PROCEEDINGS OF SCIENCE

Unified Account Management for High Performance Computing as a Service with Microservice Architecture

Rongqiang Cao[1]
Computer Network Information Center, Chinese Academy of Sciences
P.O. Box 349, Bejing 100190, China
E-mail: caor@sccas.cn

Shasha Lu
Computer Network Information Center, Chinese Academy of Sciences
P.O. Box 349, Bejing 100190, China
E-mail: lusha721@sccas.cn

Xiaoning Wang
Computer Network Information Center, Chinese Academy of Sciences
P.O. Box 349, Bejing 100190, China
E-mail: wxn@sccas.cn

Haili Xiao
Computer Network Information Center, Chinese Academy of Sciences
P.O. Box 349, Bejing 100190, China
E-mail: haili@sccas.cn

Xuebin Chi
Computer Network Information Center, Chinese Academy of Sciences
P.O. Box 349, Bejing 100190, China
E-mail: chi@sccas.cn

ABSTRACT

In recent years, High Performance Computing (HPC) has developed rapidly in China. As worked in the operation and management center of China National Grid (CNGrid) and Scientific Computing Grid (ScGrid) for many years, we notice that users prefer to manage their jobs at different supercomputers and clusters via a global account

[1]Speaker

PoS(ISGC 2018 & FCDD)020

on different remote clients such as science gateways, desktop applications and even scripts.

Therefore, we described Unified Account Management as a Service (UAMS) to access and use all HPC resources via a global account for each user in this paper. UAMS focuses on authenticaton services, and authorization services are provided by other components in middle. We addressed and solved challenges for mapping a global account to many local accounts, and provided unified account registration, management and authentication for different collaborative web gateways, command toolkits and other desktop applications. UAMS was designed in accordance with the core rules of simplicity, compatibility and reusability. In architecture design, we focused on loosely-coupled style to acquire good scalability and update internal modules transparently. In implementation, we applied widely accepted knowledge for the definitions of the RESTful API and divided them into several isolated microservices according to their usages and scenarios. For security, all sensitive data transferred in wide-network is protected by HTTPS with transport layer security outside of CNGrid and secure communication channels provided by OpenSSH inside of CNGrid. In addition, all parameters submitted to RESTful web services are strictly checked in format and variable type. For efficiency, each supercomputers provides lots accounts to UAMS at a time, and the administrators of UAMS could map different local accounts immidiately when creating global accounts.

By providing these frequently important but always challenging capabilities as a service, UAMS allows users to use tens of HPC resources and clients via only an account, and makes it easy for developers to implement clients and services related HPC with advantages of numerous users and single sign-on capability. Based on UAMS, representative clients are introduced and reviewed combined with different authentication schemes. Finally, analysis and test of UAMS shows that it can support authentication in milliseconds level and has good scalability. In future, we plan to implement federated account service that enable a local HPC account similar to a global account to login the national HPC environment, access and use all HPC resources in CNGrid.

International Symposium on Grids and Clouds (ISGC) 2018 in conjunction with Frontiers in Computational Drug Discovery
16-23 March 2018
Academia Sinica, Taipei, Taiwan

1.Introduction

In recent years, supercomputing and HPC is a key part of Chinese government's plan to shift to an innovation driven society. There are several petaflops level supercomputers built and operated at different National Supercomputer centers, such as the Sunway TaihuLight [1] and the Tianhe 2 [2]. CNGrid begun in 15 years ago and has integrated tens of HPC resources from several national supercomputing centers and other large centers distributed geographically [3]. At present, there is more than 130 petaflops computing capacity and 60 petabytes storage capacity in CNGrid. And it provides efficient and easy-to-use HPC services for user in sciences, engineering, and other areas. There are more than 800 thousands of jobs submitted through different clients built on top of CNGrid via global accounts from 2010.

As worked in the operation and management center of CNGrid for many years, we noticed that users are enjoyed to manage their jobs at different supercomputers and clusters via a global account on different remote clients such as science gateways, desktop applications and even scripts. On the one hand, they don' t like to apply for an account to each supercomputer and login into the supercomputer in specific way. On the other hand, some users already have one or more accounts belong to different supercomputers when they recognize the needs for global accounts. Then we need some ways to map global acconts to sharing accounts in each supercomputer or private accounts assigned by users. In this paper, we designed and implemented Unified Account Management as a Service (UAMS) to solve issues between global accounts of CNGrid and local accounts of heterogeneous HPC resources. UAMS provides unified account service, allows users to use tens of HPC resources and clients via only an account, and make it easy for developers to implement clients and services related HPC with advantages of numerous users and single sign-on capability. UAMS focuses on authenticaton services, and authorization services are provided by other components in middleware. There are several representative clients built and deployed to provide computing services for different disciplines and research areas.

The structure of this paper is as follows: In section 2, we present related work about account management, RESTful web services and microservice architecture in HPC. In section 3, we introduce CNGrid that provides the national HPC environment. In section 4, we discuss the architecture of UAMS and describe several microservices. In section 5, we describe RESTful web services and API. In section 6, we discuss three authentication schemes and review several representative clients authenticated by UMAS. In section 7, we discuss performance tests to verify the performance and efficiency. Finally, we summarize this paper.

2.RELATED WORKS

Supercomputing and HPC are expensive and inadequate resources so that accounts and authentication are necessary and core functions to protect these resources for using by only valid users. Password authentication scheme is the most widely used and acceptable mechanism because of its easy-operation, scalability, compatibility and low-cost advantages. However it may be prone to attacks by hackers and disclosure of sensitive information. MyProxy is a credential management system based on x.509 public key certificates scheme [4-5]. MyProxy

and its improvements are used as a primary credentialing mechanism for users in the Extreme Science and Engineering Discovery Environment (XSEDE) [6]. MyProxy is much safer than the password scheme, but it requires the ecosystem to support X.509 credentials.

