Projetos e APIs

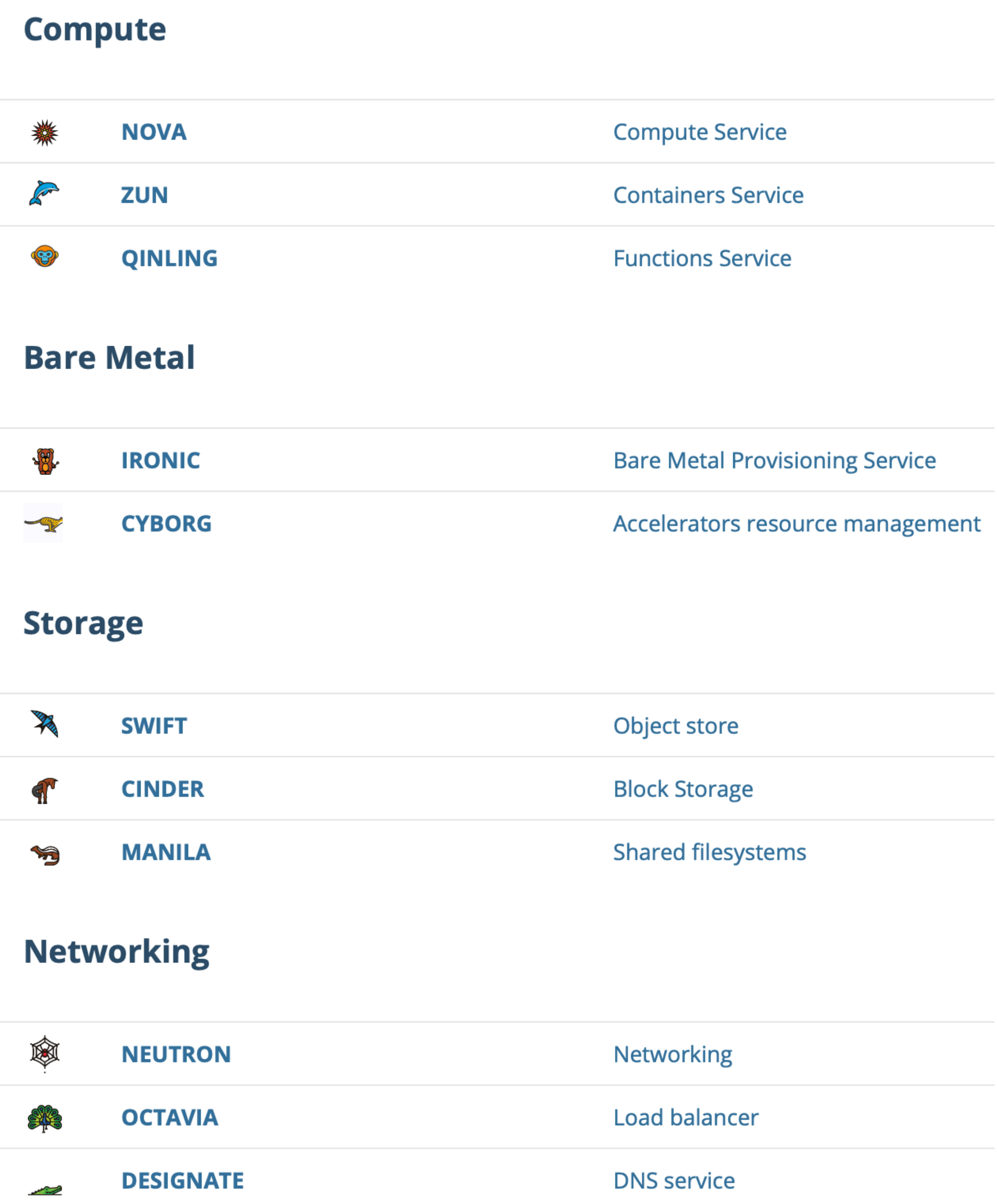
Aula Teórica nº3

2020/2021

OpenStack

- Open source cloud computing platform for all types of clouds
- Aims to be simple to implement, massively scalable, and feature rich
- Developers and cloud computing technologists from around the world create the OpenStack project
- OpenStack provides an Infrastructure-as-a-Service (IaaS) solution through a set of interrelated services
- Each service offers an application programming interface (API) that facilitates this integration
- It’s possible to install some or all services

OpenStack Services



Compute

NOVA	Compute Service
ZUN	Containers Service
QINLING	Functions Service

Bare Metal

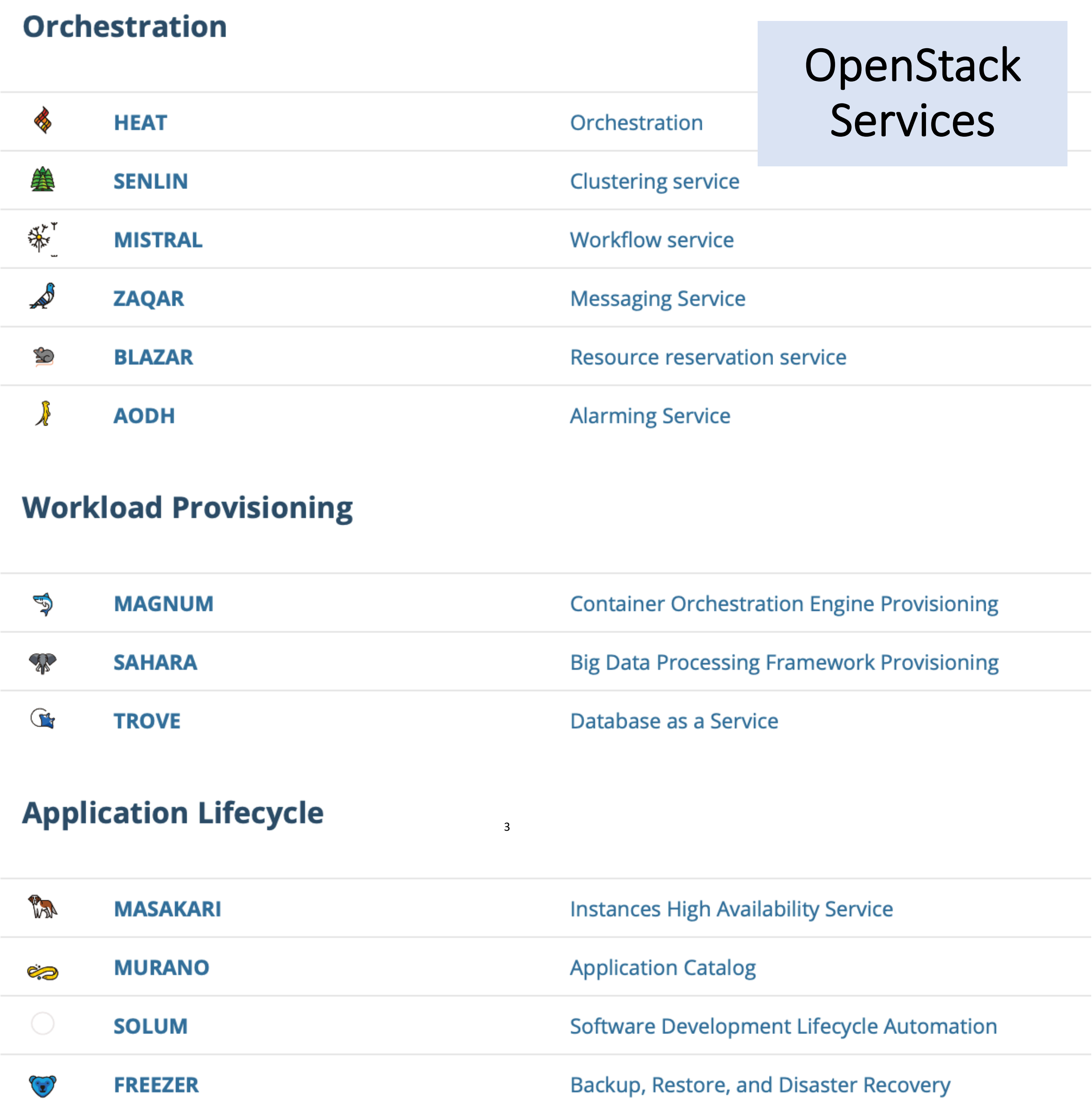
IRONIC	Bare Metal Provisioning Service
CYBORG	Accelerators resource management

Storage

SWIFT	Object store
CINDER	Block Storage
MANILA	Shared filesystems

Networking

NEUTRON	Networking
OCTAVIA	Load balancer
DESIGNATE	DNS service



Orchestration

HEAT	Orchestration
SENLIN	Clustering service
MISTRAL	Workflow service
ZAQAR	Messaging Service
BLAZAR	Resource reservation service
AODH	Alarming Service

