NIST Big Data Interoperability Framework

Volume 8: Reference Architecture Interfaces

Wo L Chang  
Gregor von Laszewski  
Editor

laszewski@gmail.com

https://github.com/cloudmesh/cloudmesh-nist

July 01, 2019 - 12:43 PM

Created by Cloudmesh & Cyberaide Bookmanager, https://github.com/cyberaide/bookmanager
NIST Big Data Interoperability Framework: Volume 8, Reference Architecture Interfaces

Editor: Gregor von Laszewski, Working Group co-Chair: Wo Chang

Copyrights and Permissions: Official publications of the National Institute of Standards and Technology are not subject to copyright in the United States. Foreign rights are reserved. Questions concerning the possibility of copyrights in foreign countries should be referred to the Office of Chief Counsel at NIST via email to nistcounsel@nist.gov.
NIST Big Data Interoperability Framework: Volume 8, Reference Architecture Interfaces

1 Introduction
   1.1 Background
   1.2 Scope and Objectives of the Reference Architectures Subgroup
   1.3 Report Production
   1.4 Report Structure

2 NBDRA Interface Requirements
   2.1 High-Level Requirements of the Interface Approach
      2.1.1 Technology- and Vendor-Agnostic
      2.1.2 Support of Plug-In Compute Infrastructure
      2.1.3 Orchestration of Infrastructure and Services
      2.1.4 Orchestration of Big Data Applications and Experiments
      2.1.5 Reusability
      2.1.6 Execution Workloads
      2.1.7 Security and Privacy Fabric Requirements
   2.2 Component-Specific Interface Requirements
      2.2.1 System Orchestrator Interface Requirements
      2.2.2 Data Provider Interface Requirements
      2.2.3 Data Consumer Interface Requirements
      2.2.4 Big Data Application Interface Provider Requirements
      2.2.5 Big Data Provider Framework Interface Requirements
      2.2.6 Big Data Application Provider to Big Data Framework Provider Interface

3 Specification Paradigm
   3.1 Hybrid and Multiple Frameworks
   3.2 Design by Resource-Oriented Architecture
   3.3 Design by Example
   3.4 Version Management
   3.5 Interface Compliancy
   3.6 Reference implementations

4 Specification
   4.1 List of specifications
   4.2 Authentication
   4.3 Status Codes and Error Responses
   4.4 Timestamp
      4.4.1 Timestamp
   4.5 Identity
      4.5.1 Organization
      4.5.2 User
      4.5.3 Account
4.5.4 Public Key Store
4.6 Variable, Default, and Alias
  4.6.1 Alias
  4.6.2 Variables
  4.6.3 Default
4.7 Data Management
  4.7.1 Filestore
  4.7.2 Replica
  4.7.3 Database
  4.7.4 Virtual Directory
4.8 Compute Management - Virtual Clusters
  4.8.1 Virtual Cluster
  4.8.2 Network of Nodes
  4.8.3 Scheduler
  4.8.4 Queue
4.9 Compute Management - Virtual Machines
  4.9.1 Image
  4.9.2 Flavor
  4.9.3 Virtual Machine
  4.9.4 Secgroup
  4.9.5 Nic
4.10 Compute Management - Containers
  4.10.1 Containers
4.11 Compute Management - Map Reduce
  4.11.1 Map Reduce
4.12 Compute Management - Functions
  4.12.1 Microservice
4.13 Reservation
  4.13.1 Reservation
4.14 Data Streams
  4.14.1 Stream
  4.14.2 Filter
4.15 Deployment
  4.15.1 Deployment
5 Acronyms and Terms
Bibliography
NIST Big Data Interoperability Framework: Volume 8, Reference Architecture Interfaces
NIST Big Data Interoperability Framework: Volume 8, Reference Architecture Interfaces

NIST Big Data Public Working Group
Reference Architecture Subgroup

July 2019

This publication is available free of charge from:
https://doi.org/10.6028/NIST.SP.1500-9
Abstract
This document summarizes interfaces that are instrumental for the interaction with Clouds, Containers, and High Performance Computing (HPC) systems to manage virtual clusters to support the NIST Big Data Reference Architecture (NBDRA). The REpresentational State Transfer (REST) paradigm is used to define these interfaces, allowing easy integration and adoption by a wide variety of frameworks.

Big Data is a term used to describe extensive datasets, primarily in the characteristics of volume, variety, velocity, and/or variability. While opportunities exist with Big Data, the data characteristics can overwhelm traditional technical approaches, and the growth of data is outpacing scientific and technological advances in data analytics. To advance progress in Big Data, the NIST Big Data Public Working Group (NBD-PWG) is working to develop consensus on important fundamental concepts related to Big Data. The results are reported in the NIST Big Data Interoperability Framework (NBDIF) series of volumes. This volume, Volume 8, uses the work performed by the NBD-PWG to identify objects instrumental for the NIST Big Data Reference Architecture (NBDRA) which is introduced in the NBDIF: Volume 6, Reference Architecture.

Keywords

Adoption; barriers; implementation; interfaces; market maturity; organizational maturity; project maturity; system modernization.

Acknowledgements

This document reflects the contributions and discussions by the membership of the NBD-PWG, cochaired by Wo Chang (NIST ITL), Bob Marcus (ET-Strategies), and Chaitan Baru (San Diego Supercomputer Center; National Science Foundation). For all versions, the Subgroups were led by the following people: Nancy Grady (SAIC), Natasha Balac (SDSC), and Eugene Luster (R2AD) for the Definitions and Taxonomies Subgroup; Geoffrey Fox (Indiana University) and Tsegereda Beyene (Cisco Systems) for the Use Cases and Requirements Subgroup; Arnab Roy (Fujitsu), Mark Underwood (Krypton Brothers; Synchrony Financial), and Akhil Manchanda (GE) for the Security and Privacy Subgroup; David Boyd (InCadence Strategic Solutions), Orit Levin (Microsoft), Don Krapohl (Augmented Intelligence), and James Ketner (AT&T) for the Reference Architecture Subgroup; and Russell Reinsch (Center for Government Interoperability), David Boyd (InCadence Strategic Solutions), Carl Buffington (Vistronix), and Dan McClary (Oracle), for the Standards Roadmap Subgroup, Gregor von Laszewski (Indiana University) for the Interface Subgroup.

The following milestone releases exist:

- Version 2.1: A previous volume used the definitions of the schema based on examples only. It was easier to read but only included the definition of the resources and not the interaction with the resources. This volume was in place until June 2018.
- Version 2.2: This version was significantly changed and used OpenAPI 2.0 to
specify the interfaces between the various services and components.

- Version 3.1.1: The version includes significant improvements of the object specifications but are still using OpenAPI 2.0.
- Version 3.2.0: All specifications have been updated to OpenAPI 3.0.2. Significant updates have been done to a number of specifications.

The editors for these documents are:

- Gregor von Laszewski (Indiana University)
- Wo Chang (NIST).

Laurie Aldape (Energetics Incorporated) and Elizabeth Lennon (NIST) provided editorial assistance across all NBDIF volumes.

NIST SP 1500-9, Draft NIST Big Data Interoperability Framework: Volume 8, Reference Architecture Interfaces, Version 2 has been collaboratively authored by the NBD-PWG. As of the date of publication, there are over six hundred NBD-PWG participants from industry, academia, and government. Federal agency participants include the National Archives and Records Administration (NARA), National Aeronautics and Space Administration (NASA), National Science Foundation (NSF), and the U.S. Departments of Agriculture, Commerce, Defense, Energy, Census, Health and Human Services, Homeland Security, Transportation, Treasury, and Veterans Affairs.

NIST would like to acknowledge the specific contributions to this volume, during Version 1 and/or Version 2 activities. **Contributors** are members of the NIST Big Data Public Working Group who dedicated great effort to prepare and gave substantial time on a regular basis to research and development in support of this document. This includes the following NBD-PWG members:

- Gregor von Laszewski, Indiana University
- Wo Chang, National Institute of Standard and Technology,
- Fugang Wang, Indiana University
- Geoffrey C. Fox, Indiana University
- Shirish Joshi, Indiana University
- Badi Abdul-Wahid, formerly Indiana University
- Alicia Zuniga-Alvarado, Consultant
- Robert C. Whetsel, DISA/NBIS
- Pratik Thakkar, Philips

---

**Executive Summary**

The **NIST Big Data Interoperability Framework (NBDIF): Volume 8, Reference Architecture Interfaces** document was prepared by the NIST Big Data Public Working Group (NBD-PWG) Reference Architecture Subgroup to identify interfaces in support of the NIST Big Data Reference Architecture (NBDRA). The interface define resources that are part of the
NBDRA. These resources are formulated in OpenAPI 3.0.2 format and can be easily integrated into a REpresentational State Transfer (REST) framework or an object-based framework.

The resources were categorized in groups that are identified by the NBDRA set forward in the NBDIF: Volume 6, Reference Architecture document. While the NBDIF: Volume 3, Use Cases and General Requirements document provides application-oriented high-level use cases, the use cases defined in this document are subsets of them and focus on interface use cases. The interface use cases are not meant to be complete examples, but showcase why the resource has been defined. Hence, the interfaces use cases are only representative, and do not encompass the entire spectrum of Big Data usage. All the interfaces were openly discussed in the working group [1]. Additions to the interfaces are welcome and the NBD-PWG is open to discuss any contributions.

The NIST Big Data Interoperability Framework (NBDIF) was released in three versions, which correspond to the three stages of the NBD-PWG work. Version 3 (current version) of the NBDIF volumes resulted from Stage 3 work with major emphasis on the validation of the NBDRA Interfaces and content enhancement. Stage 3 work built upon the foundation created during Stage 2 and Stage 1. The current effort documented in this volume reflects concepts developed within the rapidly evolving field of Big Data. The three stages (in reverse order) aim to achieve the following with respect to the NIST Big Data Reference Architecture (NBDRA):

- Stage 1: Identify the high-level Big Data reference architecture key components, which are technology-, infrastructure-, and vendor-agnostic.
- Stage 2: Define general interfaces between the NBDRA components; and
- Stage 3: Validate the NBDRA by building Big Data general applications through the general interfaces;

The NBDIF consists of nine volumes, each of which addresses a specific key topic, resulting from the work of the NBD-PWG. The nine volumes are as follows:

- Volume 1, Definitions [2]
- Volume 2, Taxonomies [3]
- Volume 3, Use Cases and General Requirements [4]
- Volume 4, Security and Privacy [5]
- Volume 6, Reference Architecture [7]
- Volume 7, Standards Roadmap [8]
- Volume 8, Reference Architecture Interfaces (this volume) [9]
- Volume 9, Adoption and Modernization [10]

During Stage 1, Volumes 1 through 7 were conceptualized, organized and written. The finalized Version 1 documents can be downloaded from the V1.0 Final Version page of the NBD-PWG website [11].

During Stage 2, the NBD-PWG developed Version 2 of the NBDIF Version 1 volumes, with the exception of Volume 5, which contained the completed architecture survey work that was
used to inform Stage 1 work of the NBD-PWG. The goals of Version 2 were to enhance the
Version 1 content, define general interfaces between the NBDRA components by aggregating
low-level interactions into high-level general interfaces, and demonstrate how the NBDRA can
be used. As a result of the Stage 2 work, the need for NBDIF Volume 8 and NBDIF Volume 9
were identified and the two new volumes were created. Version 2 of the NBDIF volumes,
resulting from Stage 2 work, can be downloaded from the V2.0 Final Version page of the
NBD-PWG website [12].

This document is the result of Stage 3 work of the NBD-PWG. Coordination of the group is
carried out on the NBD-PWG web page [1].
INTRODUCTION

1.1 BACKGROUND

There is broad agreement among commercial, academic, and government leaders about the potential of Big Data to spark innovation, fuel commerce, and drive progress. Big Data is the common term used to describe the deluge of data in today’s networked, digitized, sensor-laden, and information-driven world. The availability of vast data resources carries the potential to answer questions previously out of reach, including the following:

- How can a potential pandemic reliably be detected early enough to intervene?
- Can new materials with advanced properties be predicted before these materials have ever been synthesized?
- How can the current advantage of the attacker over the defender in guarding against cybersecurity threats be reversed?

There is also broad agreement on the ability of Big Data to overwhelm traditional approaches. The growth rates for data volumes, speeds, and complexity are outpacing scientific and technological advances in data analytics, management, transport, and data user spheres.

Despite widespread agreement on the inherent opportunities and current limitations of Big Data, a lack of consensus on some important fundamental questions continues to confuse potential users and stymie progress. These questions include the following:

- How is Big Data defined?
- What attributes define Big Data solutions?
- What is new in Big Data?
- What is the difference between Big Data and bigger data that has been collected for years?
- How is Big Data different from traditional data environments and related applications?
- What are the essential characteristics of Big Data environments?
- How do these environments integrate with currently deployed architectures?
- What are the central scientific, technological, and standardization challenges that need to be addressed to accelerate the deployment of robust, secure Big Data solutions?

Within this context, on March 29, 2012, the White House announced the Big Data Research and Development Initiative (The White House Office of Science and Technology Policy, “Big Data is a Big Deal,” OSTP Blog, accessed February 21, 2014 [13]. The initiative’s goals include helping to accelerate the pace of discovery in science and engineering, strengthening national security, and transforming teaching and learning by improving analysts’ ability to extract knowledge and insights from large and complex collections of digital data.
Six federal departments and their agencies announced more than $200 million in commitments spread across more than 80 projects, which aim to significantly improve the tools and techniques needed to access, organize, and draw conclusions from huge volumes of digital data. The initiative also challenged industry, research universities, and nonprofits to join with the federal government to make the most of the opportunities created by Big Data.

Motivated by the White House initiative and public suggestions, the National Institute of Standards and Technology (NIST) accepted the challenge to stimulate collaboration among industry professionals to further the secure and effective adoption of Big Data. As a result of NIST’s Cloud and Big Data Forum held on January 15–17, 2013, there was strong encouragement for NIST to create a public working group for the development of a Big Data Standards Roadmap. Forum participants noted that this roadmap should define and prioritize Big Data requirements, including interoperability, portability, reusability, extensibility, data usage, analytics, and technology infrastructure. In doing so, the roadmap would accelerate the adoption of the most secure and effective Big Data techniques and technology.

On June 19, 2013, the NIST Big Data Public Working Group (NBD-PWG) was launched with extensive participation by industry, academia, and government from across the nation. The scope of the NBD-PWG involves forming a community of interests from all sectors—including industry, academia, and government—with the goal of developing consensus on definitions, taxonomies, secure reference architectures, security and privacy, and, from these, a standards roadmap. Such a consensus would create a vendor-neutral, technology- and infrastructure-independent framework that would enable Big Data stakeholders to identify and use the best analytics tools for their processing and visualization requirements on the most suitable computing platform and cluster, while also allowing added value from Big Data service providers.

The **NIST Big Data Interoperability Framework (NBDIF)** was released in three versions, which correspond to the three stages of the NBD-PWG work. Version 3 (current version) of the NBDIF volumes resulted from Stage 3 work with major emphasis on the validation of the NBDRA Interfaces and content enhancement. Stage 3 work built upon the foundation created during Stage 2 and Stage 1. The current effort documented in this volume reflects concepts developed within the rapidly evolving field of Big Data. The three stages (in reverse order) aim to achieve the following with respect to the NIST Big Data Reference Architecture (NBDRA).

Stage 3: Validate the NBDRA by building Big Data general applications through the general interfaces; Stage 2: Define general interfaces between the NBDRA components; and Stage 1: Identify the high-level Big Data reference architecture key components, which are technology-, infrastructure-, and vendor-agnostic.

The NBDIF consists of nine volumes, each of which addresses a specific key topic, resulting from the work of the NBD-PWG. The nine volumes are as follows:

- Volume 1, Definitions [2]
- Volume 2, Taxonomies [3]
- Volume 3, Use Cases and General Requirements [4]
• Volume 4, Security and Privacy [5]
• Volume 5, Architectures White Paper Survey [6]
• Volume 6, Reference Architecture [7]
• Volume 7, Standards Roadmap [8]
• Volume 8, Reference Architecture Interfaces (this volume) [9]
• Volume 9, Adoption and Modernization [10]

During Stage 1, Volumes 1 through 7 were conceptualized, organized and written. The finalized Version 1 documents can be downloaded from the V1.0 Final Version page of the NBD-PWG website [11].

During Stage 2, the NBD-PWG developed Version 2 of the NBDIF Version 1 volumes, with the exception of Volume 5, which contained the completed architecture survey work that was used to inform Stage 1 work of the NBD-PWG. The goals of Version 2 were to enhance the Version 1 content, define general interfaces between the NBDRA components by aggregating low-level interactions into high-level general interfaces, and demonstrate how the NBDRA can be used. As a result of the Stage 2 work, the need for NBDIF Volume 8 and NBDIF Volume 9 were identified and the two new volumes were created. Version 2 of the NBDIF volumes, resulting from Stage 2 work, can be downloaded from the V2.0 Final Version page of the NBD-PWG website [12].

1.2 Scope and Objectives of the Reference Architectures Subgroup

Reference architectures provide “an authoritative source of information about a specific subject area that guides and constrains the instantiations of multiple architectures and solutions” [14]. Reference architectures generally serve as a foundation for solution architectures and may also be used for comparison and alignment of instantiations of architectures and solutions.

The goal of the NBD-PWG Reference Architecture Subgroup is to develop an open reference architecture for Big Data that achieves the following objectives:

• Provides a common language for the various stakeholders;
• Encourages adherence to common standards, specifications, and patterns;
• Provides consistent methods for implementation of technology to solve similar problem sets;
• Illustrates and improves understanding of the various Big Data components, processes, and systems, in the context of a vendor- and technology-agnostic Big Data conceptual model;
• Provides a technical reference for U.S. government departments, agencies, and other consumers to understand, discuss, categorize, and compare Big Data solutions; and
• Facilitates analysis of candidate standards for interoperability, portability, reusability, and extendibility.

The NBDRA is a high-level conceptual model crafted to serve as a tool to facilitate open discussion of the requirements, design structures, and operations inherent in Big Data. The
NBDRA is intended to facilitate the understanding of the operational intricacies in Big Data. It does not represent the system architecture of a specific Big Data system, but rather is a tool for describing, discussing, and developing system-specific architectures using a common framework of reference. The model is not tied to any specific vendor products, services, or reference implementation, nor does it define prescriptive solutions that inhibit innovation.

The NBDRA does not address the following:

- Detailed specifications for any organization’s operational systems;
- Detailed specifications of information exchanges or services; and
- Recommendations or standards for integration of infrastructure products.

The goals of the Subgroup were realized throughout the three planned phases of the NBD-PWG work, as outlined in Section 1.3.

1.3 REPORT PRODUCTION

The NBDIF: Volume 8, References Architecture Interfaces is one of nine volumes, whose overall aims are to define and prioritize Big Data requirements, including interoperability, portability, reusability, extensibility, data usage, analytic techniques, and technology infrastructure to support secure and effective adoption of Big Data. The overall goals of this volume are to define and specify interfaces to implement the Big Data Reference Architecture. This volume arose from discussions during the weekly NBD-PWG conference calls. Topics included in this volume began to take form in Phase 2 of the NBD-PWG work. During the discussions, the NBD-PWG identified the need to specify a variety of interfaces.

Phase 3 work, which built upon the groundwork developed during Phase 2, included an early specification based on resource object specifications that provided a simplified version of an API interface design.

1.4 REPORT STRUCTURE

To enable interoperability between the NBDRA components, a list of well-defined NBDRA interfaces is needed. These interfaces are documented in this volume. To introduce them, the NBDRA structure will be followed, focusing on interfaces that allow bootstrapping of the NBDRA. The document begins with a summary of requirements that will be integrated into our specifications. Subsequently, each section will introduce a number of objects that build the core of the interface addressing a specific aspect of the NBDRA. A selected number of interface use cases will be showcased to outline how the specific interface can be used in a reference implementation of the NBDRA. Validation of this approach can be achieved while applying it to the application use cases that have been gathered in the NBDIF: Volume 3, Use Cases and Requirements document. These application use cases have considerably contributed towards the design of the NBDRA. Hence the expectation is that: (1) the interfaces can be used to help implement a Big Data architecture for a specific use case; and (2) the proper
implementation. This approach can facilitate subsequent analysis and comparison of the use cases.

The organization of this document roughly corresponds to the process used by the NBD-PWG to develop the interfaces. Following the introductory material presented in Section 1, the remainder of this document is organized as follows:

- Section 2 presents the interface requirements;
- Section 3 presents the specification paradigm that is used;
- Section 4 presents several objects grouped by functional use while providing a summary table of selected proposed objects in Section 4.1.

While each NBDIF volume was created with a specific focus within Big Data, all volumes are interconnected. During creation, the volumes gave and/or received input from other volumes. Broad topics (e.g., definition, architecture) may be discussed in several volumes with the discussion circumscribed by the volume’s particular focus. Arrows shown in Figure 1 indicate the main flow of input/output. Volumes 2, 3, and 5 (blue circles) are essentially standalone documents that provide output to other volumes (e.g., to Volume 6). These volumes contain the initial situational awareness research. Volumes 4, 7, 8, and 9 (green circles) primarily received input from other volumes during the creation of the particular volume. Volumes 1 and 6 (red circles) were developed using the initial situational awareness research and continued to be modified based on work in other volumes. These volumes also provided input to the green circle volumes.
Figure 1: NBDIF Documents Navigation Diagram Provides Content Flow Between Volumes
The development of a Big Data reference architecture requires a thorough understanding of current techniques, issues, and concerns. To this end, the NBD-PWG collected use cases to gain an understanding of current applications of Big Data, conducted a survey of reference architectures to understand commonalities within Big Data architectures in use, developed a taxonomy to understand and organize the information collected, and reviewed existing technologies and trends relevant to Big Data. The results of these NBD-PWG activities were used in the development of the NBDRA (Figure 2) and the interfaces presented herein. Detailed descriptions of these activities can be found in the other volumes of the NBDIF.

![Figure 2: NIST Big Data Reference Architecture (NBDRA)](image)

This vendor-neutral, technology- and infrastructure-agnostic conceptual model, the NBDRA, is shown in Figure 2 and represents a Big Data system composed of five logical functional components connected by interoperability interfaces (i.e., services). Two fabrics envelop the components, representing the interwoven nature of management and security and privacy with all five of the components. These two fabrics provide services and functionality to the five main roles in the areas specific to Big Data and are crucial to any Big Data solution. Note: None of the terminology or diagrams in these documents is intended to be normative or to
imply any business or deployment model. The terms *provider* and *consumer* as used are descriptive of general roles and are meant to be informative in nature.

The NBDRA is organized around five major roles and multiple sub-roles aligned along two axes representing the two Big Data value chains: the Information Value (horizontal axis) and the Information Technology (IT; vertical axis). Along the Information Value axis, the value is created by data collection, integration, analysis, and applying the results following the value chain. Along the IT axis, the value is created by providing networking, infrastructure, platforms, application tools, and other IT services for hosting and operating the Big Data in support of required data applications. At the intersection of both axes is the Big Data Application Provider role, indicating that data analytics and its implementation provide the value to Big Data stakeholders in both value chains. The term *provider* as part of the Big Data Application Provider and Big Data Framework Provider is there to indicate that those roles provide or implement specific activities and functions within the system. It does not designate a service model or business entity.

