

OpenNode Management Server RPC API documentation

rev. 2011-02-04

OMS RPC service is based on qooxdoo RPC model (implemented in python) – utilizing JSON-RPC as the serialization and method call protocol. API usage: please acquire valid session id (SESSID) through AuthService and then just follow the query url examples.

Available RPC API methods are grouped into OMS RPC services like:

OMS RPC Services:
FuncService
KVMPlusOvzService
OpenVZService
SSHService
KVMService
ISOService
LVMService
AuthService
TagService
StatsService
NetInterfaceService
UserService

All RPC server service methods might raise the following error types:

General error types:
FuncManError
AuthError
ConfigurationReadError
DatabaseConnectionError
DatabaseSelectError
DatabaseUpdateError
DatabaseInsertError
DatabaseDeleteError
FuncMethodTimeoutError
NoFuncZabbixClientError
ZabbixAgentNotSupportedError

FuncService

<code>list_requests(SESSID, cache_timeout)</code>
returns list of request hostnames
query example
<code>https://hostname:8000/?id=1&service=funcservices&method=list_requests&params=["SESSID",0]</code>
sample response
<pre>{ "error": null, "id": 1, "result": { "type": "func", "result": [{ "name": "signed_hostname1" }, { "name": "signed_hostname2" }, { "name": "signed_hostname3" }] } }</pre>

<code>sign_requests(SESSID, cache_timeout, [hostname_list])</code>
returns list of hostnames whose type could not be identified and to be reviewed by user (these hosts had type "physical" assigned to them)

query example
<code>https://hostname:8000/?id=1&service=funcservices&method=sign_requests&params=["SESSID",0,["funcclient"]]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "func", "result": [{"name": "unknown_hostname1"}, {"name": "unknown_hostname2"}, {"name": "unknown_hostname3"}]}}</code>
raises errors
UnknownSignRequestsError - unknown hosts in the to-be-signed list, unknown hosts will be represented as comma separated text
SSHKeyCreateError - unable to create SSH RSA key pair on RPC server
SSHKeyAddError - unable to add SSH RSA public key to RPC server

remove_requests(SESSID, cache_timeout, [hostname_list])
removes certmaster sign requests for given hostnames
query example
<code>https://hostname:8000/?id=1&service=funcservices&method=remove_requests&params=["SESSID",0,["funcclient"]]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "func", "result": 0}}</code>
raises errors
UnknownRemoveRequestError - unknown hosts in the to-be-removed list, unknown hosts will be represented as comma separated text

list_signed(SESSID, cache_timeout, optional_tag_name)
returns list of signed hostnames (if optional_tag_name is given then returns hostnames filtered by tag_name)
query example
<code>https://hostname:8000/?id=1&service=funcservices&method=list_signed&params=["SESSID",0]</code> <code>https://hostname:8000/?id=1&service=funcservices&method=list_signed&params=["SESSID",0,"OpenNode"]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "func", "result": [{"name": "funcclient", "status": 1, "vmcount": 0, "tags": [{"tag": "OpenNode", "level": 1}]}]}}</code>
raises errors

remove_signed(SESSID, cache_timeout, [hostname_list])
removes certmaster signed certificates for given hostnames # remarks: hostname.cert is deleted on the host separately and func service is restarted with at command (a now +2minutes) after which the host is back in requests list
query example
<code>https://hostname:8000/?id=1&service=funcservices&method=remove_signed&params=["SESSID",0,["funcclient"]]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "func", "result": {"failed_removals": ["test-virt1"]}}}</code>
raises errors
UnknownRemoveSignedError - unknown hosts in the to-be-removed list, unknown hosts will be represented as comma separated text
SSHKeyRemoveError - unable to remove SSH RSA public key


```
{
  "percentage": "23.92", "total": "262144", "used": "62700", "cpu": {
    "percentage": "0.00", "network": {
      "ip": "192.168.1.125", "name": "venet0"
    }, "name": "suse-11.1-x86_64.template.com",
    "id": "125", {"status": 0, "tags": [], "stats": {
      "network": {
        "ip": "192.168.1.121", "name": "venet0"
      }, "name": "centos-5-x86_64.template.com",
      "id": "121", {"status": 0, "tags": [], "stats": {
        "network": {
          "ip": "192.168.1.122", "name": "venet0"
        }, "name": "centos-5-x86_64.template.com",
        "id": "122", {"status": 0, "tags": [], "stats": {
          "network": {
            "ip": "192.168.1.123", "name": "venet0"
          }, "name": "ubuntu-10.04-x86_64.template.com",
          "id": "123", {"status": 0, "tags": [], "stats": {
            "network": {
              "ip": "192.168.1.126", "name": "venet0"
            }, "name": "suse-11.1-x86_64.template.com",
            "id": "126", {"status": 0, "tags": [], "stats": {
              "network": {
                "ip": "192.168.1.127", "name": "venet0"
              }, "name": "fedora-13-x86_64.template.com",
              "id": "127", {"status": 0, "tags": [], "stats": {
                "network": {
                  "ip": "192.168.1.128", "name": "venet0"
                }, "name": "fedora-13-x86_64.template.com",
                "id": "128", {"status": 0, "tags": [], "stats": {
                  "network": {
                    "ip": "192.168.1.129", "name": "venet0"
                  }, "name": "debian-5.0-x86_64.template.com",
                  "id": "129", {"status": 0, "tags": [], "stats": {
                    "network": {
                      "ip": "192.168.1.130", "name": "venet0"
                    }, "name": "debian-5.0-x86_64.template.com",
                    "id": "130", {"status": 0, "tags": [], "stats": {
                      "network": {
                        "ip": "192.168.1.243", "name": "venet0"
                      }, "name": "test.krediidipank.ee",
                      "id": "131", {"status": 0, "tags": [], "stats": {
                        "network": {
                          "ip": "192.168.1.178", "name": "venet0"
                        }, "name": "krediidi_2",
                        "id": "132"
                      }
                    }
                  }
                }
              }
            }
          }
        }
      }
    }
  }
}
```