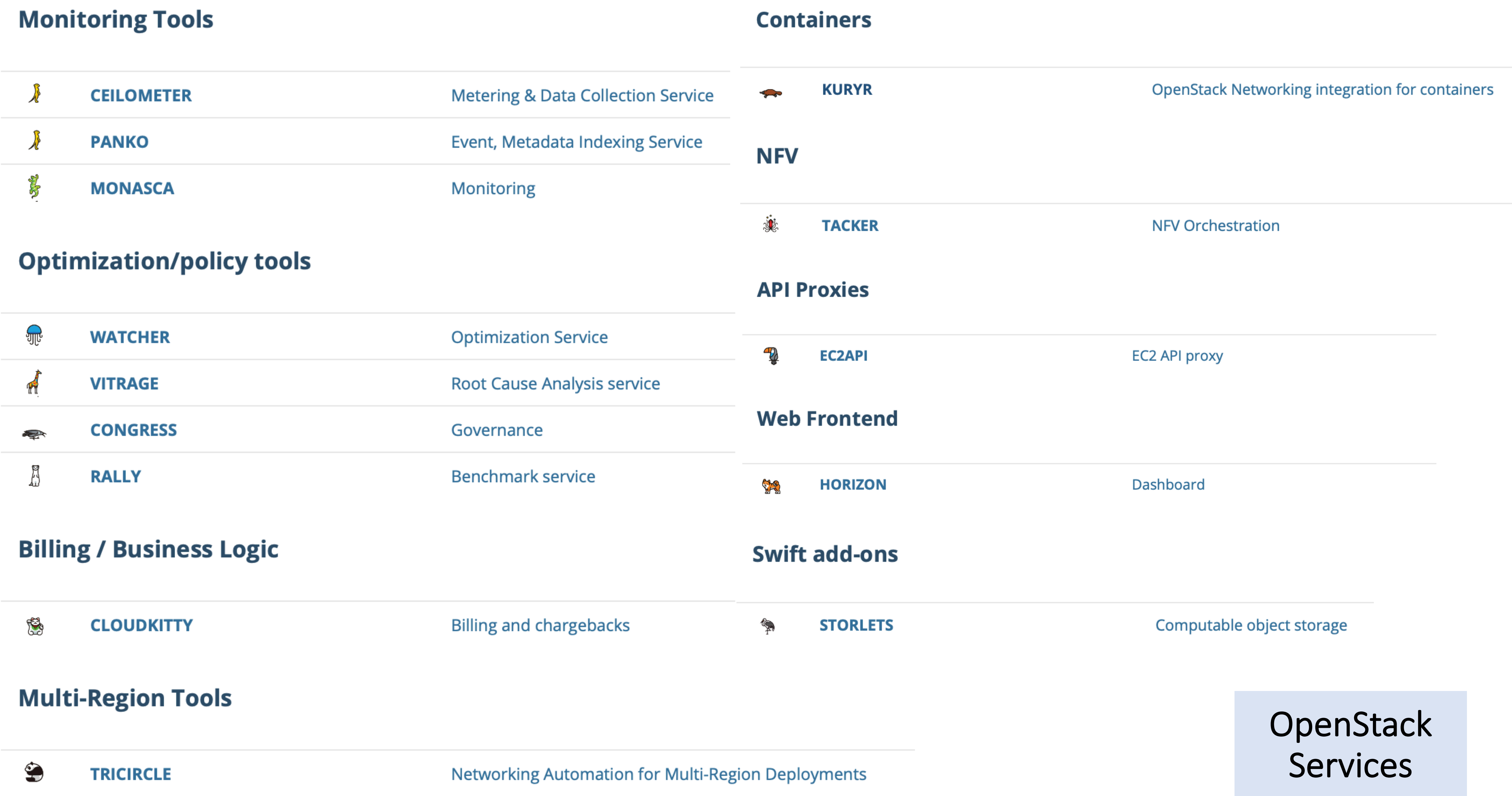
Workload Provisioning

MAGNUM	Container Orchestration Engine Provisioning
SAHARA	Big Data Processing Framework Provisioning
TROVE	Database as a Service

Application Lifecycle

MASAKARI	Instances High Availability Service
MURANO	Application Catalog
SOLUM	Software Development Lifecycle Automation
FREEZER	Backup, Restore, and Disaster Recovery

OpenStack Services



Monitoring Tools

CEILOMETER	Metering & Data Collection Service
PANKO	Event, Metadata Indexing Service
MONASCA	Monitoring

Optimization/policy tools

WATCHER	Optimization Service
VITRAGE	Root Cause Analysis service
CONGRESS	Governance
RALLY	Benchmark service

Billing / Business Logic

CLOUDKITTY	Billing and chargebacks

Multi-Region Tools

TRICIRCLE	Networking Automation for Multi-Region Deployments

Containers

KURYR	OpenStack Networking integration for containers

NFV

TACKER	NFV Orchestration

API Proxies

EC2API	EC2 API proxy

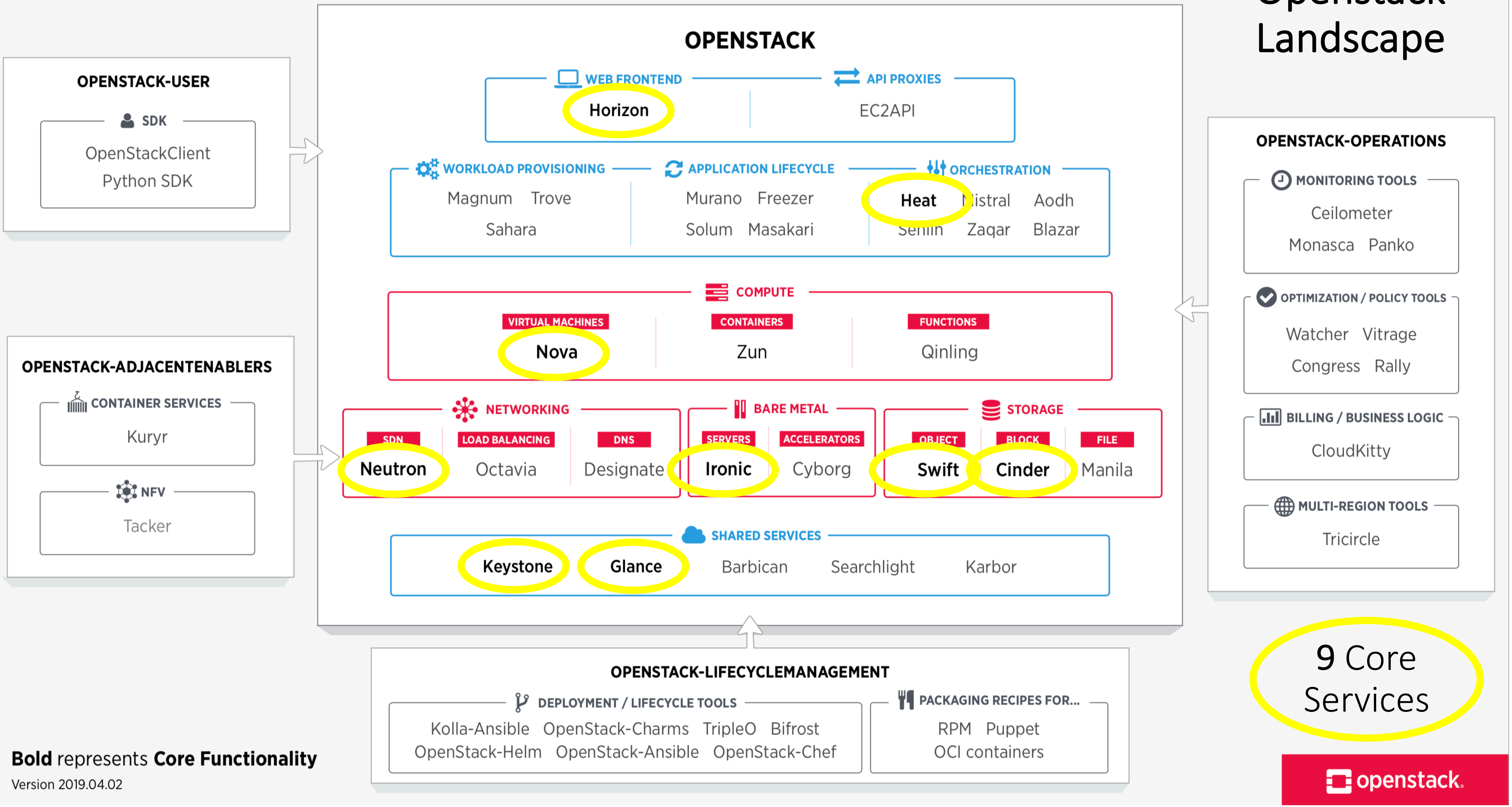
Web Frontend

HORIZON	Dashboard

Swift add-ons

STORLETS	Computable object storage

Openstack Landscape



OPENSTACK

WEB FRONTEND
- **Horizon**

API PROXIES
- EC2API

WORKLOAD PROVISIONING
- Magnum Trove
- Sahara

APPLICATION LIFECYCLE
- Murano Freezer
- Solum Masakari

ORCHESTRATION
- **Heat** Mistral Aodh
- Senlin Zaqar Blazar

COMPUTE
- VIRTUAL MACHINES: **Nova**
- CONTAINERS: Zun
- FUNCTIONS: Qinling

NETWORKING
- SDN: **Neutron**
- LOAD BALANCING: Octavia
- DNS: Designate

BARE METAL
- SERVERS: **Ironic**
- ACCELERATORS: Cyborg

STORAGE
- OBJECT: **Swift**
- BLOCK: **Cinder**
- FILE: Manila

SHARED SERVICES
- **Keystone** **Glance** Barbican Searchlight Karbor

OPENSTACK-USER

SDK
- OpenStackClient
- Python SDK

OPENSTACK-ADJACENTENABLERS

CONTAINER SERVICES
- Kuryr

NFV
- Tacker

OPENSTACK-OPERATIONS

MONITORING TOOLS
- Ceilometer
- Monasca Panko

OPTIMIZATION / POLICY TOOLS
- Watcher Vitrage
- Congress Rally

BILLING / BUSINESS LOGIC
- CloudKitty

MULTI-REGION TOOLS
- Tricircle

OPENSTACK-LIFECYCLEMANAGEMENT

DEPLOYMENT / LIFECYCLE TOOLS
- Kolla-Ansible OpenStack-Charms TripleO Bifrost
- OpenStack-Helm OpenStack-Ansible OpenStack-Chef

