ACADEMIC FORA
www.academicfora.com

FULL PAPER PROCEEDING
Multidisciplinary Studies
Available online at www.academicfora.com

Full Paper Proceeding ECBA-2017, Vol. 03 – Issue 42, 1-5

ISBN 978-969-683-369-7

ECBA-17

Identify Cloud Security Weakness related to Authentication and Identity Management (IAM) using Openstack keystone Model

Nasser Abdulla[1*], Ergun Erçelebi[2]

[1, 2] *Gaziantep University, Department of Electrical and Electronic Engineering Gaziantep, Turkey*

Abstract

The security of cloud system is one of the most important weakness in current cloud system, in this work we were able to produce a solution for two problem in current OpenStack authentication mechanism because of use unsecure password through http protocol and use 32 bit for token that need the user to login many time during the session and without certification, we solve the problem by installing PKI on keystone (openstack identity System) that can generate valid token for long period. Upon successful authentication of the openstack component (Nova, Swift, and Glance) must have a Keystone authentication to verify that authentication of the token is true, which will give us a better scalability, Keystone have main role for both authentication and authorization. Then we compare the PKI system with the current working UUID that adapting by openstack. Our design is to manage the identity across cloud applications and related services easier. Our basic emphasis is on simplicity development and integration of other services, while applying the existing authentication and authorization and privacy models. In essence, user needs to have single, faster, low cost, easy to maintain and easy in/out and around cloud environment. That we focus on identity management for users, which will be used by any application, platform and across the cloud environment for authentication, authorization services and user management

Keywords— Openstack, Authentication, Security, SSO, Authorization

Introduction

In order to access the cloud services, today we need a secure authentication, higher security policy management with secure applications, services, systems and connectivity. most of the organizations are moving towards the cloud solutions due to the fact of its Flexibility in terms of meeting their current and future needs, cheap solutions, balanced cost, (pay as you use) that solve of verity of business needs and agility. Cloud is changing the companies to achieve their business needs, but they are more concerned about security aspects than such flexible solutions. Because of dealing with highly sensitive data. Therefore, it is important to secure the identities of such sensitive data with the good secure architecture and security policy enforcement. They are trying to understand what security layers are and whether they meet their criteria or not. Cloud is open so any users and enterprise can use it. Today we need such design that accommodates verity of users and enterprises in a cloud.

Openstack Clouding system and keystone Components

OpenStack: Is an open-source cloud computing platform that can provide privet and public cloud as an infrastructure as a service (IaaS) solution. This open source system consists of a multi-projects to control the (processing, storage, and networking) that system provide a data center which managed through a web-based dashboard, command-line, or an API. OpenStack.org released the system under Apache License. Each Openstack version consist of multi-Project Components (Nova, Glance, Keystone, Swift, Cinder, Horizon, Neutron, Heat, Ceilometer) most of our work will be with Keystone that provides an identity of users mapped to the OpenStack system which provide authentication and authorization services. It provide the authentication service by integrate with existing active directory services like LDAP. It use the username and password credentials, and (UUID) as token-based systems and AWS-style logins. Also the it contain all query list of the services deployed in an OpenStack cloud in a single registry.so it provide to the Users and third-party program the way to resources after successful authentication . Therefore, identity access strategy plays a vital role in a cloud environment. It must be coordinated between various cloud services and between enterprise identity data stores. It needs Flexible, centric identity

*All correspondence related to this article should be directed to Nasser Abdulla, Gaziantep University, Department of Electrical and Electronic Engineering Turkey.
Email: Na66610@mail2.gantep.edu.tr

management that supports such identity protocols, mechanisms, various Platforms, applications, etc. Users must be able to control their own personal information

Formulation of Problem

This proposal is built on SOA that delivers a smooth way to access the services produced by openstack. Our proposal provide an open standard and secure construction for the end users, while the other security services use the web services that already defined by security system. Moreover, the authentication and authorization system that has been built are tightly coupled to provide a central Enterprise open-scale distributed cloud environments. Central security server a generate certificates and to handle the authentication by (SSO) single singe on and other authorization protocols. That contains the active contact with identity management server. In the future we can use the identity management server to provide more help by using it in federated identities