In order to enhance security, multiply levels of authentication are discussed to restrict intruders from hacking accounts due to complex authentication process [7]. Considering multiply factors authentication, a novel smart card based authentication scheme has been proposed in paper [8] and the proposed authentication scheme utilizes the biometric data embedded in a smart card along with the identity and password of the user. Another smart-card-based password authentication scheme is proposed in paper [9] to resolve the various issues arising from user corruption and server compromise, and it is formally proved secure under the harshest adversary model so far. Regarding supporting heterogeneous HPC resources and associated secure shell softwares such as OpenSSH in CNGrid, we select password authentication scheme for global users, and public key certificate scheme for CNGrid accessing HPC resources on behalf of users.

Grid computing and cloud computing provide massive HPC resources. The authentication is a basic component in the middleware and toolkits for validating the credentials. The Grid Security Infrastructure (GSI) is the basic security for the Globus Toolkit (GT 4), and involves third party verification for authorization [10]. The authentication schemes of GSI are based on the user name and the password and certificates which are generated by a secure Certificate Authority (CA). Globus Nexus is a flexible and powerful Platforms-as-a-Service to which developers can outsource identity, group, and profile management needs [11]. GSI and Nexus provide rich and flexible capabilities of account management and authentication for the Globus ecosystem. In CNGrid, we also need similar but simple services to act as an account provider and an authentication center for computing users, developers and owners of HPC resources.

In addition, many famous and large Internet companies provide account and authentication services, for example, Alipay [12], WeChat [13], and Google Accounts. These commercial applications are designed for the public and have massive accounts. However CNGrid is oriented to specific people or teams who need to deal with their works with HPC or provide HPC services for other people. China Science and Technology Network (CSTNET) passport provides similar third authentication services on top of email address and password that anyone could use to sign in to supported services [14]. CSTNET mainly covers users of CNGrid from Chinese Academy of Sciences (CAS) in difference disciplines and research areas, but doesn't contain many users mainly from many universities and institutes not belonged to CAS, and even creative companies. As discussed above, we designed UAMS for the CNGrid ecosystem that provides basic security services for the HPC environment and unified authentication services for users and developers.

At present, HPC is increasingly moving to the web, and even to mobile applications. These web gateways and applications need RESTful web API to acquire account management and other services easily and quickly in diverse programming languages and platforms. The XSEDE system provides its own REST API consisting of identity and group management API based on Globus nexus [15]. National Energy Research Scientific Computing Center (NERSC) proposed a customizable NEWT framework that enables to access to various backend resources and services through a common web API, and support to plug into whatever authentication

infrastructure already exists at a given system [16]. However, NEWT framework needs to custom implementation work for any target HPC center. CSTNET passport also provides a group of REST API for developers to integrate authentication into their clients [14]. For supporting different kinds of clients in diverse disciplines and research areas, UAMS is to be implemented in RESTful web services style to fully utilize the cross-language and cross-platform features of RESTful architecture.

Microservices are an architectural style largely based on decoupled autonomous services that can be developed, deployed and operated independently of each other. XSEDE portal API has also grown into a mature set of production quality microservices that are used both internally in the XSEDE project and externally with other clients [17]. A distributed computing model and its integrated middleware for HPC is presented based on a cooperative microservices grid computing flexible for different parallel and distributed architectures constituted over a distributed system [18]. In order to maintain agile, sustainable, and capable of handling feature, UAMS employs modular-microservice architecture that enables RESTful web services as a system to evolve incrementally.

3.NATIONAL HPC ENVIRONMENT

CNGrid [3] is a key project launched in May 2002 and supported by the China National High-Tech Research and Development Program (863 program). The goal of CNGrid is to build national HPC environment via integrating supercomputers, clusters, applications and so on into a virtual supercomputer. The mission of CNGrid is to continually promote the construction of national information capabilities and speed up the development of relevant industries by technical innovation.

During more than 15 years, CNGrid has integrated tens of HPC resources distributed geographically across China, comprising 6 National Supercomputer Centers of Tianjin, Jinan, Changsha, and Shenzhen, Guangzhou, Wuxi, and also dozens of teraflops-scale HPC resources belong to universities and institutes. At present, there are 17 supercomputers and tens of clusters connected by SCE middleware in CNGrid. SCE is a lightweight grid middleware mainly based on the OPENSSH software package, and provides grid security, job management, data management, and information service [19]. In order to make full use of the existed HPC resources and provide better computing services for users in different research areas and disciplines, several collaborative communities are built based on SCEAPI [20] in industry, medicine, and multi-media and other research areas. SCEAPI provides simple account authentication capability by login and logout functions in RESTful style. UAMS also provides a group of RESTful API for developers and part of API comes from SCEAPI.

4.ARCHITECTURES AND MICROSERVICES

UAMS is designed in accordance with the core rules of simplicity, compatibility and reusability. In architecture design, we focused on loosely-coupled style to acquire good scalability and update internal modules transparently. In implementation, we applied widely accepted knowledge for the definitions of the RESTful API and divided them into several isolated microservices according to their usages and scenarios. For security, all sensitive data transferred in wide-network is protected by HTTPS with transport layer security outside of

CNGrid and secure communication channels provided by OpenSSH inside of CNGrid. In addition, all parameters submitted to RESTful web services are strictly checked in format and variable type.

UAMS is a group of RESTful web services and implemented in several self-contained microservices based on some common components, for example, the SCE middleware, logging, memory cache, and persistence storage. In each microservice, UAMS adapted a loosely-coupled hierarchical architecture to ensure good scalability and deal with frequent changes in demands, protocols and security limits. Each RESTful API has a clear definition that specifies a general message structure, input parameters and a response structure. When updating a new version, we keep all existing definitions unchanged. If we should break the definition, a new definition will be added to keep compatibility so that existing clients do not need to be updated.

