oVirt

Applying Nova scheduler concepts for data center virtualization

Gilad Chaplik
Red Hat

FOSDEM, February 2014

What is oVirt

- oVirt is a KVM management application for data center virtualization management:
  - Manage hundreds of KVM hypervisor nodes, running thousands of VMs.
  - Features:
    - Live migration of VMs and storage
    - Templates and snapshots of running VMs.
    - Advanced network configuration.
    - Support iSCSI, FCoE, NFS, and Gluster for shared storage.

Agenda

oVirt

- The Need & Problem
- Nova Filter Scheduler concepts
  - Filter
  - Weights
- oVirt Scheduling
  - samples
- External Proxy

Intro

oVirt

- Picked up from users-list

Re: [Users] How to define max number of running VMs on a host?

….

I have 4 graphic workstations with 3 graphic cards on each. I wanna
passthrough graphic cards to the VMs one by one, since one workstation
has only 3 cards, I must limit the number of running VM on a host to 3.

What we had..

oVirt

- oVirt Scheduler
  - Executes the selected distribution algorithm on the cluster:
    - Even Distribution
    - Power Saving
  - Selects a host to run/migrate VM on.
  - Balance: Selects a VM to migrate and Host to migrate to.
  - Only 2 distribution algorithms, taking into consideration only CPU usage
  - No way to construct a user defined scheduling policy

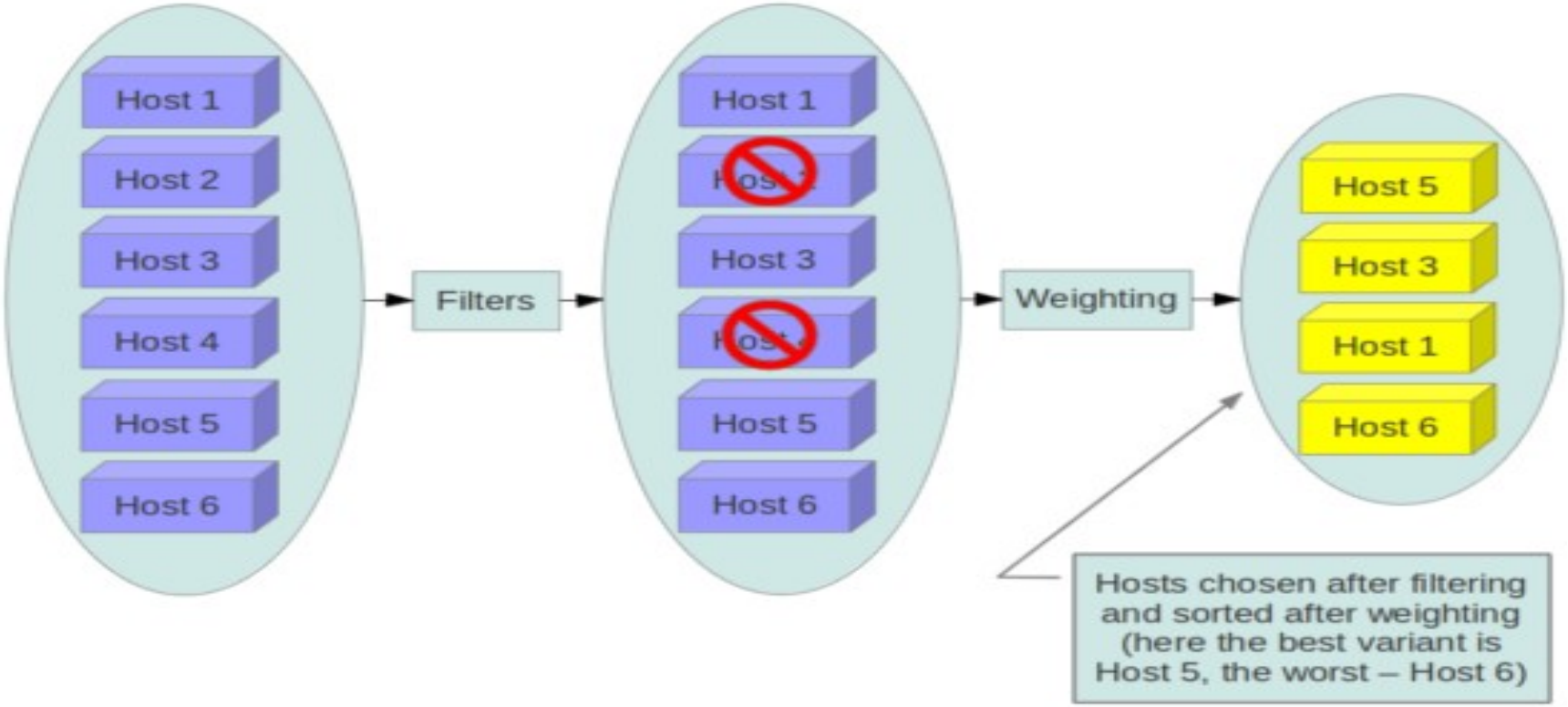
Following Nova Scheduler

oVirt

The Filter Scheduler supports filtering and weighting to make informed decisions on where a new instance should be created.

