



# Xen Management API

Version: API Revision 1.0.0

Date: 27th April 2007

Stable Release

Ewan Mellor: `ewan@xensource.com`

Richard Sharp: `richard.sharp@xensource.com`

David Scott: `david.scott@xensource.com`

## Contributors:

Stefan Berger, IBM  
Daniel Berrangé, Red Hat  
Gareth Bestor, IBM  
Hollis Blanchard, IBM  
Mike Day, IBM  
Jim Fehlig, Novell  
Jon Harrop, XenSource

Vincent Hanquez, XenSource  
John Levon, Sun Microsystems  
Jon Ludlam, XenSource  
Alastair Tse, XenSource  
Daniel Veillard, Red Hat  
Tom Wilkie, University of Cambridge

Copyright © 2006-2007 XenSource, Inc.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

# Chapter 1

## Introduction

This document contains a description of the Xen Management API—an interface for remotely configuring and controlling virtualised guests running on a Xen-enabled host.

The API is presented here as a set of Remote Procedure Calls, with a wire format based upon XML-RPC. No specific language bindings are prescribed, although examples will be given in the python programming language.

Although we adopt some terminology from object-oriented programming, future client language bindings may or may not be object oriented. The API reference uses the terminology *classes* and *objects*. For our purposes a *class* is simply a hierarchical namespace; an *object* is an instance of a class with its fields set to specific values. Objects are persistent and exist on the server-side. Clients may obtain opaque references to these server-side objects and then access their fields via *get/set* RPCs.

For each class we specify a list of fields along with their *types* and *qualifiers*. A qualifier is one of:

- *RO<sub>run</sub>*: the field is Read Only. Furthermore, its value is automatically computed at runtime. For example: current CPU load and disk IO throughput.
- *RO<sub>ins</sub>*: the field must be manually set when a new object is created, but is then Read Only for the duration of the object's life. For example, the maximum memory addressable by a guest is set before the guest boots.
- *RW*: the field is Read/Write. For example, the name of a VM.

A full list of types is given in Chapter 2. However, there are three types that require explicit mention:

- *t Ref*: signifies a reference to an object of type *t*.
- *t Set*: signifies a set containing values of type *t*.
- $(t_1, t_2)$  *Map*: signifies a mapping from values of type  $t_1$  to values of type  $t_2$ .

Note that there are a number of cases where *Refs* are *doubly linked*—e.g. a VM has a field called **VIFs** of type  $(VIF\ Ref)\ Set$ ; this field lists the network interfaces attached to a particular VM. Similarly, the VIF class has a field called **VM** of type  $(VM\ Ref)$  which references the VM to which the interface is connected. These two fields are *bound together*, in the sense that creating a new VIF causes the **VIFs** field of the corresponding VM object to be updated automatically.

The API reference explicitly lists the fields that are bound together in this way. It also contains a diagram that shows relationships between classes. In this diagram an edge signifies the existence of a pair of fields that are bound together, using standard crows-foot notation to signify the type of relationship (e.g. one-many, many-many).

## 1.1 RPCs associated with fields

Each field,  $f$ , has an RPC accessor associated with it that returns  $f$ 's value:

- “`get_f(Ref x)`”: takes a `Ref` that refers to an object and returns the value of  $f$ .

Each field,  $f$ , with attribute  $RW$  and whose outermost type is *Set* has the following additional RPCs associated with it:

- an “`add_to_f(Ref x, v)`” RPC adds a new element  $v$  to the set<sup>1</sup>;
- a “`remove_from_f(Ref x, v)`” RPC removes element  $v$  from the set;

Each field,  $f$ , with attribute  $RW$  and whose outermost type is *Map* has the following additional RPCs associated with it:

- an “`add_to_f(Ref x, k, v)`” RPC adds new pair  $(k, v)$  to the mapping stored in  $f$  in object  $x$ . Adding a new pair for duplicate key,  $k$ , overwrites any previous mapping for  $k$ .
- a “`remove_from_f(Ref x, k)`” RPC removes the pair with key  $k$  from the mapping stored in  $f$  in object  $x$ .

Each field whose outermost type is neither *Set* nor *Map*, but whose attribute is  $RW$  has an RPC accessor associated with it that sets its value:

- For  $RW$  (*Read/Write*), a “`set_f(Ref x, v)`” RPC function is also provided. This sets field  $f$  on object  $x$  to value  $v$ .

