Nova

Introduction

Nova is typically presented as the Compute API provided for OpenStack implementations. The service compares to other Computes across most cloud platforms. It serves as the primary method of instance deployment for the IaaS as a whole. Composed of almost 98% Python, the source code is openly available and contributable through Nova's GitHub repository.

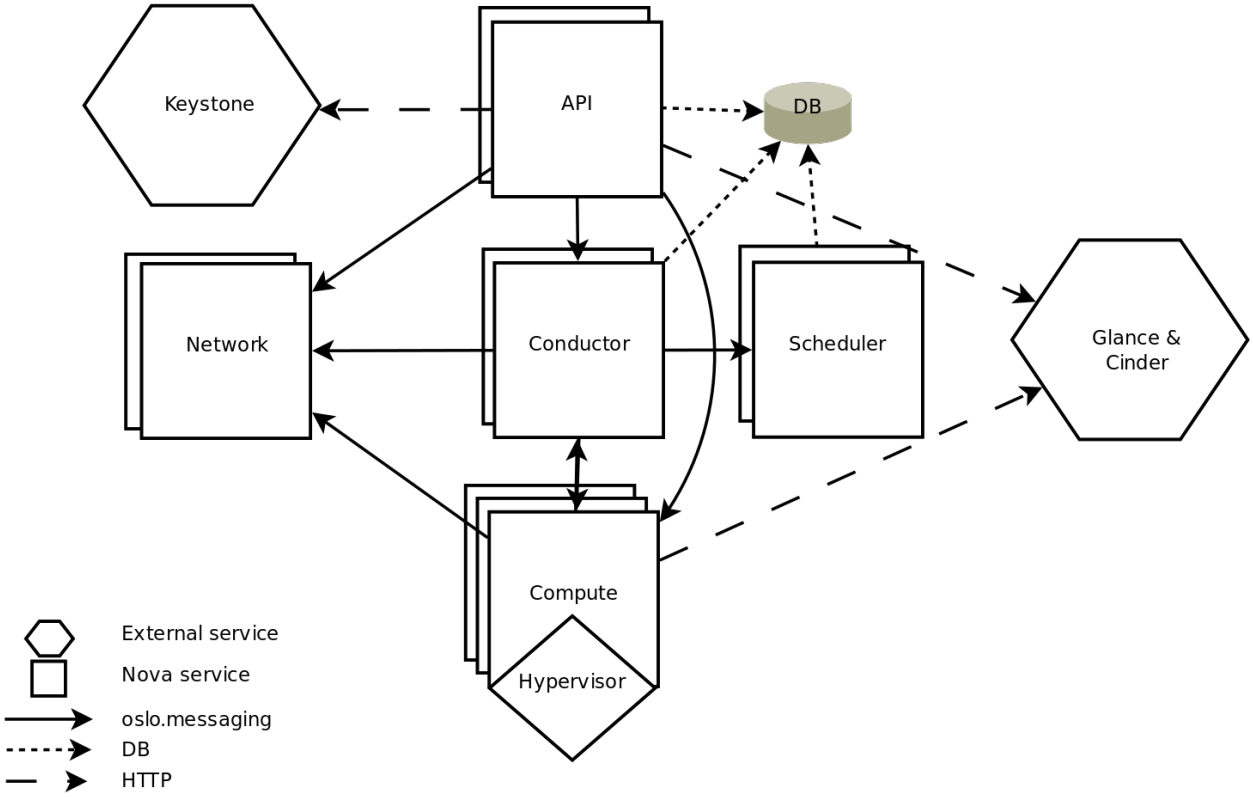
The primary method of interacting with Nova is through the publically available REST API provided as a frontend to Nova, the Compute API serving as the bulk. All interactions with Nova are typically performed from an administrative point-of-view through a client like Nova, which wraps the RESTful calls into user commands. But by nature of RESTful APIs, any number of applications can be written around Nova so that access and control may be automated.

