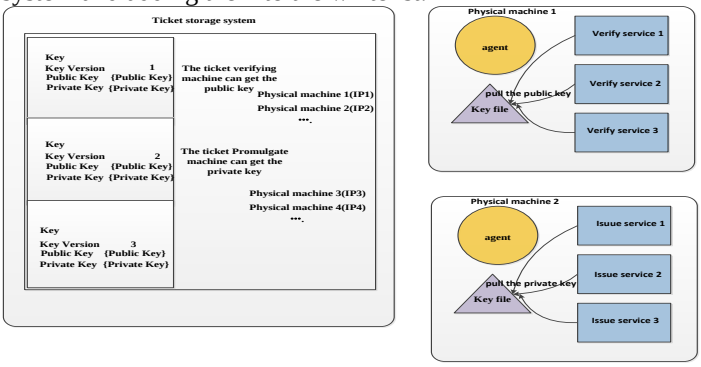

Fig. 5. Key Issuing

B. *Access Control Implementation*

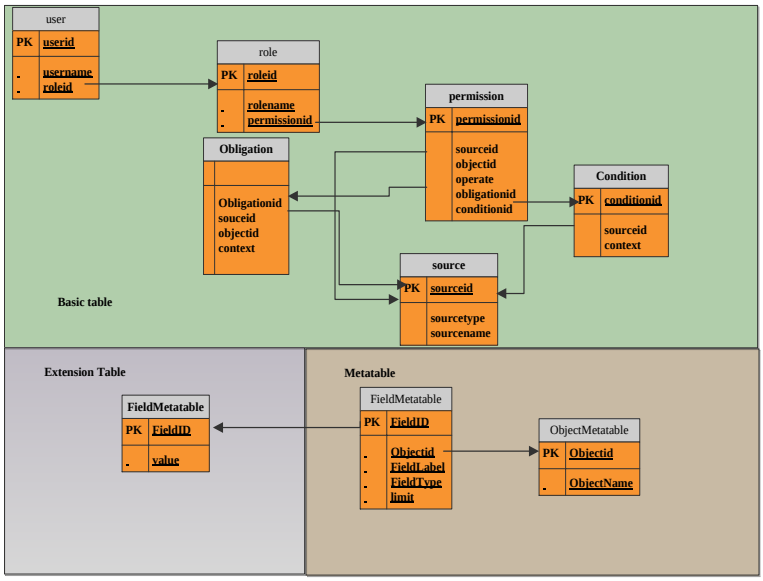

Fig. 6. Access control metadata table design

1) Metadata collection

The responsibility of the data collector is to collect metadata for the access control service engine, so that the services of the PaaS can obtain the access control information of the platform to the tenant at any time. The data collector runs on the agent of each service of the PaaS platform. Before receiving the access of the tenant, the agent collects the access control metadata under the tenant from the access control authentication center, including the data table structure, field attributes, integrity constraints, etc.

2)Metadata Processing

Access control metadata provides data support for the access control process. This type of metadata primarily consists of trigger condition metadata and access control content metadata. (1) Trigger conditions. The trigger condition is a condition for activating an access restriction, and the trigger determines the state of the environment variable according to the trigger condition. The access control of a service contains one or more limit trigger conditions, and there is a logical relationship between multiple trigger conditions. A decision rule contains an environment variable, a threshold, and a relational operator. (2) Access control template. The template defines three aspects of information: I access to resource information; II service restriction content; III trigger condition. Therefore, the metadata information collected by the metadata collector is filled into the access control template, and the template includes a comprehensive description of the access rights of the services in the PaaS.

V. EXPERIMENT AND TEST

A. *Comparison of the efficiency of encoding and decoding JCE and JWT tickets*

Lab Environment:
Container configuration:
Minimum CPU 0.5 core
Minimum Memory 500MB
Maximum CPU 16 core
Maximum Memory 24000 MB
Test ideas:
Under single thread, JCE and JWT are used to encode and decode 10000000 times for the same data structure, and the data type tested is String.
Test indicators:
Encoded length, code decoding consumption time, code decoding rate

