OpenStack Keystone 认证机制研究

尹誉衡
(华北计算技术研究所,北京 100089)

摘 要:随着云计算技术在政治、经济、科研等领域的广泛应用,对云计算平台的安全要求也越来越高。OpenStack 是一个大型开源云计算平台,被广泛应用于私有云和公有云平台的搭建中,其身份认证机制由 Keystone 组件基于用户名和密码提供,且在传输过程中以明文传输,容易受到中间人攻击,导致信息被窃取,无法适用于安全要求较高的场景。为了提高 Keystone 认证机制的安全性,对 Keystone 的两种认证机制进行了详细分析,指明其中存在的明文传输、易遭受重放攻击等安全问题,针对这些安全问题,提出一种对 Keystone 认证机制进行改进的方案。该方案结合非对称加密的方式对传输数据进行加密,加入了时间戳验证机制,有效地降低了数据在传输过程中被窃听、被篡改的风险。通过使用 Wireshark 抓包设置对比实验,证明了该方案的有效性。经分析,该方案降低了 Keystone 明文传输数据的风险,增强了 Keystone 传输数据的安全性。

关键词:云计算;Keystone;安全问题;认证机制;加密传输
中图分类号:TP309 **文献标识码**:A **文章编号**:1673-629X(2021)02-0122-05
doi:10.3969/j.issn.1673-629X.2021.02.023

Research on OpenStack Keystone Authentication Mechanism

YIN Yu-heng
(North China Institute of Computing Technology, Beijing 100089, China)

Abstract: With the wide application of cloud computing technology in politics, economy, scientific research and other fields, the security requirements of cloud computing platform are also higher and higher. OpenStack is a large-scale open-source cloud computing platform, which is widely used in the construction of private cloud and public cloud platforms. Its identity authentication mechanism is provided by keystone component based on user name and password, and it is transmitted in clear text during the transmission process, which is vulnerable to man in the middle attack, resulting in information theft, so it cannot be applied to scenarios with high security requirements. In order to improve the security of keystone authentication mechanism, two authentication mechanisms of keystone are analyzed in detail, and the security problems such as plaintext transmission and replay attack are pointed out. In this scheme, the transmission data is encrypted by asymmetric encryption, and the time stamp verification mechanism is added, which effectively reduces the risk of data eavesdropping and tampering in the transmission process. The experiment shows that the proposed scheme is effective. After analysis, the scheme reduces the risk of keystone plaintext data transmission and enhances the security of keystone data transmission.

Key words: cloud computing; Keystone; security; authentication mechanism; encrypted transmission

0 引 言

随着互联网和计算机技术的高速发展,为了满足人们对数据处理速度和数据存储容量越来越高的要求,云计算平台应运而生。云计算平台通过整合分布式资源,构建灵活、可扩展的虚拟计算环境[1],根据用户的需求提供理想的服务与资源。目前金融、政务、邮政等重要领域已经广泛应用云计算平台,大量云计算平台的用户选择将数据甚至是隐私数据保存在云端[2],因此用户迫切希望有一个安全可靠的云计算平台。

非对称加密算法是一种将数据进行加密处理的算法。数据收发双方都有一套相互匹配的公钥(public key)和私钥(private key)[3]。数据发送方和数据接收方在正式传送数据前会交换公钥,传送数据前发送方先用接收方的公钥对数据进行加密处理,接收方接收到数据后,用自己的私钥对数据进行解密,从而获取原本的数据。

OpenStack 是一个开源的、便于使用的、可扩展的

收稿日期:2020-05-07 **修回日期**:2020-09-08
基金项目:国家重点研发计划“公共安全风险防控与应急技术装备”重点专项(司法专题任务)(2018YFC0831200)
作者简介:尹誉衡(1997-),女,硕士研究生,研究方向为云计算安全、网络安全。

云计算平台,由多个功能各不相同的组件构成,各组件之间分工协作,共同构成云计算平台[4]。目前为止发布了从 A 到 T 共 20 个版本,从 Essex 版本之后开始全面支持使用 Keystone 完成身份管理、访问控制和统一授权的功能[5]。