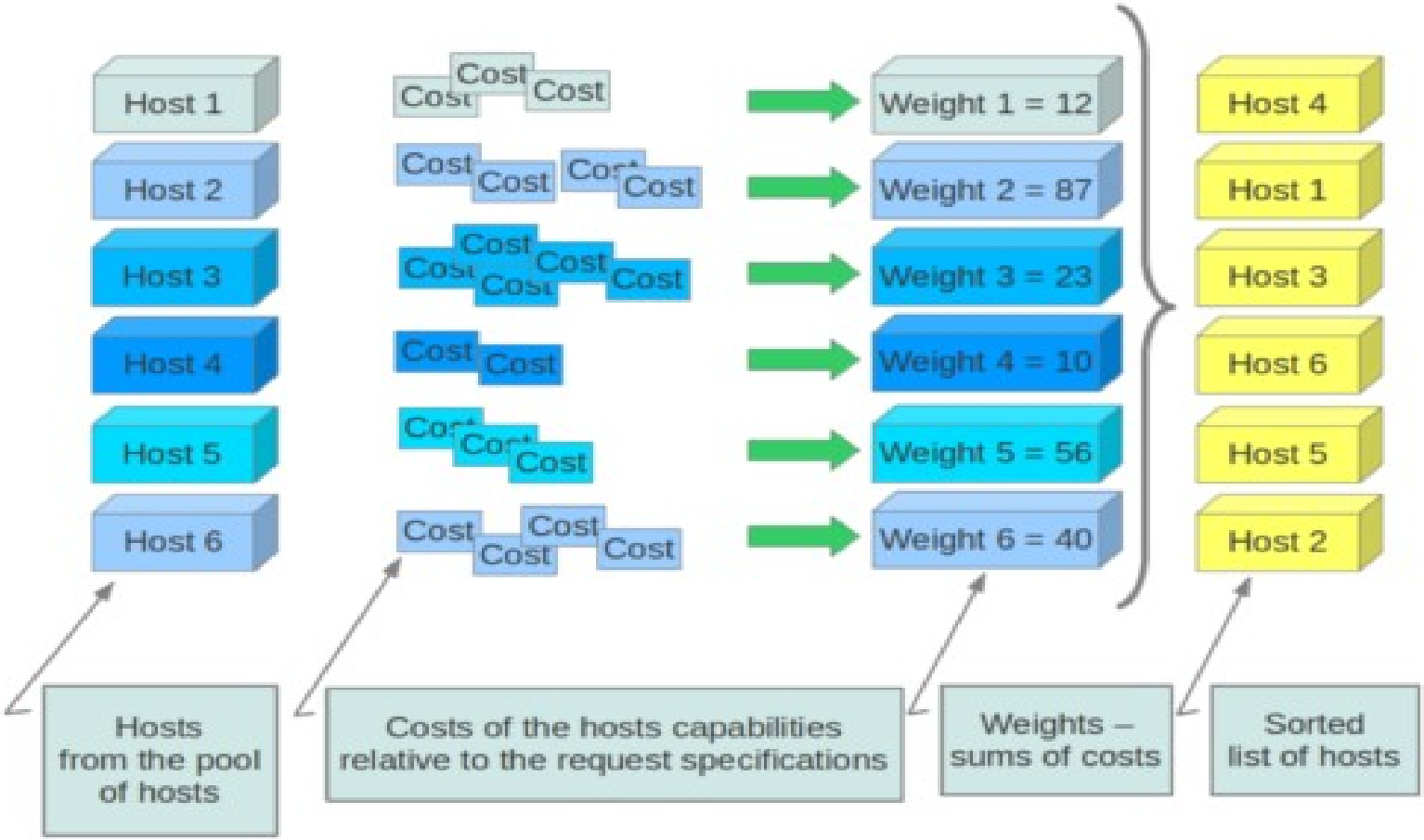
- Filters
- Weights

Following Nova Scheduler



Source: http://docs.openstack.org/developer/nova/devref/filter_scheduler.html

Following Nova Scheduler



Source: http://docs.openstack.org/developer/nova/devref/filter_scheduler.html

Filter Sample – Nova

oVirt

```
class RamFilter(filters.BaseHostFilter):
    """Ram Filter with over subscription flag"""

    def host_passes(self, host_state, filter_properties):
        """Only return hosts with sufficient available RAM."""
        instance_type = filter_properties.get('instance_type')
        requested_ram = instance_type['memory_mb']
        free_ram_mb = host_state.free_ram_mb
        total_usable_ram_mb = host_state.total_usable_ram_mb
        used_ram_mb = total_usable_ram_mb - free_ram_mb
        return total_usable_ram_mb * FLAGS.ram_allocation_ratio  - used_ram_mb >= requested_ram
```