raises errors

list_vms_status(SESSID, cache_timeout, server_hostname)

returns OpenVZ container ID-s and statuses on given server

query example

```
https://hostname:8000/?id=1&service=ovzservices&method=list_vms_status&params=["SESSID", 0, "funcclient"]
```

sample response

```
{
  "error": null, "id": 1, "result": {
    "type": "func", "result": {
      "client": "funcclient", "result": {
        "130": {
          "nrVirtCpu": 4, "state": "running", "maxMem": "0", "cpuTime": "22036000000", "memory": "0",
          "121": {
            "nrVirtCpu": 4, "state": "shutdown", "maxMem": "0", "cpuTime": "0", "memory": "0",
            "122": {
              "nrVirtCpu": 4, "state": "shutdown", "maxMem": "0", "cpuTime": "0", "memory": "0",
              "123": {
                "nrVirtCpu": 4, "state": "shutdown", "maxMem": "0", "cpuTime": "0", "memory": "0",
                "124": {
                  "nrVirtCpu": 4, "state": "shutdown", "maxMem": "0", "cpuTime": "0", "memory": "0",
                  "125": {
                    "nrVirtCpu": 4, "state": "shutdown", "maxMem": "0", "cpuTime": "0", "memory": "0",
                    "126": {
                      "nrVirtCpu": 4, "state": "shutdown", "maxMem": "0", "cpuTime": "0", "memory": "0",
                      "127": {
                        "nrVirtCpu": 4, "state": "shutdown", "maxMem": "0", "cpuTime": "0", "memory": "0",
                        "128": {
                          "nrVirtCpu": 4, "state": "shutdown", "maxMem": "0", "cpuTime": "0", "memory": "0",
                          "129": {
                            "nrVirtCpu": 4, "state": "running", "maxMem": "0", "cpuTime": "19135000000", "memory": "0",
                            "104": {
                              "nrVirtCpu": 4, "state": "running", "maxMem": "0", "cpuTime": "21050000000", "memory": "0"
                            }
                          }
                        }
                      }
                    }
                  }
                }
              }
            }
          }
        }
      }
    }
  }
}
```

raises errors

list_templates(SESSID, cache_timeout, server_hostname)

returns OpenVZ templates on given server

query example

```
https://hostname:8000/?id=1&service=ovzservices&method=list_templates&params=["SESSID", 0, "funcclient"]
```

sample response

```
{
  "error": null, "id": 1, "result": {
    "type": "func", "result": {
      "client": "funcclient", "result": {
        "template1", "template2"
      }
    }
  }
}
```

raises errors

parse_vm_settings(SESSID, cache_timeout, server_hostname, container_id)

return given OVZ VM settings

query example

```
https://hostname:8000/?id=1&service=ovzservices&method=parse_vm_settings&params=["SESSID", 0, "funcclient", "129"]
```

sample response

```
"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": {"vcpulimit": "50", "vcpu": "1", "ip_address": "192.168.1.129", "max_memory": "625", "vm_id": "129", "max_vcpu": "4", "min_vcpu": "1", "max_disk": "20480", "memory": "256", "min_disk": "1024", "nameserver": "192.168.1.11", "disk": "10240", "min_memory": "0", "min_vcpulimit": "0", "max_vcpulimit": "400", "onboot": "yes", "searchdomain": "active.ee"}}}}}
```

raises errors

update_vm_settings(SESSID, cache_timeout, server_hostname, container_id, container_settings_dictionary)

update given OVZ VM settings

query example

```
https://hostname:8000/?  
id=1&service=ovzservices&method=update_vm_settings&params=["SESSID",0,"funcclient","129",  
{ "vcpu": "2", "passowrd": "test" }]
```

sample response

```
{ "error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": {"memory": "Memory size not correct", "vcpu": "Virtual CPU count not correct", "vcpulimit": "Virtual CPU limit not correct", "disk": "Disk size not correct", "ip_address": "IP address not correct", "nameserver": "Nameserver address not correct", "password": "Password can not be zero length", "onboot": "Onboot configuration has to be yes or no"}}}}}
```

raises errors

stop_vm(SESSID, cache_timeout, server_hostname, container_id)

stops given OpenVZ container

query example

```
https://hostname:8000/?  
id=1&service=ovzservices&method=stop_vm&params=["SESSID",0,"funcclient","104"]
```

sample response

```
{ "error": null, "id": 1, "result": {"type": "tmp_cache", "result": {"client": "funcclient", "result": 0}}}
```

raises errors

start_vm(SESSID, cache_timeout, server_hostname, container_id)

starts given OpenVZ container

query example

```
https://hostname:8000/?  
id=1&service=ovzservices&method=start_vm&params=["SESSID",0,"funcclient","104"]
```

sample response

```
{ "error": null, "id": 1, "result": {"type": "tmp_cache", "result": {"client": "funcclient", "result": 0}}}
```

raises errors

parse_template_for_vm(SESSID, cache_timeout, server_hostname, template_name, vm_name)

