Expanding oVirt's horizons

Mike Kolesnik, mkolesni@redhat.com
Senior Software Engineer, Red-Hat

René Koch, rkoch@linuxland.at
Senior Solution Architect at LIS-Linuxland GmbH

FOSDEM – February 2014
Agenda

- Small oVirt Introduction
- Part 1 – Consuming oVirt
  - Introduction
  - oVirt API
  - oVirt SDK
- Part 2 – Extending oVirt
  - VDSM hooks
  - Scheduling API
  - UI Plugin API
What is oVirt?

- Large scale, centralized management for server and desktop virtualization
- Based on leading performance, scalability and security infrastructure technologies
- Focus on KVM for best integration/performance
- Provides an open source alternative to vCenter/vSphere
Who is behind it

- Integration testing
- Quality engineering
- Security hardening
- Supportability
- Documentation
- Performance
- Scalability testing

Customers
Partners and individual contributors

Red Hat
Cisco
Intel
NetApp
IBM
SUSE
Canonical

Ovirt

Fedora

Enterprise grade virtualization infrastructure
# oVirt – Web admin

## oVirt Open Virtualization Manager

Logged in user: **admin@internal** | Configure | Guide | About | Sign Out

---

**System**

### Data Centers
- Name: [Display Name]
- Host: [Host Name]
- IP Address: [IP Address]
- Cluster: [Cluster Name]
- Data Center: [Data Center]
- Memory: [Memory Percentage]
- CPU: [CPU Percentage]
- Network: [Network Percentage]
- Display: [Display Type]
- Status: [Status]

### Bookmarks
- [Bookmarks]

### Tags
- [Tags]

Last Message: 2014-Jan-26, 14:36 User admin@internal logged in.

---

### Columns
- Name
- Host
- IP Address
- Cluster
- Data Center
- Memory
- CPU
- Network
- Display
- Status

### Rows
- Name: [Name]
- Host: [Host]
- IP Address: [IP Address]
- Cluster: [Cluster]
- Data Center: [Data Center]
- Memory: [Memory]
- CPU: [CPU]
- Network: [Network]
- Display: [Display]
- Status: [Status]

### Filters
- Search: Vms:

---

### Events
- Alerts (4)
- Events
- Tasks (0)
Part 1
Consuming oVirt
What can I do via API?

- Access it via REST/SDK/Shell
- Infrastructure configuration
  - Host configuration and management
  - Network and storage configuration
- Virtual machine (VM) configuration and management
  - Networking for the Guest, Virtual disks, VM properties
- User management
- Advanced operations not available in the GUI
- And much more ...
API methods

REST
https://host:port/api/vms

Returns:
- XML/JSON/...

<vm id="ae0dbce-1591-44d4-9052-c2209b3e45b8" href="/api/vms/ae0dbce-1591-44d4-9052-c2209b3e45b8"><name>Austin</name><actions>
  <link rel="shutdown" href="/api/vms/ae0dbce-1591-44d4-9052-c2209b3e45b8/shutdown"/>
  <link rel="start" href="/api/vms/ae0dbce-1591-44d4-9052-c2209b3e45b8/start"/>

....................

SDK (Python/Java)
api.vms.list()

Returns:
- list of VM objects

Shell
list vms

Returns:
- Formatted text

id : 18df94a7-048f-4306-9cfd-a74e8ea3b907
name : Boston
description : Main service for Boston
cluster-id : 99408929-82cf-4dc7-a532-9d998063fa95
cpu-topology-cores : 2
cpu-topology-sockets : 1
API Concepts

- All APIs integrate through the oVirt engine
- All types of APIs are based on the web services interface
  - REST as the core
  - SDK on top of REST
  - Shell implemented on top the Python SDK
- Backward compatibility
- Secure access
  - Session-based access
oVirt REST API
HTTP methods in REST

- **GET**

  Requests a representation of the specified resource. Requests using GET (and a few other HTTP methods) "SHOULD NOT have the significance of taking an action other than retrieval."

- **POST**

  Submits data to be processed to the identified resource. The data is included in the body of the request.