Authentication Problems with OpenStack

The default authentication system for openstack that use Passwords for authentication and transportation method that use clear text to send password by using http (Hyper Text Transfer protocol), which give no security than https that encrypt the date before send it from user to system. And The Authentication token generated don't use 32 bit certificate, which is significantly very small when compare it to PKI server token that our proposal produce. And can have been live for longer time that offer to the user a token validity for longer time , that give the users longer token time and reduce the need to login again after the validity expires. And reduce the overhead on the system of UUID token message .in This case PKI tokens are better in security since the service can trust from where token is coming from and give us mush efficiently since the user doesn't have to validate it on every request like UUID token.in the table below the different between PKI and UUID token :

PKI(Public Key Infrastructure) Tokens:	UUID Tokens
Remove the need to call Keystone for each request.	Token ID = UUID -- a unique string used to identify the token.
Keystone holds both a public and private certificate.	The token does not contain embedded metadata
Anybody can get the public certificate from Keystone via REST API call, but the private certificate key is not exposed outside of the Keystone service itself	The endpoint service must invoke Keystone to validate the given token
Keystone creates the token JSON object and encrypts it using the private key. It then creates a signature (token ID) of the encrypted token using MD5.	In response Keystone will return the metadata associated with the token including roles, tenancy, etc
Tokens can be decrypted and validated locally by applying the public key.	

Using PKI to Extending OpenStack Authentication

PKI (Public Key Infrastructure). One of most and secure Tokens type that documents, Encrypted signed using X509 standard. It work by generation a public/private key pair who can encrypted data and check the authenticated user keys that signed with X509 certificate by set user privet key that as hidden from all public and readable by the user ,network the key is store in local Certificate Authority (LCA) certificate. These files can be generated by external server externally.

PKI can be connected to Keystone. But to have more scalability we need to have a separate CA (Certificate Authority), like CSS to provide a valid certificate that can handle the certificate requests. Authentication process work on two step in Keystone. First it check the user-name and password then if successful authentication occur, Keystone will send the token in the Second process to provide SSO (Single Sign-On) and give the user authentication to the other system service. Adding the PKI server will give a better security in the first process and increase the scalability and security of the second stage will provide signed authentication and authorization token.

In the using Keystone UUID token it required an active involvement of users for to validate the of tokens, but by using PKI we can use the privet key to encrypt the signed and encrypted information of the token, that make certainty the which services have Keystone public key and thus it can decrypt it, we minimize the need of resend request for the token validation and return data back to Keystone to check the permission

The Solution of Authorization security in openstack

In our work we isolated the access control and the authorization SAML (Security Assertion Markup Language) and XACML, to provide single sign-on to the Openstack clouding system , by integrating local Certification authority of PKI. The Model of our project work using OpenStack Platform. Depending on SSO single sign-on and PKI in

Keystone. To implement PKI in the Keystone Web server have to be SSL configured first. Because of LCA will provide the certificates for both side Web server and the Keystone.

Authentication Process

After the registration of the user in Keystone, OTP (one time password) created to start a key-pair for the user and the system. Public key that has been signed by CA (Certificate Authority) will be store in the Keystone server and the Private Key is kept in the user system.

Development of IDMS

The OpenStack clouding system are use Keystone to manage the identity and disturbed the authorization to other openstack services. That means Keystone is responsible for all user management by performing CRUD (Create, Read, Update, and Delete). One of our aim is to isolate the identity information from the users that result of that providing identity services through our CSS, the CRUD management operations will be provide by the CSS (central security system) to the users through the Web Services and we will hide Keystone in the back-end modules. This satisfy any need of separating the identity providers because IDMS will be the Central Security System that plays the role of the identity provider, the user must register in IDMS first to so it can login openstack, then, the IDMS will send API message send to Keystone inform the system of new user information and his authority. The IDMS is built on System for Cross-domain Identity Management standard, which offers users registration. After the user been registered, and depending on the credentials as ID, all the information like, email transferred to (Authorization Server) the IDMS , which will depending on his record will give corresponding user Authorization SAML ticket to make sure that the user can access to the services and resources of OpenStack. All information must be saved in PDP server that make a real time synchronized between IDMS and PDP server, so both can send message to the Keystone to identify the user. The reply of the Keystone message stored in IDMS server database, that contain all necessary information like username, authorization, password, ,the unique ID (32- bit), tenant information combined with some optional parameters required in the process of user registration.