The DATA arrows in *Figure 2* show the flow of data between the system’s main roles. Data flows between the roles either physically (i.e., by value) or by providing its location and the means to access it (i.e., by reference). The SW arrows show transfer of software tools for processing of Big Data *in situ*. The Service Use arrows represent software programmable interfaces. While the main focus of the NBDRA is to represent the run-time environment, all three types of communications or transactions can happen in the configuration phase as well. Manual agreements (e.g., service-level agreements) and human interactions that may exist throughout the system are not shown in the NBDRA.

Detailed information on the NBDRA conceptual model is presented in the *NBDIF: Volume 6, Reference Architecture* document.

Prior to outlining the specific interfaces, general requirements are introduced and the interfaces are defined.

### 2.1 High-Level Requirements of the Interface Approach

This section focuses on the high-level requirements of the interface approach that are needed to implement the reference architecture depicted in *Figure 2*.

#### 2.1.1 Technology- and Vendor-Agnostic

Due to the many different tools, services, and infrastructures available in the general area of Big Data, an interface ought to be as vendor-independent as possible, while, at the same time, be able to leverage best practices. Hence, a methodology is needed that allows extension of interfaces to adapt and leverage existing approaches, but also allows the interfaces to provide merit in easy specifications that assist the formulation and definition of the NBDRA.
2.1.2 Support of Plug-In Compute Infrastructure

As Big Data is not just about hosting data, but about analyzing data, the interfaces provided herein must encapsulate a rich infrastructure environment that is used by data scientists. This includes the ability to integrate (or plug-in) various compute resources and services to provide the necessary compute power to analyze the data. These resources and services include the following:

- Access to hierarchy of compute resources from the laptop/desktop, servers, data clusters, and clouds;
- The ability to integrate special-purpose hardware such as graphics processing units (GPUs) and field-programmable gate arrays (FPGAs) that are used in accelerated analysis of data; and
- The integration of services including microservices that allow the analysis of the data by delegating them to hosted or dynamically deployed services on the infrastructure of choice.

2.1.3 Orchestration of Infrastructure and Services

From review of the use case collection, presented in the *NBDIF: Volume 3, Use Cases and General Requirements* document [4], the need arose to address the mechanism of preparing suitable infrastructures for various use cases. As not every infrastructure is suited for every use case, a custom infrastructure may be needed. As such, this document is not attempting to deliver a single deployed NBDRA, but allow the setup of an infrastructure that satisfies the particular use case. To achieve this task, it is necessary to provision software stacks and services while orchestrating their deployment and leveraging infrastructures. It is not the focus of this document to replace existing orchestration software and services, but provide an interface to them to leverage them as part of defining and creating the infrastructure. Various orchestration frameworks and services could therefore be leveraged, even as part of the same framework, and work in orchestrated fashion to achieve the goal of preparing an infrastructure suitable for one or more applications.

2.1.4 Orchestration of Big Data Applications and Experiments

The creation of the infrastructure suitable for Big Data applications provides the basic computing environment. However, Big Data applications may require the creation of sophisticated applications as part of interactive experiments to analyze and probe the data. For this purpose, the applications must be able to orchestrate and interact with experiments conducted on the data while assuring reproducibility and correctness of the data. For this purpose, a **System Orchestrator** (either the data scientists or a service acting on behalf of the data scientist) is used as the command center to interact on behalf of the Big Data Application Provider to orchestrate dataflow from Data Provider, carry out the Big Data application life cycle with the help of the Big Data Framework Provider, and enable the Data Consumer to consume Big Data processing results. An interface is needed to describe these interactions and
to allow leveraging of experiment management frameworks in scripted fashion. A customization of parameters is needed on several levels. On the highest level, application-motivated parameters are needed to drive the orchestration of the experiment. On lower levels, these high-level parameters may drive and create service-level agreements, augmented specifications, and parameters that could even lead to the orchestration of infrastructure and services to satisfy experiment needs.

2.1.5 Reusability

The interfaces provided must encourage reusability of the infrastructure, services, and experiments described by them. This includes (1) reusability of available analytics packages and services for adoption; (2) deployment of customizable analytics tools and services; and (3) operational adjustments that allow the services and infrastructure to be adapted while at the same time allowing for reproducible experiment execution.

2.1.6 Execution Workloads

One of the important aspects of distributed Big Data services can be that the data served is simply too big to be moved to a different location. Instead, an interface could allow the description and packaging of analytics algorithms, and potentially also tools, as a payload to a data service. This can be best achieved, not by sending the detailed execution, but by sending an interface description that describes how such an algorithm or tool can be created on the server and be executed under security considerations (integrated with authentication and authorization in mind).

2.1.7 Security and Privacy Fabric Requirements

Although the focus of this document is not security and privacy, which are documented in the *NBDIF: Volume 4, Security and Privacy* [5], the interfaces defined herein must be capable of integration into a secure reference architecture that supports secure execution, secure data transfer, and privacy. Consequently, the interfaces defined herein can be augmented with frameworks and solutions that provide such mechanisms. Thus, diverse requirement needs stemming from different use cases addressing security need to be distinguished. To contrast that the security requirements between applications can vary drastically, the following example is provided. Although many of the interfaces and their objects to support Big Data applications in physics are similar to those in healthcare, they differ in the integration of security interfaces and policies. While in physics the protection of data is less of an issue, it is a stringent requirement in healthcare. Thus, deriving architectural frameworks for both may use largely similar components, but addressing security issues will be very different. The security of interfaces may be addressed in other documents. In this document, they are considered an advanced use case showcasing that the validity of the specifications introduced here is preserved, even if security and privacy requirements differ vastly among application use cases.

2.2 Component-Specific Interface Requirements
This section summarizes the requirements for the interfaces of the NBDRA components. The five components are listed in Figure 2 and addressed in Section 2.2.1 (System Orchestrator Interface Requirements) and Section 2.2.4 (Big Data Application Provider to Big Data Framework Provider Interface) of this document. The five main functional components of the NBDRA represent the different technical roles within a Big Data system and are the following:

- **System Orchestrator**: Defines and integrates the required data application activities into an operational vertical system (see Section 2.2.1);
- **Data Provider**: Introduces new data or information feeds into the Big Data system (see Section 2.2.2);
- **Data Consumer**: Includes end users or other systems that use the results of the Big Data Application Provider (see Section 2.2.3);
- **Big Data Application Provider**: Executes a data life cycle to meet security and privacy requirements as well as System Orchestrator-defined requirements (see Section 2.2.4);
- **Big Data Framework Provider**: Establishes a computing framework in which to execute certain transformation applications while protecting the privacy and integrity of data (see Section 2.2.5); and
- **Big Data Application Provider to Framework Provider Interface**: Defines an interface between the application specification and the provider (see Section 2.2.6).

### 2.2.1 System Orchestrator Interface Requirements

The System Orchestrator role includes defining and integrating the required data application activities into an operational vertical system. Typically, the System Orchestrator involves a collection of more specific roles, performed by one or more actors, which manage and orchestrate the operation of the Big Data system. These actors may be human components, software components, or some combination of the two. The function of the System Orchestrator is to configure and manage the other components of the Big Data architecture to implement one or more workloads that the architecture is designed to execute. The workloads managed by the System Orchestrator may be assigning/provisioning framework components to individual physical or virtual nodes at the lower level, or providing a graphical user interface that supports the specification of workflows linking together multiple applications and components at the higher level. The System Orchestrator may also, through the Management Fabric, monitor the workloads and system to confirm that specific quality of service requirements is met for each workload, and may elastically assign and provision additional physical or virtual resources to meet workload requirements resulting from changes/surges in the data or number of users/transactions. The interface to the System Orchestrator must be capable of specifying the task of orchestration the deployment, configuration, and the execution of applications within the NBDRA. A simple vendor-neutral specification to coordinate the various parts either as simple parallel language tasks or as a workflow specification is needed to facilitate the overall coordination. Integration of existing tools and services into the System Orchestrator as extensible interfaces is desirable.
2.2.2 Data Provider Interface Requirements

The Data Provider role introduces new data or information feeds into the Big Data system for discovery, access, and transformation by the Big Data system. New data feeds are distinct from the data already in use by the system and residing in the various system repositories. Similar technologies can be used to access both new data feeds and existing data. The Data Provider actors can be anything from a sensor, to a human inputting data manually, to another Big Data system. Interfaces for data providers must be able to specify a data provider so it can be located by a data consumer. It also must include enough details to identify the services offered so they can be pragmatically reused by consumers. Interfaces to describe pipes and filters must be addressed.

2.2.3 Data Consumer Interface Requirements

Like the Data Provider, the role of Data Consumer within the NBDRA can be an actual end user or another system. In many ways, this role is the mirror image of the Data Provider, with the entire Big Data framework appearing like a Data Provider to the Data Consumer. The activities associated with the Data Consumer role include the following:

- Search and Retrieve,
- Download,
- Analyze Locally,
- Reporting,
- Visualization, and
- Data to Use for Their Own Processes.

The interface for the data consumer must be able to describe the consuming services and how they retrieve information or leverage data consumers.

2.2.4 Big Data Application Interface Provider Requirements

The Big Data Application Provider role executes a specific set of operations along the data life cycle to meet the requirements established by the System Orchestrator, as well as meeting security and privacy requirements. The Big Data Application Provider is the architecture component that encapsulates the business logic and functionality to be executed by the architecture. The interfaces to describe Big Data applications include interfaces for the various subcomponents including collections, preparation/curation, analytics, visualization, and access. Some of the interfaces used in these subcomponents can be reused from other interfaces, which are introduced in other sections of this document. Where appropriate, application-specific interfaces will be identified and examples provided with a focus on use cases as identified in the NBDIF: Volume 3, Use Cases and General Requirements.

2.2.4.1 Collection
In general, the collection activity of the Big Data Application Provider handles the interface with the Data Provider. This may be a general service, such as a file server or web server configured by the System Orchestrator to accept or perform specific collections of data, or it may be an application-specific service designed to pull data or receive pushes of data from the Data Provider. Since this activity is receiving data at a minimum, it must store/buffer the received data until it is persisted through the Big Data Framework Provider. This persistence need not be to physical media but may simply be to an in-memory queue or other service provided by the processing frameworks of the Big Data Framework Provider. The collection activity is likely where the extraction portion of the Extract, Transform, Load (ETL)/Extract, Load, Transform (ELT) cycle is performed. At the initial collection stage, sets of data (e.g., data records) of similar structure are collected (and combined), resulting in uniform security, policy, and other considerations. Initial metadata is created (e.g., subjects with keys are identified) to facilitate subsequent aggregation or look-up methods.

2.2.4.2 Preparation

The preparation activity is where the transformation portion of the ETL/ELT cycle is likely performed, although analytics activity will also likely perform advanced parts of the transformation. Tasks performed by this activity could include data validation (e.g., checksums/hashes, format checks), cleaning (e.g., eliminating bad records/fields), outlier removal, standardization, reformatting, or encapsulating. This activity is also where source data will frequently be persisted to archive storage in the Big Data Framework Provider and provenance data will be verified or attached/associated. Verification or attachment may include optimization of data through manipulations (e.g., deduplication) and indexing to optimize the analytics process. This activity may also aggregate data from different Data Providers, leveraging metadata keys to create an expanded and enhanced data set.

2.2.4.3 Analytics

The analytics activity of the Big Data Application Provider includes the encoding of the low-level business logic of the Big Data system (with higher-level business process logic being encoded by the System Orchestrator). The activity implements the techniques to extract knowledge from the data based on the requirements of the vertical application. The requirements specify the data processing algorithms to produce new insights that will address the technical goal. The analytics activity will leverage the processing frameworks to implement the associated logic. This typically involves the activity providing software that implements the analytic logic to the batch and/or streaming elements of the processing framework for execution. The messaging/communication framework of the Big Data Framework Provider may be used to pass data or control functions to the application logic running in the processing frameworks. The analytic logic may be broken up into multiple modules to be executed by the processing frameworks which communicate, through the messaging/communication framework, with each other and other functions instantiated by the Big Data Application Provider.
2.2.4.4 Visualization

The visualization activity of the Big Data Application Provider prepares elements of the processed data and the output of the analytic activity for presentation to the Data Consumer. The objective of this activity is to format and present data in such a way as to optimally communicate meaning and knowledge. The visualization preparation may involve producing a text-based report or rendering the analytic results as some form of graphic. The resulting output may be a static visualization and may simply be stored through the Big Data Framework Provider for later access. However, the visualization activity frequently interacts with the access activity, the analytics activity, and the Big Data Framework Provider (processing and platform) to provide interactive visualization of the data to the Data Consumer based on parameters provided to the access activity by the Data Consumer. The visualization activity may be completely application-implemented, leverage one or more application libraries, or may use specialized visualization processing frameworks within the Big Data Framework Provider.

2.2.4.5 Access

The access activity within the Big Data Application Provider is focused on the communication/interaction with the Data Consumer. Like the collection activity, the access activity may be a generic service such as a web server or application server that is configured by the System Orchestrator to handle specific requests from the Data Consumer. This activity would interface with the visualization and analytic activities to respond to requests from the Data Consumer (who may be a person) and uses the processing and platform frameworks to retrieve data to respond to Data Consumer requests. In addition, the access activity confirms that descriptive and administrative metadata and metadata schemes are captured and maintained for access by the Data Consumer and as data is transferred to the Data Consumer. The interface with the Data Consumer may be synchronous or asynchronous in nature and may use a pull or push paradigm for data transfer.

2.2.5 Big Data Provider Framework Interface Requirements

Data for Big Data applications are delivered through data providers. They can be either local providers, data contributed by a user, or distributed data providers, data on the Internet. This interface must be able to provide the following functionality:

- Interfaces to files,
- Interfaces to virtual data directories,
- Interfaces to data streams, and
- Interfaces to data filters.

2.2.5.1 Infrastructures Interface Requirements

This Big Data Framework Provider element provides all the resources necessary to host/run the
activities of the other components of the Big Data system. Typically, these resources consist of some combination of physical resources, which may host/support similar virtual resources. The NBDRA needs interfaces that can be used to deal with the underlying infrastructure to address networking, computing, and storage.

2.2.5.2 Platforms Interface Requirements

As part of the NBDRA platforms, interfaces are needed that can address platform needs and services for data organization, data distribution, indexed storage, and file systems.

2.2.5.3 Processing Interface Requirements

The processing frameworks for Big Data provide the necessary infrastructure software to support implementation of applications that can deal with the volume, velocity, variety, and variability of data. Processing frameworks define how the computation and processing of the data is organized. Big Data applications rely on various platforms and technologies to meet the challenges of scalable data analytics and operation. A requirement is the ability to interface easily with computing services that offer specific analytics services, batch processing capabilities, interactive analysis, and data streaming.

2.2.5.4 Crosscutting Interface Requirements

Several crosscutting interface requirements within the Big Data Framework Provider include messaging, communication, and resource management. Often these services may be hidden from explicit interface use as they are part of larger systems that expose higher-level functionality through their interfaces. However, such interfaces may also be exposed on a lower level in case finer-grained control is needed. The need for such crosscutting interface requirements will be extracted from the *NBDIF: Volume 3, Use Cases and General Requirements* document.

2.2.5.5 Messaging/Communications Frameworks

Messaging and communications frameworks have their roots in the High Performance Computing environments long popular in the scientific and research communities. Messaging/Communications Frameworks were developed to provide application programming interfaces (APIs) for the reliable queuing, transmission, and receipt of data.

2.2.5.6 Resource Management Framework

As Big Data systems have evolved and become more complex, and as businesses work to leverage limited computation and storage resources to address a broader range of applications and business challenges, the requirement to effectively manage those resources has grown significantly. While tools for resource management and *elastic computing* have expanded and
matured in response to the needs of cloud providers and virtualization technologies, Big Data introduces unique requirements for these tools. However, Big Data frameworks tend to fall more into a distributed computing paradigm, which presents additional challenges.

2.2.6 Big Data Application Provider to Big Data Framework Provider Interface

The Big Data Framework Provider typically consists of one or more hierarchically organized instances of the components in the NBDRA IT value chain (Figure 2). There is no requirement that all instances at a given level in the hierarchy be of the same technology. In fact, most Big Data implementations are hybrids that combine multiple technology approaches to provide flexibility or meet the complete range of requirements, which are driven from the Big Data Application Provider.
3 SPECIFICATION PARADIGM

This section summarizes the elementary specification paradigm.

3.1 HYBRID AND MULTIPLE FRAMEWORKS

To avoid vendor lock-in, Big Data systems must be able to deal with hybrid and multiple frameworks. This is not only true for Clouds, containers, DevOps, but also for components of the NBDRA.

3.2 DESIGN BY RESOURCE-ORIENTED ARCHITECTURE

A resource-oriented architecture represents a software architecture and programming paradigm for designing and developing software in the form of resources. It is often associated with 

**Representational State Transfer (REST)**

interfaces. The resources are software components which can be reused in concrete reference implementations. The service specification is conducted with OpenAPI, allowing use to provide it in a very general form that is independent of the framework or computer language in which the services can be specified. Note that OpenAPI defines services in REST The previous version only specified the resource objects.

3.3 DESIGN BY EXAMPLE

To accelerate discussion among the NBD-PWG members, contributors to this document are encouraged to also provide the NBD-PWG with examples.

3.4 VERSION MANAGEMENT

Previous work that shaped the current version of this volumes and are documented in GitHub [15] with prior versions of Volume 8 [16][17] and Cloudmesh [18] in support of the NIST Big Data Architecture Framework [1].

During the design phase and development period of each version of this document, enhancements are managed through GitHub and community contributions are managed via GitHub issues. This allows preservation of the history of this document. When a new version is ready, the version will be tagged in GitHub. Older versions will, through this process, also be available as historical documents. Discussions about objects in written form are communicated as GitHub issues.

3.5 INTERFACE COMPLIANCE

Due to the easy extensibility of the resource objects specified in this document and their
interfaces, it is important to introduce a terminology that allows the definition of interface compliancy. We define three levels of interface compliance as follows:

- **Full Compliance**: These are reference implementations that provide full compliance to the objects defined in this document. A version number is added to assure that the snapshot in time of the objects is associated with the version. A full compliant framework implements all objects.

- **Partial Compliance**: These are reference implementations that provide partial compliance to the objects defined in this document. A version number will be added to assure that the snapshot in time of the objects is associated with the version. This reference implementation implements a partial list of the objects and interfaces. A document is to be added that specifies the differences to a full compliant implementation.

- **Extended Compliance**: In addition to full and partial compliance additional resources can be identified while documenting additional resource objects and interfaces that are not included in the current specification. The extended compliance document can lead to additional improvements of the current specification.

### 3.6 Reference Implementations

Documents generated during a reference implementation can be forwarded to the Reference Architecture Subgroup for further discussion and for possible future modifications based on additional practical user feedback.
4 Specification

The specifications in this section are provided through an automated document creation process using the actual OpenAPI specifications yaml files as the source. All OpenAPI specifications located in the cloudmesh/cloudmesh-nist/spec/ directory in GitHub [19].

Limitations of the current implementation are as follows. It is a demonstration that showcases the generation of a fully functioning REST service based on the specifications provided in this document. However, it is expected that scalability, distribution of services, and other advanced options need to be addressed based on application requirements.

4.1 List of Specifications

The following table lists the current set of resource objects that are defined in this draft. Additional objects are also available in GitHub [19].

Table 1 shows the list of currently included specification in this version of the document.

<table>
<thead>
<tr>
<th>Service</th>
<th>Version</th>
<th>Date</th>
<th>Reference</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alias</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.6.1</td>
</tr>
<tr>
<td>Account</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.5.3</td>
</tr>
<tr>
<td>Containers</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.10.1</td>
</tr>
<tr>
<td>Database</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.7.3</td>
</tr>
<tr>
<td>Default</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.6.3</td>
</tr>
<tr>
<td>Deployment</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.15.1</td>
</tr>
<tr>
<td>File</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.7.1</td>
</tr>
<tr>
<td>Filter</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.14.2</td>
</tr>
<tr>
<td>Flavor</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.9.2</td>
</tr>
<tr>
<td>Image</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.9.1</td>
</tr>
<tr>
<td>MapReduce</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.11.1</td>
</tr>
<tr>
<td>Microservice</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.12.1</td>
</tr>
<tr>
<td>Nic</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.9.5</td>
</tr>
<tr>
<td>Network of Nodes</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.8.2</td>
</tr>
<tr>
<td>Organization</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.5.1</td>
</tr>
<tr>
<td>Public Key Store</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.5.4</td>
</tr>
<tr>
<td>Scheduler</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td></td>
<td>Section 4.8.4</td>
</tr>
<tr>
<td>Replica</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td>Section 4.7.2</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-------</td>
<td>------------</td>
<td>---------------</td>
<td></td>
</tr>
<tr>
<td>Reservation</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td>Section 4.13.1</td>
<td></td>
</tr>
<tr>
<td>Queue</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td>Section 4.8.3</td>
<td></td>
</tr>
<tr>
<td>Secgroup</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td>Section 4.9.4</td>
<td></td>
</tr>
<tr>
<td>Stream</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td>Section 4.14.1</td>
<td></td>
</tr>
<tr>
<td>Timestamp</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td>Section 4.4.1</td>
<td></td>
</tr>
<tr>
<td>User</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td>Section 4.5.2</td>
<td></td>
</tr>
<tr>
<td>Variables</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td>Section 4.6.2</td>
<td></td>
</tr>
<tr>
<td>Virtual Cluster</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td>Section 4.8.1</td>
<td></td>
</tr>
<tr>
<td>Virtual Directory</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td>Section 4.7.4</td>
<td></td>
</tr>
<tr>
<td>Virtual Machine</td>
<td>3.2.0</td>
<td>17-06-2019</td>
<td>Section 4.9.3</td>
<td></td>
</tr>
</tbody>
</table>

**Figure 3** shows the provider view of the specification resources.

**Figure 4** shows the resources view of the specification resources.
4.2 AUTHENTICATION

Mechanisms are not included in this specification to manage authentication to external services. However, the working group has shown multiple solutions to this as part of cloudmesh. This includes the possibility of a

- **Local configuration file**: A configuration file is managed locally to allow access to the clouds. It is the designer’s responsibility not to expose such credentials.
- **Session based authentication**: No passwords are stored in the configuration file and access is granted on a per session basis where the password needs to be entered.
- **Service based authentication**: The authentication is delegated to an external process. The service that acts on behalf of the user needs to have access to the appropriate cloud provider credentials

An example for a configuration file is provided at [20].