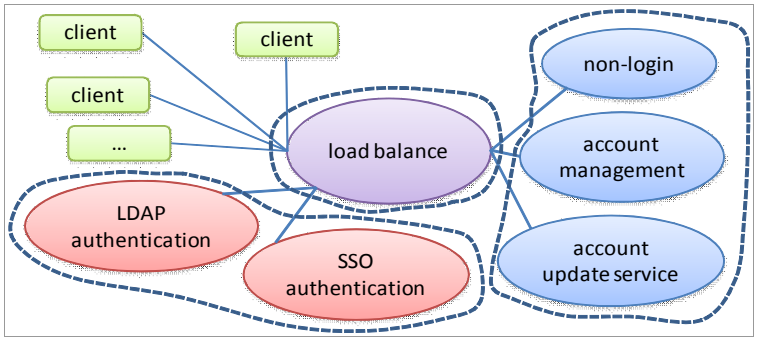

Figure 1: the microservice architecture of UAMS and clients

As shown in Figure 1, ovals delegate several microservices and rectangles stand for different remote applications that adopt any authentication scheme provided by UAMS. The pink oval represents a load balance microservice that receives all HTTP requests from multiple clients and forwards them to the other microservices. If any microservice is in heavy loaded status, one or more instances could be deployed quickly and provide service for incoming requests. If any microservice has more than one instance which is in idle or light loaded status, some instances could be shutdown to save resources in servers without affecting quality of services for clients. Considering that each request and response of account-related contain only a small number bytes of data, for example, hundreds of bytes or tens of kilobyte, only one instance of load balance microservice service is enough to forward requests related to account services and provides high quality services.

Next, we will discuss the remains microservices in figure 1 represented by 3 blue ovals that provide registration, management and update for users and administrators, and by 2 red ovals that provide several authentication schemes for different kinds of clients.

4.1.Non-login Microservice

The non-login microservice is a normal web application designed for potential users to apply for an account to access and use national HPC environment provided by CNGrid. Besides, it also provides web pages for setting and resetting passwords. As shown in Figure 1, registration process and password setting service are discussed as followed from a user-centered perspective.

On the registration web page, a user should fill in account name, at least an email address and a telephone number. The backend service would check whether the account name is

occupied or not, and make sure the email address is exist and available. In addition, other contact information such as real name, where to work and mailing address also is collected by the web page to facilitate statistics and analysis in future.

Then the information of HPC requirements is collected in two kinds of hardware and software information. If needed, applicants could describe their hardware requirements in form of how much disk storage, memory space, and computing hours. They could also specify what software stacks they are required and even a list of softwares for the target discipline and research area. Besides, we encourage applicants to provide information of research projects related to their HPC requirements and update in continually future. Finally, the applicant will receive a confirm email with an attachment which is a completed application form generated by the back end service dynamically in form of Portable Document Format (PDF).

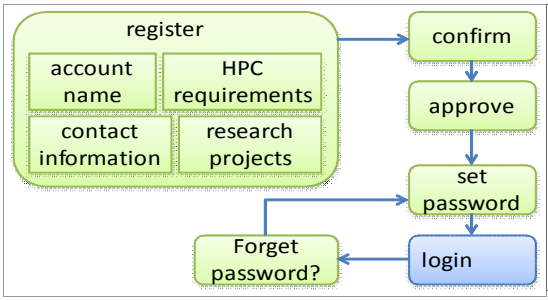

Figure2: the registration process in the non-login microservice

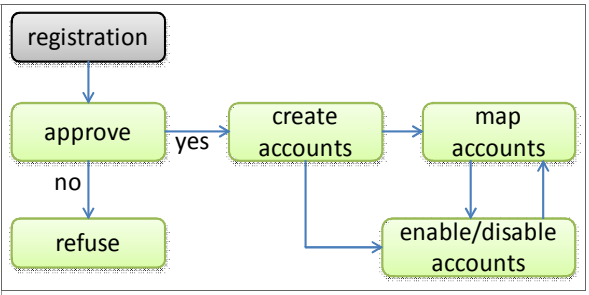

Figure3: the account workflow of full life-cycle management

When the registration is approved, the applicant will receive an email containing a Uniform Resource Locator (URL) for setting password and other information. On the web path specified by the one-time URL, the applicant set a password consisting of more than 6 letters and numbers to get a better security. Then the applicant becomes user and she/he could access and use national HPC environment provided by CNGrid. If a user forgets the password, there is no way to recover the original password because all passwords are stored in encrypted data by the Message Digest 5 (MD5) with salt. However, the user could apply for resetting the password in the password reset web page by filling account name, real name, email address and other contact information. If all information is correct, the backend service will send a one-time URL to the email address specified when applying the account. The user can poll for the email and reset a new password just like setting up the password firstly.

The non-login microservice is implemented based on Java servlet in server side and deployed in Apache Tomcat. In browser side, Asynchronous JavaScript and XML (Ajax) is used to transfer interactive data between the client and server. Most information of account, consisting of contact information, HPC requirements and research projects, is stored in MySQL database. The other information consisting of account name, password and attributes like a normal account in Linux is stored in OpenLDAP. In order to prevent information loss and provide better responsiveness, backup and read-only storages of account information are built on top of asynchronous delta update capability provided by MySQL and OpenLDAP separately. The other microservices access these main and backup storages directly to access and use account information.

4.2.Account Management Microservice

The account management microservice is a web gateway for administrators of CNGrid and it is transparent to normal users. It is an easy-to-use workbench that provides graphical and customized account workflow of full life-cycle management, especially in creating an account, mapping global accounts to multiple HPC local accounts, enabling and disabling any account logging and tracking. The account workflow is shown in figure 2 from the administrator-centered perspective and discussed as follows. Particularly, account mapping issues are discussed in context of heterogeneous and complicated national HPC environment.