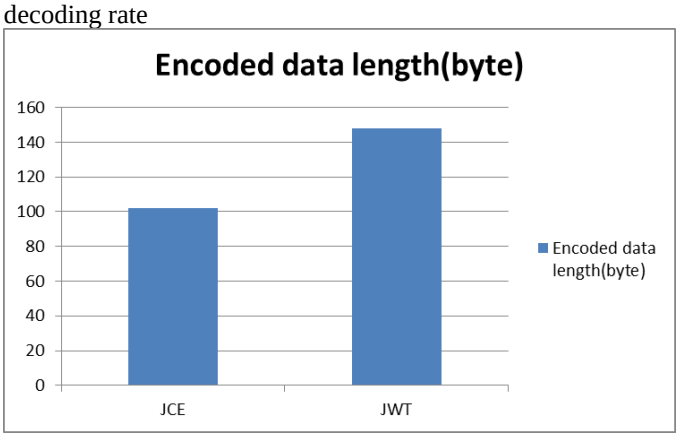

Fig. 7. Comparison of the length of tickets encoded by JCE and JWT

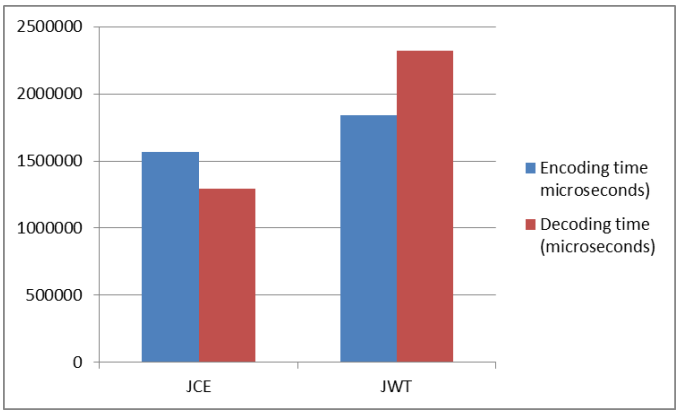

Fig. 8. Time-consuming comparison of encoding and decoding

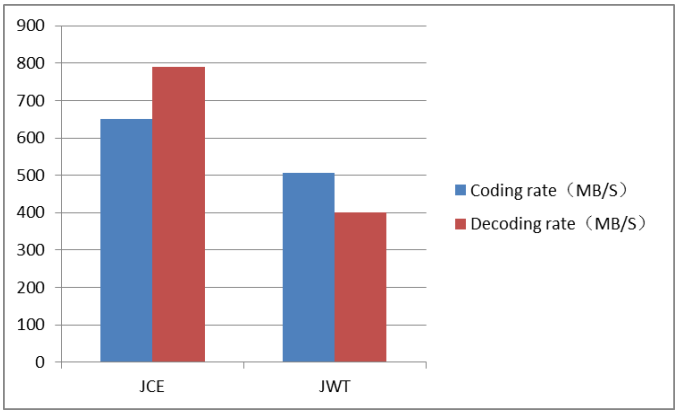

Fig. 9. Example of a figure caption. (*figure caption*)

Test conclusion:
In terms of Encoded length, code decoding consumption time, code decoding rate, JCE is better than JWT

B. *Control permissions through metadata*

Test ideas:
Scenario:
Tenant A applies for development rights to Service B through the web page form
Test indicators:
Comparison of user operation before and after permission assignment

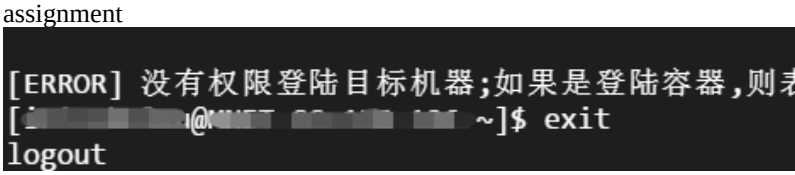

Fig. 10. Permission denied

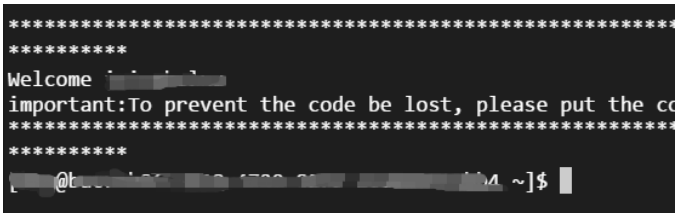

Fig. 11. Have permission

Test conclusion:
Experiments prove that the authority control can be effectively performed by means of metadata description.

VI. CONCLUSION

This paper focuses on the system authentication and access control in PaaS. Based on the analysis of the multi-tenant access to each service on the PaaS platform, the system establishes the identity authentication and access control process in the PaaS platform. The way to generate tickets using JCE is adopted. The certification is implemented. Based on the RBAC and UCON models, the metadata-driven approach is used to construct a dynamically customizable access control for PaaS multi-tenancy. Practice tests show that

the tickets generated by JCE can be efficiently returned to the identity authentication, and the metadata-driven access control model can effectively guarantee the flexible separation of permissions, user attributes and cloud resource attribute constraints, lease time constraints, usage control, etc. The cloud resource access control system, and the related constraint elements can be expanded as needed according to the business requirements, so as to better meet the cloud resource access control requirements with multi-tenant sharing and dynamic characteristics in the cloud environment.

REFERENCES

[1] Yefim V. Natis, Benoit J. Lheureux, Massimo Pezzini, David W. Cearley, Eric Knipp, Daryl C. Plummer PaaS Road Map: A Continent Emerging[J].Gartner Research, 2011.

[2] Ribas M, Lima A S, Souza J N D, et al. A Platform as a Service Billing Model for Cloud Computing Management Approaches[J]. IEEE Latin America Transactions, 2016, 14(1):267-280.

[3] QI L, ZHANG H F, ZHANG T X, et al.PaaS cloud platform based on container technology[J].Telecommunication Science,2017,33(4):177-18

[4] Luis Rodero Merino.Building safe PaaS clouds: A survey on security in multitenant software platforms[J]. Computers and Security,2012,33(1):96-108.