PACKAGING RECIPES FOR...
- RPM Puppet
- OCI containers

9 Core Services

Bold represents **Core Functionality**

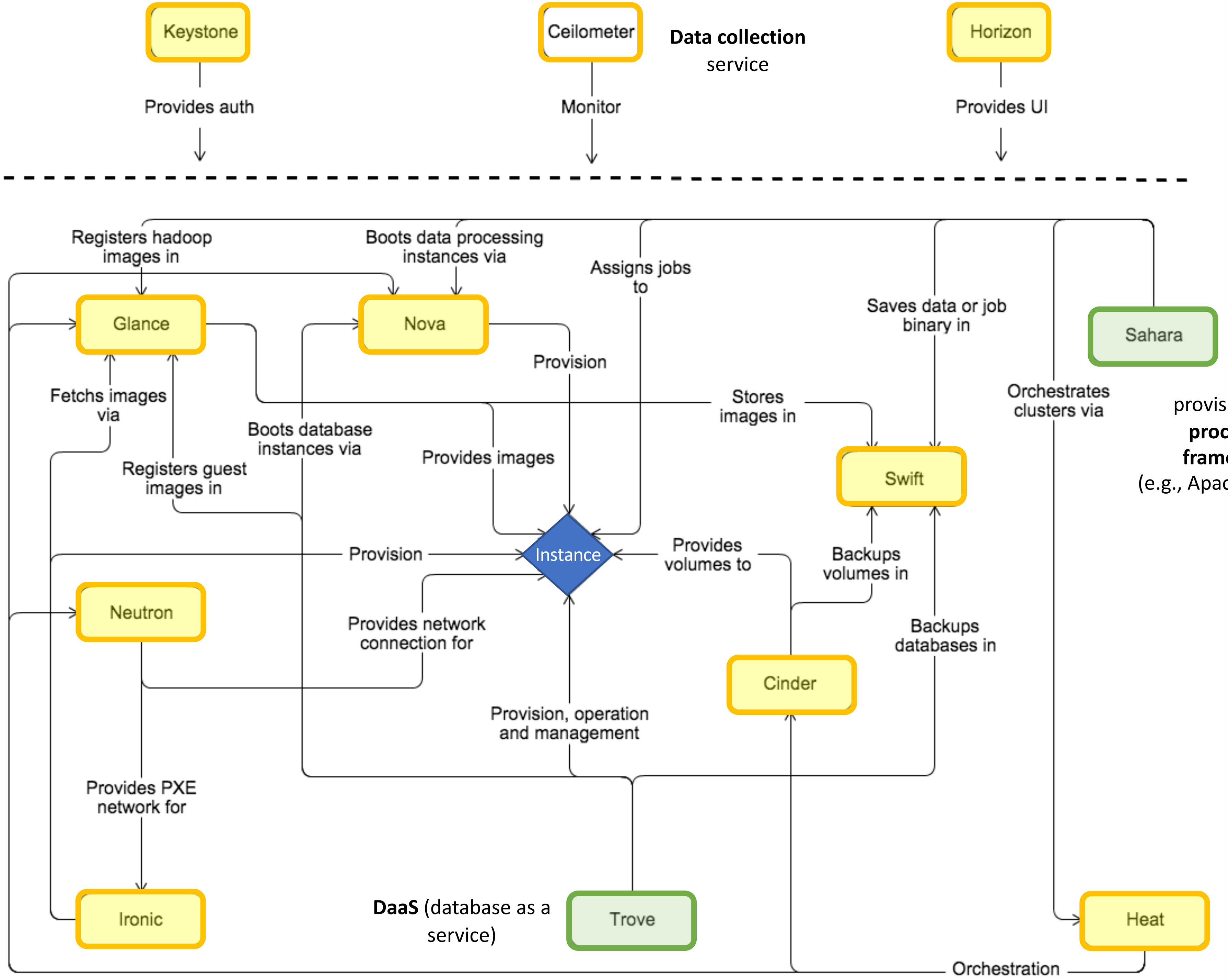
Version 2019.04.02

openstack.

OpenStack Services

- OpenStack consists of several independent parts, named the OpenStack services
- Those services work together and include the **Compute**, **Identity**, **Networking**, **Image**, **Block Storage**, **Object Storage**, **Telemetry**, **Orchestration**, and **Database** services
- It’s possible to install any of these projects separately and configure them stand-alone or as connected entities
- Individual services interact with each other through public APIs, except where privileged administrator commands are necessary
- OpenStack services are composed of several processes
- For communication between the processes of one service, an AMQP message broker is used
- The service’s state is stored in a database

OpenStack Conceptual Architecture

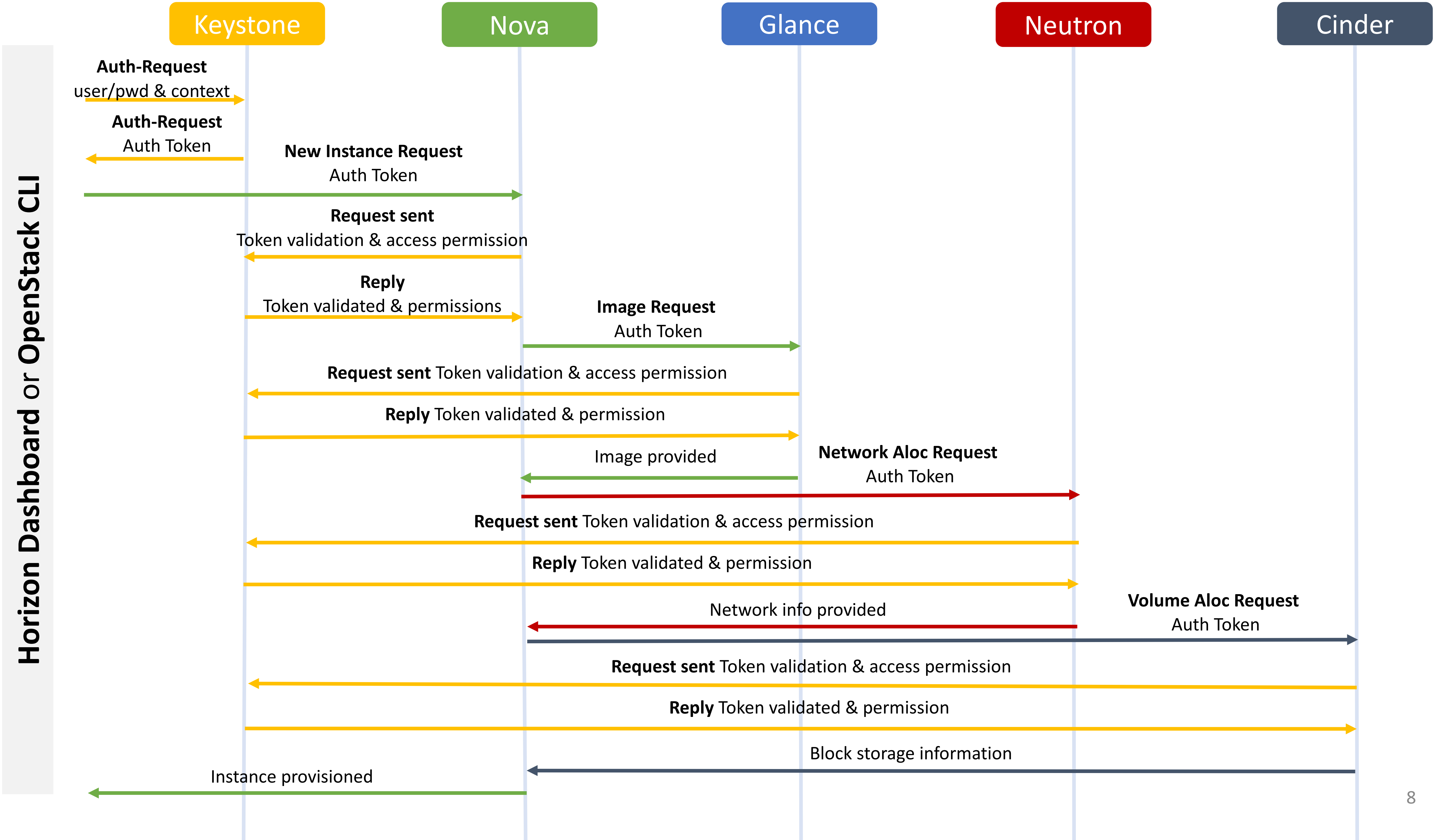


Data collection service

provision **data processing frameworks** (e.g., Apache Hadoop)