1　OpenStack Keystone 介绍

OpenStack 由一系列开源项目组成,是一种提供基础设施即服务(infrastructure as a service,IaaS)的虚拟化管理平台,能够提供可靠的云部署方案[6]。OpenStack 主要使用池化虚拟资源来构建和管理私有云和公有云,通过多种功能不同的组件提供计算、网络、存储、认证和镜像等服务,其中提供认证服务的就是 Keystone 组件。各组件之间采用 Restfull API 接口规范,实现了模块之间的低耦合,各个组件可以灵活配置,易于二次开发[7]。

1.1　Keystone 基本介绍

用户在申请 OpenStack 的资源和服务时都由 Keystone 对用户进行身份验证并对用户授权,同时还会向用户提供一个该用户可以使用的服务的列表,用以明确用户使用资源的权限范围[8]。在 OpenStack 中,无论是管理员还是用户,要想使用各项服务,都必须先通过 Keystone 的认证。并且,OpenStack 的认证工作都交由 Keystone 完成,因此,提高 Keystone 的安全性是十分必要的。

下文会介绍 Keystone 中涉及的一些基本概念。

(1)用户(User):使用 OpenStack 服务的个人或是系统,需要向 Keystone 发送凭证以通过 Keystone 的验证。Keystone 会向通过验证的用户分配一个该用户特有的令牌(Token),该令牌就是在 OpenStack 中请求其他服务时的凭证。

(2)令牌(Token):Keystone 验证用户凭证后,向用户分发特定的、在 OpenStack 各组件之间使用的通行证,用户可以以此令牌访问被授权的其他的 OpenStack 服务。令牌具有有效时间,可以随时被云平台管理员取消。Keystone 的作用就是对外提供一个可以访问资源的令牌。

(3)项目\租户(Tenant\Project):表示一组资源,资源范围由 Keystone 授权给用户的资源列表界定。

(4)凭证(Credential):用于证明用户身份或权限,也就是用户的用户名和密码,用户将其发送给 Keystone,以换取令牌。一个用户可以有多个凭证,一个凭证与一个项目关联。

Keystone 通过这些基本概念,对外提供认证、令牌、目录和安全策略四个方面的核心服务。Keystone 组件工作流程如图 1 所示。

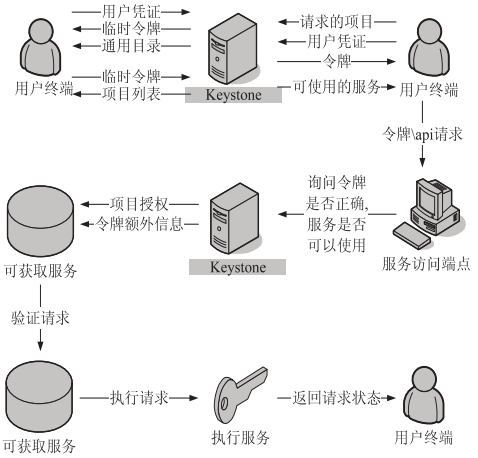

图 1　Keystone 工作流程

(1)用户发送自己的凭证(即用户名和密码)到 Keystone,Keystone 通过认证后,返回一个临时令牌和通用服务目录。

(2)用户通过临时令牌向 Keystone 查询当前环境下的项目列表。Keystone 通过验证后返回给用户一个项目列表。

(3)用户选择一个项目后,发送凭证给 Keystone 申请正式令牌。

(4)用户凭借正式令牌发送 API 请求到相应的服务端点,服务端点将令牌交由 Keystone 验证后响应请求,向用户返回请求状态。

以上是 Keystone 完整的工作流程,该文就认证过程的安全性进行研究,并给出加强安全性的方法。

1.2　Keystone 认证机制

OpenStack 身份认证分为两个部分,首先云平台用户需要先从 Keystone 申请到令牌,以令牌作为与各组件的 API 接口进行交互的凭证,使用该令牌完成单点登录和委派验证[9],从而获取其他组件提供的服务。Keystone 组件作为 OpenStack 云平台的身份认证核心[10],可以与其他后端授权系统进行集成,其身份认证机制通过 token 来实现,主要包括了 UUID token(universally unique identifier token)和 PKI(public key infrastructure token)两种认证机制。