4.3 STATUS CODES AND ERROR RESPONSES

In case of an error or a successful response, the response header contains a HTTP code (see https://tools.ietf.org/html/rfc7231). The response body usually contains the following:

- The HTTP response code;

- An accompanying message for the HTTP response code; and

- A field or object where the error occurred.
### Table 1: HTTP Response Codes

<table>
<thead>
<tr>
<th>HTTP Response</th>
<th>Operation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 Ok</td>
<td>GET, PUT, DELETE</td>
<td>No error, operation successful.</td>
</tr>
<tr>
<td>201 Created</td>
<td>POST</td>
<td>Successful creation of a resource.</td>
</tr>
<tr>
<td>204 No Content</td>
<td>GET, PUT, DELETE</td>
<td>Successful but no content.</td>
</tr>
<tr>
<td>400 Bad Request</td>
<td>GET, POST, PUT, DELETE</td>
<td>The request could not be understood.</td>
</tr>
<tr>
<td>401 Unauthorized</td>
<td>GET, POST, PUT, DELETE</td>
<td>User must authorize.</td>
</tr>
<tr>
<td>403 Forbidden</td>
<td>GET, POST, PUT, DELETE</td>
<td>The request has been refused due to authorization failure.</td>
</tr>
<tr>
<td>404 Not Found</td>
<td>GET, POST, PUT, DELETE</td>
<td>The requested resource could not be found.</td>
</tr>
<tr>
<td>405 Not Allowed</td>
<td>GET, POST, PUT, DELETE</td>
<td>The method is not allowed on the resource.</td>
</tr>
<tr>
<td>500 Server Error</td>
<td>GET, POST PUT</td>
<td>Internal Server error.</td>
</tr>
</tbody>
</table>

In the specification such responses are indicated and if an simple response is returned the term **Message** is used.

**Resources**

### 4.4 TIMESTAMP

Timestamps can be used in conjunction with any server side implementation of the interfaces. It can be useful to return information about when a particular resource has been created, updated, or accessed. To simplify the specification in the document we have not explicitly listed that a timestamp is part of the resource, but we can assume it may be added as part of the service implementation. To obtain an example timestamp a simple get function is provided.

#### 4.4.1 Timestamp

Data often needs to be time stamped to indicate when it has been accessed, created, or modified. All objects defined in this document will have, in their final version, a timestamp. The date-time string is defined in [RFC3339](https://tools.ietf.org/html/rfc3339).
4.4.1.1 Schema Timestamp

Reference: ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessed</td>
<td>string(date-time)</td>
<td>The time stamp when the object was last accessed</td>
</tr>
<tr>
<td>created</td>
<td>string(date-time)</td>
<td>The time stamp when the object was created</td>
</tr>
<tr>
<td>modified</td>
<td>string(date-time)</td>
<td>The time stamp when the object was modified</td>
</tr>
</tbody>
</table>

4.4.1.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/timestamp</td>
<td>Returns the timestamp</td>
</tr>
</tbody>
</table>

4.4.1.2.1 /timestamp

4.4.1.2.1.1 GET /timestamp

Returns the timestamp

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The current time</td>
<td>string</td>
</tr>
</tbody>
</table>

4.4.1.3 timestamp.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Timestamp
  description: |
  Data often needs to be time stamped to indicate when it has been accessed, created, or modified. All objects defined in this document will have, in their final version, a timestamp. The date-time string is defined in [RFC3339](https://xml2rfc.ietf.org/public/rfc/html/rfc3339.html#anchor14).
  contact:
    name: NIST BDRA Interface Subgroup
  url: https://cloudmesh-community.github.io/nist/spec/
  license:
    name: Apache 2.0
    url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
  servers: |
```
4.5 IDENTITY

As part of services an identity often needs to be specified. In addition, such persons [21] are often part of groups. Thus, three important terms related to the identity are distinguished as follows:

- Organization: The information representing an Organization that manages a Big Data Service (Section 4.5.1)
- Group: A group that a person may belong to that is important to define access to services (included in Section 4.5.1)
- User: The information identifying the profile of a person (Section 4.5.2)

4.5.1 Organization

An important concept in many services is the management of a group of users in an organization. Within an organization we distinguish different groups of users. Groups can be used to characterize roles users can fulfill. Users can belong to multiple groups. Such groups can also be used to specify access rights to services.

4.5.1.1 Schema Organization

Reference: ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the organization</td>
</tr>
</tbody>
</table>
users array[User] List of users

4.5.1.2 Schema Group

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the group</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the group</td>
</tr>
<tr>
<td>users</td>
<td>array[string]</td>
<td>The user names that are members of the group</td>
</tr>
</tbody>
</table>

4.5.1.3 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/organization</td>
<td>Returns a list of organizations</td>
</tr>
<tr>
<td>get</td>
<td>/organization/{name}</td>
<td>Returns the named organization</td>
</tr>
<tr>
<td>put</td>
<td>/organization/{name}</td>
<td>Uploads an organization to the list of organizations</td>
</tr>
<tr>
<td>delete</td>
<td>/organization/{name}</td>
<td>Deletes the named organization</td>
</tr>
<tr>
<td>get</td>
<td>/organization/{name}/user</td>
<td>Returns all users of the organization</td>
</tr>
<tr>
<td>get</td>
<td>/organization/{name}/user/{user}</td>
<td>Returns the specific user of that organization</td>
</tr>
<tr>
<td>put</td>
<td>/organization/{name}/user/{user}</td>
<td>Updates or adds a user in the organization</td>
</tr>
<tr>
<td>delete</td>
<td>/organization/{name}/user/{user}</td>
<td>Delete an user in the organization</td>
</tr>
<tr>
<td>get</td>
<td>/organization/{name}/group/</td>
<td>Returns all group names</td>
</tr>
<tr>
<td>get</td>
<td>/organization/{name}/group/{group}</td>
<td>Returns the specific group of that organization</td>
</tr>
<tr>
<td>put</td>
<td>/organization/{name}/group/{group}</td>
<td>Updates or adds a group in the organization</td>
</tr>
<tr>
<td>delete</td>
<td>/organization/{name}/group/{group}</td>
<td>Delete a group in the organization</td>
</tr>
</tbody>
</table>
**PUT /organization/{name}/group/{group}/{user}**  
Updates or adds a user name to the group

**DELETE /organization/{name}/group/{group}/{user}**  
Deletes a user in the group

---

### 4.5.1.3.1 /organization

#### 4.5.1.3.1.1 GET /organization

Returns a list of all organizations

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of organizations</td>
<td>array[Organization]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

---

#### 4.5.1.3.2 /organization/{name}

#### 4.5.1.3.2.1 GET /organization/{name}

Returns an organization by name

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Retruning the information of the organization</td>
<td>Organization</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named organization could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the organization</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

---

#### 4.5.1.3.2.2 PUT /organization/{name}

Uploads an organization to the list of organizations

**Responses**
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Organization created or updated</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The organization could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The organization to be uploaded</td>
<td>True</td>
<td>Organization</td>
</tr>
</tbody>
</table>

4.5.1.3.2.3 DELETE /organization/{name}

Deletes an organization by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named organization could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the organization</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.1.3.3 /organization/{name}/user

4.5.1.3.3.1 GET /organization/{name}/user

Returns all users of the organization

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The organization</td>
<td>Organization</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the organization</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.1.3.4 /organization/{name}/user/{user}

4.5.1.3.4.1 GET /organization/{name}/user/{user}

Returns the specific user of that organization

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The user</td>
<td>User</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The organization or user could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the organization</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>user</td>
<td>path</td>
<td>The user name</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.1.3.4.2 PUT /organization/{name}/user/{user}

Updates or adds a user in the organization

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>User added sucessfully</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The organization or user could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the organization</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>
4.5.1.3.4.3 DELETE /organization/{name}/user/{user}

Delete an user in the organization

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The organization or user could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the organization</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>user</td>
<td>path</td>
<td>The user name</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.1.3.5.1 GET /organization/{name}/group/

Returns all group names

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the group</td>
<td>array[String]</td>
</tr>
<tr>
<td>400</td>
<td>No group found</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The organization or group could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the organization</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.1.3.6 /organization/{name}/group/{group}

4.5.1.3.6.1 GET /organization/{name}/group/{group}

Returns the specific group of that organization

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The group</td>
<td></td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The organization or group could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the organization</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>group</td>
<td>path</td>
<td>The group name</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.1.3.6.2 PUT /organization/{name}/group/{group}

Updates or adds a group in the organization

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Group added successfully</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The organization or group could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters
4.5.1.3.6.3 DELETE /organization/{name}/group/{group}

Delete a group in the organization

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The organization or group could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the organization</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>group</td>
<td>path</td>
<td>The group name</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.1.3.7 /organization/{name}/group/{group}/{user}

4.5.1.3.7.1 PUT /organization/{name}/group/{group}/{user}

Updates or adds a user name to the group

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>User added successfully</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The organization, group, or user could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the group</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>group</td>
<td>path</td>
<td>The group name</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>
The user name

4.5.1.3.7.2 DELETE /organization/{name}/group/{group}/{user}

Delete a user in the group

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The organization, group, or user could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the organization</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>group</td>
<td>path</td>
<td>The group name</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>user</td>
<td>path</td>
<td>The user name</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.1.4 organization.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Organization
  description: |
    An important concept in many services is the management of a group
    of users in an organization. Within an organization we distinguish
    different groups of users. Groups can be used to characterize roles
    users can fulfill. Users can belong to multiple groups. Such groups can
    also be used to specify access rights to services.

  contact:
    name: NIST BDRA Interface Subgroup
    url: https://cloudmesh-community.github.io/nist/spec/
  license:
    name: Apache 2.0
    url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3
paths:
  /organization:
    get:
      tags:
        - Organization
        description: Returns a list of organizations
      summary: Returns a list of all organizations
      operationId: cloudmesh.organization.list
      responses:
        '200':
          description: The list of organizations
          content:
            application/json:
              schema:
                type: array
```

- [user](#) path [The user name](#) True String
/organization/{name}:

get:
  
  tags:
  - Organization
  summary: Returns the named organization
  description: Returns an organization by name
  operationId: cloudmesh.organization.find_by_name
  parameters:
    - name: name
      in: path
      required: true
      schema:
        type: string
      description: The name of the organization
  responses:
    '200':
      description: Returning the information of the organization
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/Organization'
    '401':
      description: Not authorized
    '404':
      description: The named organization could not be found

put:
  
  tags:
  - Organization
  summary: Uploads an organization to the list of organizations
  description: Uploads an organization to the list of organizations
  operationId: cloudmesh.organization.add
  requestBody:
    description: The organization to be uploaded
    required: true
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/Organization'
  responses:
    '200':
      description: Organization created or updated
    '401':
      description: Not authorized
    '404':
      description: The organization could not be found

delete:
  
  tags:
  - Organization
  summary: Deletes the named organization
  description: Deletes an organization by name
  operationId: cloudmesh.organization.delete_by_name
  parameters:
    - name: name
      in: path
      required: true
      schema:
        type: string
      description: The name of the organization
  responses:
    '200':
      description: Deletion successful
    '401':
      description: Not authorized
    '404':
      description: The named organization could not be found

/organization/{name}/user:

get:
  
  tags:
  - Organization
  summary: Returns all users of the organization
  description: Returns all users of the organization
  operationId: cloudmesh.organization.user.list
  parameters:
    - name: name
      in: path
      required: true
      schema:
        type: string
      description: The name of the organization
  responses:
    '200':
      description: The organization
      content:
        application/json:
          schema:
$ref: "#/components/schemas/Organization"

'401':
  description: Not authorized
/organization/{name}/user/{user}:
  get:
    tags:
      - Organization
    summary: Returns the specific user of that organization
    description: Returns the specific user of that organization
    operationId: cloudmesh.organization.user.get_by_name
    parameters:
      - name: name
        in: path
        required: true
        schema:
          type: string
          description: The name of the organization
      - name: user
        in: path
        required: true
        schema:
          type: string
    responses:
      '200':
        description: The user
        content:
          application/json:
            $ref: "user.yaml#/components/schemas/User"
      '401':
        description: Not authorized
      '404':
        description: The organization or user could not be found
  put:
    tags:
      - Organization
    summary: Updates or adds a user in the organization
    description: Updates or adds a user in the organization
    operationId: cloudmesh.organization.user.add
    parameters:
      - name: name
        in: path
        required: true
        schema:
          type: string
          description: The name of the organization
      - name: user
        in: path
        required: true
        schema:
          type: string
    requestBody:
      description: The user to be uploaded
      required: true
      content:
        application/json:
          $ref: 'user.yaml#/components/schemas/User'
    responses:
      '200':
        description: User added successfully
      '401':
        description: Not authorized
      '404':
        description: The organization or user could not be found
  delete:
    tags:
      - Organization
    summary: Delete an user in the organization
    description: Delete an user in the organization
    operationId: cloudmesh.organization.user.delete
    parameters:
      - name: name
        in: path
        required: true
        schema:
          type: string
          description: The name of the organization
      - name: user
        in: path
        required: true
        schema:
          type: string
    responses:
      '200':
        description: Deletion successful
/organization/{name}/group/:  
get:  
tag: Organization  
summary: Returns all group names  
description: Returns all group names  
operationId: cloudmesh.organization.group.list  
parameters:  
- name: name  
in: path  
required: true  
schema:  
type: string  
description: The name of the organization  
responses:  
  '200':  
description: Returning the information of the group  
content:  
  application/json:  
    schema:  
      type: array  
      items:  
        type: string  
  '400':  
description: No group found  
  '401':  
description: Not authorized  
  '404':  
description: The organization or group could not be found  
/organization/{name}/group/{group}:  
get:  
tag: Organization  
summary: Returns the specific group of that organization  
description: Returns the specific group of that organization  
operationId: cloudmesh.organization.group.get_by_name  
parameters:  
- name: name  
in: path  
required: true  
schema:  
type: string  
description: The name of the organization  
- name: group  
in: path  
required: true  
schema:  
type: string  
description: The group name  
responses:  
  '200':  
description: The group  
content:  
  application/json:  
    schema:  
      $ref: 
        "/components/schemas/Group"  
  '401':  
description: Not authorized  
  '404':  
description: The organization or group could not be found  
put:  
tag: Organization  
summary: Updates or adds a group in the organization  
description: Updates or adds a group in the organization  
operationId: cloudmesh.organization.group.add  
parameters:  
- name: name  
in: path  
required: true  
schema:  
type: string  
description: The name of the group  
- name: group  
in: path  
required: true  
schema:  
type: string  
description: The group name  
responses:  
  '200':  
description: Group added successfully  
  '401':  
description: Not authorized  
  '404':  

delete:

tags:
  - Organization
summary: Delete a group in the organization
description: Delete a group in the organization
operationId: cloudmesh.organization.group.delete
parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
      description: The name of the organization
  - name: group
    in: path
    required: true
    schema:
      type: string
      description: The group name
responses:
  '200':
    description: Deletion successful
  '401':
    description: Not authorized
  '404':
    description: The organization or group could not be found

/organization/{name}/group/{group}/{user}:

put:

tags:
  - Organization
summary: Updates or adds a user name to the group
description: Updates or adds a user name to the group
operationId: cloudmesh.organization.group.user.add
parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
      description: The name of the group
  - name: group
    in: path
    required: true
    schema:
      type: string
      description: The group name
  - name: user
    in: path
    required: true
    schema:
      type: string
      description: The user name
responses:
  '200':
    description: User added successfully
  '401':
    description: Not authorized
  '404':
    description: The organization, group, or user could not be found

delete:

tags:
  - Organization
summary: Delete a user in the group
description: Delete a user in the group
operationId: cloudmesh.organization.group.delete.user
parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
      description: The name of the organization
  - name: group
    in: path
    required: true
    schema:
      type: string
      description: The group name
  - name: user
    in: path
    required: true
    schema:
      type: string
      description: The user name
responses:
  '200':
    description: Deletion successful
  '401':
    description: Not authorized
  '404':
    description: The organization, group, or user could not be found
4.5.2 User

Services need to specify which users have access to them. User information can be reused in other services and organized in a virtual organization. A user can be added to a named list of users within this organization. A group associated with the user can be used to augment users to be part of one or more groups.

4.5.2.1 Schema User

Reference: ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>string</td>
<td>The unique username associated with the user</td>
</tr>
<tr>
<td>firstname</td>
<td>string</td>
<td>The firstname of the user</td>
</tr>
<tr>
<td>lastname</td>
<td>string</td>
<td>The lastname of the user</td>
</tr>
<tr>
<td>email</td>
<td>string</td>
<td>The email of the user</td>
</tr>
<tr>
<td>comment</td>
<td>string</td>
<td>A comment regarding the user</td>
</tr>
<tr>
<td>publickey</td>
<td>string</td>
<td>The public key of the user</td>
</tr>
</tbody>
</table>

4.5.2.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/user</td>
<td>Returns a list of users</td>
</tr>
</tbody>
</table>
get /user/{name}  Returns the named user
put /user/{name}  Uploads a user to the list of users
delete /user/{name} Deletes the named user

4.5.2.2.1 /user

4.5.2.2.1.1 GET /user

Returns a list of all users

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of users</td>
<td>array[User]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.2.2.2 /user/{name}

4.5.2.2.2.1 GET /user/{name}

Returns an user by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the user</td>
<td>User</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named user could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the user</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.2.2.2.2 PUT /user/{name}

Uploads a user to the list of users

Responses
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>User updated</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named user could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The user to be uploaded</td>
<td>True</td>
<td>User</td>
</tr>
</tbody>
</table>

4.5.2.2.2.3 DELETE /user/(name)

Deletes an user by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named user could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the user</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.2.3 user.yaml

```yaml
openapi: "3.0.2"
info:
  version: "3.2.0"
x-date: 17-06-2019
x-status: defined
title: User
description: |- Services need to specify which users have access to them. User information can be reused in other services and organized in a virtual organization. A user can be added to a named list of users within this organization. A group associated with the user can be used to augment users to be part of one or more groups.
contact:
  name: Cloudmesh User
url: https://cloudmesh-community.github.io/nist/spec/
license:
  name: Apache 2.0
url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3
```
paths:
  /user:
    get:
      tags:
        - User
      summary: Returns a list of users
      description: Returns a list of all users
      operationId: cloudmesh.user.list
      responses:
        '200':
          description: The list of users
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/User'
        '401':
          description: Not authorized

  /user/{name}:
    get:
      tags:
        - User
      summary: Returns the named user
      description: Returns an user by name
      operationId: cloudmesh.user.find_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the user
      responses:
        '200':
          description: Returning the information of the user
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/User'
        '401':
          description: Not authorized
        '404':
          description: The named user could not be found

  put:
    tags:
      - User
      summary: Uploads a user to the list of users
      description: Uploads a user to the list of users
      operationId: cloudmesh.user.add
      requestBody:
        description: The user to be uploaded
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/User'
      responses:
        '200':
          description: User updated
        '401':
          description: Not authorized
        '404':
          description: The named user could not be found

  delete:
    tags:
      - User
      summary: Deletes the named user
      description: Deletes an user by name
      operationId: cloudmesh.user.delete_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the user
      responses:
        '200':
          description: Deletion successful
        '401':
          description: Not authorized
        '404':
          description: The named user could not be found

components:
schemas:
  User:
    type: object
    properties:
4.5.3 Account

To charge the use of resources accounting can be used. Accounting can be implemented on a variety of resources, such as users, groups or organizations. It is up to the implementer to provide rules and cost for it. If needed multiple accounting resources can be implemented.

4.5.3.1 Schema Account

Reference: ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>name of account</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>the purpose of the account</td>
</tr>
<tr>
<td>charge</td>
<td>integer</td>
<td>The current charge of the account</td>
</tr>
<tr>
<td>unit</td>
<td>string</td>
<td>the unit in which the account is charged and the charge value is stored</td>
</tr>
</tbody>
</table>

4.5.3.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/account</td>
<td>Returns the accounts</td>
</tr>
<tr>
<td>get</td>
<td>/account/{name}</td>
<td>Returns the named account</td>
</tr>
<tr>
<td>put</td>
<td>/account/{name}</td>
<td>Set the value of a account</td>
</tr>
<tr>
<td>delete</td>
<td>/account/{name}</td>
<td>Deletes the named account</td>
</tr>
</tbody>
</table>

4.5.3.2.1 /account

4.5.3.2.1.1 GET /account

Returns the accounts
### Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of accounts</td>
<td>array[Account]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.3.2.2 /account/{name}

4.5.3.2.2.1 GET /account/{name}

Returns the named account

### Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the account</td>
<td>Account</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named account could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the account</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.3.2.2.2 PUT /account/{name}

Set the value of the named account

### Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Account updated or created</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error updating account</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

### Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The account and its value</td>
<td>True</td>
<td>Account</td>
</tr>
</tbody>
</table>
4.5.3.2.2.3 DELETE /account/{name}

Deletes a account by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named account could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the account</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.3.3 account.yaml

```{include=./spec/account.yaml}

4.5.4 Public Key Store

Many services and frameworks use Secure Shell (SSH) keys to authenticate. This service allows the convenient storage of the public keys.

4.5.4.1 Schema Key

Reference: 🌿

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the public key</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The value of the public key</td>
</tr>
<tr>
<td>kind</td>
<td>string</td>
<td>The key kind such as rsa, dsa</td>
</tr>
<tr>
<td>group</td>
<td>string</td>
<td>An optional group name allowing to group keys to create custom key groups within the public key store</td>
</tr>
<tr>
<td>comment</td>
<td>string</td>
<td>A comment for the public key</td>
</tr>
<tr>
<td>uri</td>
<td>string</td>
<td>The uri of the public key if any</td>
</tr>
</tbody>
</table>
4.5.4.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/key</td>
<td>Returns a list of keys</td>
</tr>
<tr>
<td>get</td>
<td>/key/{name}</td>
<td>Returns the named key</td>
</tr>
<tr>
<td>put</td>
<td>/key/{name}</td>
<td>Set a key</td>
</tr>
<tr>
<td>delete</td>
<td>/key/{name}</td>
<td>Deletes the named key</td>
</tr>
</tbody>
</table>

4.5.4.2.1 /key

4.5.4.2.1.1 GET /key

Returns a list of all keys

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of keys</td>
<td>array[Key]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.4.2.2 /key/{name}

4.5.4.2.2.1 GET /key/{name}

Returns a key by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the key</td>
<td>Key</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named key could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
</table>
4.5.4.2.2 PUT /key/{name}

Sets the named key

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Key updated</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named key could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The new key to create</td>
<td>True</td>
<td>Key</td>
</tr>
</tbody>
</table>

4.5.4.2.3 DELETE /key/{name}

Deletes a key by name

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named key could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the key</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.5.4.3 publickeystore.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Public Key Store
```
Many services and frameworks use Secure Shell (SSH) keys to authenticate. This service allows the convenient storage of the public keys.


contact:
  name: NIST BDRA Interface Subgroup
  url: https://cloudmesh-community.github.io/nist

license:
  name: Apache 2.0
  url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt

servers:
  - url: /cloudmesh/v3

paths:
  /key:
    get:
      tags: [- Key]
      summary: Returns a list of keys
      description: Returns a list of all keys
      operationId: cloudmesh.key.list
      responses:
        '200':
          description: The list of keys
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Key'
        '401':
          description: Not authorized
  /key/{name}:
    get:
      tags: [- Key]
      summary: Returns the named key
      description: Returns a key by name
      operationId: cloudmesh.key.find_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the key
      responses:
        '200':
          description: Returning the information of the key
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Key'
        '401':
          description: Not authorized
        '404':
          description: The named key could not be found
    put:
      tags: [- Key]
      summary: Set a key
      description: Sets the named key
      operationId: cloudmesh.key.add
      requestBody:
        description: The new key to create
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Key'
      responses:
        '200':
          description: Key updated
        '401':
          description: Not authorized
        '404':
          description: The named key could not be found
    delete:
      tags: [- Key]
      summary: Deletes the named key
      description: Deletes a key by name
      operationId: cloudmesh.key.delete_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
4.6 Variable, Default, and Alias

4.6.1 Alias

Often a user has the desire to create a custom name for an object. An alias allows to do that while while assosication auser defined name or alias to a previously used name. The aliases could be shared with other users. A name could have one or more aliases.