Section 1: Nova Structure

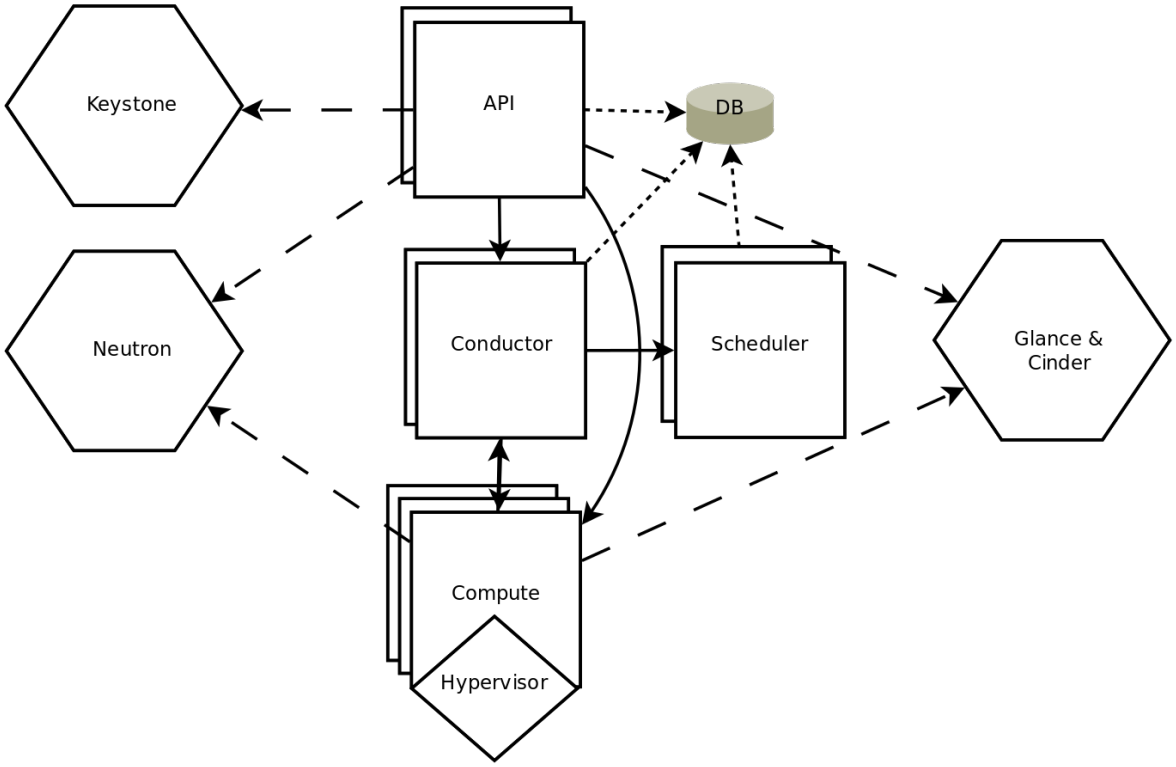
When examining the structure upon which Nova is built, the best reference material is the developer documentation. To begin with, the system architecture of Nova is made in such a way that it is split into several different components or modules. Each module then communicates with one another through a manageable messaging format that can be extended so any number of modules may be run on different hosts, all-on-one, or any combination of two. With this idea in mind comes one of the core ideas of Nova development where everything is designed to operate isolated and idependently just as the rest of the OpenStack project. This permits the mixing of various different flavors of hypervisors, networking, and anything else one might desire to create their own production OpenStack implementation.

Arguably the best way to gain a decent knowledge of the overall purpose of Nova is to examine the Scope of Nova as defined by the maintainers themselves. In summation, the document highlights a few key areas: compute resources, upgrades, api scope, & scalability. We will highlight the high-level concepts of each of these areas:

Nova-Networking



Neutron



- **Compute Resources:** Nova provides a point-of-access to the compute resources from anywhere else in OpenStack. Most widely known is the public Rest-API that the majority of applications use to communicate with Nova that covers virtually every feature available under the Compute API.

- **Upgrades:** To ensure ease of transition through updates, Nova is meant to deliver backwards compatibility, ensuring older versions function properly and their functionality will not be broken through updating. In addition, Nova attempts to maintain the current and previous iteration in terms of upgrades not to force users to upgrade immediately to gain access to new features. There exists an obvious downside to including every feature throughout implementation history in the sense that features that were a mistake to be added upstream in the past are now required to exist in the present, or even features that have been replaced. One example is Nova-Networking, where the networking used to be defined through Nova, but the switch to a fully-developed SDN suite led to Neutron's creation. The older networking is going to exist in the codebase for quite some time, but it has been deprecated. Swinging back to the first point, support will be maintained for the brief future in order to better handle migration of OpenStack deployments to newer versions.