4.2.1.Approve and Create Accounts

When opening the web page of workbench, a clear account workflow is presented and an administrator could process registration one by one according the steps provided in the workflow, as shown in figure 3. When receiving an account application, an administrator makes a decision to approve or deny the application mainly based on the parallel scalability and related research projects. CNGrid encourages users to develop and use massively parallel computation algorithms and softwares. In addition, CNGrid also preferentially supports basic research projects because it is funded by government funds partly. If the application is denied, the applicant will receive an email that contains a short and simple reason for rejection.

When creating an account, a new account is added to the Lightweight Directory Access Protocol (LDAP) account database provided by OpenLDAP. As Network Information Service (NIS) is a mechanism to centrally manage machine configuration data and was widely used in HPC resources years ago, LDAP is a better alternative because it is hierarchical, expandable and can be changed by adding schemas for authorization, contact information and so on. In the LDAP database, several important attributes are specified by the administrator, for example, the home directory attribute to specify high-speed storage in CNGrid, the start shell attribute to start up a suitable application in terminal when the user logins CNGrid by a command line client on Linux. For security, the password attribute is filled by a random sequence consisting of more than 16 numbers and letters, and then the account is disabled. When the applicant receives the notification, the applicant should set a new password to enable the account.

4.2.2.Map Global Accounts to HPC accounts

Until now, a global account is created and stored in the LDAP database. In order to access and use HPC resources in CNGrid, an administrator needs to map a global account to at least one local account provided by heterogeneous HPC resources. By bridging global accounts to HPC accounts, a user could find and use any available and required HPC resources with only one global account and don't need to take care of how to apply a local account from different HPC centers, manage several complicated credentials and login heterogeneous HPC resources in some certain but different ways. When a user accesses the HPC environment in CNGrid, the mapping mechanism is response for selecting a proper HPC account based on the computing queues, jobs applications, and switching the local HPC accounts in accordance with activities of the user. After selecting a local HPC account, the related services of SCE middleware interact with the target HPC resource on behalf of the global account under the protection of auditing and tracking policies.

Table 1: **Main Methods of Mapping Accounts**

Number of global Accounts	Number of Local Accounts in a HPC resource	Comments
One	One	Simple, needs lots of local HPC accounts.
One	Multiple	Not necessary at the same time.
Multiple	One	Complicate, needs a few of local HPC accounts.
Multiple	Multiple	Most complicate and flexible.

When discussing about mapping accounts, there are 4 main methods as shown in table 1 from the perspective of matching global accounts and local HPC accounts. In CNGrid, a global account is mapped to multiple HPC supercomputers by similar one-to-multiple method from the perspective of entire HPC environment so a user could access and use massive HPC resources via only one account. For each HPC cluster or supercomputer, a global account is mapped to the same local HPC account by one-to-one method in a period of time and switched to a different local HPC account in a year or even longer. Due to the dependency of jobs in a short time, mapping to the same local account is a good choice to improve data sharing and enable workflow among a group of related jobs. Considering the cyclical variation of HPC demands from a user, it is efficient and viable to bind a local account when the demands occur and unbind the account after a long idle period. Under these circumstances, it is possible cases that a global account is mapped to more than one local account and multiple global accounts are mapped to the same local account from the perspective of a very long time. Above, most of mapping methods are adopted in CNGrid to provide single account service for accessing all HPC resources.

For efficiency, each supercomputers provides lots accounts to UAMS at a time, and the administrators of UAMS could map different local accounts immediately when creating global accounts. In CNGrid, each HPC resource offers more than two hundrands accounts at one time. UAMS creates and maintains a virtual pool of accounts from of different HPC resources and each HPC resource owns a virtual queue in the pool. After creating an account in LDAP, an administrator creates mapping relationships with at least one HPC resource based on the different mapping methods and assistance utilities. The mapping relationships are stored as multiple text files in each HPC resource for local query launched by a front service near the HPC resource. In addition, distributed instances of the front service report each own mappings to a MySQL database for centralized query service. After setting mappings, the UAMS will send an email to tell the applicant that the application is approved and created successfully.

4.2.3.Enable and Disable Accounts

In the full life cycle of accounts, an administrator could enable and disable any account many times based on the activities of a user, the status of a HPC resource and limitations coming from security, maintenance and so on. In UAMS, several methods are implemented to manage privileges of accounts at varying degrees of granularity. The basic method is to turn on

or off the full privileges of an account. If an account is disabled by this method, the account is forbidden to login to CNGrid and the user cannot do anything in CNGrid. If malicious attacks are found by the security system, some accounts related to these safe events would be disabled by the basic method.

To add flexibility, all privileges of an account are divided to multiples categories. From the perspective of job management, an administrator could turn on or off the job submission and job query privileges separately. From the perspective of HPC resources, an administrator could set the status of availability. If the status is active of a HPC resource, a user could access and use the HPC resource. If not, all operations with the HPC resource are disabled. From the perspective of applications for different disciplines and research areas, an administrator could also set the status of availability. If the status is active of an application, a user could submit jobs with the application, or else a new submission is refused but the submitted jobs are not affected. To further enhance flexibility of privileges management, not only could be set separately each perspective for any account or all accounts, but also be set by a combination of these perspectives for any account or a group of accounts.

4.2.4.Log and Audit

The account management microservice is core and fundamental service in UAMS. Any operation of an administrator will affect at least an account and even the entire HPC environment provide by CNGrid. For stability and reliability, all operations of this microservice should be recorded in detail. In UAMS, a record should consist of timestamp, client name, operator name, target object name, operation name, and status of the operation. Besides, some optional attributes such as key parameters of the actions, are also recorded based on different operations. The client name attribute denotes on which client application an administrator does the operation. The operation name attribute defines what actions are done on the target object, such as an account, a HPC resource, an application and so on.