DaaS (database as a service)

Instance creation Step-by-Step



Horizon Dashboard or **OpenStack CLI**

Keystone

Nova

Glance

Neutron

Cinder

Auth-Request user/pwd & context

Auth-Request Auth Token

New Instance Request Auth Token

Request sent Token validation & access permission

Reply Token validated & permissions

Image Request Auth Token

Request sent Token validation & access permission

Reply Token validated & permission

Image provided

Network Aloc Request Auth Token

Request sent Token validation & access permission

Reply Token validated & permission

Network info provided

Volume Aloc Request Auth Token

Request sent Token validation & access permission

Reply Token validated & permission

Block storage information

Instance provisioned

Inside Nova: the processes (1/2)

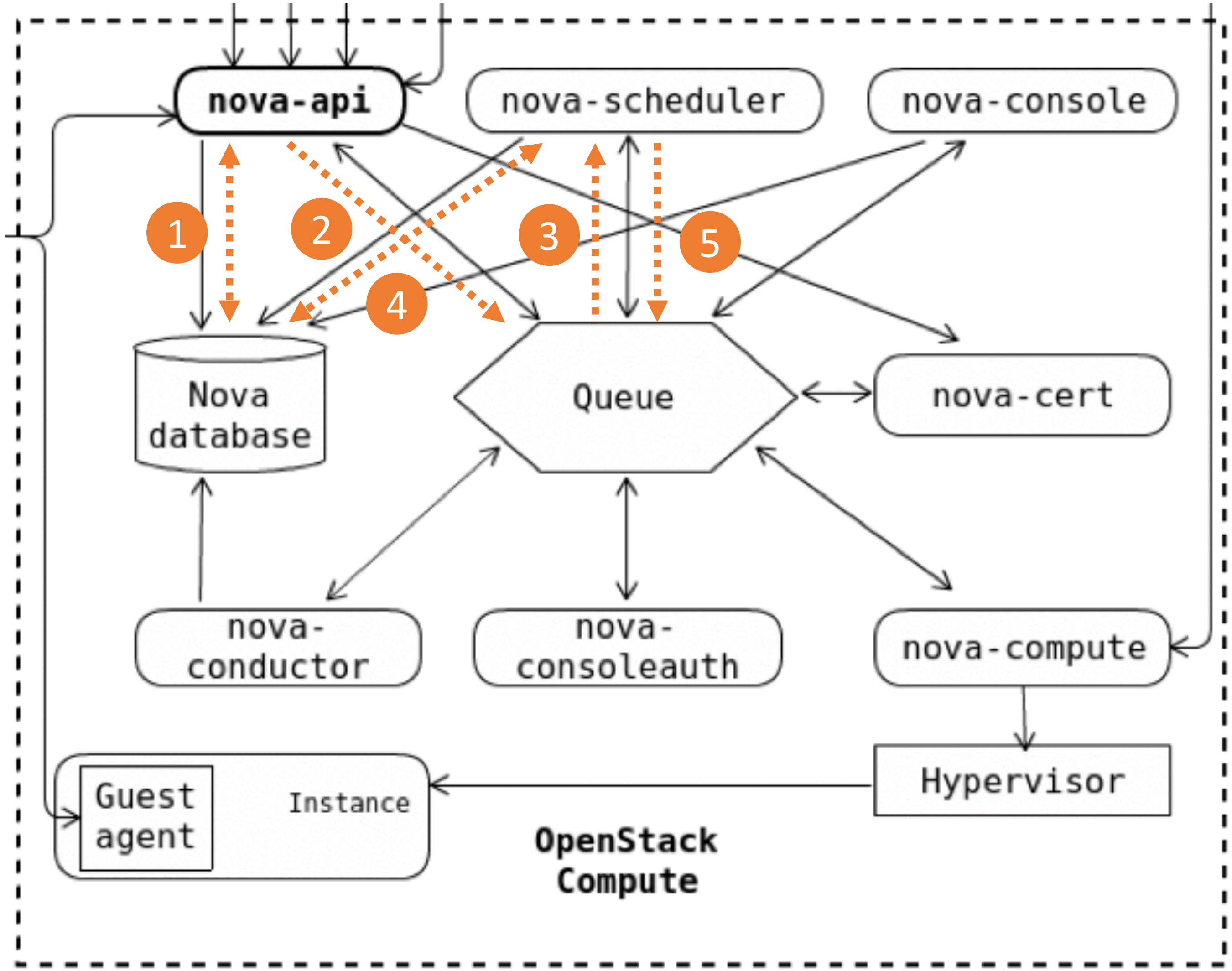
1. After getting the response from keystone, then nova-api checks for conflicts with nova-database and then it creates initial database entry for new instance or VM.

2. nova-api sends the rpc.call request to nova-scheduler expecting to get updated instance entry with host ID specified

3. nova-scheduler picks the request from the queue

4. nova-scheduler talks to nova-database to locate an appropriate host ID using filtering and weighing mechanism

5. nova-scheduler sends the rpc.cast request to nova-compute for launching an instance on the appropriate host



Inside Nova: the processes (2/2)

6. nova-compute picks the request from the queue and it sends the rpc.call request to nova-conductor to get the VM or instance info such as host ID and flavor (RAM,CPU and Disk)

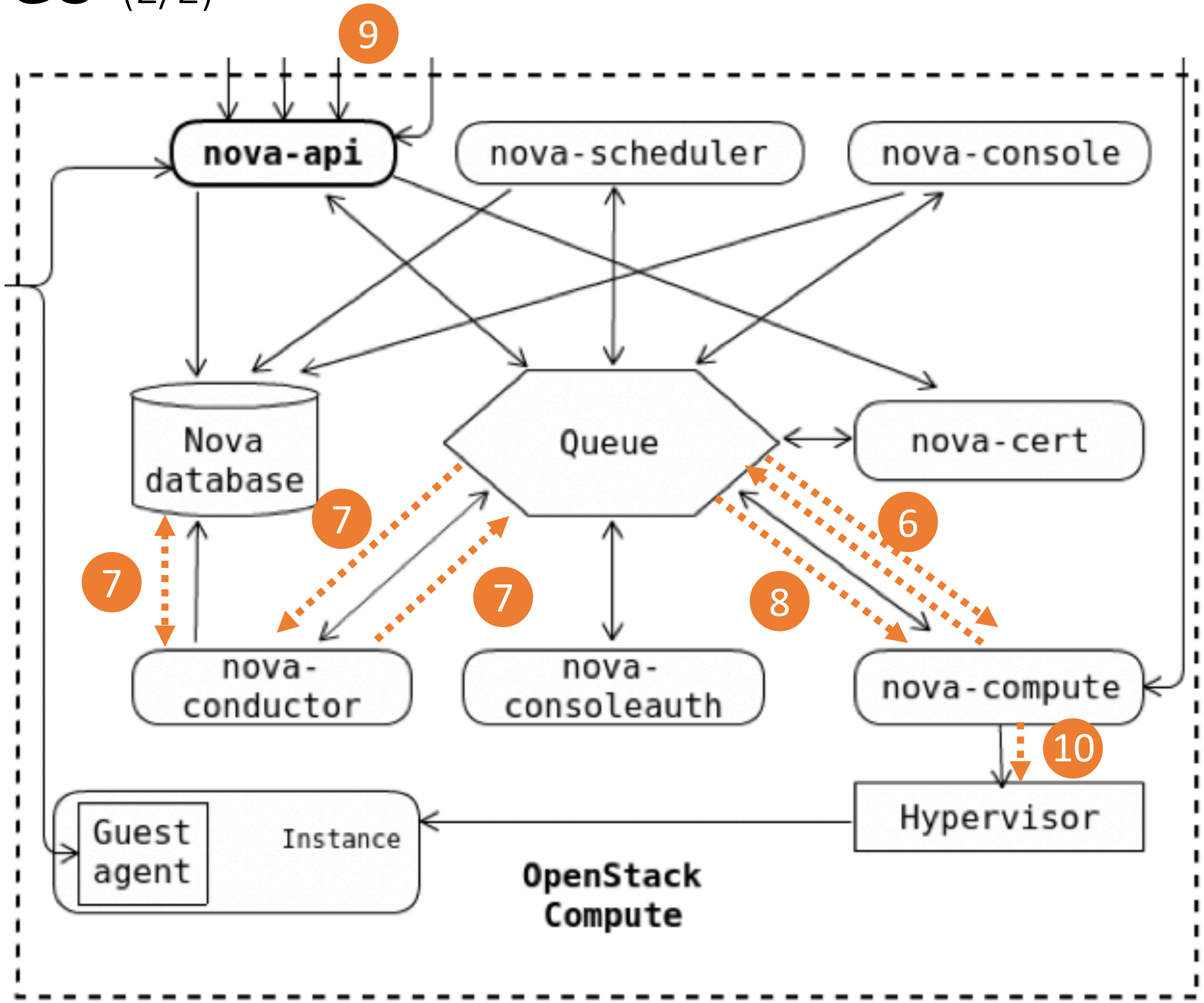
7. nova-conductor takes the request from queue and communicate with nova-database, to get the instance information and sends the information to the queue