4.6.1.1 Schema Alias

Reference: ☁️

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the alias</td>
</tr>
<tr>
<td>source</td>
<td>string</td>
<td>The original unique object name</td>
</tr>
</tbody>
</table>

4.6.1.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/alias</td>
<td>Returns a list of aliases</td>
</tr>
<tr>
<td>get</td>
<td>/alias/{name}</td>
<td>Returns the named alias</td>
</tr>
</tbody>
</table>
put /alias/{name}  Set an alias
delete /alias/{name}  Deletes the named alias

4.6.1.2.1 /alias

4.6.1.2.1.1 GET /alias

Returns a list of all aliases

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of aliases</td>
<td>array[Alias]</td>
</tr>
<tr>
<td>400</td>
<td>No alias found</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.6.1.2.2 /alias/{name}

4.6.1.2.2.1 GET /alias/{name}

Returns an alias by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the alias</td>
<td>Alias</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named alias could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the alias</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.6.1.2.2.2 PUT /alias/{name}

Sets the named alias

Responses
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Alias updated or created</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The new alias to create</td>
<td>True</td>
<td>Alias</td>
</tr>
</tbody>
</table>

4.6.1.2.3 DELETE /alias/{name}

Deletes an alias by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named alias could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the alias</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.6.1.3 alias.yaml

```yaml
openapi: '3.0.2'
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Alias
  description: |
    Often a user has the desire to create a custom name for an object. An alias allows to do that while while assosication auser defined name or "alias" to a previously used name. The aliases could be shared with other users. A name could have one or more aliases.

contact:
  name: NIST BDRA Interface Subgroup
  url: https://cloudmesh-community.github.io/nist/spec/
license:
  name: Apache 2.0
  url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3
paths:
  /alias:
    get:
tags:
- Alias

summary: Returns a list of aliases

description: Returns a list of all aliases

operationId: cloudmesh.alias.list

responses:
'200':
  description: The list of aliases
  content:
    application/json:
      schema:
        type: array
        items:
          $ref: '#/components/schemas/Alias'

'400':
  description: No alias found

'401':
  description: Not authorized

(alias/{name}):

tags:
- Alias

summary: Returns the named alias

description: Returns an alias by name

operationId: cloudmesh.alias.find_by_name

parameters:
- name: name
  in: path
  required: true
  schema:
    type: string
  description: The name of the alias

responses:
'200':
  description: Returning the information of the alias
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/Alias'

'401':
  description: Not authorized

'404':
  description: The named alias could not be found

put:

tags:
- Alias

summary: Set an alias

description: Sets the named alias

operationId: cloudmesh.alias.add

requestBody:
  description: The new alias to create
  required: true
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/Alias'

responses:
'200':
  description: Alias updated or created

'401':
  description: Not authorized

delete:

tags:
- Alias

summary: Deletes the named alias

description: Deletes an alias by name

operationId: cloudmesh.alias.delete_by_name

parameters:
- name: name
  in: path
  required: true
  schema:
    type: string
  description: The name of the alias

responses:
'200':
  description: Deletion successful

'401':
  description: Not authorized

'404':
  description: The named alias could not be found

components:

schemas:

Alias:
  type: object
  description: the alias
  properties:
    name:
      type: string
4.6.2 Variables

Variables are a simple string key value storage to store simple values. Each variable can have a datatype, so that it can be used for serialization into other formats. Internally they are stored as strings.

4.6.2.1 Schema Variable

Reference:

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the variable</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>Value of the variable</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the variable</td>
</tr>
<tr>
<td>datatype</td>
<td>string</td>
<td>The data type of the variable which can be used for serialization</td>
</tr>
</tbody>
</table>

4.6.2.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/variable</td>
<td>Returns the variables</td>
</tr>
<tr>
<td>get</td>
<td>/variable/{name}</td>
<td>Returns the named variable</td>
</tr>
<tr>
<td>put</td>
<td>/variable/{name}</td>
<td>Set the value of a variable</td>
</tr>
<tr>
<td>delete</td>
<td>/variable/{name}</td>
<td>Deletes the named variable</td>
</tr>
</tbody>
</table>

4.6.2.2.1 /variable

4.6.2.2.1.1 GET /variable

Returns the variables

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of variables</td>
<td>array[Variable]</td>
</tr>
<tr>
<td>400</td>
<td>No variable found</td>
<td>String</td>
</tr>
</tbody>
</table>
4.6.2.2 /variable/{name}

4.6.2.2.1 GET /variable/{name}

Returns the named variable

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the variable</td>
<td>Variable</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named variable could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the variable</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.6.2.2.2 PUT /variable/{name}

Set the value of the named variable

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Variable updated or created</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error updating variable</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The variable and its value</td>
<td>True</td>
<td>Variable</td>
</tr>
</tbody>
</table>

4.6.2.2.3 DELETE /variable/{name}

Deletes a variable by name
Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named variable could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the variable</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.6.2.3 variables.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Variables
  description: Variables are a simple string key value storage to store simple values. Each variable can have a datatype, so that it can be used for serialization into other formats. Internally they are stored as strings.
  contact:
    name: NIST BDRA Interface Subgroup
    url: https://cloudmesh-community.github.io/nist/spec/
  license:
    name: Apache 2.0
    url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3
paths:
  /variable:
    get:
      tags:
        - Variable
      summary: Returns a the variables
      description: Returns the variables
      operationId: cloudmesh.variable.list
      responses:
        '200':
          description: The list of variables
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Variable'
        '400':
          description: No variable found
  /variable/{name}:
    get:
      tags:
        - Variable
      summary: Returns the named variable
      description: Returns the named variable
      operationId: cloudmesh.variable.find_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
          description: The name of the variable
      responses:
        '200':
```


**4.6.3 Default**

A default is a special variable that has a context associated with it. This allows one to define values that can be easily retrieved based on the associated context. For example, a default could be the image name for a cloud where the context is defined by the cloud name.

**4.6.3.1 Schema Default**

Reference: ☁️
<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the default</td>
</tr>
<tr>
<td>value</td>
<td>string</td>
<td>The value of the default</td>
</tr>
<tr>
<td>context</td>
<td>string</td>
<td>The context of the default</td>
</tr>
</tbody>
</table>

### 4.6.3.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/default</td>
<td>Returns a list of defaults</td>
</tr>
<tr>
<td>get</td>
<td>/default/{name}</td>
<td>Returns the named default</td>
</tr>
<tr>
<td>put</td>
<td>/default/{name}</td>
<td>Set a default</td>
</tr>
<tr>
<td>delete</td>
<td>/default/{name}</td>
<td>Deletes the named default</td>
</tr>
</tbody>
</table>

#### 4.6.3.2.1 /default

**4.6.3.2.1.1 GET /default**

Returns a list of all defaults

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of defaults</td>
<td>array[Default]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

#### 4.6.3.2.2 /default/{name}

**4.6.3.2.2.1 GET /default/{name}**

Returns a default by name

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the default</td>
<td>Default</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named default could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the default</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.6.3.2.2 PUT /default/{name}

Sets the named default

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Default updated or created</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The new default to create</td>
<td>True</td>
<td>Default</td>
</tr>
</tbody>
</table>

4.6.3.2.3 DELETE /default/{name}

Deletes a default by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named default could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the default</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.6.3.3 default.yaml

openapi: "3.0.2"
info:
A default is a special variable that has a context associated with it. This allows one to define values that can be easily retrieved based on the associated context. For example, a default could be the image name for a cloud where the context is defined by the cloud name.
### 4.7 Data Management

#### 4.7.1 Filestore

A file store is a resource allowing storage of data as a traditional file. A file store can contain any number of files with additional attributes describing the file. A file store is located on the physical server. It contains access to the content of the file. This contrasts virtual directories that are just pointers to files, which could include files located in different file stores. A virtual directory also does not contain the content of the file, but just a pointer where to find the file.

#### 4.7.1.1 Schema File

**Reference:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the file</td>
</tr>
<tr>
<td>endpoint</td>
<td>string</td>
<td>The location of the file</td>
</tr>
<tr>
<td>checksum</td>
<td>string</td>
<td>The checksum of the file</td>
</tr>
<tr>
<td>size</td>
<td>integer</td>
<td>The size of the file in byte</td>
</tr>
<tr>
<td>content</td>
<td>string(binary)</td>
<td>the content of the file</td>
</tr>
</tbody>
</table>

#### 4.7.1.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>Path</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>get</td>
<td>/file</td>
<td>Returns a list of files in the file store</td>
</tr>
<tr>
<td>get</td>
<td>/file/{name}</td>
<td>Returns the named file in the file store</td>
</tr>
<tr>
<td>put</td>
<td>/file/{name}</td>
<td>Uploads a file to the list of files in the file store</td>
</tr>
<tr>
<td>delete</td>
<td>/file/{name}</td>
<td>Deletes the named file in the file store</td>
</tr>
</tbody>
</table>

### 4.7.1.2.1 /file

#### 4.7.1.2.1.1 GET /file

Returns a list of all files

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of files</td>
<td>array[File]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

### 4.7.1.2.2 /file/{name}

#### 4.7.1.2.2.1 GET /file/{name}

Returns an file by name in the file store

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the file store</td>
<td>File</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named file could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the file</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

### 4.7.1.2.2.2 PUT /file/{name}

Uploads a file to the list of files in the file store
### Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>File updated or created</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

**Request Body**

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The file to be uploaded</td>
<td>True</td>
<td>File</td>
</tr>
</tbody>
</table>

**4.7.1.2.2.3 DELETE /file/{name}**

Deletes an file by name

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named file could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the file</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

**4.7.1.3 filestore.yaml**

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: File
  description: |
  A file store is a resource allowing storage of data as a traditional file. A file store can contain any number of files with additional attributes describing the file. A file store is located on the physical server. It contains access to the content of the file. This contrasts virtual directories that are just pointers to files, which could include files located in different file stores. A virtual directory also does not contain the content of the file, but just a pointer where to find the file.
  contact:
    name: NIST BDRA Interface Subgroup
    url: https://cloudmesh-community.github.io/nist/spec/
  license:
    name: Apache 2.0
```
servers:
  - url: /cloudmesh/v3
paths:
  /file:
    get:
      tags:
        - File
      summary: Returns a list of files in the file store
      description: Returns a list of all files
      operationId: cloudmesh.file.list
      responses:
        '200':
          description: The list of files
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/File'
        '401':
          description: Not authorized
  /file/{name}:
    get:
      tags:
        - File
      summary: Returns the named file in the file store
      description: Returns an file by name in the file store
      operationId: cloudmesh.file.find_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the file
      responses:
        '200':
          description: Returning the information of the file store
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/File'
        '401':
          description: Not authorized
        '404':
          description: The named file could not be found
    put:
      tags:
        - File
      summary: Uploads a file to the list of files in the file store
      description: Uploads a file to the list of files in the file store
      operationId: cloudmesh.file.add
      requestBody:
        description: The file to be uploaded
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/File'
      responses:
        '200':
          description: File updated or created
        '401':
          description: Not authorized
        '404':
          description: The named file could not be found
    delete:
      tags:
        - File
      summary: Deletes the named file in the file store
      description: Deletes an file by name
      operationId: cloudmesh.file.delete_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the file
      responses:
        '200':
          description: Deletion successful
        '401':
          description: Not authorized
        '404':
          description: The named file could not be found
components:
  schemas:
    File:
      type: object
      description: an object representing a file
4.7.2 Replica

In many distributed systems, it is important that a file can be replicated among different systems to provide faster access. It is important to provide a mechanism to trace the pedigree of the file while pointing to its original source. A replica will point to a file in a file store and store the contents in the file store instead of the replica. The replica is just a pointer.

4.7.2.1 Schema Replica

**Reference:** ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the replica</td>
</tr>
<tr>
<td>filename</td>
<td>string</td>
<td>The original filename</td>
</tr>
<tr>
<td>endpoint</td>
<td>string</td>
<td>The location of the file</td>
</tr>
<tr>
<td>checksum</td>
<td>string</td>
<td>The checksum of the file</td>
</tr>
<tr>
<td>size</td>
<td>integer</td>
<td>The size of the file in byte</td>
</tr>
</tbody>
</table>

4.7.2.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/replica</td>
<td>Returns a list of replicas</td>
</tr>
<tr>
<td>get</td>
<td>/replica/{name}</td>
<td>Returns the named replica</td>
</tr>
<tr>
<td>put</td>
<td>/replica/{name}</td>
<td>Uploads a replica to the list of replicas</td>
</tr>
<tr>
<td>delete</td>
<td>/replica/{name}</td>
<td>Deletes the named replica</td>
</tr>
</tbody>
</table>

4.7.2.2.1 /replica

4.7.2.2.1.1 GET /replica
Returns a list of all replicas

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of replicas</td>
<td>array[Replica]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

`4.7.2.2.2 /replica/{name}`

`4.7.2.2.1 GET /replica/{name}`

Returns an replica by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the replica</td>
<td>Replica</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named replica could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the replica</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

`4.7.2.2.2.2 PUT /replica/{name}`

Uploads a replica to the list of replicas

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Replica updated or created</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body
4.7.2.2.2.3 DELETE /replica/{name}

Deletes an replica by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named replica could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the replica</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.7.2.3 replica.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Replica
  description: |
    In many distributed systems, it is important that a file can be
    replicated among different systems to provide faster access. It is
    important to provide a mechanism to trace the pedigree of the file
    while pointing to its original source. A replica will point to a file in
    a file store and store the contents in the file store instead of the
    replica. The replica is just a pointer.


contact:
  name: NIST BDRA Interface Subgroup
  url: https://cloudmesh-community.github.io/nist/spec/
  license:
    name: Apache 2.0
    url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt

servers:
- url: /cloudmesh/v3
  paths:
    /replica:
      get:
        tags: [Replica]
        summary: Returns a list of replicas
        description: Returns a list of all replicas
        operationId: cloudsmesh.replica.list
        responses:
          '200':
            description: The list of replicas
            content:
              application/json:
                schema:
                  type: array
                  items:
                    $ref: '#/components/schemas/Replica'
          '401':
            description: Not authorized
```

Body | The replica to be uploaded | True | Replica
/replica/{name}:
  get:
    tags:
      - Replica
    summary: Returns the named replica
    description: Returns an replica by name
    operationId: cloudmesh.replica.find_by_name
    parameters:
      - name: name
        in: path
        required: true
        schema:
          type: string
          description: The name of the replica
    responses:
      '200':
        description: Returning the information of the replica
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Replica'
      '401':
        description: Not authorized
      '404':
        description: The named replica could not be found
  put:
    tags:
      - Replica
    summary: Uploads a replica to the list of replicas
    description: Uploads a replica to the list of replicas
    operationId: cloudmesh.replica.add
    requestBody:
      description: The replica to be uploaded
      required: true
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/Replica'
    responses:
      '200':
        description: Replica updated or created
      '401':
        description: Not authorized
      '404':
        description: The named replica could not be found
  delete:
    tags:
      - Replica
    summary: Deletes the named replica
    description: Deletes an replica by name
    operationId: cloudmesh.replica.delete_by_name
    parameters:
      - name: name
        in: path
        required: true
        schema:
          type: string
          description: The name of the replica
    responses:
      '200':
        description: Deletion successful
      '401':
        description: Not authorized
      '404':
        description: The named replica could not be found

components:
  schemas:
    Replica:
      type: object
      description: An entry representing a file replica record
      properties:
        name:
          type: string
          description: The name of the replica
        filename:
          type: string
          description: The original filename
        endpoint:
          type: string
          description: The location of the file
        checksum:
          type: string
          description: The checksum of the file
        size:
          type: integer
          description: The size of the file in byte

4.7.3 Database
The database specification allows to register a database and perform elementary operations to use this database. We distinguish actions related to the registration, the adding of a schema, the insertion of data and the query of data. The database is defined by a name an endpoint (e.g., host, port), and a protocol used (e.g., SQL, MongoDB, graphgl, and others).

4.7.3.1 Schema Database

**Reference: ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the database</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the database</td>
</tr>
<tr>
<td>endpoint</td>
<td>string</td>
<td>Endpoint of the database</td>
</tr>
<tr>
<td>kind</td>
<td>string</td>
<td>the kind of the database</td>
</tr>
</tbody>
</table>

4.7.3.2 Schema Schema

**Reference: ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the database</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the database</td>
</tr>
<tr>
<td>kind</td>
<td>string</td>
<td>The kind of the definition</td>
</tr>
<tr>
<td>content</td>
<td>string</td>
<td>The schema associated with the table or collection</td>
</tr>
</tbody>
</table>

4.7.3.3 Schema Record

**Reference: ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>status</td>
<td>string</td>
<td>The status of the return</td>
</tr>
<tr>
<td>result</td>
<td>string</td>
<td>The result of the query in json string format</td>
</tr>
</tbody>
</table>

4.7.3.4 Schema Query

**Reference: ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
</table>
status string The query string

4.7.3.5 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/database</td>
<td>Returns all databases</td>
</tr>
<tr>
<td>get</td>
<td>/database/{name}/schema</td>
<td>Get the list of the schema</td>
</tr>
<tr>
<td>put</td>
<td>/database/{name}/schema</td>
<td>Upload a schema</td>
</tr>
<tr>
<td>delete</td>
<td>/database/{name}/schema</td>
<td>Deletes a database from the list of databases</td>
</tr>
<tr>
<td>get</td>
<td>/database/{name}</td>
<td>Query the named database</td>
</tr>
<tr>
<td>put</td>
<td>/database/{name}</td>
<td>add data to the table or collection</td>
</tr>
<tr>
<td>delete</td>
<td>/database/{name}</td>
<td>Delete the objects matching the query</td>
</tr>
</tbody>
</table>

4.7.3.5.1 /database

4.7.3.5.1.1 GET /database

Returns all databases

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>List of databases</td>
<td>array[Database]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>Named database not found</td>
<td>String</td>
</tr>
</tbody>
</table>

4.7.3.5.2 /database/{name}/schema

4.7.3.5.2.1 GET /database/{name}/schema

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>successfully returned the schema</td>
<td>array[Schema]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>Named database not found</td>
<td>String</td>
</tr>
</tbody>
</table>
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>Name of the schema</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.7.3.5.2.2 PUT /database/{name}/schema

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>successfully returned the list</td>
<td>Schema</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>Named database not found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>Name of the database</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.7.3.5.2.3 DELETE /database/{name}/schema

Deletes a database from the list of databases

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>Named database not found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>Name of the database</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.7.3.5.3 /database/{name}

4.7.3.5.3.1 GET /database/{name}
Query the named database

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successfull query</td>
<td>array[Record]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>Named database not found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>Name of the database</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>query</td>
<td>query</td>
<td>Database Query</td>
<td>True</td>
<td>Query</td>
</tr>
</tbody>
</table>

4.7.3.5.3.2 PUT /database/{name}

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>successfully uploaded</td>
<td>Record</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>Named database not found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>Name of the database</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>Record to be uploaded</td>
<td>True</td>
<td>Record</td>
</tr>
</tbody>
</table>

4.7.3.5.3.3 DELETE /database/{name}

Responses
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Successfull query</td>
<td>array[Record]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>Named database not found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>Name of the database</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>query</td>
<td>query</td>
<td>Database Query</td>
<td>True</td>
<td>Query</td>
</tr>
</tbody>
</table>

4.7.3.6 database.yaml

openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Database
  description: |

  The database specification allows to register a database and perform elementary operations to use this database. We distinguish actions related to the registration, the adding of a schema, the insertion of data and the query of data. The database is defined by a name an endpoint (e.g., host, port), and a protocol used (e.g., SQL, MongoDB, graphgl, and others).

contact:
  name: NIST BDRA Interface Subgroup
  url: https://cloudmesh-community.github.io/nist/spec/
license:
  name: Apache 2.0
  url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3
paths:
  /database:
    get:
      tags:
        - "Database Registry"
      summary: Returns all databases
      description: Returns all databases
      operationId: cloudmesh.database.get
      responses:
        '200':
          description: List of databases
          content:
            application/json:
              schema: array
              items:
                $ref: "#/components/schemas/Database"
        '401':
          description: Not authorized
        '404':
          description: Named database not found
  /database/{name}/schema:
    get:
      tags:
        - "Database Schema"
      summary: Get the list of the schema
      description: ""
      operationId: "cloudmesh.database.get.schema"
      parameters:
        - name: name
          in: path
          required: true
          schema:
type: string
responses:
  '200':
    description: "successfully returned the schema"
    content:
      application/json:
        schema:
          type: array
          items:
            $ref: "/components/schemas/Schema"
  '401':
    description: Not authorized
  '404':
    description: Named database not found
put:
  tags:
  - "Database Schema"
  summary: "Upload a schema"
  description: ""
  operationId: "cloudmesh.database.put.schema"
  parameters:
    - name: name
      description: Name of the database
      in: path
      required: true
      schema:
        type: string
  responses:
    '200':
      description: "successfully returned the list"
      content:
        application/json:
          schema:
            $ref: "/components/schemas/Schema"
    '401':
      description: Not authorized
    '404':
      description: Named database not found
delete:
  tags:
  - "Database Registry"
  summary: Deletes a database from the list of databases
  description: Deletes a database from the list of databases
  operationId: "cloudmesh.database.delete"
  parameters:
    - name: name
      description: Name of the database
      in: path
      required: true
      schema:
        type: string
  responses:
    '200':
      description: Deletion successful
    '401':
      description: Not authorized
    '404':
      description: Named database not found
/database/{name}:
get:
  tags:
  - "Database Data"
  summary: Query the named database
  description: Query the named database
  operationId: "cloudmesh.database.data.get"
  parameters:
    - name: name
      description: Name of the database
      in: path
      required: true
      schema:
        type: string
    - in: query
      name: query
      description: Database Query
      required: true
      schema:
        $ref: "/components/schemas/Query"
  responses:
    '200':
      description: Successful query
      content:
        application/json:
          schema:
            type: array
            items:
              $ref: "/components/schemas/Record"
    '401':
      description: Not authorized
```json
'404':
    description: Named database not found

put:
  tags:
  - "Database Data"
  summary: "add data to the table or collection"
  description: ""
  operationId: "cloudmesh.database.data.put"
  parameters:
    - name: name
      description: Name of the database
      in: path
      required: true
      schema:
        type: string
  requestBody:
    description: Record to be uploaded
    required: true
    content:
      application/json:
        schema:
          $ref: "#/components/schemas/Record"
  responses:
    '200':
      description: "successfully uploaded"
      content:
        application/json:
          schema:
            $ref: "#/components/schemas/Record"
    '401':
      description: Not authorized
    '404':
      description: Named database not found

delete:
  tags:
  - "Database Data"
  summary: "Delete the objects matching the query"
  description: ""
  operationId: "cloudmesh.database.data.delete"
  parameters:
    - name: name
      description: Name of the database
      in: path
      required: true
      schema:
        type: string
    - name: query
      description: Database Query
      in: query
      required: true
      schema:
        $ref: '#/components/schemas/Query'
  responses:
    '200':
      description: Successfull query
      content:
        application/json:
          schema:
            type: array
            items:
              $ref: "#/components/schemas/Record"
    '401':
      description: Not authorized
    '404':
      description: Named database not found

components:
  schemas:
    Database:
      type: object
      description: Defines a database object as an entry
      properties:
        name:
          type: string
          description: Name of the database
        description:
          type: string
          description: Description of the database
        endpoint:
          type: string
          description: Endpoint of the database
        kind:
          type: string
          description: the kind of the database
    Schema:
      type: object
      description: Defines a database
      properties:
        name:
          type: string
```

4.7.4 Virtual Directory

A virtual directory is a collection of files, replicas, streams or other virtual directories.