All records are transported to a groups of dedicated storage servers in ScGrid and CNGrid. All records are stored into a MySQL database and a logging system simultaneously. No matter an operation is executed successfully or failed because of some exceptions and reasons, a record is generated for the operation by UAMS and stored. The MySQL database is for querying by a combination of different conditions. In the web page of this microservice, there is a special area to display a list of historical operations with an account. So an administrator could check the historical operations launched by all administrators. To reduce workload of validation, an analysis system based on logs is used to find any risky event online. If any event activates a rule, a notification will be send to a group of administrations by email. All logging files are copied as backup periodically for reprocessing by new analysis models and algorithms in future.

The account management microservice is implemented in two web applications and deployed in Apache Tomcat. The one web application is RESTful web service and provides a group of account management API for administrators, detailed in section 5. The other web application is a web gateway for account management that implements the customized workflow and provides an easy-to-use workbench for administrators.

4.3.Account Update Microservice

The account update microservice is a web gateway for users already being logged in to manage their own accounts. It is well known that the HPC requirements of a user in not always the same as the time when he/she applied for an account. For example, the research projects related to the HPC requirements will be finished. Some new projects will be started and cause different and creative tasks that need more HPC resources and computing services. In addition, users often get different types of achievements on top of computing service provided CNGrid. To collect requirements changed continually and achievements contributed by computing, the account microservice is proposed and implemented to provide account-centered services for collecting and updating profile, requirements, projects and achievements information.

From the perspective of a user who has logged in this microservice, a user could maintain it's a variety of information related to the account in the web and graphical workbench provided by the account microservice. Firstly and most commonly, a user could update a new password periodically to enhance security. It only provides service for update a new password but not for reset a password. If a user forget the password and need to reset, he/she should reset a password in the non-login microservice. Another essential service is to update contact information when the original information is changed. Based on the correct contact information, the other services provided by CNGrid will send news, training notices, maintained events and other information.

It is creative to continually collect HPC requirements in full life-cycle of an account. A user could put forward demands on computing queues, compilations and libraries, memory and disk storage, applications and so on at any time. When the information is updated, an administrator will receive the information and assign available privileges to the user. To estimate the effectiveness of CNGrid, we encourage users to add their achievements, for example, articles, patents, and software copyright registrations, and related research projects in the web page provided the microservice. CNGrid collects and analyses these information year by year and make the results as part of annual report.

The account update microservice is implemented in two web applications and deployed in Apache Tomcat. The one web application is RESTful web service and provides a group of account update API for normal users, detailed in section 5. The other web application is a web gateway for normal user to update information on account profile, requirements, and achievements and so on.

4.4.LDAP Authentication Microservice

The LDAP authentication microservice is RESTful web service and provides username and password authentication scheme in multiple levels for different kinds of client applications of CNGrid, for example, desktop applications, command line tools, and scripts on Linux/Unix system. As mentioned in section 4.1, the information consisting of account name, password and attributes like a normal account in Linux is stored in OpenLDAP [21], but it doesn't mean all information is stored on a single server, because of OpenLDAP supports high availability and redundancy. OpenLDAP provides a stand-alone server, libraries implementing the LDAP protocol, and utilities, tools, and sample clients. It is easy to integrate the authentication abilities with Linux/Unix System, Apache HTTP Server, and Apache Tomcat Server. Especially in Linux/Unix System, the pam_ldap module allows (Pluggable Authentication Module) PAM-aware applications to authenticate users in LDAP, for example, console login. Similarly,

developers could implement some clients which support PAM-aware interface directly or could be protected by the console login and other PAM-aware applications.

In addition to the clients provided by OpenLDAP itself and lots of third-libraries, a group of RESTful API is implemented base on the LDAP library in Java to provide authentication service for different kinds of clients in a simple and compatible way, detailed in section 5. For manage a group accounts, a command-based LDAP toolkit is implement to access and use LDAP database directly. The toolkit provides simple commands for managing an account or a group of account. With the toolkit, it is easy to create lot of training accounts at one time and delete these accounts when the training is over. Considering CNGrid is a widely distributed HPC environment, the main node of OpenLDAP is deployed at main center of CNGrid which supports read and write operations, and several slave nodes are deployed in the backup site of CNGrid and login servers in distributed HPC environment which provide only read operations for local service and get update information from the main node in delta and synchronous way.

The account update microservice is RESTful web service implemented OpenLDAP and deployed in Apache Tomcat. This microservice provides LADP authentication, a group of RESTful API detailed in section 5, and a command-line toolkit. Unlike with the other microservices in UAMS, this microservice doesn't have graphical web pages, because it provides authentication service for developers only and normal users just interface with login process implemented by different clients.

4.5.SSO Authentication Microservice

For all kinds of web gateways, Single Sign-On (SSO) can provide better user experience than the basic username and password authentication. Under the SSO, a user login to a web gateway by filling in account, password and CAPTCHA manually. When the user wants to enter to the other gateways in the same instance of an internet browser, the user directly open the target web service without repeating login again. As above discussion, the SSO authentication microservice is proposed to provide unified login service for all web-based clients.

For best compatibility to support as many as possible web frameworks and programming languages, Apereo Central Authentication Service (Apereo-CAS) [22] is selected to build SSO service in UAMS. The Apereo-CAS is an enterprise and open source software which provides enterprise single sign-on service for the web and a friendly open source community that actively supports and contributes to the project. The Apereo-CAS supports several authentication handlers including the LDAP handler so that it is easy to bridge authentication information stored in the LDAP database. In addition, the Apereo-CAS officially provides a list of CAS client in .NET, Java and PHP languages. CAS client is a software package that can be integrated with various software platforms and applications in order to communicate with the CAS server using one or more supported protocols. The third-part also provides a list of clients in other programming languages. With these clients, several gateways have integrated the SSO service provided by the SSO microservice, detailed in section 6.