- **API Scope:** This serves more as a reminder as to not overextend Nova's functionality outside of the realm of functions contained within or directly related to the Compute engine. Somewhat related to the other concepts listed, it is important that Nova does not encompass too much to avoid stepping on the toes of other components within the OpenStack (or other platform's) framework.

- **Scalability:** Scalability refers towards Nova's ability to support everything from small to massive deployments. This often serves to deter the inclusion of specific or specialized solutions for either one end of the spectrum or the other. Nova will always favor the ambiguity of a general solution given the already defined nature of its development.

Atop all this, Nova also strives to be API-compatible with other popular cloud IaaS platforms so that it may be used in place of other existing systems, again adding to its modularity.

To provide a better overview of the current Nova layout, examine the image provided from the current developer documentation covering the architecture of Nova:

Conductor, Scheduler, & Network

This section serves to cover the lesser well-known, yet still vital modules that are present within Nova's implementation. It is important to refer to the image to see that these components are all modular and may be swapped for other compatible technologies as the situation would dictate.

- **Conductor** - The Conductor, in its most basic form acts as an intermediary between the Compute module and the Database. As its original feature, this separation of the nodes from the database served several purposes including: security, by no longer providing Nodes direct access to the database, modularity, adding a layer between the database and Nodes permits only a single point of change on the Conductor to change database details compared to trying to scale across multiple Nodes, performance, by eliminating the often blocking thread for database calls (though requests don't see much improvement), and more. As part of the design plan, Conductor is meant to slowly transform into a more complete and complex module that will steal the functionality of the Compute module over time. This is to aid both in security and ease-of-use by providing a single point of access between Compute and other modules.

- **Scheduler** - The Scheduler, serves as the driving force to actually carry out any Compute requests. Organizing requests in terms of processing as well as making decisions for host deployment for VMs based on their requests, the Scheduler is one of the important modules used in the actual provisioning of hosts and creating of the requested instances.

- **Network** - Nova-Networking, as touched on before, was the original means for handling the network implementation of OpenStack, directing the networking of the deployment. Due to the incorporation of Neutron however, Nova-Networking has now been replaced by the new SDN standard and the component is expected to eventually fade.

Section 2: Code Organization

Since Nova is written almost exclusively in Python, reading over the source code is yet another fantastic way to go about understanding the project (next to reading the documentation). Taking advantage of the object-oriented nature of Python, several of the nouns are often defined as classes. If such an object is not a class, then it is more than likely stored as a module. Examples would be objects such as Instances and Floating IPs are stored as classes compared to items such as Servers which are more conceptual and therefore kept at the module level.

At the moment, given Nova's current migration of it's Compute API form v2 to v2.1, there are many duplicate or near duplicate copies of some modules and classes, leading to some redundancy. However redundant, this inclusion also highlights the **Upgrades** concept mentioned previously where the goal is to provide a smooth transition and to maintain functionality across upgrades. Controller is a very common class to see within the Compute API directory as it is arguably meant to be the most directly accessible of the entirety of Nova. These Controllers provide a means to interact with Nova individually, providing an ease-of-access to Nova's key structural points. Navigating through the presence of controllers, one will find that the Controllers primarily belong to the Compute Operations listed above, as they are the modules called upon through the public API calls. Another convention to spot is the use of API classes when defining any kind of large API service that will span several modules such as image, console, or compute itself.

Compute Operations

Nova's Compute serving as the main purpose behind Nova, it tends to be the most well developed and focused on when comparing the many different subtrees. Within Compute, there are several different operations Nova is currently fitted with (NOTE: Only a handful of the possible operations are displayed to provide a general understanding):

- **Versions:** Simple enough, this operation provides the information about the Compute API versions.

  ```
  GET /
  ```

- **Servers:** These series of operations all have to deal with servers. Given the wide array of actions to perform against servers, the number of operations is equally as long. From creating, to destroying, to gathering data, this serves as the bulk of the functionality for the Compute API (save for Extensions).

  ```
  GET POST PUT   /v2.1/{tenant_id}/servers/
                 /v2.1/{tenant_id}/servers/{server_id}
                 /v2.1/{tenant_id}/servers/{server_id}/action
  ```

- **Flavors:** This operation focuses on maintaining the current list of flavors that is possible to support from within the OpenStack deployment at the current moment.

  ```
  GET POST PUT DELETE   /v2.1/{tenant_id}/flavors
                        /v2.1/{tenant_id}/flavors/{flavor_id}
  ```