- **PUT**

  Uploads a representation of the specified resource

- **DELETE**

  Deletes the specified resource
Media types

• XML

```xml
<vms>
  <vm id="xxx">
    <name>yyy</name>
  </vm>
</vms>
```

• JavaScript Object Notation (JSON)

```json
{
  "vms" : [
    "vm" : {
      "id" : "xxx",
      "name" : "yyy"
    }
  ]
}
```
To list all VM resources, use GET
GET http(s)://server:port/api/vms

To create a VM resource, use POST
POST http(s)://server:port/api/vms
<vm>...</vm>
To retrieve a specific VM resource, use GET

```
GET http(s)://server:port/api/vms/{ID}
```

To update the VM resource, use PUT

```
PUT http(s)://server:port/api/vms/{ID}
<vm><name>new_name</name></vm>
```

To remove the VM resource, use DELETE

```
DELETE http(s)://server:port/api/vms/{ID}
```
RSDL - RESTful Services Description Language

- Describes parameter constraints
- Easy way to understand
  - How to create the resource
  - What actions are available on a collection
  - What parameters to pass
    - Mandatory/optional/read-only
    - Type
    - Overloads
oVirt SDK
**oVirt SDK**

- Mainly used for integration or advanced automation
- Object oriented
- Current bindings
  - Java - [http://www.ovirt.org/Java-sdk](http://www.ovirt.org/Java-sdk)
  - Python - [http://www.ovirt.org/Python-sdk](http://www.ovirt.org/Python-sdk)
  - libgovirt (GObject wrapper for the oVirt REST API) - [https://github.com/GNOME/libgovirt](https://github.com/GNOME/libgovirt)
  - rbovirt – ruby binding for the oVirt REST API - [https://github.com/abenari/rbovirt](https://github.com/abenari/rbovirt)
oVirt SDK - Concepts

- Complete protocol abstraction
- Full compliance with the oVirt API architecture
- Self descriptive, intuitive and easy to use
- Auto-generated
- Auto-completion*

* On supported environments
from ovirtsdk.api import API

api = API(url='http://localhost:8080',
          username='user@domain',
          password='password')
from ovirtsdk.api import API

api = API(url='http://localhost:8080',
          username='user@domain',
          password='password')

api.v
from ovirtsdk.api import API

api = API(url='http://localhost:8080',
          username='user@domain',
          password='password')

api.vms.

- add(vm, correlation_id, exp)
- context()
- get(name, id)
- list(query, case_sensitive, max)
from ovirtsdk.api import API
from ovirtsdk.xml import params

api = API(url='http://localhost:8080',
          username='user@domain',
          password='password')

cluster = api.clusters.get(name='Default')
template = api.templates.templates.get(name='RHEL7_0')
param = params.VM(name='RHEL_VM1',
                  cluster=cluster,
                  template=template,
                  memory=4*1024**3)

vml = api.vms.add(param)
Agenda

- Small oVirt Introduction
- Part 1 – Consuming oVirt
  - Introduction
  - oVirt API
  - oVirt SDK
- Part 2 – Extending oVirt
  - VDSM hooks
  - Scheduling API
  - UI Plugin API
Part 2
Extending oVirt
VDSM Hooks
Hooks

- VDSM manages a hypervisor
- “Hook” mechanism for customization
  - Allows administrator to define scripts to modify VM/VDSM operation
    - Extend or modify VM configuration
    - Run different system scripts
Hooks

- Hook scripts are called at specific events
- Hooks can modify a virtual machines XML definition
- Hooks can run system commands – eg. Apply firewall rule to VM
- More info:
  - http://www.ovirt.org/Vdsm_Hooks
Hook Points

- Lifecycle events where you can apply hooks
  - VDSM (management agent) start
  - VDSM stop
  - VM start
  - VM stop
  - VM migration in/out
  - VM Pause
  - VM Continue
  - VM Hibernate
  - VM resume from hibernate
  - VM set ticket
  - NIC hotplug / hotunplug
  - On host networking configuration change
import os
import hooking

def removeMacSpoofingFilter(interface):
    for filterElement in interface.getElementsByTagName('filterref'):
        if isMacSpoofingFilter(filterElement):
            interface.removeChild(filterElement)

def isMacSpoofingFilter(filterElement):
    """
    Accept a filter DOM element
    and checks if it's a mac spoofing filter
    """
    filterValue = filterElement.getAttribute('filter')
    return filterValue == 'vds-m-no-mac-spoofing'

if __name__ == '__main__':
    if hooking.tobool(os.environ.get('macspoof')):
        domxml = hooking.read_domxml()

