**Nova VM Placement with Filtering and Weights**

**VM Placement -** Detemines which host the VM should be on on

* The nova-scheduler interacts with other components through the queue and central database repo
* Detemines which host the VM should be on on
* All of the compute nodes periodically publish their status, resources available and hardware capabilities to nova-scheduler through the queue
* Nova-scheduler then collects this data and uses it to make decisions when a request comes in
* By default, the compute scheduler is configured as a filter scheduler
* In a default configuration, this scheduler considers hosts that meet the following criteria:
  + Are in the requested availability zone (AvailabilityZoneFilter)
  + Have sufficient RAM available (RamFilter)
  + Are capable of servicing the request (ComputeFilter) filter

The filter schedule supports filtering and weighting to make decisions on where a new instance should be created

* This scheduler only works when working with compute nodes
* Filter scheduler makes a dictionary of unfiltered hosts
* Then it filters them using filter properties and then chooses a host for the instance(s)
* Each time it chooses the most weighted host and then appends it to the list of selected hosts

Standard filter classes to use (nova.scheduler.filters):

* AllHostsFilter – this filter does no operation – it simply passes all available hosts
* ImagePropertiesFilter – filters hosts based on properties defined on the instance’s image
  + It passes hosts that can support the specified image properties contained in the instance
* AvailabilityZoneFilter – filters hosts by availability zone
  + It passes hosts matching the availability zone specified in the instance properties
* ComputeCapabilitiesFilter – checks that the capabilities provided by the hosts Compute service satisfy any extra specifications associated with the instance type
  + It passes hosts that can create the specified instance type
  + Extra specifications can have a scope at the beginning of the key string of a key/value pair
    - The scope format is scope:key and can be nested, i.e. key\_string
      * Example\* like capabilities:cpu\_info\* features is valid scope format
    - A key string with any : is non-scope format
    - Each filter defines its valid scope, and not all filters accept non-scope format
  + Extra specifications can have an operator at the beginning of the value storing of a key/value pair
    - If there is no operator specified, then a default operator of s== is used
  + Extra specifications can have an operator at the beginning of the value storing of a key/value pair
    - Valid Operator strings:
      * \*= (equal to or greater than as a number; same as vcpus case)
      * \*==