4.7.4.1 Schema Virtualdirectory

Reference: ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the virtual directory</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Description of the virtual directory</td>
</tr>
<tr>
<td>host</td>
<td>string</td>
<td>Remote host of the virtual directory</td>
</tr>
<tr>
<td>location</td>
<td>string</td>
<td>Remote location, e.g., a directory with full path on a host</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
<td>Access protocol (e.g. HTTP, FTP, SSH, etc.)</td>
</tr>
<tr>
<td>credential</td>
<td>object</td>
<td>Credential to access</td>
</tr>
</tbody>
</table>

4.7.4.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/virtualdirectory</td>
<td>Returns a list of virtual directories</td>
</tr>
<tr>
<td>get</td>
<td>/virtualdirectory/{name}</td>
<td>Returns the named virtual directory</td>
</tr>
<tr>
<td>put</td>
<td>/virtualdirectory/{name}</td>
<td>Uploads a virtual directory to the list of virtual directories</td>
</tr>
<tr>
<td>Method</td>
<td>Uri</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------------------------</td>
<td>-------------------------------------------------------</td>
</tr>
<tr>
<td>delete</td>
<td><code>/virtualdirectory/{name}</code></td>
<td>Deletes the named virtual directory</td>
</tr>
<tr>
<td>get</td>
<td><code>/virtualdirectory/{name}/{filename}</code></td>
<td>Returns the specific file of that virtual directory</td>
</tr>
<tr>
<td>put</td>
<td><code>/virtualdirectory/{name}/{filename}</code></td>
<td>Updates or adds a virtual file in the virtual directory</td>
</tr>
<tr>
<td>delete</td>
<td><code>/virtualdirectory/{name}/{filename}</code></td>
<td>Delete an user in the virtual directory</td>
</tr>
</tbody>
</table>

4.7.4.2.1 `/virtualdirectory`  

4.7.4.2.1.1 GET `/virtualdirectory`

Returns a list of all virtual directories

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of virtual directories</td>
<td>array[Virtualdirectory]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.7.4.2.2 `/virtualdirectory/{name}`

4.7.4.2.2.1 GET `/virtualdirectory/{name}`

Returns an virtual directory by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the virtual directory</td>
<td>Virtualdirectory</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual directory could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the virtual directory</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>
4.7.4.2.2 PUT /virtualdirectory/{name}

Uploads a virtual directory to the list of virtual directories

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Virtual directory updated or created</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual directory could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The virtual directory to be uploaded</td>
<td>True</td>
<td>Virtualdirectory</td>
</tr>
</tbody>
</table>

4.7.4.2.3 DELETE /virtualdirectory/{name}

Deletes an virtual directory by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual directory could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the virtual directory</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.7.4.2.3.1 GET /virtualdirectory/{name}/{filename}

Returns the specific file of that virtual directory

Responses
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>upload successful</td>
<td>File</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual directory or file could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the virtual directory</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>filename</td>
<td>path</td>
<td>The filename</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.7.4.2.3.2 PUT /virtualdirectory/{name}/{filename}

Updates or adds a virtual file in the virtual directory

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>User added successfully</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the virtual directory</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>filename</td>
<td>path</td>
<td>The filename</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

**Request Body**

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The user to be uploaded</td>
<td>True</td>
<td>File</td>
</tr>
</tbody>
</table>

4.7.4.2.3.3 DELETE /virtualdirectory/{name}/{filename}

Delete an user in the virtual directory

**Responses**
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual directory or file could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the virtual directory</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>filename</td>
<td>path</td>
<td>The filename</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.7.4.3 virtualdirectory.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Virtual Directory
  description: |
    A virtual directory is a collection of files, replicas, streams or other virtual directories.
  contact:
    name: NIST BDRA Interface Subgroup Service
    url: https://cloudmesh-community.github.io/nist/spec/
  license:
    name: Apache 2.0
    url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
- url: /cloudmesh/v3
paths:
  /virtualdirectory:
    get:
      tags:
        - Virtual directory
      summary: Returns a list of virtual directories
      description: Returns a list of all virtual directories
      operationId: cloudmesh.virtualdirectory.list
      responses:
        '200':
          description: The list of virtual directories
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Virtualdirectory'
        '401':
          description: Not authorized
  /virtualdirectory/{name}:
    get:
      tags:
        - Virtual directory
      summary: Returns the named virtual directory
      description: Returns an virtual directory by name
      operationId: cloudmesh.virtualdirectory.find_by_name
      parameters:
        - name: name
          in: path
          required: true
      responses:
        '200':
          description: Returning the information of the virtual directory
```

content: "application/json:
  schema:
    $ref: '#/components/schemas/Virtualdirectory'
'401':
  description: Not authorized
'404':
  description: The named virtual directory could not be found

put:
  tags:
  - Virtual directory
  summary: uploads a virtual directory to the list of virtual directories
  description: Uploads a virtual directory to the list of virtual directories
  operationId: cloudmesh.virtualdirectory.add
  requestBody:
    description: The virtual directory to be uploaded
    required: true
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/Virtualdirectory'
  responses:
    '200':
      description: Virtual directory updated or created
    '401':
      description: Not authorized
    '404':
      description: The named virtual directory could not be found

delete:
  tags:
  - Virtual directory
  summary: Deletes the named virtual directory
  description: Deletes an virtual directory by name
  operationId: cloudmesh.virtualdirectory.delete_by_name
  parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
      description: The name of the virtual directory
  responses:
    '200':
      description: Deletion successful
    '401':
      description: Not authorized
    '404':
      description: The named virtual directory could not be found

/virtualdirectory/{name}/{filename}:
  get:
    tags:
    - Virtual directory
    summary: Returns the specific file of that virtual directory
    description: Returns the specific file of that virtual directory
    operationId: cloudmesh.virtualdirectory.file.get_by_name
    parameters:
    - name: name
      in: path
      required: true
      schema:
        type: string
        description: The name of the virtual directory
    - name: filename
      in: path
      required: true
      schema:
        type: string
    responses:
      '200':
        description: upload successful
        content:
          application/json:
            schema:
              $ref: "filestore.yaml#/components/schemas/File"
      '401':
        description: Not authorized
      '404':
        description: The named virtual directory or file could not be found

put:
  tags:
  - Virtual directory
  summary: Updates or adds a virtual file in the virtual directory
  description: Updates or adds a virtual file in the virtual directory
  operationId: cloudmesh.virtualdirectory.file.add
  parameters:
  - name: name
    in: path
    required: true
4.8 COMPUTE MANAGEMENT - VIRTUAL CLUSTERS

4.8.1 Virtual Cluster

A Virtual Cluster is modeled as manager node, and one or more compute nodes. The manager
node usually serves as a login node and can be accessed from outside via a public IP. The compute nodes are connected to the manager node via a private, usually high performance (high throughput and low latency) network and thus accessible only from the manager node. To use the virtual cluster, login to the manager node, and from there one can login to any compute node, or submit a job to run on the compute nodes.

4.8.1.1 Schema Virtualcluster

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the virtual cluster</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the virtual cluster</td>
</tr>
<tr>
<td>owner</td>
<td>string</td>
<td>Username of the owner of the virtual cluster</td>
</tr>
<tr>
<td>manager</td>
<td>Node</td>
<td>Manager node of the virtual cluster</td>
</tr>
<tr>
<td>nodes</td>
<td>array[Node]</td>
<td>List of nodes of the virtual cluster</td>
</tr>
</tbody>
</table>

4.8.1.2 Schema Node

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the node</td>
</tr>
<tr>
<td>state</td>
<td>string</td>
<td>Power state of the node</td>
</tr>
<tr>
<td>ncpu</td>
<td>integer</td>
<td>Number of virtual CPUs of the node</td>
</tr>
<tr>
<td>ram</td>
<td>string</td>
<td>RAM size of the node</td>
</tr>
<tr>
<td>disk</td>
<td>string</td>
<td>Disk size of the node</td>
</tr>
<tr>
<td>nics</td>
<td>array[NIC]</td>
<td>List of network interfaces of the node</td>
</tr>
</tbody>
</table>

4.8.1.3 Schema NIC

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mac</td>
<td>string</td>
<td>MAC address of the node</td>
</tr>
<tr>
<td>ip</td>
<td>string</td>
<td>IP address of the node</td>
</tr>
</tbody>
</table>

4.8.1.4 Paths
<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/virtualcluster</td>
<td>Returns a list of virtual clusters</td>
</tr>
<tr>
<td>get</td>
<td>/virtualcluster/{name}</td>
<td>Returns the named virtual cluster</td>
</tr>
<tr>
<td>put</td>
<td>/virtualcluster/{name}</td>
<td>Uploads an virtual cluster to the list of virtual clusters</td>
</tr>
<tr>
<td>delete</td>
<td>/virtualcluster/{name}</td>
<td>Deletes the named virtual cluster</td>
</tr>
<tr>
<td>get</td>
<td>/virtualcluster/{name}/{node}</td>
<td>Node of the named virtual cluster</td>
</tr>
<tr>
<td>put</td>
<td>/virtualcluster/{name}/{node}</td>
<td>Updates or adds a node to the virtual cluster</td>
</tr>
<tr>
<td>delete</td>
<td>/virtualcluster/{name}/{node}</td>
<td>Delete a node in the virtual cluster</td>
</tr>
</tbody>
</table>

4.8.1.4.1 /virtualcluster

4.8.1.4.1.1 GET /virtualcluster

Returns a list of all virtual clusters

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of virtual clusters</td>
<td>array[Virtualcluster]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.1.4.2 /virtualcluster/{name}

4.8.1.4.2.1 GET /virtualcluster/{name}

Returns an virtual cluster by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the virtual cluster</td>
<td>Virtualcluster</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual cluster could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters
### Name, Located in, Description, Required, Schema

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the virtual cluster</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.1.4.2.2 PUT `/virtualcluster/{name}`

Uploads an virtual cluster to the list of virtual clusters

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Virtual cluster updated or created</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual cluster could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

**Request Body**

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The virtual cluster to be uploaded</td>
<td>True</td>
<td>Virtualcluster</td>
</tr>
</tbody>
</table>

4.8.1.4.2.3 DELETE `/virtualcluster/{name}`

Deletes an virtual cluster by name

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual cluster could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the virtual cluster</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.1.4.3 `/virtualcluster/{name}/{node}`

4.8.1.4.3.1 GET `/virtualcluster/{name}/{node}`
Returns the specific node of the named virtual cluster. If the node name is manager, the manager node is used. A compute node cannot be named manager.

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Node info</td>
<td>Node</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual cluster or node could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the virtual cluster</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>node</td>
<td>path</td>
<td>The node name</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.1.4.3.2 PUT /virtualcluster/{name}/{node}

Updates or adds a node to the virtual cluster. If the node name is manager, the manager node is uploaded.

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Node added successfully</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual cluster or node could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the virtual cluster</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>node</td>
<td>path</td>
<td>The node name</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The node to be uploaded</td>
<td>True</td>
<td>Node</td>
</tr>
</tbody>
</table>
Delete a node in the virtual cluster

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual cluster or node could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the virtual cluster</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>node</td>
<td>path</td>
<td>The node name</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.1.5 virtualcluster.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Virtual Cluster
  description: |

  A Virtual Cluster is modeled as manager node, and one or more compute nodes. The manager node usually serves as a login node and can be accessed from outside via a public IP. The compute nodes are connected to the manager node via a private, usually high performance (high throughput and low latency) network and thus accessible only from the manager node. Login to the virtual cluster, login to the manager node, and from there one can login to any compute node, or submit a job to run on the compute nodes.

contact:
  name: NIST BDRA Interface Subgroup Service
  url: https://cloudmesh-community.github.io/nist/spec/
license:
  name: Apache 2.0
  url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3
paths:
  /virtualcluster:
    get:
      tags:
        - Virtual cluster
      summary: Returns a list of virtual clusters
      description: Returns a list of all virtual clusters
      operationId: cloudmesh.virtualcluster.list
      responses:
        "200":
          description: The list of virtual clusters
          content:
            application/json:
              schema:
                type: array
                items: $ref: '#/components/schemas/Virtualcluster'
        "401":
          description: Not authorized
```
```yaml
/virtualcluster/{name}:
  get:
    tags:
      - Virtual cluster
    summary: Returns the named virtual cluster
    description: Returns an virtual cluster by name
    operationId: cloudmesh.virtualcluster.find_by_name
    parameters:
      - name: name
        in: path
        required: true
        schema:
          type: string
        description: The name of the virtual cluster
    responses:
      '200':
        description: Returning the information of the virtual cluster
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Virtualcluster'
      '401':
        description: Not authorized
      '404':
        description: The named virtual cluster could not be found
  put:
    tags:
      - Virtual cluster
    summary: Uploads an virtual cluster to the list of virtual clusters
    description: Uploads an virtual cluster to the list of virtual clusters
    operationId: cloudmesh.virtualcluster.add
    requestBody:
      description: The virtual cluster to be uploaded
      required: true
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/Virtualcluster'
    responses:
      '200':
        description: Virtual cluster updated or created
      '401':
        description: Not authorized
      '404':
        description: The named virtual cluster could not be found
  delete:
    tags:
      - Virtual cluster
    summary: Deletes the named virtual cluster
    description: Deletes an virtual cluster by name
    operationId: cloudmesh.virtualcluster.delete_by_name
    parameters:
      - name: name
        in: path
        required: true
        schema:
          type: string
        description: The name of the virtual cluster
    responses:
      '200':
        description: Deletion successful
      '401':
        description: Not authorized
      '404':
        description: The named virtual cluster could not be found
/virtualcluster/{name}/{node}:
  get:
    tags:
      - Virtual cluster
    summary: Node of the named virtual cluster
    description: Returns the specific node of the named virtual cluster. If the node name is manager, the manager node is used. A compute node can not be named manager
    operationId: cloudmesh.virtualcluster.node.get_by_name
    parameters:
      - name: name
        in: path
        required: true
        schema:
          type: string
        description: The name of the virtual cluster
      - name: node
        in: path
        required: true
        schema:
          type: string
        description: The node name
    responses:
      '200':
        description: The node of the named virtual cluster
```

description: Node info

content:
application/json:
schema:
  $ref: "#/components/schemas/Node"

'401':
description: Not authorized
'404':
description: The named virtual cluster or node could not be found

put:
tag:
- Virtual cluster
summary: Updates or adds a node to the virtual cluster
description: Updates or adds a node to the virtual cluster. If the node name is manager, the manager node is uploaded.
operationId: cloudmesh.virtualcluster.node.add
parameters:
- name: name
  in: path
  required: true
  schema:
    type: string
    description: The name of the virtual cluster
- name: node
  in: path
  required: true
  schema:
    type: string
requestBody:
description: The node to be uploaded
required: true
content:
application/json:
schema:
  $ref: '#/components/schemas/Node'
responses:
  '200':
description: Node added successfully
  '401':
description: Not authorized
  '404':
description: The named virtual cluster or node could not be found

delete:
tag:
- Virtual cluster
summary: Delete a node in the virtual cluster
description: Delete a node in the virtual cluster
operationId: cloudmesh.virtualcluster.node.delete
parameters:
- name: name
  in: path
  required: true
  schema:
    type: string
    description: The name of the virtual cluster
- name: node
  in: path
  required: true
  schema:
    type: string
responses:
  '200':
description: Deletion successful
  '401':
description: Not authorized
  '404':
description: The named virtual cluster or node could not be found

components:
schemas:
  Virtualcluster:
    type: object
    properties:
      name:
        description: The name of the virtual cluster
        type: string
      description:
        type: string
        description: A description of the virtual cluster
      owner:
        type: string
        description: Username of the owner of the virtual cluster
      manager:
        description: Manager node of the virtual cluster
        $ref: "#/components/schemas/Node"
      nodes:
        description: List of nodes of the virtual cluster
        type: array
4.8.2 Network of Nodes

A Network of Nodes (NON) contains a number of compute nodes that are connected by a network and can be reached from each other. The concept is a generalization of the term Network of Workstations. In contrast to a Virtual Cluster it does not have a dedicated manager node. Network of nodes can be real or virtual. The same security context can be used to authenticate to all nodes in the network of nodes. This is typically done with a public keystore in which all keys are stored that allow access to the nodes.

4.8.2.1 Schema Non

Reference: ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the network of nodes</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description of the network of nodes</td>
</tr>
<tr>
<td>nodes</td>
<td>array[Node]</td>
<td>List of nodes of the network of nodes</td>
</tr>
</tbody>
</table>

4.8.2.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/non</td>
<td>Returns a list of network of nodeses</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Uploads a network of nodes to the list of</td>
</tr>
</tbody>
</table>
### 4.8.2.2.1 /non

Returns a list of all network of nodess

#### Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of network of nodess</td>
<td>array[Non]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

### 4.8.2.2.2 /non/{name}

Uploads a network of nodes to the list of network of nodess

#### Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Network of nodes updated or created.</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error updating network of nodes</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

#### Request Body
The network of nodes to be uploaded

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the network of nodes</td>
<td>string</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the network of nodes</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.2.2.3 DELETE /non/{name}/publickeystore

Deletes a network of nodes by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the network of nodes</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.2.2.3.3 PUT /non/{name}/publickeystore

Updates or adds a node to the network of nodes.
Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Node keystore added successfully</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the network of nodes</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.2.2.4 /non/{name}/node

4.8.2.2.4.1 GET /non/{name}/node

Returns a network of nodes by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the node</td>
<td>Non</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named network of nodes could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the network of nodes</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.2.2.4.2 DELETE /non/{name}/node

Deletes a network of nodes by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named network of nodes could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the network of nodes</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.2.2.5 /non/{name}/node/{node}

4.8.2.2.5.1 GET /non/{name}/node/{node}

Returns the specific node of the named network of nodes.

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Node info</td>
<td>Node</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the network of nodes</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>node</td>
<td>path</td>
<td>The node name</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.2.2.5.2 PUT /non/{name}/node/{node}

Updates or adds a node to the network of nodes

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Node added successfully</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the network of nodes</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>
node   path   The node name   True   String

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The node to be uploaded</td>
<td>True</td>
<td>Node</td>
</tr>
</tbody>
</table>

4.8.2.5.3 DELETE /non/{name}/node/{node}

Delete a node in the network of nodes

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the network of nodes</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>node</td>
<td>path</td>
<td>The node name</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.2.3 non.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Network of Nodes
  description: "A Network of Nodes (NON) contains a number of compute nodes that are connected by a network and can be reached from each other. The concept is a generalization of the term Network of Workstations. In contrast to a Virtual Cluster it does not have a dedicated manager node. Network of nodes can be real or virtual. The same security context can be used to authenticate to all nodes in the network of nodes. This is typically done with a public keystore in which all keys are stored that allow access to the nodes."

contact:
  name: NIST BDRA Interface Subgroup Service
  url: https://cloudmesh-community.github.io/nist/spec/
license:
  name: Apache 2.0
  url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3
paths:
  /non:
    get:
      tags:
- Network of nodes
  summary: Returns a list of network of nodes
  description: Returns a list of all network of nodes
  operationId: cloudmesh.non.list
  responses:
    '200':
      description: The list of network of nodes
      content:
        application/json:
          schema:
            type: array
            items:
              $ref: '#/components/schemas/Non'
    '401':
      description: Not authorized

/non/{name}:
  put:
    tags:
      - Network of nodes
      summary: Uploads a network of nodes to the list of network of nodes
      description: Uploads a network of nodes to the list of network of nodes
      operationId: cloudmesh.non.add
      requestBody:
        description: The network of nodes to be uploaded
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Non'
      responses:
        '200':
          description: Network of nodes updated or created.
        '400':
          description: Error updating network of nodes
        '401':
          description: Not authorized

/non/{name}/publickeystore:
  get:
    tags:
      - Non
      summary: Returns the information of the keystore
      description: Returns a network of nodes by name
      operationId: cloudmesh.non.keystore.find_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the network of nodes
      responses:
        '200':
          description: Returning the information of the network of nodes
          content:
            application/json:
              schema:
                type: string
                description: the endpoint of the publickeystore
        '401':
          description: Not authorized
  delete:
    tags:
      - Network of nodes
      summary: Deletes the keystore
      description: Deletes a network of nodes by name
      operationId: cloudmesh.non.keystore.delete
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the network of nodes
      responses:
        '200':
          description: Deletion successful
        '401':
          description: Not authorized
  put:
    tags:
      - Network of nodes
      summary: Adds a keystore
      description: Updates or adds a node to the network of nodes.
      operationId: cloudmesh.non.keystore.add
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
The name of the network of nodes

responses:
'200':
description: Node keystore added successfully
'401':
description: Not authorized

/non/{name}/node:

get:
tags:
  - Non
summary: Returns the named network of nodes
description: Returns a network of nodes by name
operationId: cloudmesh.non.find_by_name
parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
    description: The name of the network of nodes
responses:
'200':
description: Returning the information of the node
content:
  application/json:
    schema:
      $ref: '#/components/schemas/Non'
'401':
description: Not authorized
'404':
description: The named network of nodes could not be found

delete:
tags:
  - Network of nodes
summary: Deletes the named network of nodes
description: Deletes a network of nodes by name
operationId: cloudmesh.non.delete_by_name
parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
    description: The name of the network of nodes
responses:
'200':
description: Deletion successful
'401':
description: Not authorized
'404':
description: The named network of nodes could not be found

/non/{name}/node/{node}:

get:
tags:
  - Network of nodes
summary: Node of the named network of nodes
description: Returns the specific node of the named network of nodes.
operationId: cloudmesh.non.node.get_by_name
parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
    description: The name of the network of nodes
  - name: node
    description: The node name
    in: path
    required: true
    schema:
      type: string
    description: The node name
responses:
'200':
description: Node info
content:
  application/json:
    schema:
      $ref: "virtualcluster.yaml#/components/schemas/Node"
'401':
description: Not authorized

put:
tags:
  - Network of nodes
summary: Updates or adds a node to the network of nodes
description: Updates or adds a node to the network of nodes
operationId: cloudmesh.non.node.add
parameters:
  - name: name
    in: path
4.8.3 Scheduler

A scheduler allows to control the execution of tasks based on a policy. Schedulers may allow the assignment of different policies to define the order of the tasks. A scheduler returns the next task to be executed. Tasks can be added and deleted.