The SSO authentication microservice is implemented based on the Apereo-CAS server with some improvements and provides a group of RESTful API. In order to enhance security, the CAPTCHA is added to the login window. Some source codes of the Apereo-CAS server are modified to support the CAPTCHA. Besides, some configuration is done to bridge the Apereo-CAS to authenticate accounts by the information stored in LDAP database.

Table 2: Main Methods of Mapping Accounts

	Method	URL	Comments
Simple Authentication	POST	/users/login	Login into a client based on the LDAP library in CNGrid.
	GET	/users/logout	Logout from the client by a token in headers of a request.
Account Management	GET	/account	Get a list of accounts specified by a group of attributes.
	PUT	/account/{user}/ldap	Create an account and store it in the LDAP database.
	GET	/account/{user}/ldap	Get an account from the LDAP database specified by user.
	DELETE	/account/{user}/ldap	Delete an account from the LDAP database specified by user.
	PUT	/account/{user}/ldap/attribute	Update an attribute of a LDAP account specified by user.
	GET	/account/map	Get a list of mappings specified by a group of global accounts and name of HPC resources.
	PUT	/account/map	Add or update a group mappings specified by a data structure in the body of a request.
	GET	/admin/cluster	Get a list of privileges specified a group attributes, for example, accounts, computing queues, applications, and names of HPC resources.
	POST	/admin/cluster	Add or update a list of privileges. Each privilege is specified by a group attributes and enables/disables a rule.
	GET	/account/remarks	Get a list of remarks specified by a group of attributes.
	GET	/account/{user}/remark	Get a remark specified by user.
	PUT	/account/{user}/remark	Add a remark for the user.
Account Update	GET	/account/self/profile	Gets account information including projects and softwares.
	GET	/account/self/profile/projects	Get projects information specified by the login account.
	GET	/account/self/profile/softwares	Get projects softwares specified by the login account.
	PUT	/account/self/profile	Add or update account information including projects and softwares.
	GET	/account/self/remarks	Get a list of remarks specified by the login account.
	PUT	/account/self/remarks	Add a remark for the user specified by the login account.
SSO Authentication	POST	/casv4-service/v1/ticket	Login into a client based on the LDAP library and get a Ticket Granting Ticket.
	DELETE	/casv4-service/v1/ticket/{TGT}	Logout from the client for the user specified by the TGT.

5. RESTful WEB SERVICE AND API

The UAMS provides a group of API for developers to support the full life-cyclic account management. All of these API are from several RESTful web services implemented in different microservices based on the open source framework Jersey. Jersey is open source and production quality framework for developing RESTful web services in Java that provides support for JAX-RS API and serves as a JAX-RS (JSR 311 & JSR 339) reference implementation [23]. Besides, Jersey provides its own API that extends the JAX-RS toolkit to further simplify RESTful service and client development.

Atop of Jersey and API definitions, a group of API is implemented in several microservices separately and provides account services as a whole. As shown in table 2, a RESTful API is defined by a standard HTTP method such as GET, POST, PUT, and DELETE, a URL and a group of parameters. The symbols {} represents a variable in URL path. For example, the 'user' variable indicates the name of an account. If wants to call any API, a developer create a request in any language and platform, then sends the request. Each response is encoded in JavaScript Object Notation (JSON) format and contains a status object. The status object is a elf-explanatory data structure and has two member variables, the status code indicating the result of execution in number, and the status message describing the result in human readable format.

As shown in Table 2, all of the RESTful API can be divided into four categories. The first category provides basic username and password login service based on the OpenLDAP. The second category provides the full life-cyclic account management. The third category provides account update service for normal users. The fourth category provides SSO service .Although there are some reference implementations based on these API, for example, the general computing portal and the command-line toolkit used in CNGrid, developers still can design and implement their own services for different circumstances to provide better experience.

6. AUTHENTICATION SCHEMES AND USE CASES

UAMS provides the basic username and password authentication scheme and its variants for developers to integrate the authentication service into their clients. It is easy for developers to use authentication services of UAMS based not only on the libraries and clients provided by the OpenLDAP and the Apereo-CAS, but also on the group of RESTful API provided by UAMS. Several authentication schemes are discussed and some use cases are introduced as follows.

6.1 LDAP Authentication Scheme

In UAMS, account information consisting of username and password is stored the LDAP database. It is natural to utilize authentication capability brought by OpenLDAP. As known and popular open source software, OpenLDAP provides lots of clients and libraries for different programming languages and platforms. If a client wants to use the account information of CNGrid, the developers of this client should find a suitable client or library to configure and implement authentication service atop of the LDAP service. For security, the client could add the CAPTCHA verification, throttle limit and other policies to prevent malicious attacks.

In CNGrid, a command-line toolkit was implemented at ten years ago and provides a group of unified and easy-to-use commands for normal users to access and use all HPC resources and computing applications, as shown in figure 4. With the toolkit, a user could do data, jobs, applications and computing queues management on different and heterogeneous supercomputers with only one group of commands. The authentication service of the command-line toolkit is delegated to a PAM-aware login client that authenticates a user based on LDAP database of UAMS. Besides, the LDAP toolkit mentioned in section 4.4 also integrated the authentication service based on a LDAP library in Java..