当用户需要进行操作时,用户提供有效的用户名和口令给 Keystone,Keystone 经过认证后返回给用户一个令牌[11]。之后用户对其他组件进行其他操作时,先出示这个令牌给相应的 API,组件收到请求后,用这个令牌去向 Keystone 进行请求验证。Keystone 通过比对令牌,以及检查令牌的有效期,判断令牌的有效性,最后返回结果给相应的组件。

OpenStack 在 G 版本之前只有 UUID 令牌认证机制这一种方式。G 版本及之后版本使用 PKI 令牌机制[12],Keystone 会利用 PKI 对令牌相关的数据进行签

名,对 UUID 机制进行了改进,减少了交互开销。

1.2.1 UUID 令牌认证机制

UUID 令牌认证机制基于用户提供的用户名和口令完成认证[13]。Keystone 生成的令牌保存在后端数据库中,每个令牌都有一个独一无二的 ID。用户向 Keystone 提交验证申请,Keystone 经过验证后将令牌 ID 传给用户。用户在申请云平台资源时,会在 API 请求中附加上自己的令牌 ID,Keystone 通过比对 API 请求和后端数据库中的令牌 ID 来验证用户身份,之后再响应用户的 API 请求。这种方式的设计决定了用户每次发起 API 请求之前都要向 Keystone 发起验证请求,造成极大的开销,对网络和 Keystone 资源都是一种极大的消耗。Keystone 生成令牌和用户使用令牌发起 API 请求的过程如图 2 所示。

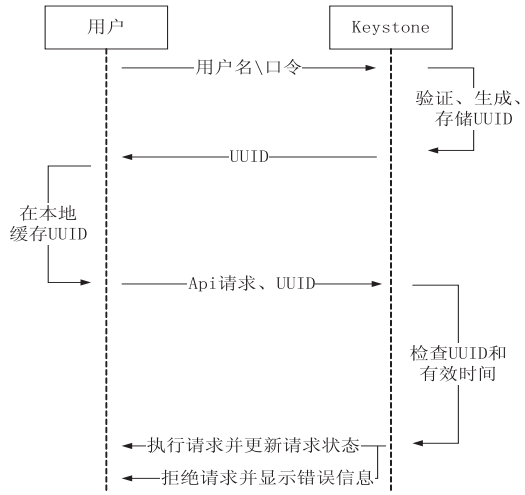

图 2 UUID 认证过程

1.2.2 PKI 令牌认证机制

PKI 令牌认证机制中 Keystone 被设计为一个数字签名认证中心。Keystone 使用签名密钥和数字证书对用户的令牌签名,每一个 API 端点都会保存一份签名公钥证书、CA 公钥证书和证书吊销列表的 Keystone 拷贝,用以验证用户的请求[13]。在这种机制下,当用户发起 API 请求时,每个 API 端点都可以使用这份拷贝离线响应请求,节省了为每个验证直接请求 Keystone 的开销,解决了 UUID 机制中客户端与 Keystone 频繁交互造成的性能瓶颈,缓解了对网络带宽和 Keystone 资源的压力。PKI 令牌认证机制的流程如图 3 所示。

2 Keystone 认证机制安全性分析

通过对 Keystone 认证机制的分析,可以发现 Keystone 在认证过程中存在一些缺陷,这些缺陷可能会使 Keystone 和其他组件遭受安全问题。本节针对 Keystone 认证机制的安全性进行分析,指出其中存在的问题。

(1)Keystone 是 OpenStack 身份认证中心,使用用户名和口令来验证用户,但在验证过程中,用户名和口令都是以明文形式传输的。Keystone 默认使用 HTTP 协议进行通信,HTTP 不会加密通信内容,攻击者可以针对 HTTP 协议发起诸如中间人攻击等的攻击行为,导致泄露用户的用户名和口令等信息。