## 1.2 RPCs associated with classes

- Each class has a *constructor* RPC named “`create`” that takes as parameters all fields marked  $RW$  and  $RO_{ins}$ . The result of this RPC is that a new *persistent* object is created on the server-side with the specified field values.
- Each class has a `get_by_uuid(uuid)` RPC that returns the object of that class that has the specified `uuid`.
- Each class that has a `name_label` field has a “`get_by_name_label(name)`” RPC that returns a set of objects of that class that have the specified `label`.
- Each class has a “`destroy(Ref x)`” RPC that explicitly deletes the persistent object specified by  $x$  from the system. This is a non-cascading delete – if the object being removed is referenced by another object then the `destroy` call will fail.

### 1.2.1 Additional RPCs

As well as the RPCs enumerated above, some classes have additional RPCs associated with them. For example, the `VM` class has RPCs for cloning, suspending, starting etc. Such additional RPCs are described explicitly in the API reference.

---

<sup>1</sup>Since sets cannot contain duplicate values this operation has no action in the case that  $v$  was already in the set.

## 1.3 Wire Protocol for Remote API Calls

API calls are sent over a network to a Xen-enabled host using the XML-RPC protocol. In this Section we describe how the higher-level types used in our API Reference are mapped to primitive XML-RPC types.

In our API Reference we specify the signatures of API functions in the following style:

```
(ref_vm Set) VM.get_all()
```

This specifies that the function with name `VM.get_all` takes no parameters and returns a Set of `ref_vms`. These types are mapped onto XML-RPC types in a straight-forward manner:

- Floats, Booleans, DateTimes and Strings map directly to the XML-RPC `double`, `boolean`, `dateTime.iso8601`, and `string` elements.
- all “`ref_`” types are opaque references, encoded as the XML-RPC’s `String` type. Users of the API should not make assumptions about the concrete form of these strings and should not expect them to remain valid after the client’s session with the server has terminated.
- fields named “`uuid`” of type “`String`” are mapped to the XML-RPC `String` type. The string itself is the OSF DCE UUID presentation format (as output by `uuidgen`, etc).
- ints are all assumed to be 64-bit in our API and are encoded as a string of decimal digits (rather than using XML-RPC’s built-in 32-bit `i4` type).
- values of enum types are encoded as strings. For example, a value of `destroy` of type `on_normal_exit`, would be conveyed as:

```
<value><string>destroy</string></value>
```

- for all our types, `t`, our type `t Set` simply maps to XML-RPC’s `Array` type, so for example a value of type `cpu_feature Set` would be transmitted like this:

```
<array>
  <data>
    <value><string>CX8</string></value>
    <value><string>PSE36</string></value>
    <value><string>FPU</string></value>
  </data>
</array>
```

- for types `k` and `v`, our type `(k, v) Map` maps onto an XML-RPC struct, with the key as the name of the struct. Note that the `(k, v) Map` type is only valid when `k` is a `String`, `Ref`, or `Int`, and in each case the keys of the maps are stringified as above. For example, the `(String, double) Map` containing the mappings `Mike → 2.3` and `John → 1.2` would be represented as:

```
<value>
  <struct>
    <member>
      <name>Mike</name>
      <value><double>2.3</double></value>
    </member>
    <member>
```

```

    <name>John</name>
    <value><double>1.2</double></value>
  </member>
</struct>
</value>

```

- our Void type is transmitted as an empty string.

### 1.3.1 Note on References vs UUIDs

References are opaque types — encoded as XML-RPC strings on the wire — understood only by the particular server which generated them. Servers are free to choose any concrete representation they find convenient; clients should not make any assumptions or attempt to parse the string contents. References are not guaranteed to be permanent identifiers for objects; clients should not assume that references generated during one session are valid for any future session. References do not allow objects to be compared for equality. Two references to the same object are not guaranteed to be textually identical.

UUIDs are intended to be permanent names for objects. They are guaranteed to be in the OSF DCE UUID presentation format (as output by `uuidgen`). Clients may store UUIDs on disk and use them to lookup objects in subsequent sessions with the server. Clients may also test equality on objects by comparing UUID strings.

The API provides mechanisms for translating between UUIDs and opaque references. Each class that contains a UUID field provides:

- A “`get.by.uuid`” method that takes a UUID,  $u$ , and returns an opaque reference to the server-side object that has `UUID=u`;
- A `get.uuid` function (a regular “field getter” RPC) that takes an opaque reference,  $r$ , and returns the UUID of the server-side object that is referenced by  $r$ .

### 1.3.2 Return Values/Status Codes

The return value of an RPC call is an XML-RPC Struct.