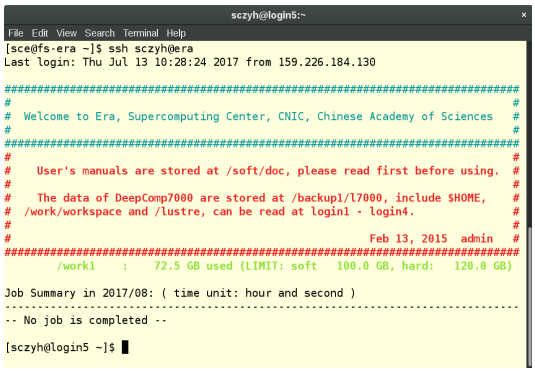

Figure 4:the screenshot of the command-line toolkit

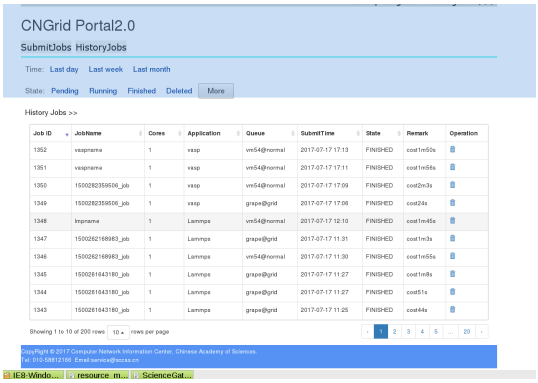

Figure 5:the screenshot of the computing portal

6.2 SSO Authentication Scheme

For different web gateways and portals, SSO is the better recommendation to achieve 'login once and access everywhere' in CNGrid. The SSO of UMAS is built mainly for the web applications, consisting of desktop and mobile web applications implemented in different languages, frameworks and platforms. Developers need to search for a suitable client of the Apereo-CAS for their web applications according to the web container, the platform, the programming language and so on. Because there are lots of different client for the Apereo-CAS as mentioned in section 4.5, it is rare for developers to implement a specialized client based on protocols provided by the Apereo-CAS. With a suitable client, developer just needs to configure parameters to make web application support the SSO authentication scheme provided by UAMS.

In CNGrid, there are several web applications supporting SSO of UAMS. The general computing portal, as shown in figure 5, is a general-purpose computing community for different disciplines and research areas [24]. The Operation and User Support (OPUS) portal is web workbench for administrators to maintain and manage the CNGrid. Besides, there are several demonstrated web communities built in the 12th five-plan for industry, chemistry, medicine, multi-media, and other research areas. All of these web applications support SSO so that a user can login any gateway then access and use other portals directly in the same instance of the internet browser.

6.3 RESTful API Authentication Scheme

The authentication API of UAMS provides flexible and compatible authentication scheme for developers because the cross-language and cross-platform features of the RESTful API and the popularity of the HTTP protocols. As mentioned in section 5, the login and logout API are provided to support the username and password authentication scheme. In fact, the login API, which is responsible for authenticating the credentials, is based on the LDAP library and the LDAP database used UAMS. The logout API is a remark function that records the logout timestamp and always returns successfully. If a client wants to use the authentication API, developers should design the complete authentication process for the client and validate the credentials based on the RESTful API provided by UAMS.

In the collaboration work with the ATLAS experiment, the customized Compute Element of the Advanced Resource Connector (ARC-CE) is used to exploit more HPC resources quickly for the ATLAS experiment based on SCEAPI [20, 25]. In the customized ARC-CE, the login and logout API is used to login to the CNGrid from the job management service of the ATLAS in a Python script. In addition, the authentication API could be used to verify the correctness quickly in prototype and will be replaced by SSO authentication scheme when the development is finished.

7. PERFORMANCE AND TEST

The UAMS is designed to provide account management and authentication service for users, administrators and developers in CNGrid. The UAMS has been implemented, deployed and maintained from 2012 and continually provides services for many years. In order to avoid affect the online services, tests were executed in the development environment for developers to implement and test different kinds of client related to CNGrid.

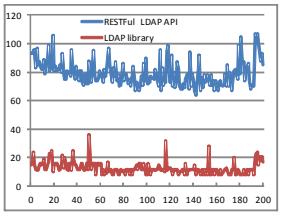

Figure 6: the time lines for creating accounts

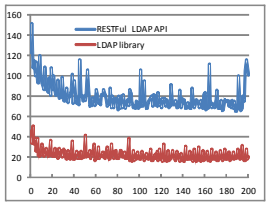

Figure 7:the time lines for changing passwords

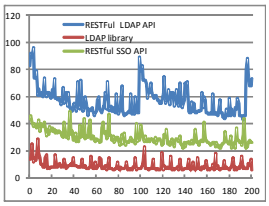

Figure 8:the time lines for authenticating accounts

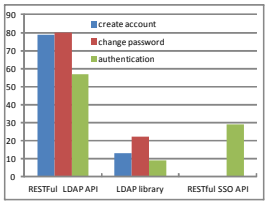

Figure 9: the average time for 3 operations in 3 implementations

The development environment is consisted of several virtual severs. Each server runs x86_64 Linux, and has a simulated CPU with two cores at 2.27 GHz, 4GB memory and 20GB disk storage. The first sever deploys the load balance microservice, the non-login service and the account update service. The second server deploys the account management microservice, the LDAP authentication microservice and the SSO authentication microservice. There are the other servers to run services provided by SCE middleware, and to simulate supercomputers and clusters. The clients for test are deployed at ordinary computers in the Computer Network Information Center (CNIC). All servers are located at the sub center of CNIC in Huairou, a small town north of Beijing and 50 kilometers away from CNIC. The networks between two centers are provided by CSTNET.

In UAMS, the important and frequently-used operations are creating an account, updating a password and authenticating an account. And the elapsed time for these 3 operations was

recorded from the perspective of clients. As well known, it is time consuming work to establish a connection in Https protocol especially for the first time. It needs about 600 to 1000 milliseconds to establish a secure connection for different clients in our tests. Because Establishing and maintaining connections are mainly handled by web containers, we established connections firstly and then we executed the test operations and recorded elapsed time consumed by each operation.

As shown in figure 6-9, the vertical axis stands for the time consumed by each operation in milliseconds, and the horizontal axis represents the sequence number of each operation for figure 6-8. For figure legends, the RESTful LDAP API is RESTful encapsulation of the LDAP library in several microservices and the SSO RESTful API is provided by the Apereo-CAS server modified in UAMS. As shown in figure 6, the operation of creating accounts is tested in two methods based on the LDAP library and the RESTful LDAP API separately. As shown in figure 7, the operation of changing passwords is tested on the same two methods of creating accounts. As shown in figure 8, the operation of authenticating accounts is tested on the two methods and the SSO RESTful API. From the perspective of the trends of lines in figure 6-8, the elapsed time of each test target is relatively smooth and the range of variation is within tens of milliseconds.