4.8.3.1 Schema Task

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>name of the scheduler</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>---------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>name of the scheduler policy</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the policy</td>
</tr>
<tr>
<td>kind</td>
<td>string</td>
<td>The kind of the policy</td>
</tr>
<tr>
<td>parameters</td>
<td>string</td>
<td>parameters to define the behaviour of the scheduler</td>
</tr>
</tbody>
</table>

### 4.8.3.2 Schema Policy

**Reference:** ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>name of the scheduler policy</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the policy</td>
</tr>
<tr>
<td>kind</td>
<td>string</td>
<td>The kind of the policy</td>
</tr>
<tr>
<td>parameters</td>
<td>string</td>
<td>parameters to define the behaviour of the scheduler</td>
</tr>
</tbody>
</table>

### 4.8.3.3 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/task</td>
<td>Returns a list of tasks</td>
</tr>
<tr>
<td>get</td>
<td>/task/{name}</td>
<td>Returns the named task</td>
</tr>
<tr>
<td>put</td>
<td>/task/{name}</td>
<td>Uploads a task to the list of tasks</td>
</tr>
<tr>
<td>delete</td>
<td>/task/{name}</td>
<td>Deletes the named task</td>
</tr>
<tr>
<td>get</td>
<td>/policy</td>
<td>Returns the policy found</td>
</tr>
<tr>
<td>put</td>
<td>/policy</td>
<td>Uploads the policy</td>
</tr>
</tbody>
</table>

#### 4.8.3.3.1 /task

**4.8.3.3.1.1 GET /task**

Returns a list of all tasks

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of tasks</td>
<td>array[Task]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

#### 4.8.3.2 /task/{name}
4.8.3.3.2.1 GET /task/{name}

Returns a task by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the task</td>
<td>Task</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named task could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the task</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>operation</td>
<td>query</td>
<td>Show the task but do not remove it from the queue</td>
<td>False</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.3.3.2.2 PUT /task/{name}

Uploads a task to the list of tasks

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Task updated</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the task</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The task to be uploaded</td>
<td>True</td>
<td>Task</td>
</tr>
</tbody>
</table>
4.8.3.3.2.3 DELETE /task/{name}

Deletes an task by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error to delete the task</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named task could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the task</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.3.3.3 /policy

4.8.3.3.3.1 GET /policy

Returns the policy

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The policy</td>
<td>array[Policy]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.3.3.3.2 PUT /policy

Uploads a task to the list of tasks

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Task updated</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error updating</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>Located in</td>
<td>Description</td>
<td>Required</td>
</tr>
<tr>
<td>------------</td>
<td>-------------</td>
<td>----------</td>
</tr>
<tr>
<td>Body</td>
<td>The policy to be uploaded</td>
<td>True</td>
</tr>
</tbody>
</table>

4.8.3.4 scheduler.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Queue
  description: |
    A scheduler allows to control the execution of tasks based on a policy. Schedulers may allow the assignment of different policies to define the order of tasks. A scheduler returns the next task to be executed. Tasks can be added and deleted.

contact:
  name: NIST BDRA Interface Subgroup
  url: https://cloudmesh-community.github.io/nist
license:
  name: Apache 2.0
  url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3/scheduler
paths:
  /task:
    get:
      tags:
        - Scheduler
      summary: Returns a list of tasks
      description: Returns a list of all tasks
      operationId: cloudmesh.scheduler.task.list
      responses:
        '200':
          description: The list of tasks
          content:
            application/json:
              schema:
                type: array
              items:
                $ref: '#/components/schemas/Task'
        '401':
          description: Not authorized
  /task/{name}:
    get:
      tags:
        - Scheduler
      summary: Returns the named task
      description: Returns an task by name
      operationId: cloudmesh.scheduler.task.find_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the task
        - name: operation
          in: query
          description: Show the task but do not remove it from the queue
          schema:
            type: string
            enum: [info]
      responses:
        '200':
          description: Returning the information of the task
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Task'
        '401':
          description: Not authorized
        '404':
          description: The named task could not be found
```

```
put:
  tags:
  - Scheduler
  summary: Uploads a task to the list of tasks
  description: Uploads a task to the list of tasks
  operationId: cloudmesh.scheduler.task.add
  parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
      description: The name of the task
  requestBody:
    description: The task to be uploaded
    required: true
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/Task'
  responses:
    '200':
      description: Task updated
    '401':
      description: Not authorized
delete:
  tags:
  - Scheduler
  summary: Deletes the named task
  description: Deletes an task by name
  operationId: cloudmesh.scheduler.task.delete_by_name
  parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
      description: The name of the task
  responses:
    '200':
      description: Deletion successful
    '400':
      description: Error to delete the task
    '401':
      description: Not authorized
    '404':
      description: The named task could not be found
/policy:
get:
  tags:
  - Scheduler
  summary: Returns the policy found
  description: Returns the policy
  operationId: cloudmesh.scheduler.policy.list
  responses:
    '200':
      description: The policy
      content:
        application/json:
          schema:
            type: array
            items:
              $ref: '#/components/schemas/Policy'
    '401':
      description: Not authorized
put:
  tags:
  - Scheduler
  summary: Uploads the policy
  description: Uploads a task to the list of tasks
  operationId: cloudmesh.scheduler.policy.add
  requestBody:
    description: The policy to be uploaded
    required: true
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/Policy'
  responses:
    '200':
      description: Task updated
    '400':
      description: Error updating
    '401':
      description: Not authorized
components:
schemas:
  Task:
    type: object
    description: the scheduler
4.8.4 Queue

The queue is a special scheduler that allows tasks to be scheduled while doing queue policies, such as LIFO, FIFO, and so on. A queue returns the next task to be executed. Tasks can be added and deleted.

4.8.4.1 Schema Task

Reference:

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the scheduler</td>
</tr>
<tr>
<td>user</td>
<td>string</td>
<td>The username the task belongs to</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the task</td>
</tr>
<tr>
<td>kind</td>
<td>string</td>
<td>The kind of the task</td>
</tr>
<tr>
<td>content</td>
<td>string</td>
<td>The content of the task</td>
</tr>
</tbody>
</table>

4.8.4.2 Schema Policy

Reference:

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the scheduler policy</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the policy</td>
</tr>
<tr>
<td>kind</td>
<td>string</td>
<td>The kind of the policy</td>
</tr>
</tbody>
</table>
parameters  string  parameters to define the behaviour of the scheduler

4.8.4.3 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/task</td>
<td>Returns a list of tasks</td>
</tr>
<tr>
<td>get</td>
<td>/task/{name}</td>
<td>Returns the named task</td>
</tr>
<tr>
<td>put</td>
<td>/task/{name}</td>
<td>Uploads a task to the list of tasks</td>
</tr>
<tr>
<td>delete</td>
<td>/task/{name}</td>
<td>Deletes the named task</td>
</tr>
<tr>
<td>get</td>
<td>/policy</td>
<td>Returns the policy</td>
</tr>
<tr>
<td>put</td>
<td>/policy</td>
<td>Uploads the policy</td>
</tr>
</tbody>
</table>

4.8.4.3.1 /task

4.8.4.3.1.1 GET /task

Returns a list of all tasks

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of tasks</td>
<td>array[Task]</td>
</tr>
<tr>
<td>400</td>
<td>No tasks found</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.4.3.2 /task/{name}

4.8.4.3.2.1 GET /task/{name}

Returns an task by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the task</td>
<td>Task</td>
</tr>
<tr>
<td>400</td>
<td>No task found</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named task could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>
Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the task</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>operation</td>
<td>query</td>
<td>ERROR: description missing</td>
<td>False</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.4.3.2.2 PUT /task/{name}

Uploads a task to the list of tasks

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Task updated</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error updating task.</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the task</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The task to be uploaded</td>
<td>True</td>
<td>Task</td>
</tr>
</tbody>
</table>

4.8.4.3.2.3 DELETE /task/{name}

Deletes an task by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>No task found</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>
The named task could not be found

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the task</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.4.3.3 /policy

4.8.4.3.3.1 GET /policy

Returns the policy

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The policy</td>
<td>array[Policy]</td>
</tr>
<tr>
<td>400</td>
<td>No tasks found</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.8.4.3.3.2 PUT /policy

Uploads a task to the list of tasks

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Task updated</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error updating task</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

**Request Body**

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The Policy</td>
<td>True</td>
<td>Policy</td>
</tr>
</tbody>
</table>

4.8.4.4 queue.yaml

```yaml
openapi: "3.0.2"
info:
```
The queue is a special scheduler that allows tasks to be scheduled while doing queue policies, such as LIFO, FIFO, and so on. A queue returns the next task to be executed. Tasks can be added and deleted.


contact:
  name: NIST BDRA Interface Subgroup
  url: https://cloudmesh-community.github.io/nist

license:
  name: Apache 2.0
  url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt

servers:
  - url: /cloudmesh/v3/scheduler

paths:
  /task:
    get:
      tags:
        - Task
      summary: Returns a list of tasks
      description: Returns a list of all tasks
      operationId: cloudmesh.task.list
      responses:
        '200':
          description: The list of tasks
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Task'
        '400':
          description: No tasks found
        '401':
          description: Not authorized

  /task/{name}:
    get:
      tags:
        - Task
      summary: Returns the named task
      description: Returns an task by name
      operationId: cloudmesh.task.find_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the task
        - in: query
          name: operation
          schema:
            type: string
            enum:
              - info
              - pop
      responses:
        '200':
          description: Returning the information of the task
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Task'
        '400':
          description: No task found
        '401':
          description: Not authorized
        '404':
          description: The named task could not be found

    put:
      tags:
        - Task
      summary: Uploads a task to the list of tasks
      description: Uploads a task to the list of tasks
      operationId: cloudmesh.task.add
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the task
requestBody:
  description: The task to be uploaded
  required: true
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/Task'
  responses:
    '200':
      description: Task updated
    '400':
      description: Error updating task.
    '401':
      description: Not authorized

delete:
  tags:
    - Task
  summary: Deletes the named task
  description: Deletes an task by name
  operationId: cloudmesh.task.delete_by_name
  parameters:
    - name: name
      in: path
      required: true
      schema:
        type: string
        description: The name of the task
  responses:
    '200':
      description: Deletion successful
    '400':
      description: No task found
    '401':
      description: Not authorized
    '404':
      description: The named task could not be found

/policy:
  get:
    tags:
      - Task
    summary: Returns the policy
    description: Returns the policy
    operationId: cloudmesh.task.policy.list
    responses:
      '200':
        description: The policy
        content:
          application/json:
            schema:
              type: array
              items:
                $ref: '#/components/schemas/Policy'
      '400':
        description: No tasks found
      '401':
        description: Not authorized

  put:
    tags:
      - Task
    summary: Uploads the policy
    description: Uploads a task to the list of tasks
    operationId: cloudmesh.task.policy.add
    requestBody:
      description: The Policy
      required: true
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/Policy'
    responses:
      '200':
        description: Task updated
      '400':
        description: Error updating task
      '401':
        description: Not authorized

components:
  schemas:
    Task:
      type: object
      description: The scheduler
      properties:
        name:
          type: string
          description: Name of the scheduler
        user:
          type: string
          description: The username the task belongs to
4.9 COMPUTE MANAGEMENT - VIRTUAL MACHINES

This section summarizes a basic interface specification of virtual machines.

4.9.1 Image

To execute virtual machines, we need an image that specifies the details of the operating system.

4.9.1.1 Schema Image

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>A unique name of the image</td>
</tr>
<tr>
<td>cloud</td>
<td>string</td>
<td>The name of the cloud</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>A label that can be defined by the user for the image</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description for the image</td>
</tr>
<tr>
<td>osType</td>
<td>string</td>
<td>The OS type of the image</td>
</tr>
<tr>
<td>osVersion</td>
<td>string</td>
<td>The OS version of the image</td>
</tr>
<tr>
<td>status</td>
<td>string</td>
<td>The status of the image</td>
</tr>
<tr>
<td>progress</td>
<td>integer</td>
<td>The loading progress percentage of the image</td>
</tr>
<tr>
<td>visibility</td>
<td>string</td>
<td>The visibility of the image</td>
</tr>
<tr>
<td>requirement</td>
<td>Requirements</td>
<td>Minimum requirement to run the image</td>
</tr>
</tbody>
</table>

4.9.1.2 Schema Requirements
### Reference:

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>size</td>
<td>integer</td>
<td>Minimum disk size in bytes required for the image</td>
</tr>
<tr>
<td>ram</td>
<td>integer</td>
<td>Minimum ram size in bytes to run the image</td>
</tr>
<tr>
<td>cpu</td>
<td>string</td>
<td>CPU required to run the image</td>
</tr>
<tr>
<td>cores</td>
<td>integer</td>
<td>Minimum number of cores</td>
</tr>
</tbody>
</table>

#### 4.9.1.3 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/image/{cloud}</td>
<td>Returns a list of images for the cloud</td>
</tr>
<tr>
<td>get</td>
<td>/image/{cloud}/{name}</td>
<td>Returns the named image</td>
</tr>
<tr>
<td>put</td>
<td>/image/{cloud}/{name}</td>
<td>Add a image</td>
</tr>
<tr>
<td>delete</td>
<td>/image/{cloud}/{name}</td>
<td>Deletes the named image</td>
</tr>
</tbody>
</table>

#### 4.9.1.3.1 /image/{cloud}

4.9.1.3.1.1 GET /image/{cloud}

Returns a list of all images

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of images</td>
<td>array[Image]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>path</td>
<td>The name of the cloud</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>
Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the image</td>
<td>Image</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named image could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>path</td>
<td>The name of the cloud</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the image</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.9.1.3.2.2 PUT /image/{cloud}/{name}

Sets the named image

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Image updated or created</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named image could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>path</td>
<td>The name of the cloud</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The image to add or modify</td>
<td>True</td>
<td>Image</td>
</tr>
</tbody>
</table>

4.9.1.3.2.3 DELETE /image/{cloud}/{name}

Deletes a image by name
Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named image could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>path</td>
<td>The name of the cloud</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the image</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.9.1.4 image.yaml

```
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Image
  description: |
    To execute virtual machines, we need an image that specifies the
doals of the operating system.

contact:
  name: NIST BDRA Interface Subgroup
  url: https://cloudmesh-community.github.io/nist/spec/
license:
  name: Apache 2.0
  url: https://github.com/cloudmeh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3
paths:
  /image/{cloud}:
    get:
      tags:
        - Image
      summary: Returns a list of images for the cloud
      description: Returns a list of all images
      operationId: cloudmesh.image.list
      parameters:
        - name: cloud
          in: path
          required: true
          schema:
            type: string
            description: The name of the cloud
      responses:
        '200':
          description: The list of images
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Image'
        '401':
          description: Not authorized
  /image/{cloud}/{name}:
    get:
      tags:
        - Image
      summary: Returns the named image
      description: Returns a image by name
      operationId: cloudmesh.image.find_by_name
```
parameters:
- name: cloud
  in: path
  description: The name of the cloud
  required: true
  schema:
    type: string
- name: name
  in: path
  required: true
  schema:
    type: string
  description: The name of the image

responses:
'200':
  description: Returning the information of the image
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/Image'
'401':
  description: Not authorized
'404':
  description: The named image could not be found

put:
  tags:
  - Image
  summary: Add a image
  description: Sets the named image
  operationId: cloudmesh.image.add
  parameters:
  - name: cloud
    in: path
    required: true
    schema:
      type: string
  requestBody:
    description: The image to add or modify
    required: true
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/Image'
  responses:
    '200':
      description: Image updated or created
    '401':
      description: Not authorized
    '404':
      description: The named image could not be found

delete:
  tags:
  - Image
  summary: Deletes the named image
  description: Deletes a image by name
  operationId: cloudmesh.image.delete_by_name
  parameters:
  - name: cloud
    in: path
    required: true
    schema:
      type: string
  - name: name
    in: path
    required: true
    schema:
      type: string
  responses:
    '200':
      description: Deletion successful
    '401':
      description: Not authorized
    '404':
      description: The named image could not be found

components:
schemas:
  Image:
    type: object
    properties:
      name:
        type: string
        description: A unique name of the image
      cloud:
        type: string
        description: The name of the cloud
      label: 
4.9.2 Flavor

The flavor specifies elementary information about a virtual machine or compute node. This information includes name, id, label, ram size, swap size, disk space, availability of ephemeral disk, available bandwidth, price value, cloud name. Flavors and the corresponding information are essential to size a virtual cluster appropriately.

4.9.2.1 Schema Flavor

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the flavor</td>
</tr>
<tr>
<td>id</td>
<td>string</td>
<td>The id of the flavor for the named cloud</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>A label that a user can set for this flavor</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>A description for the flavor</td>
</tr>
<tr>
<td>ram</td>
<td>integer</td>
<td>Number of bytes used for the image in RAM</td>
</tr>
<tr>
<td>swap</td>
<td>integer</td>
<td>Number of bytes used for the image in SWAP</td>
</tr>
<tr>
<td>disk</td>
<td>integer</td>
<td>Number of bytes used for the disk</td>
</tr>
<tr>
<td>ephemeral_disk</td>
<td>boolean</td>
<td>Specifies whether the flavor features an ephemeral disk</td>
</tr>
<tr>
<td>bandwidth</td>
<td>integer</td>
<td>Bandwidth of the node</td>
</tr>
<tr>
<td>price</td>
<td>number</td>
<td>Price for the flavor</td>
</tr>
<tr>
<td>-------</td>
<td>--------</td>
<td>----------------------</td>
</tr>
<tr>
<td>cloud</td>
<td>string</td>
<td>Name of the cloud this flavor is used in</td>
</tr>
</tbody>
</table>

### 4.9.2.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/flavor/{cloud}</td>
<td>Returns a list of flavors for the cloud</td>
</tr>
<tr>
<td>get</td>
<td>/flavor/{cloud}/{name}</td>
<td>Returns the named flavor</td>
</tr>
<tr>
<td>put</td>
<td>/flavor/{cloud}/{name}</td>
<td>Add a flavor</td>
</tr>
<tr>
<td>delete</td>
<td>/flavor/{cloud}/{name}</td>
<td>Deletes the named flavor</td>
</tr>
</tbody>
</table>

#### 4.9.2.2.1 /flavor/{cloud}

**GET /flavor/{cloud}**

Returns a list of all flavors

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of flavors</td>
<td>array[Flavor]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>path</td>
<td>The name of the cloud</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

#### 4.9.2.2.2 /flavor/{cloud}/{name}

**GET /flavor/{cloud}/{name}**

Returns a flavor by name

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the flavor</td>
<td>Flavor</td>
</tr>
</tbody>
</table>

401  Not authorized  String
404  The named flavor could not be found  String

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>path</td>
<td>The name of the cloud</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the flavor</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.9.2.2.2.2 PUT /flavor/{cloud}/{name}

Sets the named flavor

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Flavor updated</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named flavor could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>path</td>
<td>The name of the cloud</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The flavor to add or modify</td>
<td>True</td>
<td>Flavor</td>
</tr>
</tbody>
</table>