[5] Shen, J., Liu, D., Liu, Q., Sun, X., & Zhang, Y. (2017). Secure Authentication in Cloud Big Data with Hierarchical Attribute Authorization Structure. IEEE Transactions on Big Data, 1–1.doi:10.1109/tbdata.2017.2705048 authority

[6] Shen, J., Liu, D., Liu, Q., Sun, X., & Zhang, Y. (2017). Secure Authentication in Cloud Big Data with Hierarchical Attribute Authorization Structure. IEEE Transactions on Big Data

[7] Lin Guoyuan，He Shan，Huang Hao，et al. Access controlsecurity model based on behavior in cloud computing en-vironment［J］. Journal on Communications，2013，33(3) :59 － 66. (in Chinese)

[8] Abdulrahman A A, Muhammad I S, Saleh B, et al. A dis-tributed access control architecture for cloud computing［J］. IEEE Software, 2012, 29(2) : 36 － 44.

[9] Thomas R, Sandhu R. Task-Based authorization controls (TBAC): A family of models for active and enterprise oriented

[10] authorization management. In: Proc. of the 11th IFIP WG11.3 Conf. on Database Security. Lake Tahoe, 1997. 166-181.

Deng JB, Hong F. Task-Based access control model. Ruan Jian Xue Bao/Journal of Software, 2003,14(1):76-82 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/14/76.htm

[11] Li FH, Su M, Shi GZ, Ma JF. Research status and development trends of access control model. Chinese Journal of Electronics,2012,40(4):805-813 (in Chinese with English abstract).

[12] Huang JW, David MN, Rakesh B, Jun HH. A framework integrating attribute-based policies into role-based access control. In: Proc.of the SACMAT 2012. 2012. 187-196. [doi: 10.1145/2295136.2295170]

[13] Bertino E, Bonatti P, Ferrari E. TRBAC: A temporal role-based access control model. ACM Trans. on Information and System Security, 2001,4(3):191-223. [doi: 10.1145/501978.501979]

[14] Wang XM, Zhao ZT. Role-Based access control model of temporal object. Acta Electronica Sinica, 2005,33(9):1634-1638 (in Chinese with English abstract).

[15] Park J, Sandhu R. Towards usage control models: Beyond traditional access control. In: Proc. of the 7th ACM Symp. on Access Control Models and Technologies (SACMAT 2002). 2002. 57-64. [doi: 10.1145/507711.507722]

[16] Krautsevich L, Lazouski A, Martinelli F, Yautsiukhin A. Risk-Aware usage decision making in highly dynamic systems. In: Proc.of the 5th Int'l Conf. on Internet Monitoring and Protection. Barcelona: IEEE Computer Society, 2010. 29-34. [doi: 10.1109/ICIMP.2010.13]

[17] Chu XB, Qin Y. A distributed usage control system based on trusted computing. Chinese Journal of Computers, 2010,33(1):93-102(in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2010.00093]

[18] Lin GY, He S, Huang H, Wu JY, Chen W. Access control security model based on behavior in cloud computing environment. Journal on Communications, 2012,33(3):59-66 (in Chinese with English abstract).

[19] Weng CL, Luo Y, Li ML, Lu XD. A BLP-based access control mechanism for the virtual machine system. In: Proc. of the 9th Int'lConf. for Young Computer Scientists (ICYCS 2008). 2008. [doi: 10.1109/ICYCS.2008.503]

[20] Kritikos, K., Kirkham, T., Kryza, B., & Massonet, P. (2017). Towards a security-enhanced PaaS platform for multi-cloud applications. Future Generation Computer Systems, 67, 206–226.doi:10.1016/j.future.2016.10.008

[21] MT Sandikkaya, AE Harmanci, "Security problems of platform-as-a-service (paas) clouds and practical solutions to the problems," 2012 IEEE 31st Symposium on Reliable Distributed Systems,2012 ieeexplore.ieee.org

[22] Cabuk S, Dalton C I, Eriksson K, et al. Towards automated security policy enforcement in multi-tenant virtual data centers［J］. Journal of Computer Security, 2010, 18(1) : 89-121. (11)

[23] Gerges S, Khattab S, Hassan H, et al. Scalable multi-tenant authorization in highly-collaborative cloud applications［J］. International Journal of Cloud Computing and Services Science (IJ-Closer) , 2013, 2 (2) : 106-115.

[24] Lu Zhi-gang, Jiang Zheng-wei, Liu Bao-xu. A virtual network access control method based on VxLAN［J］. Computer Engineering, 2014, 40(8) : 86-90.

[25] Paladi N, Michalas A, Gehrmann C. Domain based storage protection with secure access control for the cloud［C］. Proceedings of the 2nd International Workshop on Security in Cloud Computing, ACM, 2014: 35-42.

[26] JSON Web Token(JWT). RFC7519. https://tools.ietf.org/html/rfc7519. 2015

[27] 24Badr Eddine Sabir; Mohamed Youssfi; Omar Bouattane; Authentication and load balancing scheme based on JSON Token For Multi-Agent Systems. Hakim Allali Procedia Computer Science 2019-10.1016/j.procs.2019.01.029