Unpacks (if necessary) given OpenVZ template. Parses template settings to a dictionary and returns it to user

query example

```
https://hostname:8000/?  
id=1&service=ovzservices&method=parse_template_for_vm&params=["SESSID",0,"funcclient","templatename",  
"virtual_machine.example.com"]
```

sample response

```
{ "error": null, "id": 1, "result": { "type": "func", "result": { "client" : "funcclient", "result" :
  { "template_name" : "template_name",
    "vm_name" : "vm_name.active.ee",
    "vm_id" : "120",
    "domain_type" : "openvz",
    "memory": "256",
    "min_memory": "256",
    "max_memory": "256",
    "vcpu" : "1",
    "min_vcpu" : "1",
    "max_vcpu" : "1",
    "vcpulimit" : "50",
    "min_vcpulimit" : "0",
    "max_vcpulimit" : "100",
    "disk" : "10240",
    "min_disk" : "10240",
    "max_disk" : "10240",
    "ip_address" : "192.168.0.1",
    "nameserver" : "192.168.0.1",
    "passwd" : ""
  }
}
}}
```

raises errors

deploy_template_for_vm(SESSID, cache_timeout, server_hostname, template_name, vm_name, settings_dictionary)

Checks given template settings, updates template settings on server (if necessary) and deploys the template to a OpenVZ CT. If the settings conflict with template or system limits then errors are reported to user. User must then fix these settings and call this function again with the updated/fixed template settings.

query example

```
https://hostname:8000/?
id=1&service=ovzservices&method=deploy_template_for_vm&params=["SESSID",0,"funcclient","templatename","virtual_machine.example.com",{}]
```

sample response

```
{ "error": null, "id": 1, "result": { "type": "func", "result": { "client" : "funcclient", "result" :
  { "memory" : "memory_error",
    "vcpu" : "vcpu_error",
    "vcpulimit" : "vcpulimit_error",
    "disk" : "disk_error",
    "ip_address" : "ip_address_error",
    "nameserver" : "nameserver_error",
    "passwd" : "root_password_error"
  }
}
}}
```

raises errors

parse_vm_for_template(SESSID, cache_timeout, server_hostname, ct_id, template_name)

Parse CT settings for template creation. CT settings are returned as a dictionary.

query example

```
https://hostname:8000/?
id=1&service=ovzservices&method=parse_vm_for_template&params=["SESSID",0,"funcclient","119","templatename"]
```

sample response

```
{ "error": null, "id": 1, "result": { "type": "func", "result": { "client" : "funcclient", "result" :
  { 'vm_id': "119",
    'template_name': 'templeit',
    'domain_type': 'openvz',
    'vcpu': "1",
    'min_vcpu': "1",
    'memory': "256",
    'min_memory': "256"
  }
}
}}
```

```
}}}
```

raises errors

```
create_template_from_vm(SESSID, cache_timeout, server_hostname,  
ct_id, template_name, settings_dictionary)
```

Create template from OpenVZ CT according to given CT settings. If any of sent settings conflict with CT or system then errors are reported back. Otherwise it is assumed that the template creation was successful.

query example

```
https://hostname:8000/?  
id=1&service=ovzservices&method=create_template_from_vm&params=["SESSID",0,"funcclient","119","tem  
platenamename",{}}
```

sample response

```
{  
  "error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": :  
    "memory": "possible error",  
    "vcpu": "possible error",  
  }}  
}
```

raises errors

```
delete_vm(SESSID, cache_timeout, server_hostname, container_id)
```

Shuts down given OpenVZ container and deletes its private area

query example

```
https://hostname:8000/?  
id=1&service=ovzservices&method=delete_vm&params=["SESSID",0,"funcclient",118]
```

sample response

```
{  
  "error": null, "id": 1, "result": {"type": "tmp_cache", "result": {"client": "funcclient",  
    "result": 0 }}  
}
```

raises errors

```
migrate_vm(SESSID, cache_timeout, server1_hostname, container_id,  
server2_hostname)
```

migrates given OpenVZ container from server1 to server2

query example

```
https://hostname:8000/?  
id=1&service=ovzservices&method=migrate_vm&params=["SESSID",0,"funcclient",110,"funcclient2"]
```

sample response

raises errors

"OpenVZMigrationNotCapable" - host is not capable to be source or target of CT migration

SSHService

```
set_up_ssh_connection(SESSID, cache_timeout, server_hostname)
```

set up SSH connection to given server

query example

```
https://hostname:8000/?
```

<code>id=1&service=sshservices&method=set_up_ssh_connection&params=["SESSID",0,"funcclient"]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "func", "result": "6900"}}</code> # After the VNC connection is set up GUI should open this URL in popup # <code>https://hostname:6900/?SESSID=SESSID</code>
raises errors
SSHAlreadySetUpError
SSHAlreadySetUpAnotherSessionError
SSHRemoteSetUpError
SSHPoolSpotSetUpError
SSHTunnelSetUpError

remove_session_ssh_connections(SESSID, cache_timeout)
remove SSH connections associated with current session
query example
<code>https://hostname:8000/?id=1&service=sshservices&method=remove_session_ssh_connections&params=["SESSID",0]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "func", "result": 0}}</code>
raises errors
SSHRemoveError

list_session_ssh_connections(SESSID, cache_timeout)
list SSH connections associated with current session
query example
<code>https://hostname:8000/?id=1&service=sshservices&method=list_session_ssh_connections&params=["SESSID",0]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "func", "result": [{"host": "funcclient", "port": 6900}]}}</code>
raises errors

check_session_ssh_connection(SESSID, cache_timeout, port)
list VNC connections associated with current session
query example
<code>https://hostname:8000/?id=1&service=sshservices&method=check_session_ssh_connection&params=["SESSID",0,6900]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "func", "result": "0"}}</code>
raises errors
SSHConnectionCheckError