8. nova-compute picks the instance information from the queue

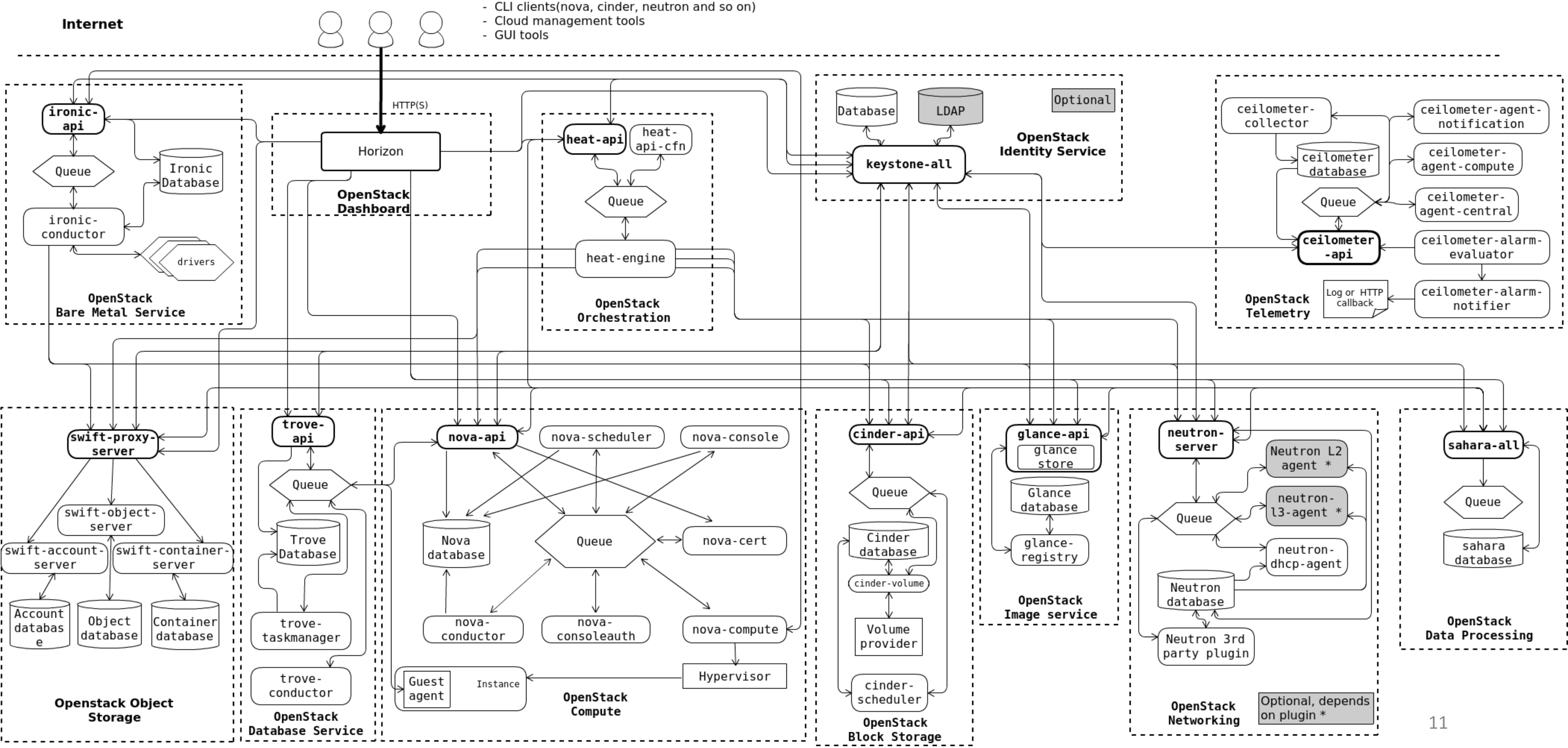
9. Through API nova-compute gets the image, networking and block storage information needed for the instance:

Glance Neutron Cinder

10. nova-compute generates data for the hypervisor driver and executes the request on the hypervisor using libvirt or API and then finally a VM is created on the hypervisor.

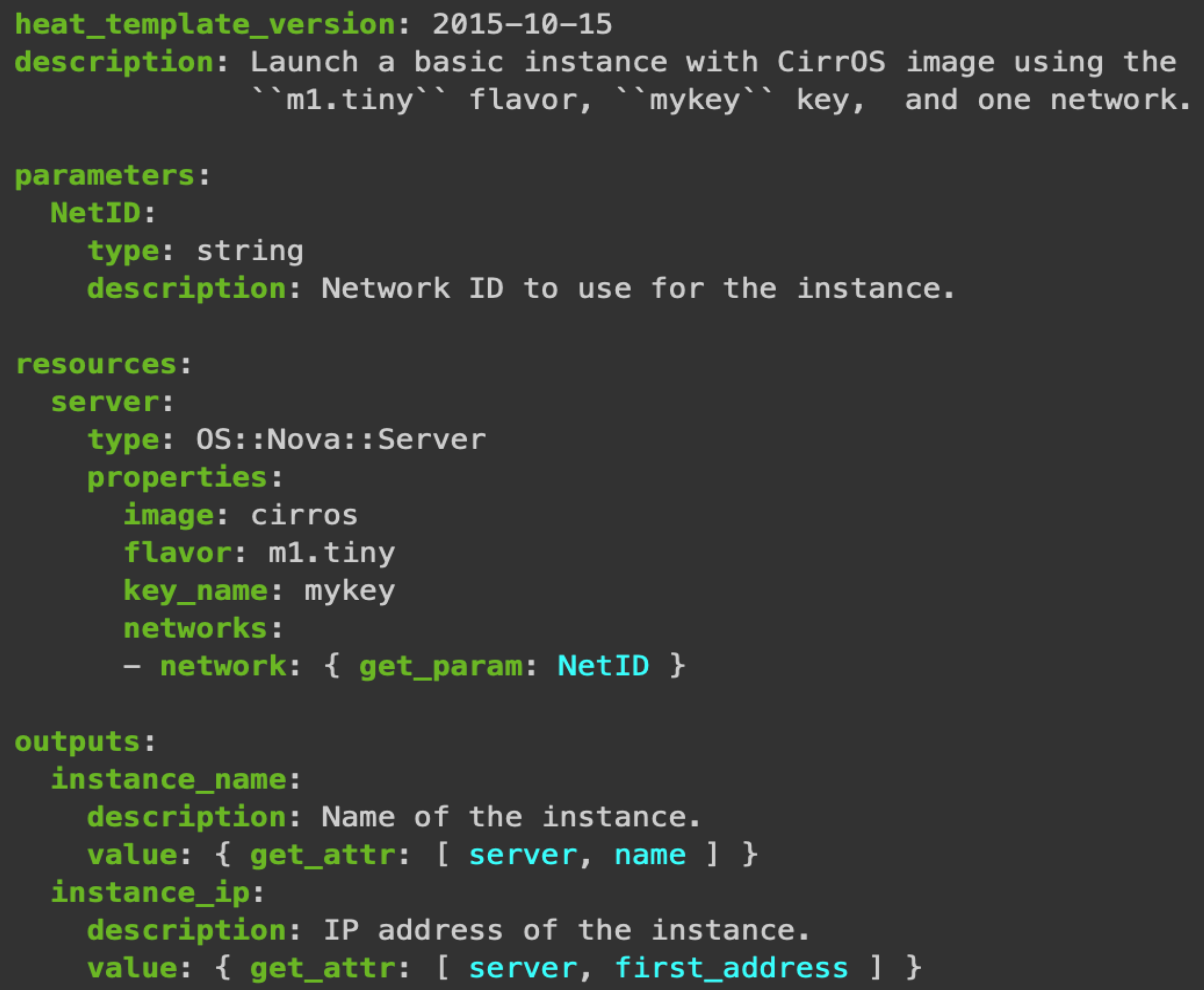


OpenStack Logical Architecture



Orchestration Service (Heat)

- Heat provides a template based orchestration for describing a cloud application
- A Heat template describes the infrastructure for a cloud application in text files which are readable and writable by humans (YAML)
- Templates specify the relationships between resources (e.g. this volume is connected to this server)
- The templates allow creation of most OpenStack resource types (such as instances, floating ips, volumes, security groups, users, etc)
- Heat primarily manages infrastructure, but the templates integrate well with SW configuration management tools such as Puppet and Ansible