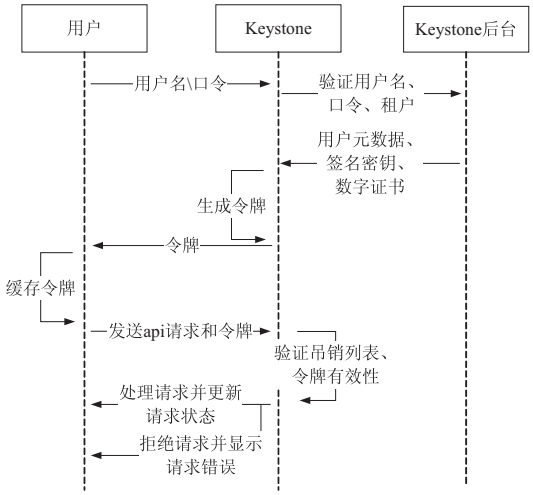

图 3 PKI 认证过程

(2)Keystone 会向经过认证的用户发送一个令牌,用户凭此令牌进行后续的资源访问和使用,其他的组件不会再对用户的身份加以验证。这种做法意味着攻击者一旦通过某些途径拿到某个用户的令牌,就相当于拥有了此用户在云平台的所有权利,攻击者可以不再经过 Keystone 的身份验证而直接以该用户的身份使用云平台的资源,甚至对云平台本身发动攻击。

(3)用户在请求其他组件的服务时,发送的 API 请求包括了令牌和请求信息。由于 API 请求在传输过程中是以明文传输的,所以攻击者一旦截获了用户的任意一条 API 请求,都可以通过该请求中的信息对 API 接口发动重放攻击。

(4)Keystone 作为 OpenStack 平台唯一一个强制服务,同时还具备管理认证的功能,这使得它成为攻击者的首选攻击目标。在 Keystone 的基础上结合其他安全手段无疑是提高其安全性的一个重要方式。

综上所述,Keystone 组件的认证机制中仍然存在很多安全问题[14],尤其是通信过程中的明文传输,这是一个很大的隐患,使得 OpenStack 无法应用于对数据保密有极高要求的场景,因此需要将 Keystone 结合其他安全手段来保障 Keystone 数据传输的隐蔽性,而非对称加密作为一种经过验证的可靠方式,无疑是最佳的选择。

3 Keystone 改进方案及分析

3.1 Keystone 改进方案

针对 Keystone 现有的安全问题,该文介绍一种基

于非对称加密的认证机制改进方案,该方法可以有效地解决上文中指出的问题。

非对称加密算法使用公钥和私钥对数据进行加密和解密。公钥作为对信息进行加密的密钥,对数据发送方公布,同时数据接收方应妥善保管私钥,任何人都无法通过公钥推测出私钥。被数据接收方的公钥加密过的数据只能通过接收方的私钥解密,其他人无法解密获取其中的信息,由此可见,使用非对称加密算法可以有效地防止信息泄露,将其与 Keystone 认证机制相结合,可以大大提高 Keystone 认证机制的安全性。

表 1 给出了改进方案相关的符号记法。

表 1　改进方案符号记法

符号	描述
C	用户
K	Keystone
K_PUB	用户或 Keystone 持有的公钥。K_PUB_C 表示用户持有的公钥,K_PUB_K 表示 Keystone 持有的公钥
K_PRI	用户或 Keystone 持有的私钥。K_PRI_C 表示用户持有的私钥,K_PRI_K 表示 Keystone 持有的公钥
$E_K(M)$	用 k 加密消息 M 后生成的密文
A->B:M	A 向 B 发送消息 M

在设计改进方案时,考虑到需要交换公钥,并且让用户验证 Keystone 的身份,所以在认证过程的前三个阶段仍然使用明文传输。在交换了公钥并验证了 Keystone 的身份之后再使用非对称加密算法对传输的敏感信息进行加密处理,完成认证工作,增强认证机制的安全性。