4.9.2.2.2.3 DELETE /flavor/{cloud}/{name}

Deletes a flavor by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
</tbody>
</table>
4.9.2.3 flavor.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Flavor
  description: |
    The flavor specifies elementary information about a virtual machine or compute node. This information includes name, id, label, ram size, swap size, disk space, availability of ephemeral disk, available bandwidth, price value, cloud name. Flavors and the corresponding information are essential to size a virtual cluster appropriately.
contact:
  name: NIST BDRA Interface Subgroup
  url: https://cloudmesh-community.github.io/nist
license:
  name: Apache 2.0
  url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers: |
  - url: /cloudmesh/v3
paths:
  /flavor/{cloud}:
    get:
      tags:
        - Flavor
      summary: Returns a list of flavors for the cloud
      description: Returns a list of all flavors
      operationId: cloudmesh.flavor.list
      parameters:
        - name: cloud
          in: path
          required: true
          schema:
            type: string
      responses:
        '200':
          description: The list of flavors
          content:
            application/json:
              schema:
                type: array
              items:
                $ref: '#/components/schemas/Flavor'
        '401':
          description: Not authorized
  /flavor/{cloud}/{name}:
    get:
      tags:
        - Flavor
      summary: Returns the named flavor
      description: Returns a flavor by name
      operationId: cloudmesh.flavor.find_by_name
      parameters:
        - name: cloud
          in: path
          required: true
          schema:
            type: string
      responses:
        '200':
          description: Flavor found
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Flavor'
        '404':
          description: The named flavor could not be found
          content:
            text/plain:
              schema:
                type: string
  ''
```

401 Not authorized String
404 The named flavor could not be found String

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>path</td>
<td>The name of the cloud</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the flavor</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

- type: string
  - name: name
    in: path
    required: true
    schema:
      type: string
      description: The name of the flavor

responses:
  '200':
    description: Returning the information of the flavor
    content:
      application/json:
        schema: $ref: '#/components/schemas/Flavor'
  '401':
    description: Not authorized
  '404':
    description: The named flavor could not be found

put:
  tags:
    - Flavor
  summary: Add a flavor
  description: Sets the named flavor
  operationId: cloudmesh.flavor.add
  parameters:
    - name: cloud
      in: path
      required: true
      schema:
        type: string
        description: The name of the cloud
  requestBody:
    description: The flavor to add or modify
    required: true
    content:
      application/json:
        schema: $ref: '#/components/schemas/Flavor'

responses:
  '200':
    description: Flavor updated
  '401':
    description: Not authorized
  '404':
    description: The named flavor could not be found

delete:
  tags:
    - Flavor
  summary: Deletes the named flavor
  description: Deletes a flavor by name
  operationId: cloudmesh.flavor.delete_by_name
  parameters:
    - name: cloud
      in: path
      required: true
      description: The name of the cloud
    - name: name
      in: path
      required: true
      schema:
        type: string
        description: The name of the flavor

responses:
  '200':
    description: Deletion successful
  '401':
    description: Not authorized
  '404':
    description: The named flavor could not be found

components:
  schemas:
    Flavor:
      type: object
      description: The flavor
      properties:
        name:
          type: string
          description: Name of the flavor
        id:
          type: string
          description: The id of the flavor for the named cloud
        label:
          type: string
          description: A label that a user can set for this flavor
        description:
          type: string
          description: A description for the flavor
4.9.3 Virtual Machine

Vm is used to manage virtual machines.

4.9.3.1 Schema Vm

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>provider</td>
<td>string</td>
<td>Name of the provider</td>
</tr>
<tr>
<td>name</td>
<td>string</td>
<td>the unique name of the virtual machine</td>
</tr>
<tr>
<td>image</td>
<td>string</td>
<td>the image name for the virtual machine</td>
</tr>
<tr>
<td>flavor</td>
<td>string</td>
<td>the flavor name for the virtual machine</td>
</tr>
<tr>
<td>region</td>
<td>string</td>
<td>an optional region</td>
</tr>
<tr>
<td>state</td>
<td>string</td>
<td>The state of the virtual machine</td>
</tr>
<tr>
<td>private_ips</td>
<td>string</td>
<td>The private IPs</td>
</tr>
<tr>
<td>public_ips</td>
<td>string</td>
<td>The public IPS</td>
</tr>
<tr>
<td>metadata</td>
<td>string</td>
<td>The meta data passed along to the virtual machine</td>
</tr>
</tbody>
</table>

4.9.3.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/vm/{cloud}</td>
<td>Returns a list of virtual machines for the cloud</td>
</tr>
<tr>
<td>get</td>
<td>/vm/{cloud}/{name}</td>
<td>Returns the named virtual machine</td>
</tr>
<tr>
<td>put</td>
<td>/vm/{cloud}/{name}</td>
<td>Add a virtual machine</td>
</tr>
<tr>
<td>delete</td>
<td>/vm/{cloud}/{name}</td>
<td>Deletes the named virtual machine</td>
</tr>
</tbody>
</table>
4.9.3.2.1 /vm/{cloud}

4.9.3.2.1.1 GET /vm/{cloud}

Returns a list of all virtual machines

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of virtual machines</td>
<td>array[Vm]</td>
</tr>
<tr>
<td>400</td>
<td>No Vm found</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>path</td>
<td>The name of the cloud</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.9.3.2.2 /vm/{cloud}/{name}

4.9.3.2.2.1 GET /vm/{cloud}/{name}

Returns a virtual machine by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the virtual machine</td>
<td>Vm</td>
</tr>
<tr>
<td>400</td>
<td>Error updating virtual machine</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual machine or cloud could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>path</td>
<td>The name of the cloud</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the virtual machine</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.9.3.2.2.2 PUT /vm/{cloud}/{name}
Sets the named virtual machine

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Vm updated</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error updating virtual machine</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual machine or cloud could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>path</td>
<td>The name of the cloud</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The virtual machine to add or modify</td>
<td>True</td>
<td>Vm</td>
</tr>
</tbody>
</table>

4.9.3.2.2.3 DELETE /vm/{cloud}/{name}

Deletes a virtual machine by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error updating virtual machine</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named virtual machine or cloud could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>cloud</td>
<td>path</td>
<td>The name of the cloud</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the virtual machine</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Virtual Machine
  description: |
    Vm is used to manage virtual machines.
  termsOfService: https://github.com/cloudmesh-community/nist/blob/master/LICENSE.txt
  contact:
    name: NIST BDRA Interface Subgroup Service
    url: https://cloudmesh-community.github.io/nist/spec/
  license:
    name: Apache 2.0
    url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3
paths:
  /vm/{cloud}:
    get:
      tags: [Vm]
      summary: Returns a list of virtual machines for the cloud
      description: Returns a list of all virtual machines
      operationId: cloudmesh.vm.list
      parameters: [name: cloud]
        in: path
        required: true
        schema:
          type: string
      responses:
        '200':
          description: The list of virtual machines
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Vm'
        '400':
          description: No Vm found
        '401':
          description: Not authorized

  /vm/{cloud}/{name}:
    get:
      tags: [Vm]
      summary: Returns the named virtual machine
      description: Returns a virtual machine by name
      operationId: cloudmesh.vm.find_by_name
      parameters: [name: cloud]
        in: path
        required: true
        schema:
          type: string
      - name: name
        in: path
        required: true
        schema:
          type: string
      responses:
        '200':
          description: Returning the information of the virtual machine
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Vm'
        '400':
          description: Error updating virtual machine
        '401':
          description: Not authorized
        '404':
          description: The named virtual machine or cloud could not be found

    put:
      tags: [Vm]
      summary: Add a virtual machine
      description: Sets the named virtual machine
operationId: cloudmesh.vm.add
parameters:
  - name: cloud
    in: path
    required: true
    schema:
      type: string
      description: The name of the cloud
requestBody:
  description: The virtual machine to add or modify
  required: true
  content:
    application/json:
      schema:
        $ref: '#/components/schemas/Vm'
responses:
  '200':
    description: Vm updated
  '400':
    description: Error updating virtual machine
  '401':
    description: Not authorized
  '404':
    description: The named virtual machine or cloud could not be found

delete:
  tags:
    - Vm
  summary: Deletes the named virtual machine
  description: Deletes a virtual machine by name
  operationId: cloudmesh.vm.delete_by_name
parameters:
  - name: cloud
    description: The name of the cloud
    in: path
    required: true
    schema:
      type: string
  - name: name
    description: The name of the virtual machine
    in: path
    required: true
    schema:
      type: string
responses:
  '200':
    description: Deletion successful
  '400':
    description: Error updating virtual machine
  '401':
    description: Not authorized
  '404':
    description: The named virtual machine or cloud could not be found

components:
  schemas:
    Vm:
      type: object
      properties:
        provider:
          type: string
          description: Name of the provider
        name:
          type: string
          description: the unique name of the virtual machine
        image:
          type: string
          description: the image name for the virtual machine
        flavor:
          type: string
          description: the flavor name for the virtual machine
        region:
          type: string
          description: an optional region
        state:
          type: string
          description: The state of the virtual machine
        private_ips:
          type: string
          description: The private IPs
        public_ips:
          type: string
          description: The public IPs
        metadata:
          type: string
          description: The meta data passed along to the virtual machine

4.9.4 Secgroup
A security group defines the incoming and outgoing security rules which can then be assigned to a node. The connection to and from the node will be determined by the security group rules, in addition to any other possible rules applied on network devices or from the instance’s firewall settings. A security group may have one or multiple rules and a node may be associated with one or more security groups.

4.9.4.1 Schema Secgroup

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the security group</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>Describes what the security group is for</td>
</tr>
<tr>
<td>rules</td>
<td>array[SecGroupRule]</td>
<td>List of Security group rules</td>
</tr>
</tbody>
</table>

4.9.4.2 Schema SecGroupRule

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Unique name of the rule</td>
</tr>
<tr>
<td>ingress</td>
<td>boolean</td>
<td>The defined security group rule is for ingress if True</td>
</tr>
<tr>
<td>egress</td>
<td>boolean</td>
<td>The defined security group rule is for egress if True</td>
</tr>
<tr>
<td>remote_group</td>
<td>string</td>
<td>Name of the group if the rule is defined by group instead of IP range</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
<td>The protocol used such as TCP, UDP, ICMP</td>
</tr>
<tr>
<td>from_port</td>
<td>integer</td>
<td>Port range starting port</td>
</tr>
<tr>
<td>to_port</td>
<td>integer</td>
<td>Port range ending port</td>
</tr>
<tr>
<td>cidr</td>
<td>string</td>
<td>The source or destination network in CIDR notation,</td>
</tr>
</tbody>
</table>

4.9.4.3 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/secgroup</td>
<td>Returns all security groups</td>
</tr>
<tr>
<td>get</td>
<td>/secgroup/{name}</td>
<td>Return the security group by name</td>
</tr>
<tr>
<td>post</td>
<td>/secgroup/{name}</td>
<td>Create the named security group</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Get an existing rule from the specified</td>
</tr>
</tbody>
</table>
get /secgroup/{name}/rule/{rule} security group

put /secgroup/{name}/rule/{rule} Create or update specified security group

delete /secgroup/{name}/rule/{rule} Delete an existing rule from the specified security group

4.9.4.3.1 /secgroup

4.9.4.3.1.1 GET /secgroup

Returns all security groups

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>security group information</td>
<td>array[Secgroup]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.9.4.3.2 /secgroup/{name}

4.9.4.3.2.1 GET /secgroup/{name}

Return the security group by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>security group information</td>
<td>Secgroup</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named security group could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>name of the security group</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.9.4.3.2.2 POST /secgroup/{name}
Create a new named security group

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>The group could not be created</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the security group to create</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.9.4.3.3 /secgroup/{name}/rule/{rule}

Create a new rule in security group

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The security group rule definition info</td>
<td>Secgrouprule</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named security group or role could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The named of the security group from which the rule will be deleted</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>rule</td>
<td>path</td>
<td>The rule to be added</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.9.4.3.3.2 PUT /secgroup/{name}/rule/{rule}

Create a new rule in security group
**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Created</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named security group or role could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the new security group to create</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

**Request Body**

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The new security group rule to create</td>
<td>True</td>
<td>SecGroupRule</td>
</tr>
</tbody>
</table>

4.9.4.3.3 DELETE /secgroup/{name}/rule/{rule}

Create a new rule in security group

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deleted successfully</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named security group or role could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The named secgroup</td>
<td>True</td>
<td>String</td>
</tr>
<tr>
<td>rule</td>
<td>path</td>
<td>The secgroup rule</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.9.4.4 secgroup.yaml
A security group defines the incoming and outgoing security rules which can then be assigned to a node. The connection to and from the node will be determined by the security group rules, in addition to any other possible rules applied on network devices or from the instance's firewall settings. A security group may have one or multiple rules and a node may be associated with one or more security groups.

paths:
  /secgroup:
    get:
      tags: ["Security group"]
      summary: Returns all security groups
      description: Returns all security groups
      operationId: cloudmesh.secgroup.get
      responses:
        '200':
          description: security group information
          content:
            application/json:
              schema:
                type: array
        '401':
          description: Not authorized
  /secgroup/{name}:
    get:
      tags: ["Security group"]
      summary: Return the security group by name
      description: Return the security group by name
      operationId: cloudmesh.secgroup.get_by_name
      parameters:
        - name: name
          description: name of the security group
          in: path
          required: true
          schema:
            type: string
      responses:
        '200':
          description: security group information
          content:
            application/json:
              schema:
                $ref: "#/components/schemas/Secgroup"
        '401':
          description: Not authorized
        '404':
          description: The named security group could not be found
    post:
      tags: ["Security group"]
      summary: Create the named security group
      description: Create a new named security group
      operationId: cloudmesh.secgroup.create
      parameters:
        - in: path
          name: name
          required: true
          description: The name of the security group to create
          schema:
            type: string
      responses:
        '201':
          description: Created
        '400':
          description: The group could not be created
        '401':
          description: Not authorized
  /secgroup/{name}/rule/{rule}:
tags:
- Security group
summary: Get an existing rule from the specified security group
description: Create a new rule in security group
operationId: cloudmesh.secgroup.get_rule
parameters:
- in: path
  name: name
  required: true
  description: The named of the security group from which the rule will be deleted
  schema:
    type: string
- in: path
  name: rule
  required: true
  description: The rule to be added
  schema:
    type: string
responses:
  '200':
    description: The security group rule definition info
    content:
      application/json:
        schema:
          $ref: "#/components/schemas/SecGroupRule"
  '401':
    description: Not authorized
  '404':
    description: The named security group or role could not be found
put:
  tags:
  - Security group
summary: Create or update specified security group
description: Create a new rule in security group
operationId: cloudmesh.secgroup.add_rule
parameters:
  - in: path
    name: name
    required: true
    description: The name of the new security group to create
    schema:
      type: string
  requestBody:
    description: The new security group rule to create
    required: true
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/SecGroupRule'
responses:
  '200':
    description: Created
  '401':
    description: Not authorized
  '404':
    description: The named security group or role could not be found
delete:
  tags:
  - Security group
summary: Delete an existing rule from the specified security group
description: Create a new rule in security group
operationId: cloudmesh.secgroup.delete_rule
parameters:
- in: path
  name: name
  required: true
  description: The named secgroup
  schema:
    type: string
- in: path
  name: rule
  required: true
  description: The secgroup rule
  schema:
    type: string
responses:
  '200':
    description: Deleted sucessfully
  '401':
    description: Not authorized
  '404':
    description: The named security group or role could not be found
components:
schemas:
  Secgroup:
    type: object
    description: the security group object
    properties:
      name:
4.9.5 Nic

A resource store Network Interface Controller (NIC) information.

4.9.5.1 Schema Nic

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the network interface controller</td>
</tr>
<tr>
<td>kind</td>
<td>string</td>
<td>Kind of the network interface controller (wifi, WAN, …)</td>
</tr>
<tr>
<td>mac</td>
<td>string</td>
<td>The mac address</td>
</tr>
<tr>
<td>ip</td>
<td>string</td>
<td>The IP address</td>
</tr>
<tr>
<td>mask</td>
<td>string</td>
<td>The network mask</td>
</tr>
<tr>
<td>broadcast</td>
<td>string</td>
<td>The broadcast address</td>
</tr>
<tr>
<td>gateway</td>
<td>string</td>
<td>The gateway address</td>
</tr>
<tr>
<td>mtu</td>
<td>integer</td>
<td>The MTU of the NIC</td>
</tr>
<tr>
<td>bandwidth</td>
<td>integer</td>
<td>The bandwidth in bps</td>
</tr>
</tbody>
</table>
4.9.5.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/nic</td>
<td>Returns a list of network interface controllers</td>
</tr>
<tr>
<td>get</td>
<td>/nic/{name}</td>
<td>Returns the named network interface controller</td>
</tr>
<tr>
<td>put</td>
<td>/nic/{name}</td>
<td>Set a network interface controller</td>
</tr>
<tr>
<td>delete</td>
<td>/nic/{name}</td>
<td>Deletes the named network interface controller</td>
</tr>
</tbody>
</table>

4.9.5.2.1 /nic

4.9.5.2.1.1 GET /nic

Returns a list of all network interface controllers

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of network interface controllers</td>
<td>array[<strong>Nic</strong>]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.9.5.2.2 /nic/{name}

4.9.5.2.2.1 GET /nic/{name}

Returns a network interface controller by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the network interface controller</td>
<td><strong>Nic</strong></td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named network interface controller could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the network interface controller</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>
4.9.5.2.2 PUT /nic/{name}

Sets the named network interface controller

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Nic updated</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The new nic to create or update</td>
<td>True</td>
<td>Nic</td>
</tr>
</tbody>
</table>

4.9.5.2.3 DELETE /nic/{name}

Deletes a network interface controller by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named network interface controller could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the network interface controller</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.9.5.3 nic.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Nic
  description: |

  A resource store Network Interface Controller (NIC) information.
```
/nic:
  get:
    tags: [Nic]
    summary: Returns a list of network interface controllers
    description: Returns a list of all network interface controllers
    operationId: cloudmesh.nic.list
    responses:
      '200':
        description: The list of network interface controllers
        content:
          application/json:
            schema:
              type: array
              items:
                $ref: '#/components/schemas/Nic'
      '401':
        description: Not authorized

/nic/{name}:
  get:
    tags: [Nic]
    summary: Returns the named network interface controller
    description: Returns a network interface controller by name
    operationId: cloudmesh.nic.find_by_name
    parameters:
      - name: name
        in: path
        required: true
        schema:
          type: string
        description: The name of the network interface controller
    responses:
      '200':
        description: Returning the information of the network interface controller
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Nic'
      '401':
        description: Not authorized
      '404':
        description: The named network interface controller could not be found
  put:
    tags: [Nic]
    summary: Set a network interface controller
    description: Sets the named network interface controller
    operationId: cloudmesh.nic.add
    requestBody:
      description: The new nic to create or update
      required: true
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/Nic'
    responses:
      '200':
        description: Nic updated
      '401':
        description: Not authorized
      '404':
        description: Not authorized
  delete:
    tags: [Nic]
    summary: Deletes the named network interface controller
    description: Deletes a network interface controller by name
    operationId: cloudmesh.nic.delete_by_name
    parameters:
      - name: name
        in: path
        required: true
        schema:
          type: string
        description: The name of the network interface controller
    responses:
      '200':
        description: Deletion successful
      '401':
        description: Not authorized
      '404':
        description: Not authorized
The named network interface controller could not be found

components:
schemas:
  Nic:
    type: object
description: The network interface controller
    properties:
      name:
        type: string
description: Name of the network interface controller
      kind:
        type: string
description: Kind of the network interface controller (wifi, WAN, ...)
      mac:
        type: string
description: The mac address
      ip:
        type: string
description: The IP address
      mask:
        type: string
description: The network mask
      broadcast:
        type: string
description: The broadcast address
      gateway:
        type: string
description: The gateway address
      mtu:
        type: integer
description: The MTU of the NIC
      bandwidth:
        type: integer
description: The bandwidth in bps

4.10 Compute Management - Containers

4.10.1 Containers

Numerous different containers are likely to be created and handling them becomes more and more time consuming as their number increases. This service helps to solve that issue by storing containers and their corresponding information.

4.10.1.1 Schema Container

Reference: 🧫

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the container</td>
</tr>
<tr>
<td>version</td>
<td>string</td>
<td>Version of the container</td>
</tr>
<tr>
<td>label</td>
<td>string</td>
<td>Label of the container</td>
</tr>
<tr>
<td>type</td>
<td>string</td>
<td>Type of the container</td>
</tr>
<tr>
<td>definition</td>
<td>string</td>
<td>Definition or manifest of the container</td>
</tr>
<tr>
<td>imgURI</td>
<td>string</td>
<td>URI of the container</td>
</tr>
<tr>
<td>tags</td>
<td>array[string]</td>
<td>Tags of the container</td>
</tr>
</tbody>
</table>

4.10.1.2 Paths
<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/container</td>
<td>Returns a list of containers</td>
</tr>
<tr>
<td>get</td>
<td>/container/{name}</td>
<td>Returns the named container</td>
</tr>
<tr>
<td>put</td>
<td>/container/{name}</td>
<td>Set an container</td>
</tr>
<tr>
<td>delete</td>
<td>/container/{name}</td>
<td>Deletes the named container</td>
</tr>
</tbody>
</table>

4.10.1.2.1 /container

4.10.1.2.1.1 GET /container

Returns a list of all containers

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of container</td>
<td>array[Container]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.10.1.2.2 /container/{name}

4.10.1.2.2.1 GET /container/{name}

Returns an container by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the container</td>
<td>Container</td>
</tr>
<tr>
<td>400</td>
<td>No Container found</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named container could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the container</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.10.1.2.2.2 PUT /container/{name}
Sets the named container

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Container updated</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error updating container</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The new container to create</td>
<td>True</td>
<td>Container</td>
</tr>
</tbody>
</table>

4.10.1.2.3 DELETE /container/{name}

Deletes an container by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named container could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the container</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.10.1.3 containers.yaml

openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Containers
  description: |
  Numerous different containers are likely to be created and handling them becomes more and more time consuming as their number increases. This service helps to solve that issue by storing containers and their corresponding information.

contact:
paths:
  /container:
    get:
      tags:
        - Container
      summary: Returns a list of containers
      description: Returns a list of all containers
      operationId: cloudmesh.container.list
      responses:
        '200':
          description: The list of containers
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Container'
        '401':
          description: Not authorized
  /container/{name}:
    get:
      tags:
        - Container
      summary: Returns the named container
      description: Returns a container by name
      operationId: cloudmesh.container.find_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the container
      responses:
        '200':
          description: Returning the information of the container
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Container'
        '400':
          description: No Container found
        '401':
          description: Not authorized
        '404':
          description: The named container could not be found
    put:
      tags:
        - Container
      summary: Set an container
      description: Sets the named container
      operationId: cloudmesh.container.add
      requestBody:
        description: The new container to create
        required: true
        content:
          application/json:
            schema:
              $ref: '#/components/schemas/Container'
      responses:
        '200':
          description: Container updated
        '401':
          description: Not authorized
        '400':
          description: Error updating container
    delete:
      tags:
        - Container
      summary: Deletes the named container
      description: Deletes an container by name
      operationId: cloudmesh.container.delete_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the container
      responses:
        '200':
          description: Deletion successful
4.11 **COMPUTE MANAGEMENT - MAP REDUCE**

4.11.1 Map Reduce

A service to store the information of a mapreduce deployment definition. All of the attributes are stored as Strings.

4.11.1.1 Schema Map

**Reference:** 📦

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the map function</td>
</tr>
<tr>
<td>kind</td>
<td>string</td>
<td>The kind in which the specification is provided</td>
</tr>
<tr>
<td>content</td>
<td>string</td>
<td>The kind in which the specification is provided</td>
</tr>
</tbody>
</table>

4.11.1.2 Schema Reduce

**Reference:** 📦

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the reduce function</td>
</tr>
<tr>
<td>kind</td>
<td>string</td>
<td>The kind in which the specification is provided</td>
</tr>
</tbody>
</table>
4.11.1.3 Schema Data

**Reference:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the data</td>
</tr>
<tr>
<td>content</td>
<td>string</td>
<td>The content of the data</td>
</tr>
</tbody>
</table>

4.11.1.4 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/mapreduce</td>
<td>Returns the data identified by the mapreduce resource</td>
</tr>
<tr>
<td>get</td>
<td>/mapreduce/{name}</td>
<td>Returns the data identified by the map and function</td>
</tr>
<tr>
<td>put</td>
<td>/mapreduce/map/{name}</td>
<td>Create or update the map function</td>
</tr>
<tr>
<td>get</td>
<td>/mapreduce/map/{name}</td>
<td>Returns the data identified by the map function</td>
</tr>
<tr>
<td>put</td>
<td>/mapreduce/reduce/{name}</td>
<td>Create or update the reduce function</td>
</tr>
<tr>
<td>get</td>
<td>/mapreduce/reduce/{name}</td>
<td>Returns the data identified by the reduce function</td>
</tr>
</tbody>
</table>

4.11.1.4.1 /mapreduce

4.11.1.4.1.1 GET /mapreduce

Returns the data identified by the mapreduce resource

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>mapreduce names</td>
<td>array[String]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.11.1.4.2 /mapreduce/{name}

4.11.1.4.2.1 GET /mapreduce/{name}

Returns the data identified by the mapreduce resource
Returns the data identified by the reduce function.

### Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The data identified by reduce</td>
<td>array[Data]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The resource could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the function</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.11.1.4.3 /mapreduce/map/{name}

4.11.1.4.3.1 PUT /mapreduce/map/{name}

Create or update the map function

### Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The map function was created or updated</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The resource could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the function</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

### Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The new default to create</td>
<td>True</td>
<td>Map</td>
</tr>
</tbody>
</table>

4.11.1.4.3.2 GET /mapreduce/map/{name}
Returns the data identified by the map function

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The data identified by map</td>
<td>array[Data]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The resource could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the function</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.11.1.4.4 /mapreduce/reduce/{name}

4.11.1.4.4.1 PUT /mapreduce/reduce/{name}

Create or update the reduce function

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The reduce function was created or updated</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The resource could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the function</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The new default to create</td>
<td>True</td>
<td>Reduce</td>
</tr>
</tbody>
</table>

4.11.1.4.4.2 GET /mapreduce/reduce/{name}
Returns the data identified by the reduce function

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The data identified by reduce</td>
<td>array[Data]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The resource could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the function</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.11.1.5 mapreduce.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: MapReduce
  description: A service to store the information of a mapreduce deployment definition.
  All of the attributes are stored as Strings.
  contact:
    name: NIST BDRA Interface Subgroup
    url: https://cloudmesh-community.github.io/nist
  license:
    name: Apache 2.0
    url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
- url: /cloudmesh/v3
paths:
  /mapreduce:
    get:
      tags:
        - mapreduce
      summary: Returns the data identified by the mapreduce resource
      description: Returns the data identified by the mapreduce resource
      operationId: cloudmesh.mapreduce.list
      responses:
        '200':
          description: mapreduce names
          content:
            application/json:
              schema:
                type: array
                items:
                  type: string
        '401':
          description: Not authorized
  /mapreduce/{name}:
    get:
      tags:
        - mapreduce
      summary: Returns the data identified by the map and function
      description: Returns the data identified by the reduce function.
      operationId: cloudmesh.mapreduce.get
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
```
description: The name of the function

responses:
  '200':
    description: The data identified by reduce
    content:
      application/json:
        schema:
          type: array
          items:
            $ref: '#/components/schemas/Data'

  '401':
    description: Not authorized
  '404':
    description: The resource could not be found

/mapreduce/map/{name}:
put:
  tags:
  - mapreduce
  summary: Create or update the map function
  description: Create or update the map function
  operationId: cloudmesh.mapreduce.map.put
  parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
      description: The name of the function
  requestBody:
    description: The new default to create
    required: true
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/Map'
  responses:
    '200':
      description: The map function was created or updated
    '401':
      description: Not authorized
    '404':
      description: The resource could not be found

get:
  tags:
  - mapreduce
  summary: Returns the data identified by the map function
  description: Returns the data identified by the map function
  operationId: cloudmesh.mapreduce.map.get
  parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
      description: The name of the function
  responses:
    '200':
      description: The data identified by map
      content:
        application/json:
          schema:
            type: array
            items:
              $ref: '#/components/schemas/Data'
    '401':
      description: Not authorized
    '404':
      description: The resource could not be found

/mapreduce/reduce/{name}:
put:
  tags:
  - mapreduce
  summary: Create or update the reduce function
  description: Create or update the reduce function
  operationId: cloudmesh.mapreduce.reduce.put
  parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
      description: The name of the function
  requestBody:
    description: The new default to create
    required: true
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/Reduce'
4.12 Compute Management - Functions

4.12.1 Microservice

As part of microservices, a function with parameters that can be invoked has been defined.

4.12.1.1 Schema Microservice
<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the microservice</td>
</tr>
<tr>
<td>endpoint</td>
<td>string</td>
<td>The end point of the microservice</td>
</tr>
<tr>
<td>function</td>
<td>string</td>
<td>The function the microservice represents</td>
</tr>
<tr>
<td>description</td>
<td></td>
<td>The description of the microservice</td>
</tr>
</tbody>
</table>

4.12.1.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/microservice</td>
<td>Returns a list of microservices</td>
</tr>
<tr>
<td>get</td>
<td>/microservice/{name}</td>
<td>Returns the named microservice</td>
</tr>
<tr>
<td>put</td>
<td>/microservice/{name}</td>
<td>Set an microservice</td>
</tr>
<tr>
<td>delete</td>
<td>/microservice/{name}</td>
<td>Deletes the named microservice</td>
</tr>
</tbody>
</table>

4.12.1.2.1 /microservice

4.12.1.2.1.1 GET /microservice

Returns a list of all microservices

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of microservices</td>
<td>array[Microservice]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.12.1.2.2 /microservice/{name}

4.12.1.2.2.1 GET /microservice/{name}

Returns the named microservice

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returns the microservice</td>
<td>Microservice</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
<td>Schema</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------</td>
<td>--------</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named microservice could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the microservice</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.12.1.2.2 PUT /microservice/{name}

Sets the named microservice

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Microservice updated</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error updating microservice</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The new microservice to create</td>
<td>True</td>
<td>Microservice</td>
</tr>
</tbody>
</table>

4.12.1.2.3 DELETE /microservice/{name}

Deletes an microservice by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error deleting microservice</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named microservice could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters
As part of microservices, a function with parameters that can be invoked has been defined.

Terms of Service: "https://github.com/cloudmesh-nist/blob/master/LICENSE.txt"

Contact:
name: NIST BDRA Interface Subgroup
url: https://cloudmesh-community.github.io/nist
License:
name: Apache 2.0
url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt

Servers:
- url: /cloudmesh/v3

Paths:
/microservice:
get:
tags:
  - Microservice
summary: Returns a list of microservices
description: Returns a list of all microservices
operationId: cloudmesh.microservice.list
responses:
  '200':
    description: The list of microservices
    content:
      application/json:
        schema: $ref: '#/components/schemas/Microservice'
  '401':
    description: Not authorized

/microservice/{name}:
get:
tags:
  - Microservice
summary: Returns the named microservice
description: Returns the named microservice
operationId: cloudmesh.microservice.find_by_name
parameters:
  - name: name
    in: path
    required: true
    schema:
      type: string
    description: The name of the microservice
responses:
  '200':
    description: Returns the microservice
    content:
      application/json:
        schema: $ref: '#/components/schemas/Microservice'
  '401':
    description: Not authorized
  '404':
    description: The named microservice could not be found

put:
tags:
  - Microservice
summary: Set an microservice
description: Sets the named microservice
operationId: cloudmesh.microservice.add
requestBody:
  description: The new microservice to create
  required: true
  content:
    application/json:
      schema: $ref: '#/components/schemas/Microservice'
responses:
4.13 Reservation

4.13.1 Reservation

Some services may consume a considerable amount of resources, necessitating the reservation of resources.

4.13.1.1 Schema Reservation

Reference: ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the reservation</td>
</tr>
<tr>
<td>service</td>
<td>string</td>
<td>The name of the service for which the reservation is applied</td>
</tr>
<tr>
<td>description</td>
<td>string</td>
<td>The description of the reservation</td>
</tr>
<tr>
<td>start</td>
<td>string(date)</td>
<td>The start time and date</td>
</tr>
<tr>
<td>end</td>
<td>string(date)</td>
<td>The end time and date</td>
</tr>
</tbody>
</table>
4.13.1.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/reservation</td>
<td>Returns a list of reservations</td>
</tr>
<tr>
<td>get</td>
<td>/reservation/{name}</td>
<td>Returns the named reservation</td>
</tr>
<tr>
<td>put</td>
<td>/reservation/{name}</td>
<td>Uploads a reservation to the list of reservations</td>
</tr>
<tr>
<td>delete</td>
<td>/reservation/{name}</td>
<td>Deletes the named reservation</td>
</tr>
</tbody>
</table>

4.13.1.2.1 /reservation

4.13.1.2.1.1 GET /reservation

Returns a list of all reservations

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of reservations</td>
<td>array[Reservation]</td>
</tr>
<tr>
<td>400</td>
<td>No Reservations found</td>
<td>String</td>
</tr>
</tbody>
</table>

4.13.1.2.2 /reservation/{name}

4.13.1.2.2.1 GET /reservation/{name}

Returns an reservation by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the reservation</td>
<td>Reservation</td>
</tr>
<tr>
<td>400</td>
<td>No reservation found</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named reservation could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the reservation</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>
4.13.1.2.2 PUT /reservation/{name}

Uploads a reservation to the list of reservations

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Reservation updated</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error updating reservation</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The reservation to be uploaded</td>
<td>True</td>
<td>Reservation</td>
</tr>
</tbody>
</table>

4.13.1.2.3 DELETE /reservation/{name}

Deletes an reservation by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>No reservation found</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named reservation could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the reservation</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.13.1.3 reservation.yaml

openapi: '3.0.2'
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Reservation
  description: |
  - Some services may consume a considerable amount of resources, necessitating the reservation of resources.
paths:
/reservation:
get:
  tags:
    - Reservation
  summary: Returns a list of reservations
  description: Returns a list of all reservations
  operationId: cloudmesh.reservation.list
  responses:
    '200':
      description: The list of reservations
      content:
        application/json:
          schema:
            type: array
            items:
              $ref: '#/components/schemas/Reservation'
    '400':
      description: No Reservations found
/reservation/{name}:
get:
  tags:
    - Reservation
  summary: Returns the named reservation
  description: Returns an reservation by name
  operationId: cloudmesh.reservation.find_by_name
  parameters:
    - name: name
      in: path
      required: true
      schema:
        type: string
        description: The name of the reservation
  responses:
    '200':
      description: Returning the information of the reservation
      content:
        application/json:
          schema:
            $ref: '#/components/schemas/Reservation'
    '400':
      description: No reservation found
    '401':
      description: Not authorized
    '404':
      description: The named reservation could not be found
put:
  tags:
    - Reservation
  summary: Uploads a reservation to the list of reservations
  description: Uploads a reservation to the list of reservations
  operationId: cloudmesh.reservation.add
  requestBody:
    description: The reservation to be uploaded
    required: true
    content:
      application/json:
        schema:
          $ref: '#/components/schemas/Reservation'
  responses:
    '200':
      description: Reservation updated
    '400':
      description: Error updating reservation
delete:
  tags:
    - Reservation
  summary: Deletes the named reservation
  description: Deletes an reservation by name
  operationId: cloudmesh.reservation.delete_by_name
  parameters:
    - name: name
      in: path
      required: true
      schema:
        type: string
        description: The name of the reservation
  responses:
    '200':
      description: Deletion successful
    '400':
      description:
4.14 Data Streams

4.14.1 Stream

The stream object describes a data flow, providing information about the rate and number of items exchanged while issuing requests to the stream. A stream may return data items in a specific format that is defined by the stream.

4.14.1.1 Schema Stream

Reference: ☁

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the stream</td>
</tr>
<tr>
<td>format</td>
<td>string</td>
<td>Format of the stream</td>
</tr>
<tr>
<td>rate</td>
<td>integer</td>
<td>The rate of messages</td>
</tr>
<tr>
<td>limit</td>
<td>integer</td>
<td>The limit of items send</td>
</tr>
<tr>
<td>endpoint</td>
<td>string</td>
<td>The endpoint of the stream</td>
</tr>
<tr>
<td>protocol</td>
<td>string</td>
<td>DThe definition of the protocol used</td>
</tr>
</tbody>
</table>

4.14.1.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/stream</td>
<td>Returns a list of streams</td>
</tr>
<tr>
<td>Method</td>
<td>Path</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-----------------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>get</td>
<td>/stream/{name}</td>
<td>Returns the named stream</td>
</tr>
<tr>
<td>put</td>
<td>/stream/{name}</td>
<td>Set an stream</td>
</tr>
<tr>
<td>delete</td>
<td>/stream/{name}</td>
<td>Deletes the named stream</td>
</tr>
</tbody>
</table>

4.14.1.2.1 /stream

4.14.1.2.1.1 GET /stream

Returns a list of all streams

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of streams</td>
<td>array[Stream]</td>
</tr>
<tr>
<td>400</td>
<td>No Stream found</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.14.1.2.2 /stream/{name}

4.14.1.2.2.1 GET /stream/{name}

Returns an stream by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the stream</td>
<td>Stream</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named stream could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the stream</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.14.1.2.2.2 PUT /stream/{name}

Sets the named stream
Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Stream updated</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The new stream to create</td>
<td>True</td>
<td><code>Stream</code></td>
</tr>
</tbody>
</table>

4.14.1.2.2.3 DELETE /stream/{name}

Deletes an stream by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named stream could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the stream</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.14.1.3 stream.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Stream
description: |
  The stream object describes a data flow, providing information about the rate and number of items exchanged while issuing requests to the stream. A stream may return data items in a specific format that is defined by the stream.

contact:
  name: NIST BDRA Interface Subgroup
  url: https://cloudmesh-community.github.io/nist
license:
  name: Apache 2.0
  url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3
```
paths:
  /stream:
    get:
      tags:
        - Stream
      summary: Returns a list of streams
      description: Returns a list of all streams
      operationId: cloudmesh.stream.list
      responses:
        '200':
          description: The list of streams
          content:
            application/json:
              schema: $ref: '#/components/schemas/Stream'
        '400':
          description: No Stream found
        '401':
          description: Not authorized
  /stream/{name}:
    get:
      tags:
        - Stream
      summary: Returns the named stream
      description: Returns an stream by name
      operationId: cloudmesh.stream.find_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the stream
      responses:
        '200':
          description: Returning the information of the stream
          content:
            application/json:
              schema: $ref: '#/components/schemas/Stream'
        '401':
          description: Not authorized
        '404':
          description: The named stream could not be found
    put:
      tags:
        - Stream
      summary: Set an stream
      description: Sets the named stream
      operationId: cloudmesh.stream.add
      requestBody:
        description: The new stream to create
        required: true
        content:
          application/json:
            schema: $ref: '#/components/schemas/Stream'
      responses:
        '200':
          description: Stream updated
        '401':
          description: Not authorized
        '404':
          description: The named stream could not be found
    delete:
      tags:
        - Stream
      summary: Deletes the named stream
      description: Deletes an stream by name
      operationId: cloudmesh.stream.delete_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the stream
      responses:
        '200':
          description: Deletion successful
        '401':
          description: Not authorized
        '404':
          description: The named stream could not be found
components:
schemas:
  Stream:
    type: object
    description: The stream
    properties:
4.14.2 Filter

Filters can operate on a variety of objects and reduce the information received based on a search criterion.

4.14.2.1 Schema Filter

Reference: ☁️

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>Name of the filter</td>
</tr>
<tr>
<td>function</td>
<td>string</td>
<td>The function used to filter the data in the stream</td>
</tr>
<tr>
<td>kind</td>
<td>string</td>
<td>The filter kind or type</td>
</tr>
</tbody>
</table>

4.14.2.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/filter</td>
<td>Returns a list of filters</td>
</tr>
<tr>
<td>get</td>
<td>/filter/{name}</td>
<td>Returns the named filter</td>
</tr>
<tr>
<td>put</td>
<td>/filter/{name}</td>
<td>Set an filter</td>
</tr>
<tr>
<td>delete</td>
<td>/filter/{name}</td>
<td>Deletes the named filter</td>
</tr>
</tbody>
</table>

4.14.2.2.1 /filter

4.14.2.2.1.1 GET /filter

Returns a list of all filters

Responses
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of filterses</td>
<td>array[Filter]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.14.2.2.2 /filter/{name}

4.14.2.2.2.1 GET /filter/{name}

Returns an filter by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the filter</td>
<td>Filter</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named filter could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the filter</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.14.2.2.2 PUT /filter/{name}

Sets the named filter

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Filter updated</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>400</td>
<td>Error updating filter</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The new filter to create</td>
<td>True</td>
<td>Filter</td>
</tr>
</tbody>
</table>
### 4.14.2.2.2.3 DELETE /filter/{name}

Deletes a filter by name.

**Responses**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named filter could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

**Parameters**

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the filter</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

### 4.14.2.3 filter.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Filter
  description: |
    Filters can operate on a variety of objects and reduce the information received based on a search criterion.
  contact:
    name: NIST BDRA Interface Subgroup
    url: https://cloudmesh-community.github.io/nist
  license:
    name: Apache 2.0
    url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
- url: /cloudmesh/v3
paths:
  /filter:
    get:
      tags:
        - Filter
      summary: Returns a list of filters
      description: Returns a list of all filters
      operationId: cloudmesh.filter.list
      responses:
        '200':
          description: The list of filters
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Filter'
        '401':
          description: Not authorized
  /filter/{name}:
    get:
      tags:
        - Filter
      summary: Returns the named filter
      description: Returns an filter by name
      operationId: cloudmesh.filter.find_by_name
      parameters:
        - name: name
```
4.15 **DEPLOYMENT**

4.15.1 Deployment

A resource to store software deployments. The deployment is formulated in a specification file. To distinguish the format of the specification file a string is used that defines the kind of the deployment. In case the specification uses an external service an endpoint to the service can be
used and the name of the specification is used to identify the deployment.

4.15.1.1 Schema Deployment

**Reference:**

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>string</td>
<td>The name of the deployment</td>
</tr>
<tr>
<td>kind</td>
<td>string</td>
<td>The kind of the deployment</td>
</tr>
<tr>
<td>specification</td>
<td>string</td>
<td>The specification of the deployment</td>
</tr>
<tr>
<td>endpoint</td>
<td>string</td>
<td>The location of the deployment service</td>
</tr>
<tr>
<td>endpointname</td>
<td>string</td>
<td>in case an endpoint is used, the endpointname is used to uniquely identify the deployment within the endpoint defined service</td>
</tr>
</tbody>
</table>

4.15.1.2 Paths

<table>
<thead>
<tr>
<th>HTTP</th>
<th>Path</th>
<th>Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>get</td>
<td>/deployment</td>
<td>Returns a list of deployments</td>
</tr>
<tr>
<td>get</td>
<td>/deployment/{name}</td>
<td>Returns the named deployment</td>
</tr>
<tr>
<td>put</td>
<td>/deployment/{name}</td>
<td>Set an deployment</td>
</tr>
<tr>
<td>delete</td>
<td>/deployment/{name}</td>
<td>Deletes the named deployment</td>
</tr>
</tbody>
</table>

4.15.1.2.1 /deployment

4.15.1.2.1.1 GET /deployment

Returns a list of all deployments

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The list of deployments</td>
<td>array[Deployment]</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

4.15.1.2.2 /deployment/{name}

4.15.1.2.2.1 GET /deployment/{name}
Returns an deployment by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Returning the information of the deployment</td>
<td>Deployment</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named deployment could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>

Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the deployment</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

4.15.1.2.2 PUT /deployment/{name}

Sets the named deployment

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deployment updated</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Body

<table>
<thead>
<tr>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>The new deployment to create</td>
<td>True</td>
<td>Deployment</td>
</tr>
</tbody>
</table>

4.15.1.2.2.3 DELETE /deployment/{name}

Deletes an deployment by name

Responses

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Deletion successful</td>
<td>String</td>
</tr>
<tr>
<td>401</td>
<td>Not authorized</td>
<td>String</td>
</tr>
<tr>
<td>404</td>
<td>The named deployment could not be found</td>
<td>String</td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Name</th>
<th>Located in</th>
<th>Description</th>
<th>Required</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>path</td>
<td>The name of the deployment</td>
<td>True</td>
<td>String</td>
</tr>
</tbody>
</table>

#### 4.15.1.3 deployment.yaml

```yaml
openapi: "3.0.2"
info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Deployment
description: |
  A resource to store software deployments. The deployment is formulated in a specification file. To distinguish the format of the specification file a string is used that defines the kind of the deployment. In case the specification uses an external service an endpoint to the service can be used and the name of the specification is used to identify the deployment.
contact:
  name: NIST BDRA Interface Subgroup
  url: https://cloudmesh-community.github.io/nist
license:
  name: Apache 2.0
  url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3
paths:
  /deployment:
    get:
      tags:
        - Deployment
      summary: Returns a list of deployments
      description: Returns a list of all deployments
      operationId: cloudmesh.deployment.list
      responses:
        '200':
          description: The list of deployments
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Deployment'
        '401':
          description: Not authorized
  /deployment/{name}:
    get:
      tags:
        - Deployment
      summary: Returns the named deployment
      description: Returns an deployment by name
      operationId: cloudmesh.deployment.find_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the deployment
      responses:
        '200':
          description: Returning the information of the deployment
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Deployment'
        '401':
          description: Not authorized
        '404':
          description: The named deployment could not be found
    put:
      tags:
        - Deployment
```

- openapi: "3.0.2"
- info:
  version: 3.2.0
  x-date: 17-06-2019
  x-status: defined
  title: Deployment
description: |
  A resource to store software deployments. The deployment is formulated in a specification file. To distinguish the format of the specification file a string is used that defines the kind of the deployment. In case the specification uses an external service an endpoint to the service can be used and the name of the specification is used to identify the deployment.
contact:
  name: NIST BDRA Interface Subgroup
  url: https://cloudmesh-community.github.io/nist
license:
  name: Apache 2.0
  url: https://github.com/cloudmesh/cloudmesh-nist/blob/master/LICENSE.txt
servers:
  - url: /cloudmesh/v3
paths:
  /deployment:
    get:
      tags:
        - Deployment
      summary: Returns a list of deployments
      description: Returns a list of all deployments
      operationId: cloudmesh.deployment.list
      responses:
        '200':
          description: The list of deployments
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Deployment'
        '401':
          description: Not authorized
  /deployment/{name}:
    get:
      tags:
        - Deployment
      summary: Returns the named deployment
      description: Returns an deployment by name
      operationId: cloudmesh.deployment.find_by_name
      parameters:
        - name: name
          in: path
          required: true
          schema:
            type: string
            description: The name of the deployment
      responses:
        '200':
          description: Returning the information of the deployment
          content:
            application/json:
              schema:
                $ref: '#/components/schemas/Deployment'
        '401':
          description: Not authorized
        '404':
          description: The named deployment could not be found
    put:
      tags:
        - Deployment
```
summary: Set an deployment
description: Sets the named deployment
operationId: cloudmesh.deployment.add
requestBody:
description: The new deployment to create
required: true
content:
application/json:
schema:
  $ref: '#/components/schemas/Deployment'
responses:
  '200':
    description: Deployment updated
  '401':
    description: Not authorized
delete:
tags:
  - Deployment
summary: Deletes the named deployment
description: Deletes an deployment by name
operationId: cloudmesh.deployment.delete_by_name
parameters:
  - name: name
    in: path
    required: true
    type: string
    description: The name of the deployment
responses:
  '200':
    description: Deletion successful
  '401':
    description: Not authorized
  '404':
    description: The named deployment could not be found
components:
schemas:
  Deployment:
    type: object
    description: the deployment
    properties:
      name:
        type: string
        description: The name of the deployment
      kind:
        type: string
        description: The kind of the deployment
      specification:
        type: string
        description: The specification of the deployment
      endpoint:
        type: string
        description: The location of the deployment service
      endpointname:
        type: string
        description: in case an endpoint is used, the endpointname is used to uniquly identify the deployment within the endpoint defined service
The following acronyms and terms are used in this volume.

**ACID**

Atomicity, Consistency, Isolation, Durability

**API**

Application Programming Interface

**ASCII**

American Standard Code for Information Interchange

**BASE**

Basically Available, Soft state, Eventual consistency

**Container**


**Cloud Computing**

The practice of using a network of remote servers hosted on the Internet to store, manage, and process data, rather than a local server or a personal computer. See [http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf](http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf)

**DevOps**

A clipped compound of software DEVELOpment and information technology OPerationS

**Deployment**

The action of installing software on resources

**HTTP**

HyperText Transfer Protocol HTTPS HTTP Secure

**Hybrid**

IaaS

Infrastructure as a Service SaaS Software as a Service

ITL

Information Technology Laboratory

Microservice Architecture

Is an approach to build applications based on many smaller modular services. Each module supports a specific goal and uses a simple, well-defined interface to communicate with other sets of services.

NBD-PWG

NIST Big Data Public Working Group

NBDRA

NIST Big Data Reference Architecture

NBDRAI

NIST Big Data Reference Architecture Interface

NIST

National Institute of Standards and Technology

OS

Operating System

REST

REpresentational State Transfer

Replica

A duplicate of a file on another resource to avoid costly transfer costs in case of frequent access.

Serverless Computing

Serverless computing specifies the paradigm of function as a service (FaaS). It is a cloud computing code execution model in which a cloud provider manages the function deployment and utilization while clients can utilize them. The charge model is based on
execution of the function rather than the cost to manage and host the VM or container.

Software Stack

A set of programs and services that are installed on a resource to support applications.

Virtual File System

An abstraction layer on top of a distributed physical file system to allow easy access to the files by the user or application.

Virtual Machine

A VM is a software computer that, like a physical computer, runs an operating system and applications. The VM is composed of a set of specification and configuration files and is backed by the physical resources of a host.

Virtual Cluster

A virtual cluster is a software cluster that integrate either VMs, containers, or physical resources into an agglomeration of compute resources. A virtual cluster allows users to authenticate and authorize to the virtual compute nodes to utilize them for calculations. Optional high-level services that can be deployed on a virtual cluster may simplify interaction with the virtual cluster or provide higher-level services.

Workflow

The sequence of processes or tasks

WWW

World Wide Web
BIBLIOGRAPHY


https://bigdatawg.nist.gov/show_InputDoc.php

[10] W. L. Chang (Co-Chair), R. Reinsch (Subgroup Co-chair), C. Austin (Editor), and NIST Big Data Public Working Group, “NIST Big Data Interoperability Framework: Volume 9, Adoption and Modernization (NIST SP 1500-10 VERSION 2),” National Institute of Standards; Technology (NIST), Gaithersburg, MD, Jul. 2019 [Online]. Available: 
https://bigdatawg.nist.gov/show_InputDoc.php

https://bigdatawg.nist.gov/V1_output_docs.php

https://bigdatawg.nist.gov/V2_output_docs.php