As shown in figure 9, the native LDAP library has the best performance. The RESTful LDAP API needs much more time to do the same operations than the native LDAP library but the average time is less than 80 milliseconds. For the SSO RESTful API, it takes an average 30 milliseconds to authenticate an account. From the perspective of the average time, the elapsed time of each operation is acceptable and could bring good experiences.

8.CONCLUSIONS

In this paper, UAMS is proposed to provide account management and authentication schemes for accessing and using national HPC environment via a global account of CNGrid. UAMS is implemented in self-contained microservices and solved challenges for mapping a global account to many local accounts, and provides unified account registration, management and authentication for administrators and users. It is creative for users to update the requirements of HPC in the full life-cycle of accounts. Correspondingly, administrators adjust assign available privileges to satisfy the demands.

UAMS provides the basic username and password authentication scheme and its variants for developers to integrate the authentication service into their clients. With UAMS, multiple clients, consisting of command-line toolkits, web gateways and scripts, have integrated authentication service based different authentication schemes. In future, we plan to implement federated account service that enable a local HPC account similar to a global account to login the national HPC environment, access and use all HPC resources in CNGrid.

ACKNOWLEDGMENTS

This work was partially supported by National Natural Science Foundation of China under grant No. 61702476.

References

[1] Fu, Haohuan, et al, *The Sunway TaihuLight supercomputer: system and applications,* Science China Information Sciences 59.7 (2016): 072001.
[2] Lu, Yutong, *Overview of tianhe-2 (MilkyWay-2) supercomputer*, Tech. Rep.(2012).
[3] Depei, Qian, *CNGrid: A test-bed for grid technologies in China,* Distributed Computing Systems, 2004. FTDCS 2004.
[4] Basney, Jim, Marty Humphrey, and Von Welch, *The MyProxy online credential repository,* Software: Practice and Experience 35.9 (2005): 801-816..
[5] Dooley, Rion, Joe Stubbs, and Jim Basney. "The MyProxy Gateway."Science Gateways (IWSG), 2014 6th International Workshop on. IEEE, 2014.
[6] Basney, Jim, et al, *Integrating science gateways with xsede security: A survey of credential management approaches*, Proceedings of the 2014 Annual Conference on Extreme Science and Engineering Discovery Environment. ACM, 2014..
[7] Anantula, Pranayanath Reddy, and G. Manoj Someswar, *Authenticating users with multiple levels of validations in a secure Cloud Computing Environment,* i-manager's Journal on Cloud Computing 3.3 (2016): 9.
[8] Gnanaraj, Jaspher Willsie Kathrine, Kirubakaran Ezra, and Elijah Blessing Rajsingh, *Smart card based time efficient authentication scheme for global grid computing,* Human-centric Computing and Information Sciences 3.1 (2013): 16.
[9] Wang, Ding, and Ping Wang, *Two birds with one stone: Two-factor authentication with security beyond conventional bound,* IEEE Transactions on Dependable and Secure Computing (2016).
[10] Welch, V, *Globus toolkit version 4 grid security infrastructure: A standards perspective,* The Globus Alliance (2005).
[11] Ananthakrishnan, Rachana, et al, *Globus Nexus: An identity, profile, and group management platform for science gateways and other collaborative science applications,* Cluster Computing (CLUSTER), 2013 IEEE International Conference on. IEEE, 2013.
[12] Alipay, *Antfin Open Platform,* https://open.alipay.com/platform/home.htm, 2017.
[13] WeChat, *WeChat Open Platform,* https://open.weixin.qq.com, 2017
[14] CNIC, *CSTNET passport,*
http://english.cnic.cas.cn/RS/support/201611/t20161121_170806.html, 2016.
[15] Bachmann, Felix, et al, *XSEDE architecture overview. Technical Report,* 2014.
[16] Cholia, Shreyas, and Terence Sun, *The NEWT platform: an extensible plugin framework for creating ReSTful HPC APIs,* Concurrency and Computation: Practice and Experience 27.16 (2015): 4304-4317.
[17] Scarborough, Walter, Carrie Arnold, and Maytal Dahan, *Case Study: Microservice Evolution and Software Lifecycle of the XSEDE User Portal API*, Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale. ACM, 2016.
[18] Benchara, Fatéma Zahra, et al, *A new efficient distributed computing middleware based on cloud micro-services for HPC,* Multimedia Computing and Systems (ICMCS), 2016 5th International Conference on. IEEE, 2016.
[19] Dai, Zhihui, et al, *A lightweight grid middleware based on OPENSSH-SCE,* Grid and Cooperative Computing, 2007. GCC 2007. Sixth International Conference on. IEEE, 2007.
[20] Cao Rongqiang, Wang Xiaoning, et al, *SCEAPI: A unified Restful Web API for High-Performance Computing ,* CHEP 2016 Conference, San Francisco, October 8-14, 2016.
[21] OpenLDAP Foundation, *OpenLDAP*, https://www.openldap.org, 2017.
[22] Apereo, *Enterprise Single Sign-On - CAS*, https://www.apereo.org/projects/cas, 2017.
[23] Oracle Corporation, *Jersey- RESTful Web Services in Java,* https://jersey.java.net, 2017.
[24] CNIC, *CNGrid Portal,* http://test.cngrid.net or http://portal.cngrid.net, 2017.
[25] Collaboration, A. T. L. A. S, *Integration of the Chinese HPC Grid in ATLAS Distributed Computing,* CHEP 2016 Conference, San Francisco, October 8-14, 2016.