KVMService

list_vms(SESSID, cache_timeout, server_hostname, stats_argument1, stats_argument2, ...)
returns KVM VM names on given server
query example

```
https://hostname:8000/?
id=1&service=kvmservices&method=list_vms&params=["SESSID",0,"funcclient","cpu","memory","df"]
```

sample response

```
{
  "error": null,
  "id": 1,
  "result": {
    "type": "func",
    "result": {
      "client": "funcclient",
      "result": [
        {
          "status": 1,
          "tags": [],
          "stats": {},
          "name": "openfiler23",
          "id": "openfiler23",
          "status": 1,
          "tags": [],
          "stats": {},
          "name": "centos54-2",
          "id": "centos54-2",
          "status": 1,
          "tags": [],
          "stats": {},
          "name": "centos54",
          "id": "centos54",
          "status": 0,
          "tags": [],
          "stats": {},
          "name": "virtual_machine.example.com",
          "id": "virtual_machine.example.com",
          "status": 0,
          "tags": [],
          "stats": {},
          "name": "openfiler2",
          "id": "openfiler2",
          "status": 0,
          "tags": [],
          "stats": {},
          "name": "winxpl",
          "id": "winxpl",
          "status": 0,
          "tags": [],
          "stats": {},
          "name": "vm_name",
          "id": "vm_name",
          "status": 0,
          "tags": [],
          "stats": {},
          "name": "test_vm_1",
          "id": "test_vm_1",
          "status": 0,
          "tags": [],
          "stats": {},
          "name": "winxp2",
          "id": "winxp2",
          "status": 0,
          "tags": [],
          "stats": {},
          "name": "openfiler24",
          "id": "openfiler24"
        ]
      ]
    }
  }
}
```

raises errors

list_vms_status(SESSID, cache_timeout, server_hostname)

returns KVM VM names and their statuses

query example

```
https://hostname:8000/?
id=1&service=kvmservices&method=list_vms_status&params=["SESSID",0,"funcclient"]
```

sample response

```
{
  "error": null,
  "id": 1,
  "result": {
    "type": "func",
    "result": {
      "client": "funcclient",
      "result": [
        {
          "Test1": {
            "nrVirtCpu": 1,
            "state": "shutdown",
            "maxMem": "524288",
            "cpuTime": "0",
            "memory": "524288",
            "vm_name": {
              "nrVirtCpu": 1,
              "state": "shutdown",
              "maxMem": "524288",
              "cpuTime": "0",
              "memory": "524288",
              "centos54": {
                "nrVirtCpu": 1,
                "state": "shutdown",
                "maxMem": "1048576",
                "cpuTime": "0",
                "memory": "1048576",
                "openfiler2": {
                  "nrVirtCpu": 1,
                  "state": "shutdown",
                  "maxMem": "524288",
                  "cpuTime": "0",
                  "memory": "524288",
                  "winxpl": {
                    "nrVirtCpu": 1,
                    "state": "shutdown",
                    "maxMem": "1048576",
                    "cpuTime": "0",
                    "memory": "1048576",
                    "winxp2": {
                      "nrVirtCpu": 1,
                      "state": "shutdown",
                      "maxMem": "1048576",
                      "cpuTime": "0",
                      "memory": "1048576",
                      "openfiler23": {
                        "nrVirtCpu": 1,
                        "state": "shutdown",
                        "maxMem": "524288",
                        "cpuTime": "0",
                        "memory": "524288",
                        "centos54-2": {
                          "nrVirtCpu": 1,
                          "state": "shutdown",
                          "maxMem": "1048576",
                          "cpuTime": "0",
                          "memory": "1048576",
                          "openfiler24": {
                            "nrVirtCpu": 1,
                            "state": "shutdown",
                            "maxMem": "524288",
                            "cpuTime": "0",
                            "memory": "524288"
                          }
                        }
                      }
                    }
                  }
                }
              }
            }
          }
        ]
      ]
    }
  }
}
```

raises errors

list_templates(SESSID, cache_timeout, server_hostname)

returns KVM templates on given server

query example

```
https://hostname:8000/?id=1&service=kvmservices&method=list_templates&params=["SESSID",0,"funcclient"]
```

sample response

```
{
  "error": null,
  "id": 1,
  "result": {
    "type": "func",
    "result": {
      "client": "funcclient",
      "result": [
        "template1",
        "template2"
      ]
    }
  }
}
```

raises errors

parse_system_min_max(SESSID, cache_timeout, server_hostname)

return system min-max for KVM VM settings

query example

```
https://hostname:8000/?
id=1&service=kvmservices&method=parse_system_min_max&params=["SESSID",0,"funcclient"]
```

sample response

```
{
  "error": null,
  "id": 1,
  "result": {
    "type": "func",
    "result": {
      "client": "funcclient",
      "result": {
        "max_vcpu": "4",
        "min_vcpu": "0",
        "min_memory": "0",
        "max_memory": "9740"
      }
    }
  }
}
```

raises errors

parse_vm_settings(SESSID, cache_timeout, server_hostname, kvm_vm_name)

return given KVM VM settings

query example

```
https://hostname:8000/?  
id=1&service=kvmservices&method=parse_vm_settings&params=["SESSID",0,"funcclient","openfiler"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": {"vcpu": "2", "max_memory": "20480", "max_vcpu": "16", "min_vcpu": "1", "memory": "1024", "min_memory": "0"}}}}
```

raises errors

update_vm_settings(SESSID, cache_timeout, server_hostname, kvm_vm_name, kvm_vm_settings_dictionary)

update given KVM VM settings

query example

```
https://hostname:8000/?  
id=1&service=kvmservices&method=update_vm_settings&params=["SESSID",0,"funcclient","openfiler", {"vcpu": "2"}]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": {"memory": "Memory size not correct", "vcpu": "Virtual CPU count not correct"}}}}
```

raises errors

set_up_vnc_connection(SESSID, cache_timeout, server_hostname, kvm_vm_name)