public cloud vs. data center virtualization

oVirt

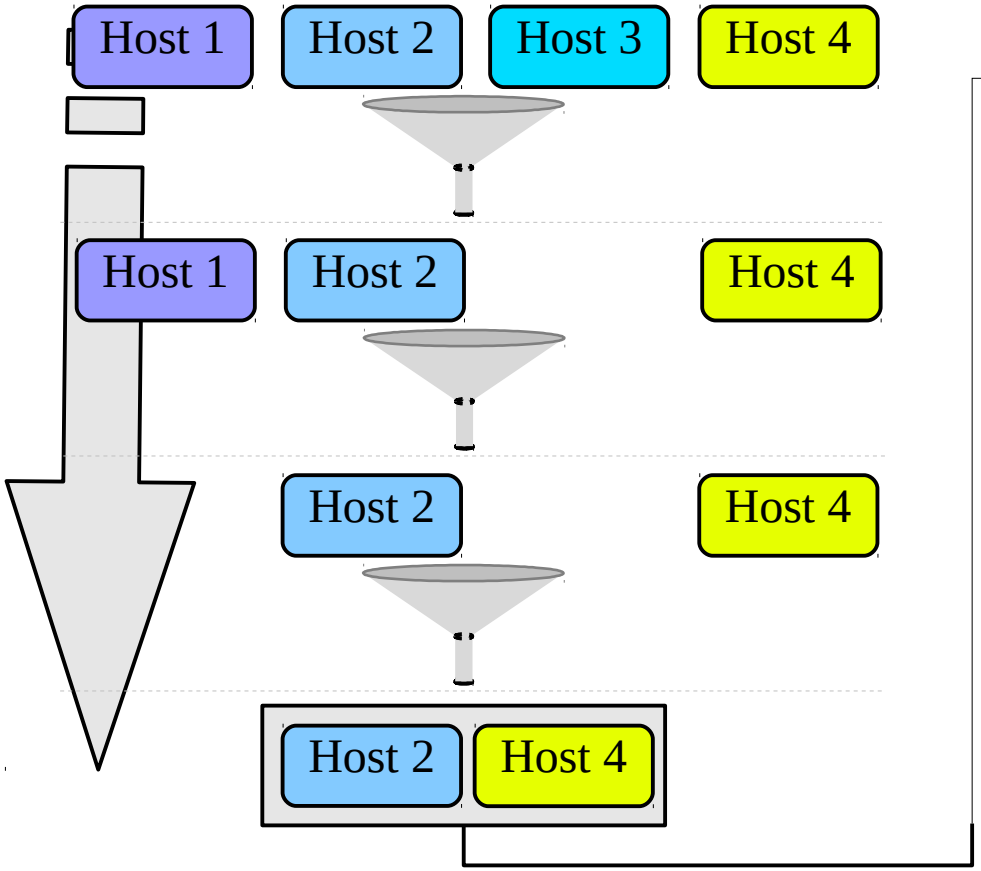
- Each Filter and Weight is applied on a single host
- Migration Domain
- Load balancing

- Implementation Details
  - Policy container

oVirt VM Scheduler

- Use internal/external Filters and Weights.
- Apply a method on all Hosts in Cluster.
- Containers - Cluster Policies.
- Load balancing for each cluster policy.
- Custom Properties.

oVirt - The new Model



	func 1	func 2	sum
Factor	5	2	
Host 2	10	2	54
Host 4	3	12	**39***

***Host 4 sum: 3*5+12*2 = 39**

oVirt -Filters

- Filters Hosts of a given migration domain aka Cluster.
- Existing logic (pin-to-host, memory limitations, etc.) is translated into built-in Filters.
- External Filters written in python can be loaded into oVirt engine.

Filter Sample

```
import datetime

class shut_down_hosts_filter():
    '''filters out hosts according to shutdown hour'''

    properties_validation = 'shut_down_hour=[0-9]*;wake_up_hour=[0-9]*;'

    def do_filter(self, hosts_ids, vm_id, args_map):
        wake_up_hour = int(args_map.get('wake_up_hour', -1))
        shut_down_hour = int(args_map.get('shut_down_hour', 25))
        current_hour = datetime.datetime.now().hour
        if(current_hour < wake_up_hour and current_hour > shut_down_hour):
            hosts_ids = []
        print(hosts_ids)
```

Filter Sample

oVirt

```
import datetime

class shut_down_hosts_filter():
    '''filters out hosts according to 'shutdown hour'''

    properties_validation = 'shut_down_hour=[0-9]*;wake_up_hour=[0-9]*;'

    def do_filter(self, hosts_ids, vm_id, args_map):
        wake_up_hour = int(args_map.get('wake_up_hour', -1))
        shut_down_hour = int(args_map.get('shut_down_hour', 25))
        current_hour = datetime.datetime.now().hour
        if(current_hour < wake_up_hour and current_hour > shut_down_hour):
            hosts_ids = []
        print(hosts_ids)
```

Filter Sample

```
import datetime

class shut_down_hosts_filter():
    '''filters out hosts according to shutdown hour'''

properties_validation = 'shut_down_hour=[0-9]*;wake_up_hour=[0-9]*;'

    def do_filter(self, hosts_ids, vm_id, args_map):
        wake_up_hour = int(args_map.get('wake_up_hour', -1))
        shut_down_hour = int(args_map.get('shut_down_hour', 25))
        current_hour = datetime.datetime.now().hour
        if(current_hour < wake_up_hour and current_hour > shut_down_hour):
            hosts_ids = []
        print(hosts_ids)
```

Filter Sample

```
import datetime

class shut_down_hosts_filter():
    '''filters out hosts according to shutdown hour'''

    properties validation = 'shut_down_hour=[0-9]*;wake_up_hour=[0-9]*;'

    def do_filter(self, hosts_ids, vm_id, args_map):
        wake_up_hour = int(args_map.get('wake_up_hour', -1))
        shut_down_hour = int(args_map.get('shut_down_hour', 25))
        current_hour = datetime.datetime.now().hour
        if(current_hour < wake_up_hour and current_hour > shut_down_hour):
            hosts_ids = []
        print(hosts_ids)
```