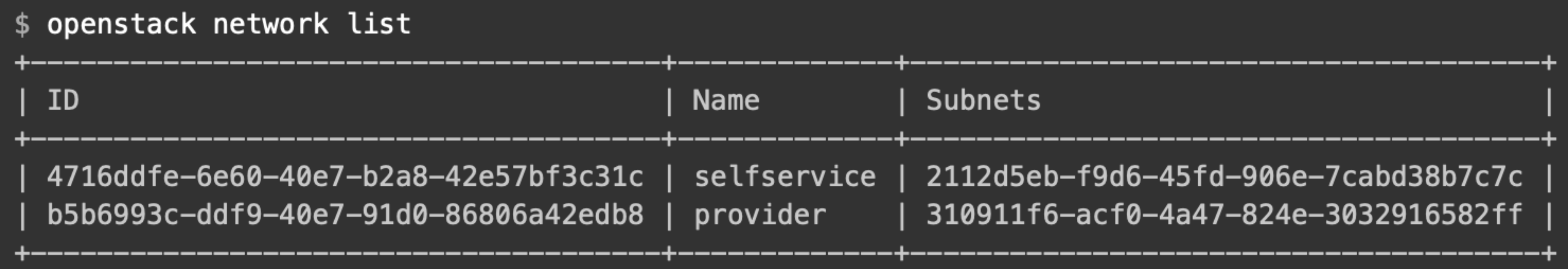


```yaml
heat_template_version: 2015-10-15
description: Launch a basic instance with CirrOS image using the
             ``m1.tiny`` flavor, ``mykey`` key,  and one network.

parameters:
  NetID:
    type: string
    description: Network ID to use for the instance.

resources:
  server:
    type: OS::Nova::Server
    properties:
      image: cirros
      flavor: m1.tiny
      key_name: mykey
      networks:
      - network: { get_param: NetID }

outputs:
  instance_name:
    description: Name of the instance.
    value: { get_attr: [ server, name ] }
  instance_ip:
    description: IP address of the instance.
    value: { get_attr: [ server, first_address ] }
```

Heat – Launch an instance

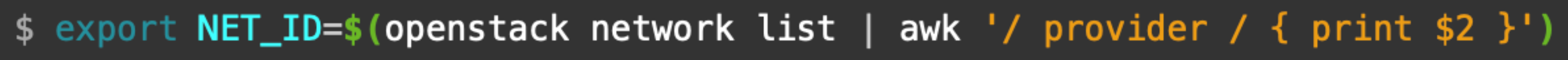
1. Create the `demo-template.yml` file with the content in this slide

2. Determine available networks:



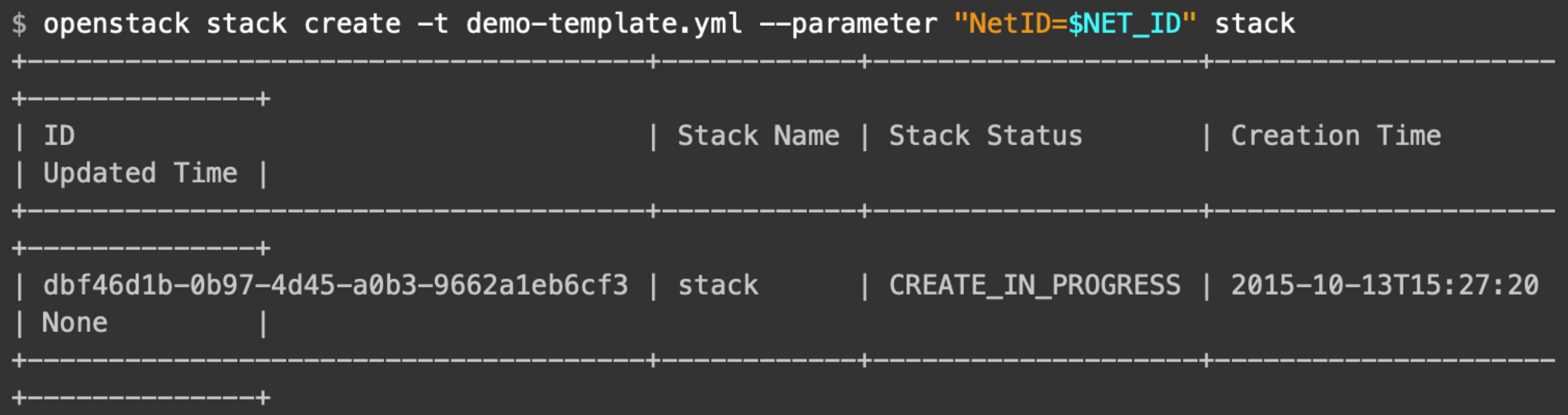
```
$ openstack network list
+--------------------------------------+-------------+--------------------------------------+
| ID                                   | Name        | Subnets                              |
+--------------------------------------+-------------+--------------------------------------+
| 4716ddfe-6e60-40e7-b2a8-42e57bf3c31c | selfservice | 2112d5eb-f9d6-45fd-906e-7cabd38b7c7c |
| b5b6993c-ddf9-40e7-91d0-86806a42edb8 | provider    | 310911f6-acf0-4a47-824e-3032916582ff |
+--------------------------------------+-------------+--------------------------------------+
```

3. Set the NET_ID environment variable to reflect the ID of a network. For example, using the provider network:



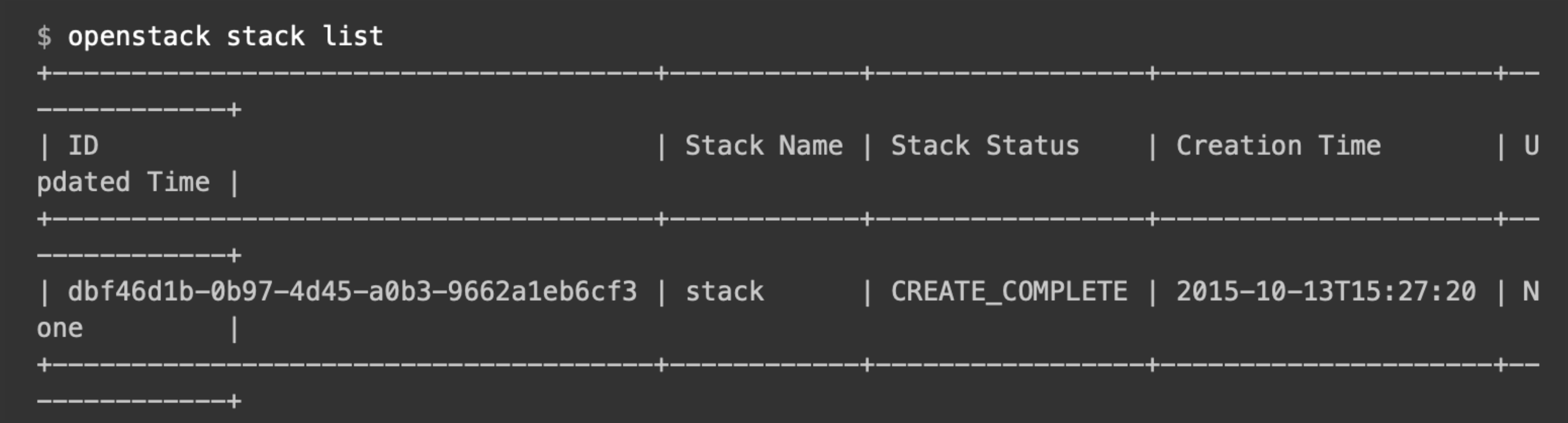
```
$ export NET_ID=$(openstack network list | awk '/ provider / { print $2 }')
```

4. Create a stack of one CirrOS instance on the provider network:



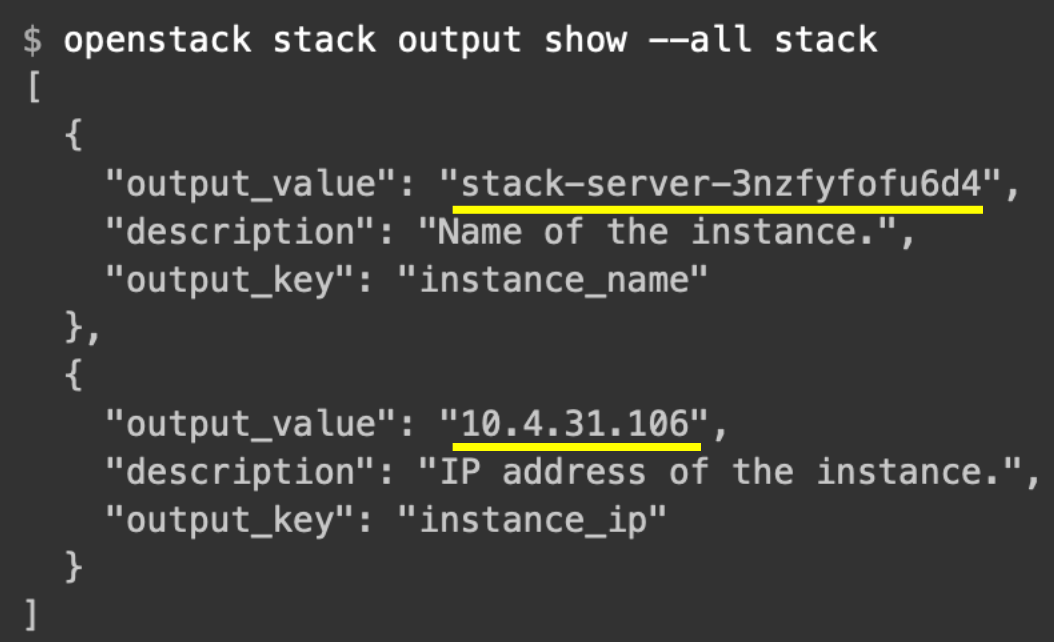
```
$ openstack stack create -t demo-template.yml --parameter "NetID=$NET_ID" stack
+--------------------------------------+------------+--------------------+---------------------
+--------------+
| ID                                   | Stack Name | Stack Status       | Creation Time
| Updated Time |
+--------------------------------------+------------+--------------------+---------------------
+--------------+
| dbf46d1b-0b97-4d45-a0b3-9662a1eb6cf3 | stack      | CREATE_IN_PROGRESS | 2015-10-13T15:27:20
| None         |
+--------------------------------------+------------+--------------------+---------------------
+--------------+
```