Authentication using PKI

We implement LCA server to be the authentication server of Keystone, the methodology is to make all connected component certificated based authentication. Local Certification Authority server and IDMS server must be synchronized to make successful user authentication based on certificates. LCA server is a part of the Pre-shared key hierarchy. User information will be in IDMS to generating key-pair, public key of the user is used to create the certificate request. And the privet key still hidden in the user system then the certificate and can be verify by the IDMS server.

SAML Token with SSO / Authentication

Keystone are used as interface for OpenStack provide integration of SAML protocol for SSO into OpenStack. That means after we connect openstack libraries and plug-ins with Keystone identity, we can use SSO/authentication based on SAML token.

We have to synchronized all component of openstack environment to run authentication/SSO based on a SAML token, IDMS server is registered all openstack services and components. And even all request made by user with identity verification. LCA server will generate X509 certification after successful identity verification user. And send it to the SAML server, then depending on the login information that determine the authority and identity information passed to the SAML server a new ticket will be send . SAML ticket consists of user information include (identification information, Tenant, the keystone establish a secure channel with any openstack services request to verifies SAML ticket and provide response

Authorization based on the XACML Policy

In our work the Authorization is depend on the XACML policy that worked on the idea of make all the Security system well-organized and efficient in a cooperative situation. That mean all the configuration and component for the keystone identity must be synchronize with Central Security System. We choose a single Policy Decision Point to have the authority of controlling the authorization processes. The responsibility of management of groups is made by PDP server, the roles and the XACML policies and the policy are defined by the security administrator, depending on authority and how access to the system. This give the process that when the user needs a services PEP will check the request and send XACML authorization request. The message contains Tenant information, and the permission. Depending on the security administrator policy, PDP server check the request. And the answer is sent back to the PEP server. PEP now accept or refuse the demand, depending on the answer of the PDP

Combination of Keystone and PKI

PKI system are depending on private and public key using the standard certification of X509. Keystone save the Public and Private Keys and certificates. After the system is connected and synchronized, any connected user will receive certificate from the Keystone by using REST web services. And the certificates can be hack in the public network (internet). But if openstack service connect to PKI mode, all related service gets public key from Keystone that can be used by the services to communicate .when the user use OTP to authenticate to openstack system, Keystone generated a token in a PKI mode. Keystone creates token JSON object which contains token's metadata, that is more secure by using Keystone’s private key to encrypt it and then signs it by MD5. When the user have Keystone public key it can decrypt and verify the data. Because the data information was inside encrypted token, on this project we don’t require for the user to Interfering Keystone message to have the token and therefore we provide both scalability and delegation of the authentication protocol. The one time password in Ubuntu Runtime authentication of user with Keystone in Figure 1.

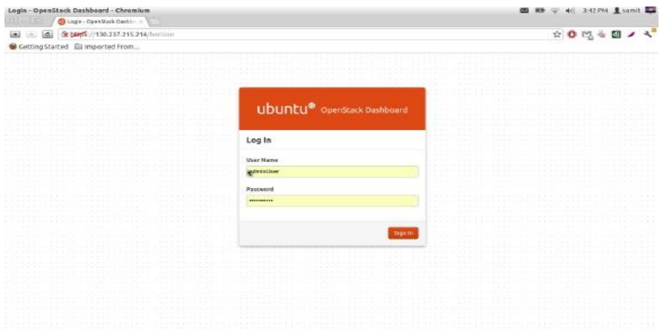
Login - OpenStack Dashboard - Chromium
ubuntu OpenStack Dashboard
Log In
User Name
Password
Sign In