        for interface in domxml.getElementsByTagName('interface'):
            removeMacSpoofingFilter(interface)

        hooking.write_domxml(domxml)
Hook Example – VM level
import os
import hooking

def removeMacSpoofingFilter(interface):
    for filterElement in interface.getElementsByTagName('filterref'):
        if isMacSpoofingFilter(filterElement):
            interface.removeChild(filterElement)

def isMacSpoofingFilter(filterElement):
    ""
    Accept a filter DOM element
    and checks if it's a mac spoofing filter
    ""
    filterValue = filterElement.getAttribute('filter')
    return filterValue == 'vdsm-no-mac-spoofing'

if __name__ == '__main__':
    if hooking.tobool(os.environ.get('ifacemacspoof')):
        domxml = hooking.read_domxml()
        interface, = domxml.getElementsByTagName('interface')
        removeMacSpoofingFilter(interface)
        hooking.write_domxml(domxml)
Hook Example – Device level
Writing a hook

- To write a hook you need:
  - Hook script(s)
  - README
    - What the hook does
    - How to configure the system for the hook
    - How to use the hook
  - Sudoers file*
  - Makefile to install hook
    - Hooks usually installed in /usr/libexec/vdsm/hooks

* In case your hook needs sudo
Scheduling API
Introduction

- The need - **construct user-defined scheduling policy**

*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.*
Old Scheduling Mechanism

- Executes the selected distribution algorithm on the Cluster (by CPU only)
  - Evenly Distributed
  - Power Saving
- Scheduling
  - Selects a host to run/migrate VM
- Load balancing
  - Selects a VM to migrate and Host to migrate to
- No way to extend by users
New Scheduling Mechanism

• Scheduling policy consists of
  • Filter modules
  • Weight modules
  • Load balancing module
• External modules developed in Python
• Existing (legacy) logic translated to modules
• Set the desired policy for a cluster
• More info:
  • http://goo.gl/senjQA - Existing policy units catalog
  • http://www.ovirt.org/External_Scheduler_Samples
New Scheduling Mechanism

<table>
<thead>
<tr>
<th>Host 1</th>
<th>Host 2</th>
<th>Host 3</th>
<th>Host 4</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Host 1</th>
<th>Host 2</th>
<th>Host 4</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Host 2</th>
<th>Host 4</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>func 1</th>
<th>func 2</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Host 2</td>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>Host 4</td>
<td>3</td>
<td>12</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Host</th>
<th>Host 1</th>
<th>Host 2</th>
<th>Host 3</th>
<th>Host 4</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>func 1</th>
<th>func 2</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Host 2</td>
<td>10</td>
<td>2</td>
</tr>
<tr>
<td>Host 4</td>
<td>3</td>
<td>12</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Host 1</th>
<th>Host 2</th>
<th>Host 3</th>
<th>Host 4</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Host 1</th>
<th>Host 2</th>
<th>Host 4</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Host 2</th>
<th>Host 4</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Host 2</th>
<th>Host 4</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Factor</th>
<th>func 1</th>
<th>func 2</th>
<th>sum</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2</td>
<td></td>
<td>54</td>
</tr>
<tr>
<td>Host 2</td>
<td>10</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>Host 4</td>
<td>3</td>
<td>12</td>
<td>39*</td>
</tr>
</tbody>
</table>

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

- Logical unit which filters out hosts
  - Clear cut logic
  - Easy to write and maintain
  - Chained up-dependently to allow complete filtering
  - Allows custom parameters
- Existing logic (pin-to-host, memory limitations, etc.) is translated into filters
- External filters written in python can be loaded into engine
Let's go back to the example

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.