set up vnc connection to given KVM VM

returns port number where noVNC client should connect on the Funcman RPC server

query example

```
https://hostname:8000/?  
id=1&service=kvmservices&method=set_up_vnc_connection&params=["SESSID",0,"funcclient","winxp2"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": "6801"}}  
# After the VNC connection is set up GUI should open this URL in popup  
# https://hostname:8002/vnc.html?SESSID=SESSID&host=hostname&port=6801  
# This is HTML5 VNC window that automatically connects on window opening. Opened window should not  
# have any task/option bars.
```

raises errors

VNCAreadySetUpError

VNCAreadySetUpAnotherSessionError

VNCRemoteSetUpError

VNCPoolSpotSetUpError

VNCTunnelSetUpError

remove_vnc_connection(SESSID, cache_timeout, server_hostname, kvm_vm_name)

remove vnc connection to given KVM VM

returns port number of the removed VNC connection

query example

```
https://hostname:8000/?  
id=1&service=kvmservices&method=remove_vnc_connection&params=["SESSID",0,"funcclient","winxp2"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": "6801"}}
```

raises errors

VNCRemoveError

remove_session_vnc_connections(SESSID, cache_timeout)

remove VNC connections associated with current session

query example

```
https://hostname:8000/?  
id=1&service=kvmservices&method=remove_session_vnc_connections&params=["SESSID",0]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": 0}}
```

raises errors

VNCRemoveError

list_session_vnc_connections(SESSID, cache_timeout)

list VNC connections associated with current session

query example

```
https://hostname:8000/?  
id=1&service=kvmservices&method=list_session_vnc_connections&params=["SESSID",0]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": [{"host": "funcclient", "port":  
6801, "vm": "winxp2"}]}}
```

raises errors

check_session_vnc_connection(SESSID, cache_timeout, port)

list VNC connections associated with current session

query example

```
https://hostname:8000/?  
id=1&service=kvmservices&method=check_session_vnc_connection&params=["SESSID",0,6801]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": "0"}}
```

raises errors

VNCConnectionCheckError

stop_vm(SESSID, cache_timeout, server_hostname, kvm_vm_name)

stops given KVM VM

query example

```
https://hostname:8000/?  
id=1&service=kvmservices&method=stop_vm&params=["SESSID",0,"funcclient","openfiler2"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":  
0}}}
```

raises errors

```
start_vm(SESSID, cache_timeout, server_hostname, kvm_vm_name)
```

```
# starts given KVM VM
```

```
# query example
```

```
https://hostname:8000/?  
id=1&service=kvm&method=start_vm&params=["SESSID",0,"funcclient","openfiler"]
```

```
# sample response
```

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": 0}}}
```

```
# raises errors
```

```
parse_template_for_vm(SESSID, cache_timeout, server_hostname,  
template_name, kvm_vm_name)
```

```
# Unpacks (if necessary) given KVM template. Parses template settings to a dictionary and returns it to user
```

```
# query example
```

```
https://hostname:8000/?  
id=1&service=kvm&method=parse_template_for_vm&params=["SESSID",0,"funcclient","templatename","test_vm_1"]
```

```
# sample response
```

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": {"template_name": "template_name",  
"vm_name": "vm_name",  
"domain_type": "kvm",  
"virt_type": "hvm",  
"memory": "256",  
"arch": "x86_64",  
"vcpu": "vcpu_nr",  
"boot_dev": "hd",  
"features": ["acpi", "apic", "pae"],  
"clock_offset": "utc",  
"emulator": "/usr/libexec/qemu-kvm",  
"on_poweroff": "destroy",  
"on_reboot": "reboot",  
"on_crash": "reboot",  
"interface": {"type": "bridge",  
"source_bridge": "vibr0"},  
"serial": {"type": "pty",  
"target_port": "0"},  
"console": {"type": "pty",  
"target_port": "0"},  
"mouse_bus": "ps2",  
"graphics": {"type": "vnc",  
"port": "-1",  
"autoport": "yes",  
"keymap": "et"},  
"disks": [{"template_name": "diskimage.img",  
"template_format": "qcow2",  
"template_capacity": "disk_image_size",  
"deploy_type": "physical",  
"type": "block",  
"device": "disk",  
"source_dev": "/dev/VolGroup00/openfiler",  
"target_dev": "vda",  
"target_bus": "virtio"},  
{"template_name": "diskimage.img",  
"template_format": "qcow2",  
"template_capacity": "disk_image_size",  
"deploy_type": "lvm",  
"type": "block",  
"device": "disk",  
"source_dev": "/dev/sde",
```

```

        "target_dev" : "vdc",
        "target_bus" : "virtio"
    }
    {
        "template_name" : "diskimage.img",
        "template_format" : "qcow2",
        "template_capacity" : "disk_image_size",
        "deploy_type" : "file",
        "type" : "file",
        "device" : "disk",
        "source_file" : "nasdisk.img",
        "target_dev" : "vda",
        "target_bus" : "virtio"
    }
}
}}}

```

raises errors

deploy_template_for_vm(SESSID, cache_timeout, server_hostname, template_name, kvm_name, template_settings_dictionary)

Checks given template settings, updates template settings on server (if necessary) and
deploys the template to a KVM VM.
If the settings conflict with template or system then errors are reported back. User must then fix these settings
and call this function again with the updated/fixed template settings.
If no errors are reported back it is assumed that the template deployment was successful.

query example

```

https://hostname:8000/?
id=1&service=kvm&method=deploy_template_for_vm&params=["SESSID",0,"funcclient","templeit",
"test_vm_1",{}]