Filter Sample

```
import datetime

class shut_down_hosts_filter():
    '''filters out hosts according to shutdown hour'''

    properties_validation = 'shut_down_hour=[0-9]*;wake_up_hour=[0-9]*;'

    def do_filter(self, hosts_ids, vm_id, args_map):
        wake_up_hour = int(args_map.get('wake_up_hour', -1))
        shut_down_hour = int(args_map.get('shut_down_hour', 25))
        if(current_hour < wake_up_hour and current_hour > shut_down_hour):
            hosts_ids = []
        print(hosts_ids)
```

Filter Sample

oVirt

```
import datetime

class shut_down_hosts_filter():
    '''filters out hosts according to shutdown hour'''

    properties_validation = 'shut_down_hour=[0-9]*;wake_up_hour=[0-9]*;'

    def do_filter(self, hosts_ids, vm_id, args_map):
        wake_up_hour = int(args_map.get('wake_up_hour', -1))
        shut_down_hour = int(args_map.get('shut_down_hour', 25))
current_hour = datetime.datetime.now().hour
if(current_hour < wake_up_hour and current_hour > shut_down_hour):
    hosts_ids = []
print(hosts_ids)
```

oVirt - Weights

- Weights hosts that passed all filters.
- Predefined Weights:
  - Even Distribution
  - Power Saving
  - [3.4] VM Affinity, Even Guest Distribution, HA, etc.
- Factors.
- External Weights.

Weight Sample

```
class even_vm_distribution():
    '''rank hosts by the number of running vms on them, with the least first'''

    properties_validation = ''

    def do_score(self, hosts_ids, vm_id, args_map):
        #open a connection to the rest api
        try:
            connection = API(url='http://host:port',
                             username='user@domain', password='')
        except BaseException as ex:
            #letting the external proxy know there was an error
            print >> sys.stderr, ex
            return

        #get all the hosts with the given ids
        engine_hosts = \
            connection.hosts.list(
                query=" or ".join(["id=%s" % u for u in hosts_ids]))

        #iterate over them and score them based on the number of vms running
        host_scores = []
        for engine_host in engine_hosts:
            if(engine_host and
                    engine_host.summary):
                host_scores.append((engine_host.id, engine_host.summary.active))
        print host_scores
```

Weight Sample

oVirt

```
class even_vm_distribution():
    '''rank hosts by the number of running vms on them, with the least first'''

    properties_validation = ''

    def do_score(self, hosts_ids, vm_id, args_map):
        #open a connection to the rest api
        try:
            connection = API(url='http://host:port',
                             username='user@domain', password='')
        #letting the external proxy know there was an error
        print >> sys.stderr, ex
        return

    #get all the hosts with the given ids
    engine_hosts = \
        connection.hosts.list(
            query=" or ".join(["id=%s" % u for u in hosts_ids]))

    #iterate over them and score them based on the number of vms running
    host_scores = []
    for engine_host in engine_hosts:
        if(engine_host and
                engine_host.summary):
            host_scores.append((engine_host.id, engine_host.summary.active))
    print host_scores
```

Weight Sample

oVirt

```
class even_vm_distribution():
    '''rank hosts by the number of running vms on them, with the least first'''

    properties_validation = ''

    def do_score(self, hosts_ids, vm_id, args_map):
        #open a connection to the rest api
        try:
            connection = API(url='http://host:port',
                             username='user@domain', password='')
        except BaseException as ex:
            #letting the external proxy know there was an error
            print >> sys.stderr, ex
            return
```

```
#iterate over them and score them based on the number of vms runn
host_scores = []
for engine_host in engine_hosts:
    if(engine_host and
            engine_host.summary):
        host_scores.append((engine_host.id, engine_host.summary.a
print host_scores
```

```
        if(engine_host and
                engine_host.summary):
            host_scores.append((engine_host.id, engine_host.summary.active))
    print host_scores
```

Load Balancing

oVirt

- Triggers a scheduled task to perform actions on a given cluster.
- A balancing logic can return a single VM, and oVirt Engine will migrate according to current scheduling logic.
- A single load balancing logic is allowed per cluster.

Load Balancing

oVirt

- Predefined Load Balancing logic include:
  - CPU power saving
  - CPU even distribution
  - Even VM distribution
- External Balancing logic can be written in python and loaded into the engine.

Balancing Sample

oVirt

```
def do_balance(self, hosts_ids, args_map):
    wake_up_hour = int(args_map.get('wake_up_hour', -1))
    shut_down_hour = int(args_map.get('shut_down_hour', 25))
    current_hour = datetime.datetime.now().hour
    wake_up = current_hour > wake_up_hour and current_hour < shut_down_hour
    conn = self._get_connection()
    if conn is None:
        return
    for host_id in hosts_ids:
        host = conn.hosts.get(id=host_id)
        if(wake_up and host.status.state != 'up'):
            host.activate()
        elif(not(wake_up) and host.status.state != 'maintenance'):
            vms = conn.vms.list(query='host=' +host.name)
            for vm in vms:
                vm.stop()
            host.deactivate()
```