Figure 1: (OTP) One-Time Password using HTTPS /Authentications

Figure 2: Token transferred are Encrypted and Signed using PKI in open stack

Conclusions

Based on the new design of our research that will shifting of all security service to application layer (the 7th layer) we provide a secure environments can be apply to the SSO (Single sign-on) and even the control management and identity will be certificate based authentication all the security service are known use web services within the central security system this give a new add value to usability, interoperability, deploy ability with scalability for open stack system ,further , due to the modularity of central security system architecture and new component can add or modified to the system so in future a new research can be add without the need to affected the hole system, and as the cloud system still new there is a lot of new consideration issue need to be checked like the read/write permission and SSL connection .

Our work is to improve the design a secure architecture for the openstack cloud computing that reduce the To evaluate our system that has been proposal to improve the security of cloud system and use openstack environment we must indicate how we can protect the system from outside threats.

Authentication/SSO Evaluation: as we use a public network to exchange the SAML ticket between the end user in the internet and application services for single sign-on , we need a mechanism to protect the date from multi-type of threat like man in middle ,unknown replay , Dos attack, so need the application to make a replay depending on the PDP server decision ,this can be done by pre-established of the a trust relation with the user before transfer the data by using PKI model with user digital signing with privet and public need to be predefined so both side can verify the data by the signature of the other side so we provide the authentication and integrity for SMAL ticket and the exchange of ticket become over a secure communication channel using SSL the solve all the threat of hacker .with keystone services .

Authorization Evaluation: after we protect the authentication process we need to protect the authorization message to prevent illegal permission we protect the message of request-response of PDP service that communicate with other application services through internet theses message are subject to threat like man in middle , modification ,and Dos attack .the implementation of secure SSL channel that we proposal give us a way to stop these attack. And the authenticate of PDP and PEP before and exchange of XAXML message so they transfer on secure channel .a randomly session ID is produce to prevent the replay attack. all the openstack component are now connect through PKI model to ensure the that the authentication between XACML message and between PDP and PEP server are secure and trust relationship are trust

References

Halpert, B. (2011). *Auditing cloud computing: A security and privacy guide* (Vol. 21). New Jersy, NJ: John Wiley & Sons.

Rittinghouse, J. W., & Ransome, J. F. (2016). *Cloud computing: implementation, management, and security*. Florida, US: CRC press.

Hamlen, K., Kantarcioglu, M., Khan, L., & Thuraisingham, B. (2012). Security issues for cloud computing. *Optimizing Information Security and Advancing Privacy Assurance: New Technologies*, *150*.

Yang, K., & Jia, X. (2014). *Security for cloud storage systems*. Berlin, Germany: Springer-Verlag.

Krutz, R. L., & Vines, R. D. (2010). *Cloud security: A comprehensive guide to secure cloud computing*. New Jersy, NJ: Wiley Publishing.

Rjaibi, W., & Wilding, M. (2011). *Best practices data protection in the cloud*.

ISC. (n.d.). *Underscoring cloud security issues*. Retrieved from: http://www.virtualization.co.kr/reference/Underscoring-Cloud-Security-Issues.pdf

Lim, I., Coolidge, E. C., & Hourani, P. (2013). *Securing Cloud and Mobility: A practitioner's guide*. Florida, US: CRC Press.

Newcombe, L. (2012). *Securing Cloud services: A pragmatic approach to security architecture in the Cloud*. Ely, Cambridgeshire , UK: IT Governance Publishing.

Sarkar, P. (2013). *VMware VCloud Security*. Birmingham, UK: Packt Publishing Ltd.

Mitchell, I., & Alcock, J. (2011). *Cloud Security: The definitive guide to managing risk in the new ICT*. Tokyo, Japan: Fujitsu Services Ltd

Cooper, J. (2013). *Analysis of security in cloud platforms using openstack as case study* (Master thesis). University of Agder, Kristiansand, Norway.