- **Keypairs:** Keypairs

```
GET POST DELETE   /v2.1/{tenant_id}/os-keypairs
                  /v2.1/{tenant_id}/os-keypairs/{keypair_name}
```

- **Limits:** Similar to **Versions**, Limits serves as a resource for information about the current deployment focusing on the current global and rate limits.

  ```
  GET   /v2.1/{tenant_id}/limits
  ```

- **Extensions:** This is left broad in the sense that extensions are not static across any standard OpenStack deployments. These do house very familiar operations such as Images, Host Aggregates, Cells, and the like.

  ```
  GET   /v2.1/extensions
        /v2.1/extensions/{alias}
  ```

Section 3: Walking Through Entire Nova Workflow/Cheat Sheet

Nova Instances

Instances are the running virtual machines within an OpenStack cloud. This section deals with how to work with them and their underlying images, their network properties, and how they are represented in the database. To launch an instance, you need to select an image, a flavor, and a name. The name needn't be unique.

For example:

nova list

```
stack@osic-utsa-osdev01: /opt/stack/nova/nova $ nova list
+--------------------------+----------------+--------+-------+-------------+------------------+
| ID                       | Name           | Status | State | Power State | Networks         |
+--------------------------+----------------+--------+-------+-------------+------------------+
| 381f0bec-3c55-4baa-adba265 | MyFirstInstance | ACTIVE | -     | Running     | private=10.0.0.9  |
| 3ac597bc-20b7-4cf7-891fbfa | Test           | ACTIVE | -     | Running     | private=10.0.0.4  |
| 4c5a3a9e-9b3e-4ae3-bc74cdf | Test           | ACTIVE | -     | Running     | private=10.0.0.5  |
| d18e20d9-91c6-49140b4ccdad | Test           | ACTIVE | -     | Running     | private=10.0.0.6  |
| f53bacb3-697e-434e-bdec0d2 | Test           | ACTIVE | -     | Running     | private=10.0.0.3  |
+--------------------------+----------------+--------+-------+-------------+------------------+
```

Listing Image

To list the image of the instance.

nova image-list

```
stack@osic-utsa-osdev01: /opt/stack/nova/nova $ nova image-list
+--------------------------------------+---------------------------------+--------+--------+
| ID                                   | Name                            | Status | Server |
+--------------------------------------+---------------------------------+--------+--------+
| 0688db6e-1cec-4038-8768-5908e582e989 | cirros-0.3.4-x86_64-uec         | ACTIVE |        |
| ac4ab80a-95f2-4327-a1a7-2f559225629e | cirros-0.3.4-x86_64-uec-kernel  | ACTIVE |        |
| b9859daf-a4b9-4754-a962-21a970289dea | cirros-0.3.4-x86_64-uec-ramdisk | ACTIVE |        |
+--------------------------------------+---------------------------------+--------+--------+
```

Listing Flavors

We use the following command to list the flavors.

nova flavor-list

```
stack@osic-utsa-osdev01: ~ $ nova flavor-list
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| ID | Name      | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor | Is_Public |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| 1  | m1.tiny   | 512       | 1    | 0         |      | 1     | 1.0         | True      |
| 2  | m1.small  | 2048      | 20   | 0         |      | 1     | 1.0         | True      |
| 3  | m1.medium | 4096      | 40   | 0         |      | 2     | 1.0         | True      |
| 4  | m1.large  | 8192      | 80   | 0         |      | 4     | 1.0         | True      |
| 42 | m1.nano   | 64        | 0    | 0         |      | 1     | 1.0         | True      |
| 5  | m1.xlarge | 16384     | 160  | 0         |      | 8     | 1.0         | True      |
| 84 | m1.micro  | 128       | 0    | 0         |      | 1     | 1.0         | True      |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
```

Boot Instances

First create a network that virtual machines can use. Do this once for the entire installation and not on each compute node. Run the **nova network-create** command and on the controller:

```
$ cd /opt/stack/devstack
$ source openrc
```