Balancing Sample

oVirt

```
def do_balance(self, hosts_ids, args_map):
    wake_up_hour = int(args_map.get('wake_up_hour', -1))
    shut_down_hour = int(args_map.get('shut_down_hour', 25))
```

```
def do_balance(self, hosts_ids, args_map):
    wake_up_hour = int(args_map.get('wake_up_hour', -1))
    shut_down_hour = int(args_map.get('shut_down_hour', 25))
    current_hour = datetime.datetime.now().hour
    wake_up = current_hour > wake_up_hour and current_hour < shut_down_hour
    conn = self._get_connection()
    if conn is None:
        return
```

```
vms = conn.vms.list(query='host=' +host.name)
for vm in vms:
    vm.stop()
host.deactivate()
```

Balancing Sample

```
def do_balance(self, hosts_ids, args_map):
    wake_up_hour = int(args_map.get('wake_up_hour', -1))
```

```
for host_id in hosts_ids:
    host = conn.hosts.get(id=host_id)
    if(wake_up and host.status.state != 'up'):
        host.activate()
    elif(not(wake_up) and host.status.state != 'maintenance'):
        vms = conn.vms.list(query='host=' +host.name)
        for vm in vms:
            vm.stop()
        host.deactivate()
```

```
host.deactivate()
```

Balancing Sample 2

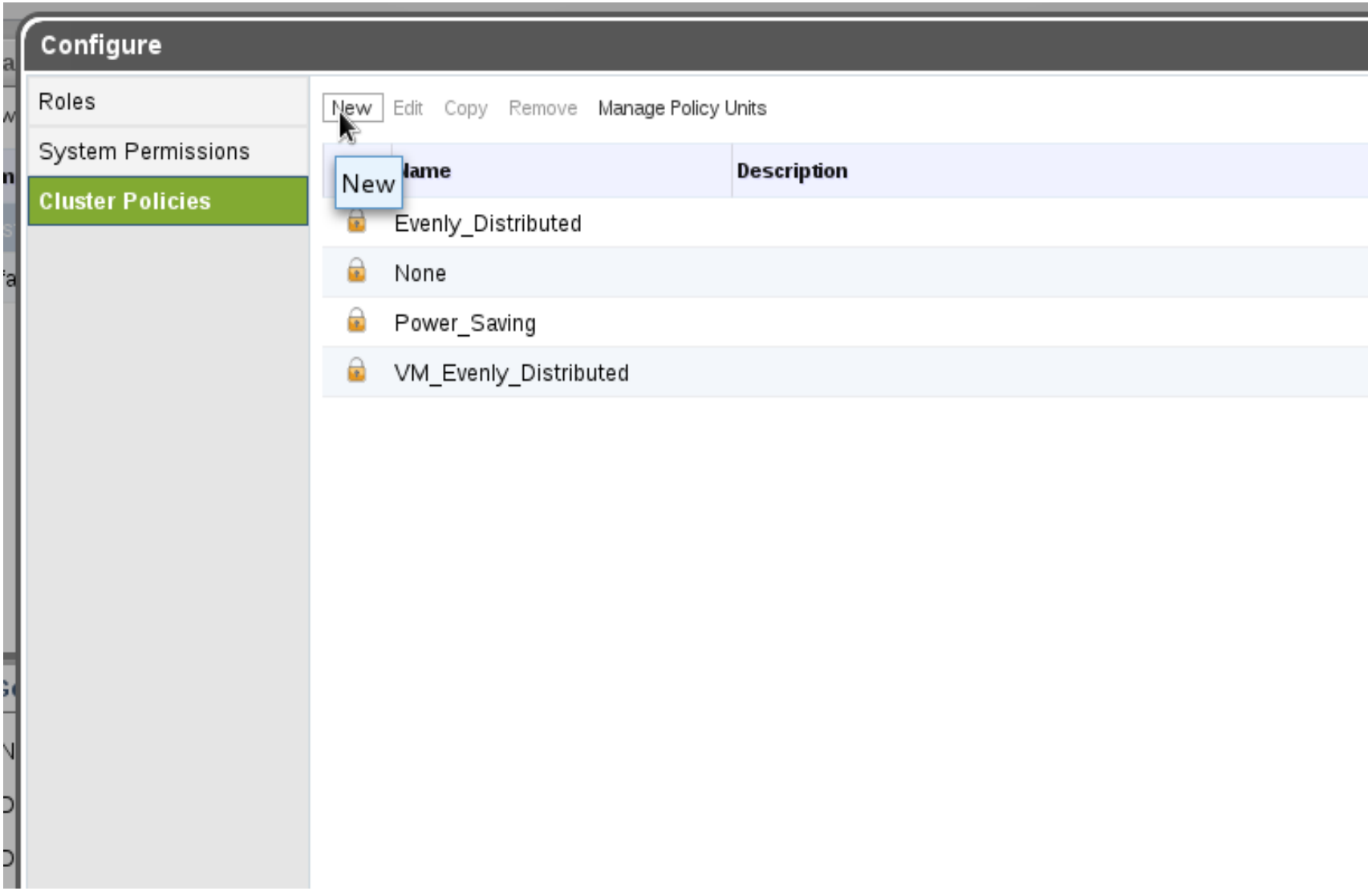
```
selected_vm = None
#just pick the first we find
host_vms = connection.vms.list('host='+over_loaded_host.name)
if host_vms:
    selected_vm = host_vms[0].id
else:
    return

print (selected_vm, white_listed_hosts)
```

Cluster Policy

oVirt

- Container for Filters, Weights and a single load balancing logic.
- Can be defined in oVirt Engine, and attached to a Cluster.
- Optimizations:
  - Speed
  - Overbooking