5. After a short time, verify successful creation of the stack:



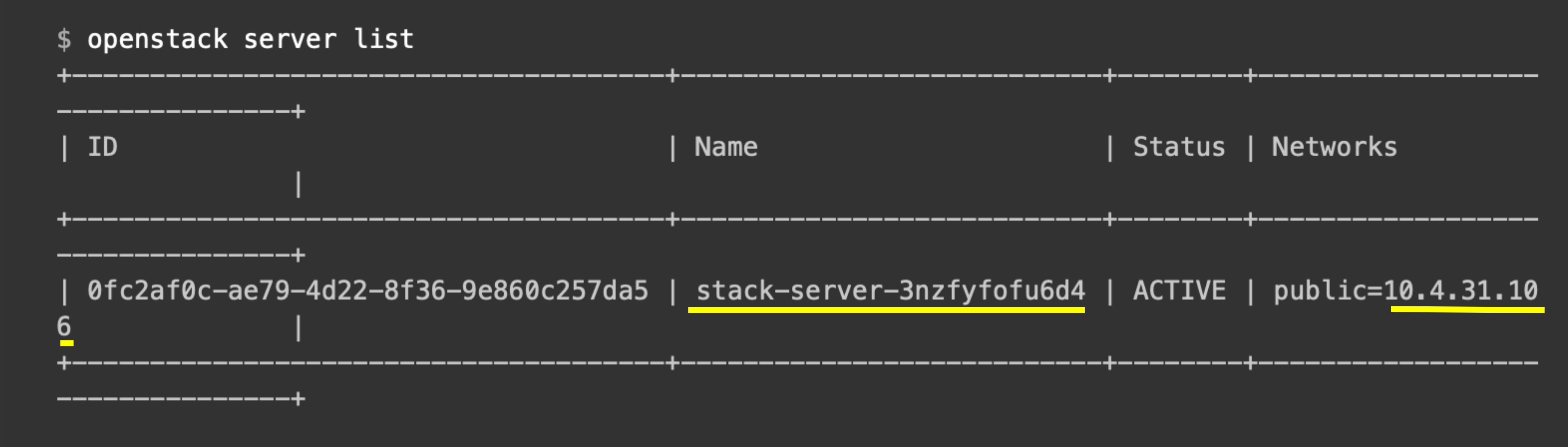
```
$ openstack stack list
+--------------------------------------+------------+-----------------+---------------------+--
------------+
| ID                                   | Stack Name | Stack Status    | Creation Time       | U
pdated Time |
+--------------------------------------+------------+-----------------+---------------------+--
------------+
| dbf46d1b-0b97-4d45-a0b3-9662a1eb6cf3 | stack      | CREATE_COMPLETE | 2015-10-13T15:27:20 | N
one         |
+--------------------------------------+------------+-----------------+---------------------+--
------------+
```

6. Show the name and IP address of the instance…



```
$ openstack stack output show --all stack
[
  {
    "output_value": "stack-server-3nzfyfofu6d4",
    "description": "Name of the instance.",
    "output_key": "instance_name"
  },
  {
    "output_value": "10.4.31.106",
    "description": "IP address of the instance.",
    "output_key": "instance_ip"
  }
]
```

7. … and compare with the output of the OpenStack client:



```
$ openstack server list
+--------------------------------------+---------------------------+--------+-------------------
---------------+
| ID                                   | Name                      | Status | Networks
               |
+--------------------------------------+---------------------------+--------+-------------------
---------------+
| 0fc2af0c-ae79-4d22-8f36-9e860c257da5 | stack-server-3nzfyfofu6d4 | ACTIVE | public=10.4.31.10
6              |
+--------------------------------------+---------------------------+--------+-------------------
---------------+
```

Methods to send API requests

- cURL

  A command-line tool that lets you send HTTP requests and receive responses.

- OpenStack command-line client

  The OpenStack project provides a command-line client that enables you to access APIs through easy-to-use commands

- REST clients

  Browser-based graphical interfaces for REST in Firefox and Chrome, AdvancedRestClient, Postman, etc

- OpenStack Python Software Development Kit (SDK)

  Use this SDK to write Python automation scripts that create and manage resources in your OpenStack cloud. The SDK implements Python bindings to the OpenStack API, which enables you to perform automation tasks in Python by making calls on Python objects rather than making REST calls directly

Openstack Toolkits 1/2

- **Go**
  - Gophercloud provides a Go binding to OpenStack cloud APIs.
- **Java**
  - OpenStack4j A fluent Java OpenStack API.
  - OpenStack Java SDK is a Java binding for the OpenStack APIs.
  - Apache jclouds is a Multi-cloud SDK for Java with OpenStack support
- **JavaScript**
  - pkgcloud is a Multi-cloud library for Node.js that supports OpenStack
  - jstack is a JavaScript client library for the OpenStack API.
  - js-openclient is a very opinionated core client which can be used in either Node.js or in the browser (browser support not yet complete) to communicate with a RESTful APIs, including but not limited to any OpenStack-compatible API. (last updated 2015)
  - node-openstack-wrapper is a convenience wrapper for many of Openstack's common features with a focus on projects/tenants.
- **.NET**
  - OpenStack.NET is a .NET SDK for OpenStack.

Openstack Toolkits 2/2

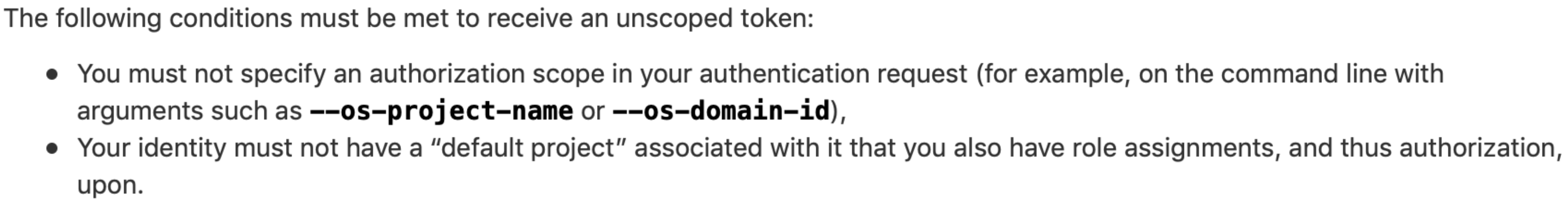
- **PHP**
  - php-opencloud/openstack is a PHP SDK for OpenStack. It is actively maintained and supports latest OpenStack identity api (Keystone v3).
  - SDK covers following api: BlockStorage v2, Compute v2, Identity Keystone v2 and v3, Images v2, Networking v2 including Layer3 extension and Security Groups extension, Object Storage v1, Gnocchi v1
- **Python**
  - OpenStack Shade shade is a simple client library for operating OpenStack clouds that abstracts deployer differences
  - The SDK-Development/PythonOpenStackSDK project is a proposed solution to offering an SDK that provides a single point of entry for consumers, and a base from which other tools can be built upon, such as command-line interfaces.
  - The OpenStackClients are the native Python bindings for the OpenStack APIs. They are used to implement the command-line interfaces (which ship with the library).
  - Apache libcloud is a Python library that abstracts away differences among multiple cloud provider APIs.
- **Ruby**
  - fog is a Multi-cloud Ruby library with support for OpenStack
  - Misty is a dedicated HTTP client for OpenStack APIs
  - Aviator An elegantly designed OpenStack SDK for Ruby (hasn't been updated since 2015)

Keystone Tokens

- Tokens are used to authenticate and authorize interactions with the various OpenStack APIs
- Tokens can express your authorization in different scopes
- A user may have different sets of roles, in different projects, and in different domains
- While tokens always express one identity, they may only ever express one set of roles in one authorization scope at a time

Unscoped Tokens

- An unscoped token contains neither roles, a project scope, nor a domain scope
- Their primary use case is simply to prove your identity to keystone at a later time, usually to generate scoped tokens, without repeatedly presenting your original credentials



The following conditions must be met to receive an unscoped token:

- You must not specify an authorization scope in your authentication request (for example, on the command line with arguments such as **`--os-project-name`** or **`--os-domain-id`**),
- Your identity must not have a “default project” associated with it that you also have role assignments, and thus authorization, upon.