图 4 给出改进方案的工作流程。

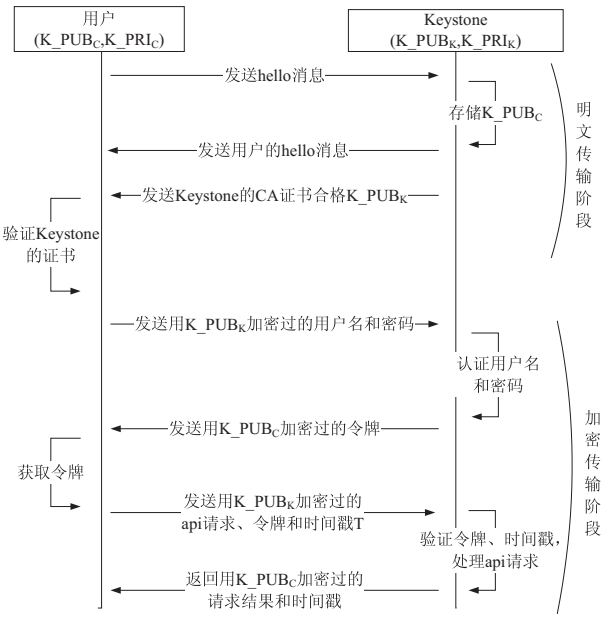

图 4　改进方案工作流程

(1)用户向 Keystone 发送 hello 消息:用户发送包含了用户公钥 K_PUB_C 的 hello 消息给 Keystone。

(2)Keystone 向用户发送 hello 消息:Keystone 收到用户发送的 hello 消息后,将用户公钥和用户随机数保存在本地,然后将用户的 hello 消息再发还给用户,作为收到 hello 消息的确认。

(3)Keystone 发送证书:Keystone 将自己的 CA 证书和公钥 K_PUB_K 发给用户。

(4)用户发送身份信息:用户验证证书通过后,将自己的用户名和密码用 Keystone 的公钥 K_PUB_K 加密,发送给 Keystone。

(5)发放令牌:Keystone 用私钥 K_PRIK 解密消息,获取用户名和密码并验证后,发送用户公钥 K_PUB_C 加密过的令牌给用户。

(6)发送 API 请求和令牌:用户基于随机数生成会话密钥,将 API 请求、令牌和时间戳 T 用 Keystone 的公钥 K_PUB_K 加密,并发送给 Keystone。

(7)API 请求处理:Keystone 用私钥 K_PRIK 解密后,验证令牌和 T,验证通过后,将 API 请求的结果和时间戳 T 用 K_PUB_C 加密后发送给用户。

3.2　实验分析

实验环境:物理机 win 7,使用 VMware Workstations 安装两台 CentOS 7.7 虚拟机,分别作为 Controller 节点和 Compute 节点,并在 Controller 节点和 Compute 节点上安装 OpenStack Stein。

在物理机上使用 wireshark 抓包,追踪 http 流,未加密流量中可以直接获取用户名和密码,而在对数据流进行加密之后,无法解密数据流,只能获取到乱码,从而有效地保障了通信过程的机密性。

3.3　Keystone 改进方案分析

上文所述的改进方案通过在 Keystone 认证过程中增加非对称加密机制,对认证过程中所涉及的用户名、密码和令牌等重要信息进行加密传输,对这些重要信息的传输进行了保护。该方案实现了一个安全有效的改进方案,解决了第 2 节提出的安全问题。以下从三个方向对该改进方案进行了详细具体的安全分析。

3.3.1　防消息泄漏

从改进方案的工作流程可以看出,只有在前三个阶段没有用公钥加密,而前三个阶段包含的有效信息只有用户公钥、Keystone 公钥和 Keystone 的 CA 证书,没有用户名和密码,在后续向 Keystone 发送认证请求的过程中无法通过 Keystone 的验证,从而就无法获取令牌和时间戳,因此即使攻击者截获了前三个阶段交互的信息,也不会泄露用户名和密码、令牌、API 请求等信息。

通过对认证过程中的敏感信息使用安全性较高的非对称加密算法进行加密,有效地防止了敏感信息的

泄露,攻击者即使截获了信息也无法破解出有效信息。

3.3.2 防重放攻击

重放攻击是指攻击者向目标主机发送一个合法的包,但是这个包是该主机事先已经接收过的包,利用他人的合法身份来欺骗目标主机[15]。这种攻击方式主要用于身份认证过程,从而造成目标主机对攻击者错误的身份认证。

发动重放攻击的可以是普通的合法用户,也可以是将合法用户的请求拦截了的攻击者。攻击者可以通过窃听合法用户的网络流量或其他途径获取到认证过程中的凭证,然后再将这个凭证发给认证服务器,从而获取经过认证的合法身份。

若将重放攻击运用在欺骗 Keystone,攻击者就可以在没有破解出传输数据的真实含义时,也能够用加密过的认证信息欺骗 Keystone,从而获取到令牌。

在改进方案中,在用户发送 API 请求时,同时会发送一个时间戳。由于攻击者无法破解传输的加密,所以无法定点修改时间戳,因此信息中的时间戳必定是合法用户发送的。所以当 Keystone 收到两个相同的消息时,Keystone 可以通过对比信息中包含的时间戳和接收到信息的时间来确定哪一个 API 请求是由合法用户发出的,从而防止有效的重放攻击。