Filter: filters out hosts with number running of vms > 3
class max_vms():
    '''returns only hosts with less running vms then the maximum'''
    #What are the values this module will accept, used to present
    #the user with options
    properties_validation = 'maximum_vm_count=[0-9]*'

def do_filter(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 our parameters from the map
    maximum_vm_count = int(args_map.get('maximum_vm_count', 100))

    #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 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
class max_vms():
    '''returns only hosts with less running vms then the maximum'''

    #What are the values this module will accept, used to present
    #the user with options
    properties_validation = 'maximum_vm_count=[0-9]*'

def do_filter(self, hosts_ids, vm_id, args_map):
    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 our parameters from the map
    maximum_vm_count = int(args_map.get('maximum_vm_count', 100))

    #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 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
Filter Example

class max_vms():
    '''returns only hosts with less running vms then the maximum'''

    # What are the values this module will accept, used to present
    # the user with options
    properties_validation = 'maximum_vm_count=[0-9]*'

def do_filter(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 our parameters from the map
    maximum_vm_count = int(args_map.get('maximum_vm_count', 100))

    # 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 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
class max_vms():
    '''returns only hosts with less running vms then the maximum'''

    #What are the values this module will accept, used to present
    #the user with options
    properties_validation = 'maximum_vm_count=[0-9]*'

def do_filter(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 our parameters from the map
    maximum_vm_count = int(args_map.get('maximum_vm_count', 100))

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

    #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
class max_vms():
    '''returns only hosts with less running vms then the maximum'''

    #What are the values this module will accept, used to present
    #the user with options
    properties_validation = 'maximum_vm_count=[0-9]+'

def do_filter(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 our parameters from the map
    maximum_vm_count = int(args_map.get('maximum_vm_count', 100))

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

    #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
External Policy Units

• External process is scanning directory /usr/share/ovirt-scheduler-proxy/plugins for python source files
• Analyze for filter / weight / balance functions
• Cache results
• Expose source files as external policy units
Cluster Policy Management
Cluster Policy Management

Edit Cluster Policy

Name: max_vms
Description:

Filter Modules

Enabled Filters
- CPU
- Network
- (EXT) max_vms

Disabled Filters
- (EXT) dummy
- (EXT) example

Weights Modules

Enabled Weights & Factors

Disabled Weights

Load Balancer

Type: vm_balance

Attached Clusters

vm_balance

Properties

maximum_vm_count: 3

+ -
Apply Cluster Policy

Select Policy: max_vms

Properties:
- maximum_vm_count: 2
UI Plugins
Web Admin user interface

- Extend oVirt Web Admin user interface
- Included in oVirt 3.2 release

- http://www.ovirt.org/Features/UIPlugins
- http://www.ovirt.org/Tutorial/UIPlugins/Crash Course
- http://www.ovirt.org/Features/UIPlugins#Real-world_UI_plugins
Web Admin user interface
UI plugin basics

• Plugin host page
  • Hosts actual plugin code (JavaScript)
    /usr/share/ovirt-engine/ui-plugins/<resourcePath>/<hostPage>.html

• Plugin descriptor
  • Meta-data + default configuration
    /usr/share/ovirt-engine/ui-plugins/<descriptorName>.json

• Plugin user configuration
  • Override default configuration, tweak runtime behavior
    /etc/ovirt-engine/ui-plugins/<descriptorName>-config.json
Writing plugins

<!-- Fetch additional resources if necessary -->
<script type="text/javascript" src="jquery-min.js"></script>

<!-- Actual plugin code -->
<script>
  // Access plugin API from iframe context
  var api = parent.pluginApi('myPlugin');

  // Register plugin event handler functions
  api.register({
    UiInit: function() {
      api.addMainTab('Foo Tab', 'foo-tab', 'http://foo.com/');
    }
  });

  // Tell plugin infrastructure that we are ready
  api.ready();
</script>

<!-- HTML body is intentionally empty -->
</body>
</html>
Supported API functions

- addMainTab(label, historyToken, contentUrl)
- addSubTab(entityTypeName, label, historyToken, contentUrl)
- setTabContentUrl(historyToken, contentUrl)
- setTabAccessible(historyToken, tabAccessible)
Supported API functions

- `addMainTabActionButton(entityTypeName, label, buttonInterface, options)`
  - Can add the button at the toolbar, context menu, or both
- `addSubTabActionButton(mainTabEntityName, subTabEntityName, label, buttonInterface, options)`
- `showDialog(title, dialogToken, contentUrl, width, height)`
- `SetDialogContentUrl(dialogToken, contentUrl)`
- `CloseDialog(dialogToken)`
- `loginUserName()`
- `loginUserId()`

String  
Boolean  
Number  
Object
Supported API events

-UiInit
-\{entity\}SelectionChange(selectedItems[])
-UserLogin(userNameWithDomain, userId)
-UserLogout()
-RestApiSessionAcquired(sessionId)
-MessageReceived (data, sourceWindow)
  - allows Plugin HTML to interact with the UI plugin by sending messages via HTML5 postMessage API
  - Requires allowedMessageOrigins set in Plugin Descriptor file