Cluster Policy – Define



Cluster Policy – Define

oVirt

New Cluster Policy

Name shut_down Description

Filter Modules Drag or use context menu to make changes

Enabled Filters

(EXT) shut_down_hosts_filter

Disabled Filters

VmAffinityGroups
HA
Network
(EXT) shut_down_hosts_filter

Weights Modules Drag or use context menu to make changes

Enabled Weights & Factors

1 OptimalForEvenDistribution

Disabled Weights

None
OptimalForEvenGuestDistribution
VmAffinityGroups
OptimalForPowerSaving

Load Balancer

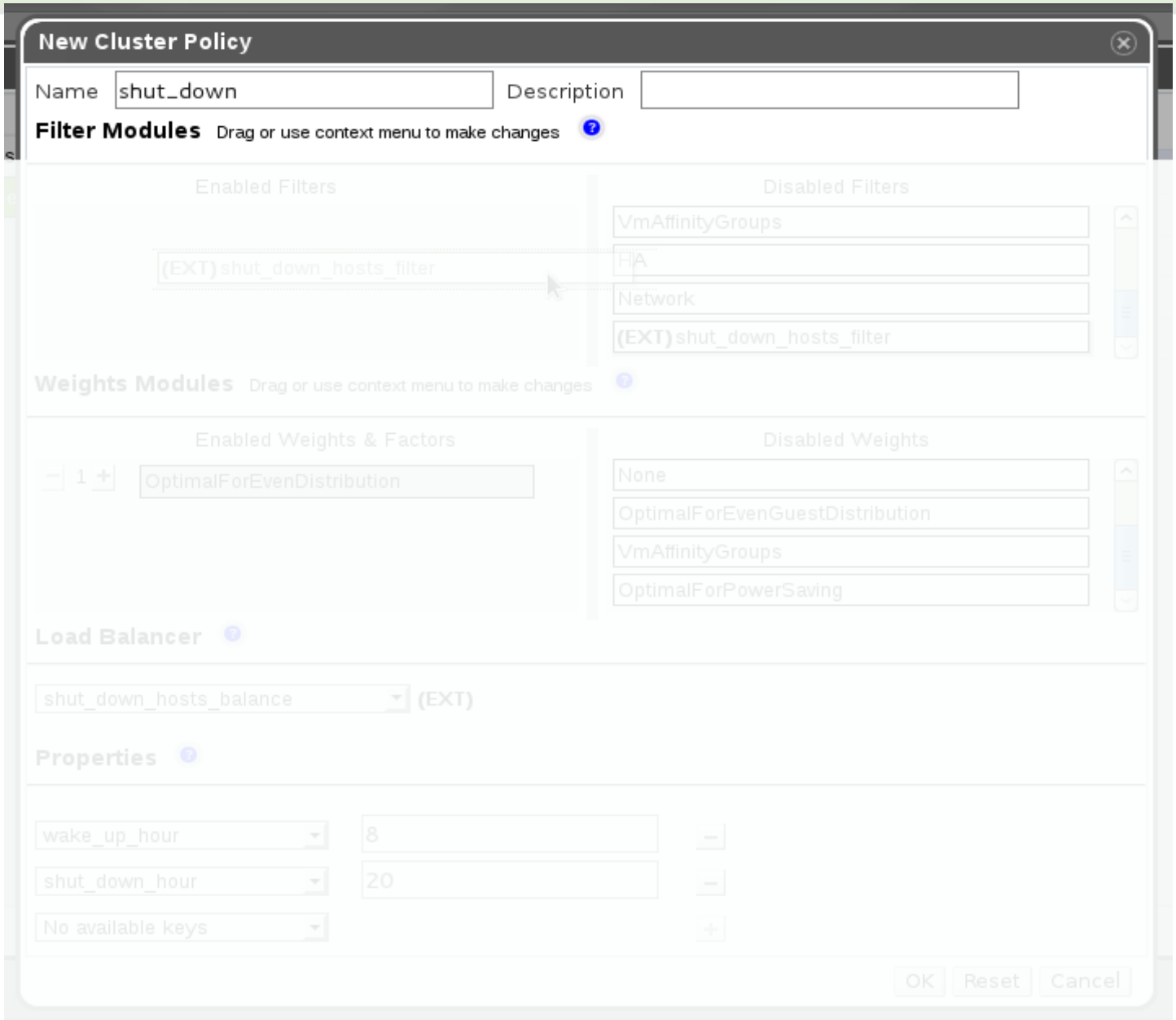
shut_down_hosts_balance (EXT)

Properties

wake_up_hour 8
shut_down_hour 20
No available keys

OK Reset Cancel

Cluster Policy – Define



Cluster Policy – Define

oVirt

New Cluster Policy

Name shut_down Description

Filter Modules Drag or use context menu to make changes

Enabled Filters

(EXT) shut_down_hosts_filter

Disabled Filters

VmAffinityGroups

HA

Network

(EXT) shut_down_hosts_filter

Weights Modules Drag or use context menu to make changes

Enabled Weights & Factors

1 OptimalForEvenDistribution

Disabled Weights

None

OptimalForEvenGuestDistribution

VmAffinityGroups

OptimalForPowerSaving

Load Balancer

shut_down_hosts_balance (EXT)