Later we execute the boot instance command by typing the following:

nova boot –image cirros-0.3.4-x86_64-uec –flavor m1.tiny MyFirstInstance

```
stack@osic-utsa-osdev01: ~ $ nova boot --image cirros-0.3.4-x86_64-uec \
 --flavor m1.tiny MyFirstInstance
+--------------------------------------+-----------------------------------------------------+
| Property                             | Value                                               |
+--------------------------------------+-----------------------------------------------------+
| OS-DCF:diskConfig                    | MANUAL                                              |
| OS-EXT-AZ:availability_zone          | nova                                                |
| OS-EXT-STS:power_state               | 0                                                   |
| OS-EXT-STS:task_state                | scheduling                                          |
| OS-EXT-STS:vm_state                  | building                                            |
| OS-SRV-USG:launched_at               | -                                                   |
| OS-SRV-USG:terminated_at             | -                                                   |
| accessIPv4                           |                                                     |
| accessIPv6                           |                                                     |
| adminPass                            | u7wKFV3m77Bv                                        |
| config_drive                         |                                                     |
| created                              | 2015-09-22T03:24:05Z                                |
| flavor                               | m1.tiny (1)                                         |
| hostId                               |                                                     |
| id                                   | 123b9ecc-9081-40dc-b9e6-de2a1bd10fac                |
| image                                | cirros-0.3.4-x86_64-uec (0688db6e-1cec-5908e582e989) |
```

```
| key_name                            | -                                                |
| metadata                            | {}                                               |
| name                                | MyFirstInstance                                  |
| os-extended-volumes:volumes_attached | []                                               |
| progress                            | 0                                                |
| security_groups                     | default                                          |
| status                              | BUILD                                            |
| tenant_id                           | c2e15e8077f54fc5b55ec8987cdff28d                 |
| updated                             | 2015-09-22T03:24:05Z                             |
| user_id                             | ce6e05f1b85e42948daf1cf91dd51c81                 |
+-------------------------------------+--------------------------------------------------+
```

Login to Instance

We will type the following commands to login to an instance using the user key:

```
stack@osic-utsa-osdev01: ~ $ ip netns
qrouter-fdcc879f-4976-4b16-815d-834a273f2b51
qdhcp-7b293cb1-d958-409a-b310-78d58320ff2c
stack@osic-utsa-osdev01: ~ $ sudo ip netns exec \
 qdhcp-7b293cb1-d958-409a-b310-78d58320ff2c ssh cirros@10.0.0.9
cirros@10.0.0.9's password: //Password is "cubswin:)"
```

Show Details of the Instances

By typing the command below it will give the information about the instance in detail.

nova show MyFirstInstance

```
stack@osic-utsa-osdev01: ~ $ nova show MyFirstInstance
+-------------------------------------+--------------------------------------------------+
| Property                            | Value                                            |
+-------------------------------------+--------------------------------------------------+
| OS-DCF:diskConfig                   | MANUAL                                           |
| OS-EXT-AZ:availability_zone         | nova                                             |
| OS-EXT-STS:power_state              | 1                                                |
| OS-EXT-STS:task_state               | -                                                |
| OS-EXT-STS:vm_state                 | active                                           |
| OS-SRV-USG:launched_at              | 2015-09-22T03:24:34.000000                       |
| OS-SRV-USG:terminated_at            | -                                                |
| accessIPv4                          |                                                  |
| accessIPv6                          |                                                  |
| config_drive                        | True                                             |
| created                             | 2015-09-22T03:24:29Z                             |
| flavor                              | m1.tiny (1)                                      |
| hostId                              | ebaf3a2f0b66226924d1f7d6bc317dcdeab6a3a5d797ec   |
| id                                  | 840c18ac-0046-49dc-b809-7d83447c6b63             |
| image                               | cirros-0.3.4-x86_64-uec (0688db6e-1cec-5908e582e989) |
| key_name                            | -                                                |
| metadata                            | {}                                               |
| name                                | MyFirstInstance                                  |
| os-extended-volumes:volumes_attached | []                                               |
| private network                     | 10.0.0.18                                        |
```

```
| progress                             | 0                                                |
| security_groups                      | default                                          |
| status                               | ACTIVE                                           |
| tenant_id                            | c2e15e8077f54fc5b55ec8987cdff28d                 |
| updated                              | 2015-09-22T03:24:34Z                             |
| user_id                              | ce6e05f1b85e42948daf1cf91dd51c81                 |
+--------------------------------------+--------------------------------------------------+
```

View Console Log of Instances

It gives information like CPU, Ram size, disk, ssh host key, ip route, instance id, instance name, availability-zone, & local hostname, etc.