String
Boolean
Number
Object
Plugin descriptor

• **Meta-data + default configuration**
/usr/share/ovirt-engine/ui-plugins/<descriptorName>.json

{

    // A name that uniquely identifies the plugin (required)
    "name": "foo",

    // URL of plugin host page that invokes the plugin code (required)
    "url": "/webadmin/webadmin/plugin/foo/start.html",

    // Default configuration object associated with the plugin (optional)
    "config": { "band": "ZZ Top", "classic": true, "score": 10 },

    // Path to plugin static resources (optional)
    // Used when serving plugin files through PluginResourceServlet
    // This path is relative to /usr/share/ovirt-engine/ui-plugins
    "resourcePath": "foo-files"

}
Main steps in plugin development

(1) Write plugin descriptor
(2) Write plugin host page
(3) See plugin in action
Example: Monitoring UI-Plugin

Author: René Koch <rkoch@linuxland.at>
Project Page: https://github.com/monitoring-ui-plugin
# Monitoring UI-Plugin – VM Graphs

## General

<table>
<thead>
<tr>
<th>Service</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP Check</td>
<td>HTTP OK; HTTP/1.1 200 OK - 282 bytes in 0.011 second response</td>
</tr>
<tr>
<td>Load Check</td>
<td>OK - load average: 0.09, 0.69, 0.80</td>
</tr>
<tr>
<td>Memory Check</td>
<td>MEMORY OK - 32% Used (330 MB of 998 MB)</td>
</tr>
<tr>
<td>MySQL Threads connected</td>
<td>OK - 1 client connection threads</td>
</tr>
<tr>
<td>MySQL connection time</td>
<td>OK - 0.03 seconds to connect as monitoring</td>
</tr>
<tr>
<td>MySQL long running proc</td>
<td>OK - 0 long running processes</td>
</tr>
<tr>
<td>MySQL open files</td>
<td>OK - 12.98% of the open files limit reached (128 of max. 1024)</td>
</tr>
<tr>
<td>MySQL slow queries</td>
<td>OK - 0 slow queries in 310 seconds (0.09/sec)</td>
</tr>
<tr>
<td>NRPE Alive</td>
<td>NRPE v2.14</td>
</tr>
<tr>
<td>PINGs</td>
<td>OK - 10 0.200 19 ms; 2.524 ms; 100.0%</td>
</tr>
<tr>
<td>SSH Check</td>
<td>SSH OK - OpenSSH 5.3 (protocol 2.0)</td>
</tr>
<tr>
<td>Swap Check</td>
<td>SWAP OK - 96% free (2018 MB out of 2047 MB)</td>
</tr>
</tbody>
</table>

## Network Interfaces

<table>
<thead>
<tr>
<th>Name</th>
<th>Host</th>
<th>IP Address</th>
<th>Cluster</th>
<th>Data Center</th>
<th>Memory</th>
<th>CPU</th>
<th>Network</th>
<th>Display</th>
<th>Status</th>
<th>Uptime</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>sol11</td>
<td>centos-hyp01.lab.ovido.at</td>
<td></td>
<td>ovido-local</td>
<td>ovido-local</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>SPICE</td>
<td>Up</td>
<td></td>
<td></td>
</tr>
<tr>
<td>bathtub7</td>
<td>centos-hyp01.lab.ovido.at</td>
<td></td>
<td>ovido-local</td>
<td>ovido-local</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
<td>SPICE</td>
<td>Up</td>
<td></td>
<td></td>
</tr>
<tr>
<td>spidebian</td>
<td>centos-hyp01.lab.ovido.at</td>
<td></td>
<td>ovido-local</td>
<td>ovido-local</td>
<td>0%</td>
<td>1%</td>
<td>0%</td>
<td>SPICE</td>
<td>Up</td>
<td>2 min</td>
<td></td>
</tr>
</tbody>
</table>

## Disk Performance

![Disk Performance Graph](image)

## Network Performance

![Network Performance Graph](image)
Monitoring UI-Plugin – Details

- UI Plugin infrastructure
- Perl with various Modules
- Template Toolkit
- jQuery
- jQuery UI
- jQuery.loadTemplate
- GNU Autotools, Spec, SELinux-Policy
Monitoring UI-Plugin – monitoring-ui.json