```

sample response

```

{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": {
    "virt_type": "possible error",
    "memory": "possible error",
    "arch": "possible error",
    "vcpu": "possible error",
    "boot_dev": "possible error",
    "features": "possible error",
    "clock_offset": "possible error",
    "emulator": "possible error",
    "on_poweroff": "possible error",
    "on_reboot": "possible error",
    "on_crash": "possible error",
    "interface": "possible error",
    "serial": "possible error",
    "console": "possible error",
    "mouse_bus": "ps2",
    "graphics": "possible error",
    "disks": [
        {
            "template_name": "diskimage.img",
            "error": "possible error"
        }
    ]
}}}

```

raises errors

parse_vm_for_template(SESSID, cache_timeout, server_hostname, kvm_vm_name, template_name)

Parse VM settings for template creation. VM settings are returned as a dictionary.

query example

```

https://hostname:8000/?
id=1&service=kvm&method=parse_vm_for_template&params=["SESSID",0,"funcclient","openfiler23",
"templatename"]

```

sample response

```
{
  "error": null, "id": 1, "result": {
    "type": "func", "result": {
      "client": "funcclient", "result": {
        'vm_name': 'openfiler23',
        'vcpu': "1",
        'min_vcpu': "1",
        'template_name': 'templeit',
        'interfaces':
        [
          {
            'source_bridge': 'vmbr0',
            'type': 'bridge',
            'mac_address': '54:52:00:07:df:7a'
          }
        ],
        'disks':
        [
          {
            'disk_capacity': "8878644224",
            'new_path': '/storage/templates/kvm/deploy/templeit/templeit1.img',
            'disk_id': 'vmdisk1.img',
            'filename': 'templeit1.img',
            'file_id': 'diskfile1',
            'disk_path': '/storage/images/openfiler23.img',
            'file_size': "1074040832"
          }
        ],
        'domain_type': 'kvm',
        'memory': "256",
        'min_memory': "256",
        'arch': 'x86_64'
      }
    }
  }
}
```

raises errors

create_template_from_vm(SESSID, cache_timeout, server_hostname, kvm_vm_name, template_name, vm_settings_dictionary)

Create template from KVM VM according to given VM settings. If any of sent settings conflict with VM or system then errors are reported back. Otherwise it is assumed that the template creation was successful.

query example

```
https://hostname:8000/?
id=1&service=kvm&method=create_template_from_vm&params=["SESSID",0,"funcclient","openfiler23","templeit",{}]
```

sample response

```
{
  "error": null, "id": 1, "result": {
    "type": "func", "result": {
      "client": "funcclient", "result": {
        "memory": "possible error",
        "arch": "possible error",
        "vcpu": "possible error",
      }
    }
  }
}
```

raises errors

create_vm_from_iso(SESSID, cache_timeout, server_hostname, kvm_vm_name, os_variant, os_type, cdrom_iso, file_based_disk_size, block_device_based_disk_path, memory_size, noapic_on_off, noacpi_on_off)

install KVM VM from an ISO image

query example

```
https://hostname:8000/?
id=1&service=kvm&method=create_vm_from_iso&params=["SESSID",0,"funcclient","new_test_vm","winxp","windows","winxpsp2corp.iso","10","","512","yes",""]
```

sample response

```
{
  "error": null, "id": 1, "result": {
    "type": "func", "result": {
      "client": "funcclient", "result": {
        0
      }
    }
  }
}
```

currently returns an exception or keeps running until a first reboot is done on the created VM (then returns the given output)

raises errors

delete_vm(SESSID, cache_timeout, server_hostname, kvm_vm_name)

shuts down and destroys given KVM VM

query example

```
https://hostname:8000/?  
id=1&service=kvm&method=delete_vm&params=["SESSID",0,"funcclient","openfiler2"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": 0}}}
```

raises errors

create_file_based_image(SESSID, cache_timeout, server_hostname, diskimage_name, diskimage_size_megabytes)

creates file based disk image

query example

```
https://hostname:8000/?  
id=1&service=kvm&method=create_file_based_image&params=["SESSID",0,"funcclient","testimage.img",10]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": 0}}}
```

raises errors

delete_file_based_image(SESSID, cache_timeout, server_hostname, diskimage_name)

deletes file based disk image

query example

```
https://hostname:8000/?  
id=1&service=kvm&method=delete_file_based_image&params=["SESSID",0,"funcclient","testimage.img"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": [0, "", ""]}}}
```

raises errors

list_file_based_images(SESSID, cache_timeout, server_hostname)

returns list of file based disk images

query example

```
https://hostname:8000/?  
id=1&service=kvm&method=list_file_based_image&params=["SESSID",0,"funcclient"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": ["openfiler23.img", "openfiler24.img", "openfiler2.img", "openfiler2.img.backup", "Test1.img", "testimage.img", "test_vm-templeit1.img", "winxpl.img"]}}}
```

raises errors

list_file_based_images_used(SESSID, cache_timeout, server_hostname)

returns list of file based disk images that are defined in KVM VM configurations (that are in use currently)

query example

```
https://hostname:8000/?id=1&service=kvmservices&method=list_file_based_image_used&params=["SESSID",0,"funcclient"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": ["openfiler23.img", "openfiler2.img", "winxpl1.img", "storage_file_name.img", "openfiler24.img"]}}}
```

raises errors

ISOService

list_images(SESSID, cache_timeout, server_hostname)

returns ISO images on given server

query example

```
https://hostname:8000/?id=1&service=isoservices&method=list_images&params=["SESSID",0,"funcclient"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": ["openfiler-2.3-x86_64-discl.iso", "winxpsp2corp.iso", "CentOS-5.4-x86_64-bin-DVD.iso"]}}}
```

raises errors

LVMService

list_vg(SESSID, cache_timeout, server_hostname)