3.3.3 双向认证

在 Keystone 过程中,不仅是 Keystone 需要确定用户的身份,用户同样也需要确定 Keystone 的身份,防止攻击者通过伪装成 Keystone 来欺骗用户。

在该文的改进方案中,用户会在通信过程前期收到 Keystone 的 CA 证书,从而确认 Keystone 的身份,为下一步的认证过程提供保障。CA 证书由第三方可信机构签署颁发,具有极高的可信度,用户通过 CA 证书验证身份后可确定对方身份。同时因为该证书是公开证书,所以可以在认证过程的明文传输阶段就对 Keystone 进行验证。

而 Keystone 验证用户的身份则是通过用户名和密码,在信息不被泄露的前提下,使用用户特有的用户名和密码来验证用户身份不失为一个传统但有效的方法。

在认证机制中加入这样的双向认证可以保障用户和 Keystone 双方都不被欺骗,极大地增强了认证机制的安全性。

4 结束语

通过分析 Keystone 的工作流程以及 UUID token 和 PKI token 两种认证机制,指出其中存在的缺陷和可能受到的攻击,再针对这些缺陷,给出一种改进方案,将非对称加密机制与 Keystone 的认证机制结合,并加入双向认证和时间戳证明,保证通信过程的数据安全。然后对这种改进方案进行安全分析,可以看出该方案对信息泄露和重放攻击有很好的防御,同时增加的双向认证也加强了通信双方的互信度,不失为一个一举多得的方案。下一步工作是对该方案的性能进行评估,并通过适当的策略降低认证服务器负载量,提高整个系统的工作效率。

参考文献:

[1] 王斌锋,苏金树,陈 琳. 云计算数据中心网络设计综述[J]. 计算机研究与发展,2016,53(9):2085-2106.

[2] 张玉清,王晓菲,刘雪峰,等. 云计算环境安全综述[J]. 软件学报,2016,27(6):1328-1348.

[3] 卓先德,赵 菲,曾德明. 非对称加密技术研究[J]. 四川理工学院学报:自然科学版,2010,23(5):562-564.

[4] SEFRAOUI O,AISSAOUI M,ELEULDJ M. Openstack:toward an open-source solution for cloud computing[J]. International Journal of Computer Applications,2012,55(3):38-42.

[5] KHAN R H,YLITALO J,AHMED A S. OpenID authentication as a service in OpenStack[C]//7th international conference on information assurance and security. Melacca,Malaysia:IEEE,2011.

[6] WEN X,GU G,LI Q,et al. Comparison of open-source cloud management platforms:OpenStack and OpenNebula[C]//2012 9th international conference on fuzzy systems and knowledge discovery. Sichuan:IEEE,2012:2457-2461.

[7] 朱智强,林韧昊,胡翠云. 基于数字证书的 openstack 身份认证协议[J]. 通信学报,2019,40(2):188-196.

[8] 英特尔开源技术中心. OpenStack 设计与实现[M]. 北京:电子工业出版社,2017.

[9] YANG X,ZHAO L L. Research on OpenStack authentication security[J]. Audio Engineering,2019,43(2):39-41.

[10] MARTINELLI S,NASH H,TOPOL B. Identity,authentication,and access management in openstack:implementing and deploying Keystone[M]. [s. l.]:O'Reilly Media,2015.

[11] 熊 微,房秉毅,张云勇,等. OpenStack 认证安全问题研究[J]. 邮电设计技术,2014(7):21-25.

[12] NASH A,DUANE W,JOSEPH C. PKI:implementing and managing e-security[M]. [s. l.]:McGraw-Hill, Inc. ,2001.

[13] 席 涛. OpenStack 云计算平台安全缺陷分析技术研究[D]. 北京:北京邮电大学,2016.

[14] 杨 幸,赵丽莉. OpenStack 认证安全问题研究[J]. 电声技术,2019,43(2):39-41.

[15] SINGH M,PATI D. Usefulness of linear prediction residual for replay attack detection[J]. International Journal of Electronics and Communications,2019,110:152837.