Properties

wake_up_hour 8

shut_down_hour 20

No available keys

OK Reset Cancel

Cluster Policy – Define

New Cluster Policy

Name shut_down Description

Filter Modules Drag or use context menu to make changes

Enabled Filters

(EXT) shut_down_hosts_filter

Disabled Filters

VmAffinityGroups

HA

Network

(EXT) shut_down_hosts_filter

Weights Modules Drag or use context menu to make changes

Enabled Weights & Factors

1 OptimalForEvenDistribution

Disabled Weights

None

OptimalForEvenGuestDistribution

VmAffinityGroups

OptimalForPowerSaving

Load Balancer

shut_down_hosts_balance (EXT)

Properties

wake_up_hour 8

shut_down_hour 20

No available keys

OK Reset Cancel

Cluster Policy – Define

New Cluster Policy

Name shut_down Description

Filter Modules Drag or use context menu to make changes

Enabled Filters

(EXT) shut_down_hosts_filter

Disabled Filters

VmAffinityGroups

HA

Network

(EXT) shut_down_hosts_filter

Weights Modules Drag or use context menu to make changes

Enabled Weights & Factors

1 OptimalForEvenDistribution

Disabled Weights

None

OptimalForEvenGuestDistribution

VmAffinityGroups

OptimalForPowerSaving

Load Balancer

shut_down_hosts_balance (EXT)