returns LVM Volume Groups on server

query example

```
https://hostname:8000/?id=1&service=lvmservices&method=list_vg&params=["SESSID",0,"funcclient"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": [[['VG', 'Name', 'VolGroup00'], ['System', 'ID'], ['Format', 'lvm2'], ['Metadata', 'Areas', '1'], ['Metadata', 'Sequence', 'No', '17'], ['VG', 'Access', 'read/write'], ['VG', 'Status', 'resizable'], ['MAX', 'LV', '0'], ['Cur', 'LV', '8'], ['Open', 'LV', '5'], ['Max', 'PV', '0'], ['Cur', 'PV', '1'], ['Act', 'PV', '1'], ['VG', 'Size', '135,09', 'GB'], ['PE', 'Size', '32,00', 'MB'], ['Total', 'PE', '4323'], ['Alloc', 'PE', '/', 'Size', '3776', '/', '118,00', 'GB'], ['Free', 'PE', '/', 'Size', '547', '/', '17,09', 'GB'], ['VG', 'UUID', 'Sb1lok-R2y8-cOvr-mDqL-vn0z-AZzs-zFdyNw']]]}}}
```

raises errors

list_lv(SESSID, cache_timeout, server_hostname)

returns LVM Volumes on server

query example

```
https://hostname:8000/?id=1&service=lvmservices&method=list_lv&params=["SESSID",0,"funcclient"]
```

sample response

AuthService

authenticate(username, password)
authenticates user # returns SESSID
query example
<code>https://hostname:8000/?id=1&service=authservices&method=authenticate&params=["username","password"]</code>
sample response
<code>"error": null, "id": 1, "result": {"type": "auth", "result": "8BA9DAE5D36E46B5E13DD035D4BBEE763ADB280"}}</code>
raises errors
AuthError - username or password were not valid or the authentication with those credentials failed

deauthenticate(SESSID)
deauthenticate user, destroys session
query example
<code>https://hostname:8000/?id=1&service=authservices&method=deauthenticate&params=["SESSID"]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "auth", "result": 0}}</code>
raises errors
DeathError - the given session can not be deauthenticated because it probably does not exist

checksessid(SESSID)
check user's session validity
query example
<code>https://hostname:8000/?id=1&service=authservices&method=checksessid&params=["SESSID"]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "auth", "result": 0}}</code>
raises errors
AuthError

TagService

change_host_type(SESSID, cache_timeout, [{"name":host_name, "type":new_host_type}, ...])
changes host's types (it means hosts level 1 tag name). hosts are passed as a list of dictionaries
query example
<code>https://hostname:8000/?id=1&service=tagservices&method=change_host_type&params=["SESSID", 0, [{"name":"host_name", "type":"new_host_type"}]]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "func", "result": 0}}</code>
raises errors
UnknownHostException - unknown host type assigned to the host (not within default host types)

list_tags(SESSID, cache_timeout)
returns list of tags stored on OMS RPC server

query example
<code>https://hostname:8000/?id=1&service=tagservices&method=list_tags&params=["SESSID", 0]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "func", "result": [{"tag": "OpenNode", "level": 1}]}}</code>
raises errors

edit_freeform_tag_name(SESSID, cache_timeout, old_tag_name, new_tag_name)
changes freeform (level 3) tag name
query example
<code>https://hostname:8000/?id=1&service=tagservices&method=edit_freeform_tag_name&params=["SESSID", 0, "old_name", "new_name"]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "func", "result": 0}}</code>
raises errors

StatsService

get_stats(SESSID, cache_timeout, server_hostname, "cpu_load", "memory", "df", "uptime", "kernel", "network", "bandwidth", "swap", "iowait")
returns stats from given physical server_hostname # arguments define which stats will be returned, for example # cpu - CPU load # memory - memory information # df - usage of disks/partitions # uptime - server uptime # kernel - kernel version # network - network interfaces status # bandwidth - network bandwidth statistics (represented in following ways XXB, XXk, XXM) # swap - server's swap space path and usage statistics # iowait - I/O wait statistics
query example
<code>https://hostname:8000/?id=1&service=statsservices&method=get_stats&params=["SESSID", 0, "funcclient"]</code>
sample response
<code>{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result": {"cpu": {"percentage": "0.00", "cpu_cache": "4096 KB", "cpu_type": "Intel(R) Xeon(R) CPU 5110 @ 1.60GHz", "cpu_mhz": "1596.040"}, "df": {"percentage": "0.00", "total": "0", "used": "0"}, "details": [{"mount": "/", "used": "5436672", "dev": "/dev/mapper/VolGroup00-root", "free": "4196408"}, {"mount": "/storage", "used": "128717868", "dev": "/dev/mapper/VolGroup00-storage", "free": "63946828"}, {"mount": "/vz", "used": "7009104", "dev": "/dev/mapper/VolGroup00-vz", "free": "89322960"}, {"mount": "/backup", "used": "1579788", "dev": "/dev/mapper/VolGroup00-backup", "free": "27319536"}, {"mount": "/boot", "used": "113240", "dev": "/dev/sda1", "free": "853360"}, {"mount": "/dev/shm", "used": "0", "dev": "tmpfs", "free": "5620020"}]}, "kernel": "2.6.18-164.15.1.el5.028stab068.9", "memory": {"percentage": "0.00", "total": "0", "used": "0"}, "network": {"ip": "192.168.1.40", "name": "vmbro"}}</code>

```

        "uptime": "855087",
        "bandwidth": { 'percentage': '0.00', "received": 5447, "maximum": 104857600,
"sent": 6619},
        "osrelease": "Build 28",
        "swap": { "path": "/dev/mapper/VolGroup00-swap", "percentage": "0.00", "total":
"8388600", "used": "0"},
        "iowait" : { "percentage": "0.30"}
    }}