• Plugin descriptor

```
{
  "name": "monitoring",
  // URL of plugin host page that invokes the plugin code (required)
  "url": "/webadmin/webadmin/plugin/monitoring/start.html",

  // Default configuration object associated with the plugin
  "config": {
    "url": "/monitoring-ui/cgi/monitoring-ui.cgi",
    "monitoringDetailsLabel": "Monitoring Details",
    "monitoringDashboardLabel": "Monitoring Dashboard"
  },

  // Path to plugin static resources (optional)
  // This path is relative to /usr/share/ovirt-engine/ui-plugins
  "resourcePath": "monitoring-files"
}
```
Monitoring UI-Plugin – start.html

• Register section - register the following event handlers
  
  • UiInit – add the main/sub tabs

```javascript
// Register an event handler, for the UI Plugin init
api.register({
  UiInit: function() {
//    // Dashboard Main Tab
    api.addMainTab(conf.icingaDashboardLabel, 'ovirt-monitoring', 'https://www.icinga.org');

    // Sub Tabs
    api.addSubTab('DataCenter', conf.monitoringDetailsLabel, 'datacenters-monitoring', conf.url + '?results=datacenters');
    api.addSubTab('Cluster', conf.monitoringDetailsLabel, 'clusters-monitoring', conf.url + '?results=clusters');
    api.addSubTab('Host', conf.monitoringDetailsLabel, 'hosts-monitoring', conf.url + '?results=hosts');
    api.addSubTab('Storage', conf.monitoringDetailsLabel, 'storage-monitoring', conf.url + '?results=storage');
    api.addSubTab('VirtualMachine', conf.monitoringDetailsLabel, 'vms-monitoring', conf.url + '?results=vms');
    api.addSubTab('Pools', conf.monitoringDetailsLabel, 'pools-monitoring', conf.url + '?results=pools');
  },
```

Monitoring UI-Plugin – start.html

*SelectionChange – set sub-tabs URL

```javascript
// Get name for changed selections
DataCenterSelectionChange: function() {
  if (arguments.length == 1) {
    var dataCenterName = arguments[0].name;
    api.setTabContentUrl('datacenters-monitoring', conf.url + '?results=datacenters&host=' + encodeURIComponent(dataCenterName));
  }
},
ClusterSelectionChange: function() {
  if (arguments.length == 1) {
    var clusterName = arguments[0].name;
    api.setTabContentUrl('clusters-monitoring', conf.url + '?results=clusters&host=' + encodeURIComponent(clusterName));
  }
},
HostSelectionChange: function() {
  if (arguments.length == 1) {
    var hostName = arguments[0].name;
    api.setTabContentUrl('hosts-monitoring', conf.url + '?results=hosts&host=' + encodeURIComponent(hostName));
  }
},
StorageSelectionChange: function() {
  if (arguments.length == 1) {
    var storageName = arguments[0].name;
    api.setTabContentUrl('storage-monitoring', conf.url + '?results=storage&host=' + encodeURIComponent(storageName));
  }
},
```
Monitoring - check_rhev3

- Icinga/Nagios plugin to check:
  - Datacenters
  - Clusters
  - Hosts
  - Storagedomains
  - Virtual Machines
  - Virtual Machine Pools

Plugin: **check_rhev3**
Author: **René Koch** <rkoch@linuxland.at>
Project Page: https://github.com/ovido/check_rhev3

```
$ ./check_rhev3.pl -H ovirt-engine \
-a admin@internal:password -D "*" -l status
RHEV CRITICAL: Datacenters critical - 1/2
Datacenters with state UP |up=1;2;2;0; contend=0;;;0; problematic=0;;;0; not_operational=0;;;0; uninitialized=1;;;0; maintenance=0;;;0;
```
More info

- oVirt
  - http://www.ovirt.org

- Mailing lists
  - users@ovirt.org
  - arch@ovirt.org
  - engine-devel@ovirt.org
  - vdsm-devel@lists.fedorahosted.org

- IRC Channel
  - #ovirt channel on irc.OFTC.net
THANK YOU!

Mike Kolesnik
mkolesni@redhat.com

René Koch
rkoch@linuxland.at