Properties

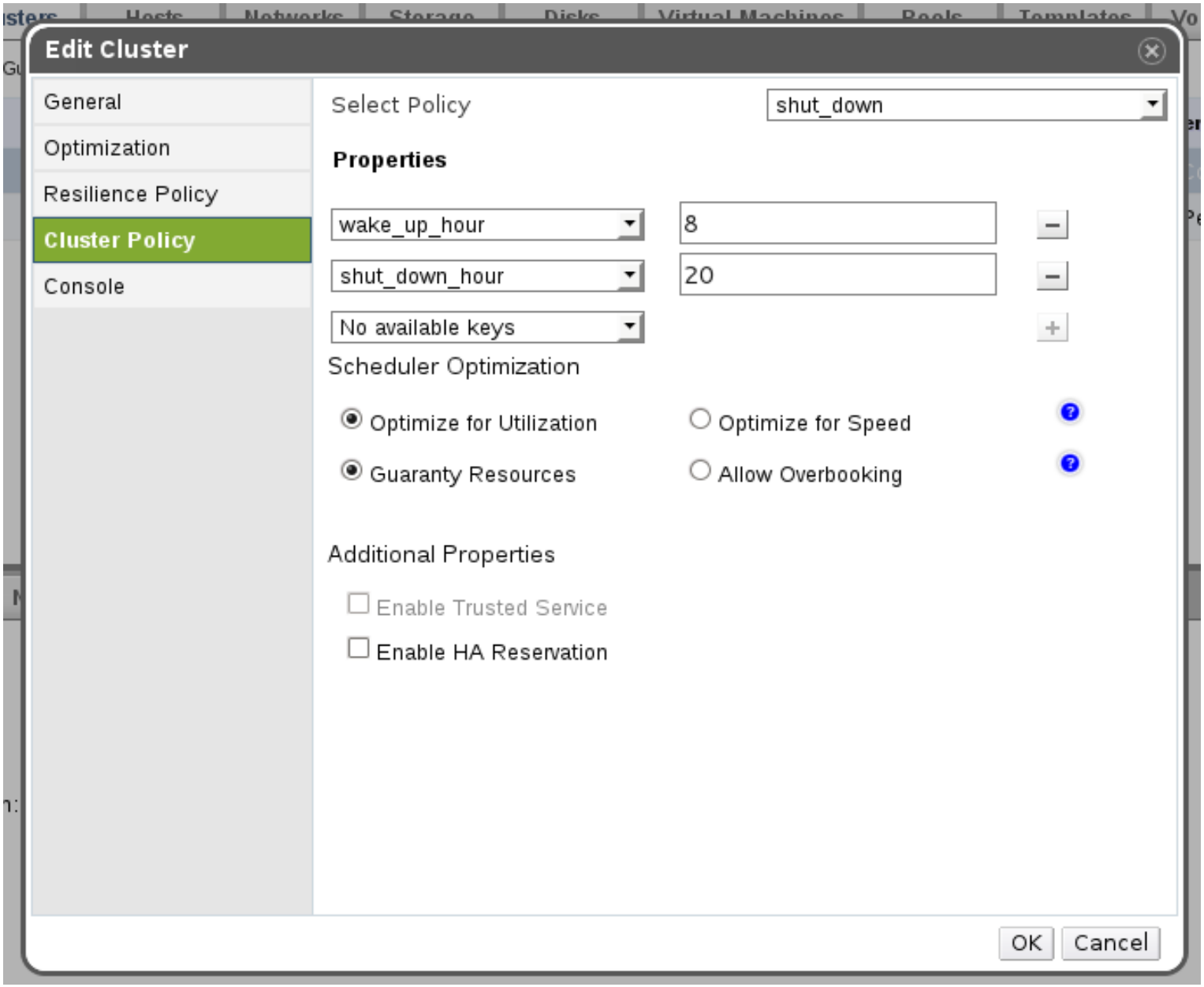
wake_up_hour 8

shut_down_hour 20

No available keys

OK Reset Cancel

Cluster Policy – Attach to Cluster



External Scheduler

oVirt

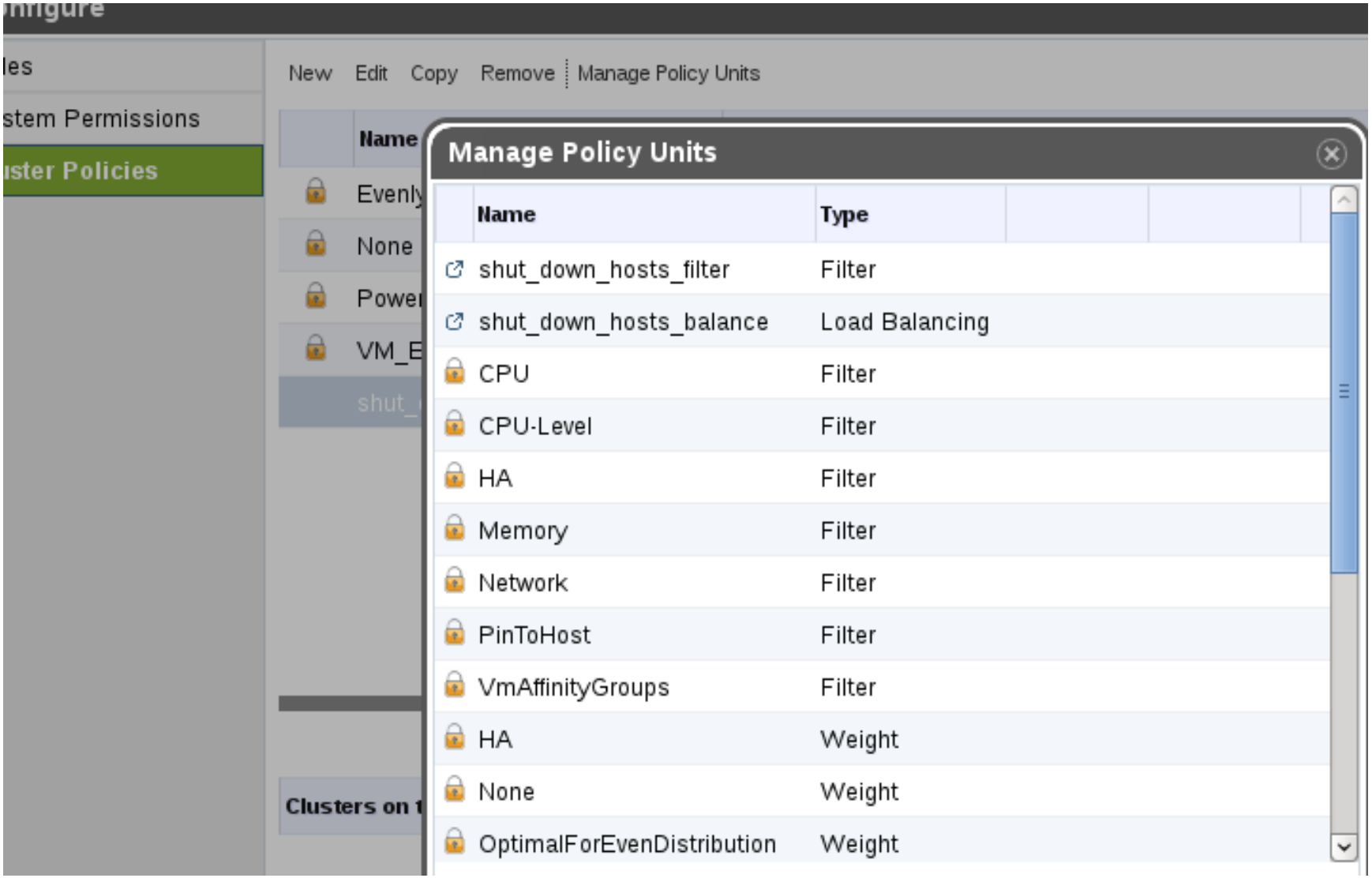
- Disabled by default (ExternalSchedulerEnabled = false)
- External service written in python and run as a separate process from the engine
- Why do we need it?
  - Engine safety
  - Should allow other languages
  - Going forward we may suggest SaaS (Scheduling as a Service)

External Scheduler

oVirt

- Packaged as ovirt-scheduler-proxy RPM, which is optional (not installed by default).
- Initialization
  - Service Start
  - Analyze
  - Publishing Internal API (Starting XML-RPC Server)
  - Waiting for engine calls
    - Discover
    - ...

External Modules in oVirt



Back to the users-list...

```
#iterate over them and decide which to accept
accepted_host_ids = []
for engine_host in engine_hosts:
    if(engine_host and
            engine_host.summary.active < maximum_vm_count):
        accepted_host_ids.append(engine_host.id)
print accepted_host_ids
```

To sum it up..

oVirt

- oVirt supports easy python plugins for VM scheduling
- Manage your scheduling policy separately for each migration domain (Cluster).
- New modules (features) are added in each version.

oVirt

Questions?

oVirt

THANK YOU !

http://www.ovirt.org
http://www.ovirt.org/Features/oVirtScheduler
http://www.ovirt.org/Features/oVirt_External_Scheduling_Proxy
http://www.ovirt.org/External_Scheduler_Samples

http://lists.ovirt.org/mailman/listinfo
vdsm-devel@lists.fedorahosted.org

#ovirt irc.oftc.net

gchaplik@redhat.com