nova console-log MyFirstInstance

```
stack@osic-utsa-osdev01: ~ $ nova console-log MyFirstInstance
[    0.000000] Initializing cgroup subsys cpuset
[    0.000000] Initializing cgroup subsys cpu
[    0.000000] Linux version 3.2.0-80-virtual (buildd@batsu) (gcc version 4.6.3 (Ub
[    0.000000] Command line: root=/dev/vda console=tty0 console=ttyS0 no_timer_check
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ // Several lines omitted.
launch-index: 0
=== cirros: current=0.3.4 uptime=14.23 ===
  ____               ____  ____
 / __/ __ ____ ____ / __ \/ __/
/ /__ / // __// __// /_/ /\ \
\___//_//_/  /_/   \____/___/
   http://cirros-cloud.net


login as 'cirros' user. default password: 'cubswin:)'. use 'sudo' for root.
myfirstinstance login:
```

To Pause/Unpause/Suspend/Unsuspend, etc.:

Various commands used to control the current status of an instance.

```
stack@osic-utsa-osdev01: ~ $ nova pause MyFirstInstance
stack@osic-utsa-osdev01: ~ $ nova unpause MyFirstInstance
stack@osic-utsa-osdev01: ~ $ nova suspend MyFirstInstance
stack@osic-utsa-osdev01: ~ $ nova resume MyFirstInstance
stack@osic-utsa-osdev01: ~ $ nova stop MyFirstInstance
 stop server MyFirstInstance has been accepted.
stack@osic-utsa-osdev01: ~ $ nova start MyFirstInstance
 start server MyFirstInstance has been accepted.
```

Rebuild the Server

OpenStack, and in particular the compute service, Nova, has a useful rebuild function that allows you to rebuild an instance from a fresh image while maintaining the same fixed and floating IP addresses.

nova rebuild

```
stack@osic-utsa-osdev01: ~ $ nova rebuild MyFirstInstance cirros-0.3.4-x86_64-uec-kernel
+--------------------+--------------------------------------------------------------------------+
| Property           | Value                                                                    |
+--------------------+--------------------------------------------------------------------------+
| OS-DCF:diskConfig | MANUAL                                                                   |
| accessIPv4         |                                                                          |
| accessIPv6         |                                                                          |
| adminPass          | dy2x3TwoeFN7                                                             |
| created            | 2015-09-22T03:24:29Z                                                     |
| flavor             | m1.tiny (1)                                                              |
| hostId             | ebaf3a2f0b66226924d1f7d6bc317dcdeab6a3ac4257185675d797ec                 |
| id                 | 840c18ac-0046-49dc-b809-7d83447c6b63                                     |
| image              | cirros-0.3.4-x86_64-uec-kernel (ac4ab80a-95f2-4327-a1a7-2f559225629e)    |
| metadata           | {}                                                                       |
| name               | MyFirstInstance                                                          |
| private network    | 10.0.0.18                                                                |
| progress           | 0                                                                        |
| status             | REBUILD                                                                  |
| tenant_id          | c2e15e8077f54fc5b55ec8987cdff28d                                         |
| updated            | 2015-09-22T03:26:13Z                                                     |
| user_id            | ce6e05f1b85e42948daf1cf91dd51c81                                         |
+--------------------+--------------------------------------------------------------------------+
```

Resize the Server Capacity:

Resizes the current instance to a new flavor capacity.

nova resize my-pem-server m1.small
nova resize confirm server1

```
stack@osic-utsa-osdev01: ~ $ nova resize MyFirstInstance m1.small
stack@osic-utsa-osdev01: ~ $ nova resize confirm MyFirstInstance
```

Rescue Instance

Rescue is the procedure to recover the short instance data in the event of failed configuration or the inability to access a previously accessible instance. In this operation all the rescue data will appear on the rescue instance that is created. This instance uses a specially build Ubuntu image.

```
stack@osic-utsa-osdev01: ~ $ nova rescue MyFirstInstance
+-----------+--------------+
| Property  | Value        |
+-----------+--------------+
| adminPass | 2v2J4dPBz27d |
+-----------+--------------+
```

Reboot

Will execute a reboot on the current instance.

nova reboot

```
stack@osic-utsa-osdev09: ~ $ nova reboot OSIC-work
```

Inject

Allows a user to, upon booting of an instance, to inject customized data into the user_data (or user-data) field.

nova boot –user-data

```
stack@osic-utsa-osdev09: ~ $ echo "My user data goes here" > test.file
stack@osic-utsa-osdev09: ~ $ nova boot --user-data ./test.file --image cirros-0.3.4-x86_64-uec \
 --flavor m1.tiny injectTest
```