```

raises errors

NetInterfaceService

show_routing_table(SESSID, cache_timeout, server_name)

show servers routing table

query example

```

https://hostname:8000/?
id=1&service=netinterfaceservices&method=show_routing_table&params=["SESSID", 0, "funcclient"]

```

sample response

```

{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":
[{"count": "0", "use": "0", "destination": "192.168.1.211", "metrics": "0", "netmask":
"255.255.255.255", "flags": "UH", "interface": "venet0", "router": "*"}, {"count": "0", "use":
"0", "destination": "192.168.1.122", "metrics": "0", "netmask": "255.255.255.255", "flags": "UH",
"interface": "venet0", "router": "*"}, {"count": "0", "use": "0", "destination": "192.168.1.0",
"metrics": "0", "netmask": "255.255.255.0", "flags": "U", "interface": "vibr0", "router": "*"},
{"count": "0", "use": "0", "destination": "192.168.122.0", "metrics": "0", "netmask":
"255.255.255.0", "flags": "U", "interface": "virbr0", "router": "*"}, {"count": "0", "use": "0",
"destination": "169.254.0.0", "metrics": "0", "netmask": "255.255.0.0", "flags": "U", "interface":
"vibr0", "router": "*"}, {"count": "0", "use": "0", "destination": "default", "metrics": "0",
"netmask": "0.0.0.0", "flags": "UG", "interface": "vibr0", "router": "192.168.1.1"}]}}}

```

raises errors

add_static_route(SESSID, cache_timeout, server_name, destination, netmask, gateway, interface)

add static route entry to server

query example

```

https://hostname:8000/?
id=1&service=netinterfaceservices&method=add_static_route&params=["SESSID",0,"funcclient",
"192.168.3.0","255.255.255.0","0.0.0.0", "vibr0"]

```

sample response

```

{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":
0}}}

```

raises errors

edit_static_route(SESSID, cache_timeout, server_name, destination_old, netmask_old, gateway_old, interface, destination_new, netmask_new, gateway_new)

edit servers static route entry

query example

```

https://hostname:8000/?
id=1&service=netinterfaceservices&method=edit_static_route&params=["SESSID",0,"funcclient",
"192.168.3.0","255.255.255.0","0.0.0.0", "vibr0","192.168.4.0","255.255.255.0","0.0.0.0"]

```

sample response

```

{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":
0}}}

```

raises errors

delete_static_route(SESSID, cache_timeout, server_name, destination, netmask, gateway, interface)

delete static route entry to server

query example

```
https://hostname:8000/?  
id=1&service=netinterfaceservices&method=delete_static_route&params=["SESSID",0,"funcclient",  
"192.168.3.0","255.255.255.0","0.0.0.0","vubr0"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":  
0}}}
```

raises errors

edit_static_gateway(SESSID, cache_timeout, server_name, gateway, interface)

edit servers static gateway

query example

```
https://hostname:8000/?  
id=1&service=netinterfaceservices&method=edit_static_gateway&params=["SESSID",0,"funcclient",  
"192.168.1.1","vubr0"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":  
0}}}
```

raises errors

get_default_gateway(SESSID, cache_timeout, server_name, interface)

get servers default gateway for corresponding interface

query example

```
https://hostname:8000/?  
id=1&service=netinterfaceservices&method=get_default_gateway&params=["SESSID",0,"funcclient","vubr  
0"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":  
"192.168.1.1"}}}
```

raises errors

list_bridges(SESSID, cache_timeout, server_name)

list server's bridge interfaces with their members

query example

```
https://hostname:8000/?  
id=1&service=netinterfaceservices&method=list_bridges&params=["SESSID",0,"funcclient"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":  
[{"bridge": "vubr0", "members": ["eth0", "vnet3", "vnet2", "vnet0"]}]}}
```

raises errors

add_bridge(SESSID, cache_timeout, server_name, bridge_name, ip_address, netmask, gateway)

add new bridge interface to server

query example

```
https://hostname:8000/?  
id=1&service=netinterfaceservices&method=add_bridge&params=["SESSID",0,"funcclient","testbridge",  
192.168.1.41","255.255.255.0","192.168.1.1"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":  
0}}}
```

raises errors

add_bridge_member(SESSID, cache_timeout, server_name, bridge_name, bridge_member)

add new member to bridge interface on server

query example

```
https://hostname:8000/?  
id=1&service=netinterfaceservices&method=add_bridge_member&params=["SESSID",0,"funcclient","testbr  
idge","eth3"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":  
0}}}
```

raises errors

delete_bridge_member(SESSID, cache_timeout, server_name, bridge_name, bridge_member)

delete member from bridge interface on server

query example

```
https://hostname:8000/?  
id=1&service=netinterfaceservices&method=delete_bridge_member&params=["SESSID",0,"funcclient","tes  
tbridge","eth3"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":  
0}}}
```

raises errors

delete_bridge(SESSID, cache_timeout, server_name, bridge_name)

delete bridge interface from server

query example

```
https://hostname:8000/?  
id=1&service=netinterfaceservices&method=delete_bridge&params=["SESSID",0,"funcclient","testbridge  
"]
```

sample response

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":  
0}}}
```

raises errors

```
edit_bridge(SESSID, cache_timeout, server_name, bridge_name,  
ip_address, netmask, gateway)
```

```
# edit bridge interface on server
```

```
# query example
```

```
https://hostname:8000/?  
id=1&service=netinterfaceservices&method=edit_bridge&params=["SESSID",0,"funcclient","testbridge",  
"192.168.1.41","255.255.255.0","192.168.1.1"]
```

```
# sample response
```

```
{"error": null, "id": 1, "result": {"type": "func", "result": {"client": "funcclient", "result":  
0}}}
```

```
# raises errors
```