Scoped Tokens

- Project-scoped
  - they express the authorization to operate in a specific tenancy (project) of the cloud and are useful to authenticate when working with most other services
- Domain-scoped
  - they express the authorization to operate a domain-level, above that of the user and projects contained therein (domain-level administrator)
  - They contain a limited-service catalog
  - They can also be used to work with domain-level concerns in other services, such as to configure domain-wide quotas that apply to all users or projects in a specific domain

Identity Concepts

- **Authentication.** The process of confirming the identity of a user. To confirm an incoming request, OpenStack Identity validates a set of credentials that users supply. Initially, these credentials are a username and password, or a username and API key. When OpenStack Identity validates user credentials, it issues an authentication token. Users provide the token in subsequent requests.
- **Endpoint.** A network-accessible address, usually a URL, through which you can access a service.
- **Regions, Domains and Groups.** Identity services API v3 entities. A region represents a general division in an OpenStack deployment. Domains are a collection of projects and users that define administrative boundaries for managing Identity entities. Groups are a collection of users owned by a domain.
- **Project.** A container that groups or isolates resources or identity objects. Depending on the service operator, a project might map to a customer, account, organization, or tenant.
- **Service.** An OpenStack service, such as Compute (nova), Object Storage (swift), or Image service (glance), that provides one or more endpoints through which users can access resources and perform operations.
- **Token.** An alpha-numeric text string that enables access to OpenStack APIs and resources. A token may be revoked at any time and is valid for a finite duration.
- **User.** A digital representation of a person, system, or service that uses OpenStack cloud services.

Using the OpenStack APIs

1. Issue an authentication request with a payload of credentials to OpenStack Identity to get an authentication token

Credentials are usually a combination of your username and password, and optionally, the name or ID of the project of your cloud.

2. When you send API requests, you include the token in the X-Auth-Token header.

A token is valid for a limited time before it expires. A token can also become invalid for other reasons. For example, if the roles for a user change, existing tokens for that user are no longer valid.

https://developer.openstack.org/api-guide/quick-start/api-quick-start.html

Scoped Token – Example

Request

```
POST /identity/v3/auth/tokens HTTP/1.1
HOST: 127.0.0.1:8080
content-type: application/json
content-length: 402

{
  "auth": {
    "identity": {
      "methods": [
        "password"
      ],
      "password": {
        "user": {
          "name": "demo",
          "domain": {
                        "name": "Default"
          },
          "password": "devstack"
        }
      }
    },
    "scope": {
      "project": {
        "id": "9176c2c2a1144a97bcf0de413d7a3cd9"
      }
    }
  }
}
```

Project *demo*

Response

```
date: Tue, 01 May 2018 09:50:53 GMT
server: Apache/2.4.29 (Ubuntu)
x-subject-token: gAAAAABa6Dh-eyy0aHgVnbTXAu48Y3mPA6c6pMuhuX4HhEqhYkD6jEm1YIA28kYD
RebozMgYWRzchfKVWVOpv8AA_UoAh2BfVV2THGgW3MmDiVesVS9oAZ6_HCWueMI11rlWZeEZ8pbL40Seu
HbwrSM8xekyM6xOwuw1ZA_k5QDB5JiqXv-UdzA
vary: X-Auth-Token
content-type: application/json
content-length: 3397
x-openstack-request-id: req-02d627b9-2a6b-44f8-8d04-66671ad0a35d
connection: close
{
body
}
```

Catalog that includes endpoints to access services, e.g.:

http://10.0.2.15/compute/v2.1

Catalog received in the response to the Token request

```
9176c2c2a1144a9
7bcf0de413d7a3c
d9 = Project_ID
```

```
devstack@openstack:~/devstack$ openstack catalog list
```

```
+-------------+----------------+------------------------------------------------------------------------+
| Name        | Type           | Endpoints                                                              |
+-------------+----------------+------------------------------------------------------------------------+
| keystone    | identity       | RegionOne                                                              |
|             |                |   admin: http://10.0.2.15/identity                                     |
|             |                | RegionOne                                                              |
|             |                |   public: http://10.0.2.15/identity                                    |
|             |                |                                                                        |
| cinderv2    | volumev2       | RegionOne                                                              |
|             |                |   public: http://10.0.2.15/volume/v2/9176c2c2a1144a97bcf0de413d7a3cd9  |
|             |                |                                                                        |
| cinderv3    | volumev3       | RegionOne                                                              |
|             |                |   public: http://10.0.2.15/volume/v3/9176c2c2a1144a97bcf0de413d7a3cd9  |
|             |                |                                                                        |
| placement   | placement      | RegionOne                                                              |
|             |                |   public: http://10.0.2.15/placement                                   |
|             |                |                                                                        |
| nova        | compute        | RegionOne                                                              |
|             |                |   public: http://10.0.2.15/compute/v2.1                                |
|             |                |                                                                        |
| cinder      | volume         | RegionOne                                                              |
|             |                |   public: http://10.0.2.15/volume/v1/9176c2c2a1144a97bcf0de413d7a3cd9  |
|             |                |                                                                        |
| nova_legacy | compute_legacy | RegionOne                                                              |
|             |                |   public: http://10.0.2.15/compute/v2/9176c2c2a1144a97bcf0de413d7a3cd9 |
|             |                |                                                                        |
| neutron     | network        | RegionOne                                                              |
|             |                |   public: http://10.0.2.15:9696/                                       |
|             |                |                                                                        |
| glance      | image          | RegionOne                                                              |
|             |                |   public: http://10.0.2.15/image                                       |
|             |                |                                                                        |
| cinder      | block-storage  | RegionOne                                                              |
|             |                |   public: http://10.0.2.15/volume/                                     |
|             |                |                                                                        |
+-------------+----------------+------------------------------------------------------------------------+
```

Scoped Token – Example

Request

```
GET /compute/v2.1/servers HTTP/1.1
HOST: 127.0.0.1:8080
x-auth-token: gAAAABa6Dh-eyy0aHgVnbTXAu48Y3mPA6c6pMuhuX4HhEqhYkD6jEm1YIA28kYDRebozMgYWRzchfKVWVOpv8AA_UoAh2BfVV2THGgW3MmDiVesVS9oAZ6_HCWueMI11rlWZeEZ8pbL40Se
uHbwrSM8xekyM6xOwuw1ZA_k5QDB5JiqXv-UdzA
```

Response

```
date: Tue, 01 May 2018 09:55:50 GMT
server: Apache/2.4.29 (Ubuntu)
content-length: 919
content-type: application/json
openstack-api-version: compute 2.1
x-openstack-nova-api-version: 2.1
vary: OpenStack-API-Version,X-OpenStack-Nova-API-Version
x-openstack-request-id: req-c6ec4efe-619a-4eaf-af4c-125ba93db35f
x-compute-request-id: req-c6ec4efe-619a-4eaf-af4c-125ba93db35f
connection: close
```

```
{
...
"servers": [
  {
"id": "5622e010-538e-49f4-85e2-560727253b21",
"links": [
  {
"href": "http://127.0.0.1:8080/compute/v2.1/servers/5622e010-538e-49f4-85e2-
560727253b21",
"rel": "self"
},
  {
"href": "http://127.0.0.1:8080/compute/servers/5622e010-538e-49f4-85e2-
560727253b21",
"rel": "bookmark"
}
],
"name": "test_instance_2"
],
}
```

- A **self** link contains a versioned link to the resource. Use these links when the link is followed immediately.
- A **bookmark** link provides a permanent link to a resource that is appropriate for long term storage.

Cinder and Nova APIs Usage Examples

```
GET /volume/v3/9176c2c2a1144a97bcf0de413d7a3cd9/volumes HTTP/1.1
HOST: 127.0.0.1:8080
x-auth-token: gAAAAABa6Cu5x9O_6YWjPW9DYthkMFYVU7lfnLDkERcLnzq
UZ7a3PJ8-TzxH5GtLhzJHSI9m_u5X6JL41sRwhaF27wTSRH9pwu3T8f-wZTNn
2mhdvBSva0lJ_7ISIAE8mPorDNEO4SzxvIMGhZtIzVIYKKn9EVNXhT3QmpsZR
sIB6YE95dL5Ox4
```

Volumes associated to **user** ***demo*****,** **project** ***demo*** in Cinder

Create a new instance in Nova

<u>Response</u>: 202 Accepted

```
POST /compute/v2.1/servers HTTP/1.1
HOST: 127.0.0.1:8080
x-auth-token: gAAAAABa6Cu5x9O_6YWjPW9DYthkMFYVU7lfnLDkERcLnzqUZ7a3PJ8-TzxH5GtLhzJH
SI9m_u5X6JL41sRwhaF27wTSRH9pwu3T8f-wZTNn2mhdvBSva0lJ_7ISIAE8mPorDNEO4SzxvIMGhZtIzV
IYKKn9EVNXhT3QmpsZRsIB6YE95dL5Ox4
content-type: application/json
content-length: 191
{
    "server": {
        "name": "test_instance_auto",
        "imageRef": "6eda4305-d711-4686-85bc-c47856c2e2c8",
        "flavorRef": http://openstack.example.com/flavors/1
    }
}
```