- The first element of the struct is named `Status`; it contains a string value indicating whether the result of the call was a “Success” or a “Failure”.

If `Status` was set to `Success` then the Struct contains a second element named `Value`:

- The element of the struct named `Value` contains the function’s return value.

In the case where `Status` is set to `Failure` then the struct contains a second element named `ErrorDescription`:

- The element of the struct named `ErrorDescription` contains an array of string values. The first element of the array is an error code; the remainder of the array are strings representing error parameters relating to that code.

For example, an XML-RPC return value from the `host.get_resident_VMs` function above may look like this:

```

<struct>
  <member>
    <name>Status</name>
    <value>Success</value>
  </member>

```

```

<member>
  <name>Value</name>
  <value>
    <array>
      <data>
        <value>81547a35-205c-a551-c577-00b982c5fe00</value>
        <value>61c85a22-05da-b8a2-2e55-06b0847da503</value>
        <value>1d401ec4-3c17-35a6-fc79-cee6bd9811fe</value>
      </data>
    </array>
  </value>
</member>
</struct>

```

## 1.4 Making XML-RPC Calls

### 1.4.1 Transport Layer

The following transport layers are currently supported:

- HTTP/S for remote administration
- HTTP over Unix domain sockets for local administration

### 1.4.2 Session Layer

The XML-RPC interface is session-based; before you can make arbitrary RPC calls you must login and initiate a session. For example:

```
session_id session.login_with_password(string uname, string pwd)
```

Where `uname` and `password` refer to your username and password respectively, as defined by the Xen administrator. The `session_id` returned by `session.login_with_password` is passed to subsequent RPC calls as an authentication token.

A session can be terminated with the `session.logout` function:

```
void session.logout(session_id session)
```

### 1.4.3 Synchronous and Asynchronous invocation

Each method call (apart from methods on “Session” and “Task” objects and “getters” and “setters” derived from fields) can be made either synchronously or asynchronously. A synchronous RPC call blocks until the return value is received; the return value of a synchronous RPC call is exactly as specified in Section 1.3.2.

Only synchronous API calls are listed explicitly in this document. All asynchronous versions are in the special `Async` namespace. For example, synchronous call `VM.clone(...)` (described in Chapter 2) has an asynchronous counterpart, `Async.VM.clone(...)`, that is non-blocking.

Instead of returning its result directly, an asynchronous RPC call returns a `task-id`; this identifier is subsequently used to track the status of a running asynchronous RPC. Note that an asynchronous call may fail immediately, before a `task-id` has even been created—to represent this eventuality, the returned `task-id` is wrapped in an XML-RPC struct with a `Status`, `ErrorDescription` and `Value` fields, exactly as specified in Section 1.3.2.

The `task-id` is provided in the `Value` field if `Status` is set to `Success`.

The RPC call

```
(ref_task Set) Task.get_all(session_id s)
```

returns a set of all task IDs known to the system. The status (including any returned result and error codes) of these tasks can then be queried by accessing the fields of the Task object in the usual way. Note that, in order to get a consistent snapshot of a task's state, it is advisable to call the "get\_record" function.

## 1.5 Example interactive session

This section describes how an interactive session might look, using the python XML-RPC client library.

First, initialise python and import the library `xmlrpclib`:

```
\$ python2.4
...
>>> import xmlrpclib
```

Create a python object referencing the remote server:

```
>>> xen = xmlrpclib.Server("http://test:4464")
```

Acquire a session token by logging in with a username and password (error-handling omitted for brevity; the session token is pointed to by the key 'Value' in the returned dictionary)

```
>>> session = session.login_with_password("user", "passwd")['Value']
```

When serialised, this call looks like the following:

```
<?xml version='1.0'?>
<methodCall>
  <methodName>session.login_with_password</methodName>
  <params>
    <param>
      <value><string>user</string></value>
    </param>
    <param>
      <value><string>passwd</string></value>
    </param>
  </params>
</methodCall>
```

Next, the user may acquire a list of all the VMs known to the host: (Note the call takes the session token as the only parameter)

```
>>> all_vms = host.get_resident_VMs(session)['Value']
>>> all_vms
['OpaqueRef:1', 'OpaqueRef:2', 'OpaqueRef:3', 'OpaqueRef:4' ]
```

The VM references here have the form `OpaqueRef:X`, though they may not be that simple in the future, and you should treat them as opaque strings. Once a reference to a VM has been acquired a lifecycle operation may be invoked:

```
>>> xen.VM.start(session, all_vms[3], False)
{'Status': 'Failure', 'ErrorDescription': ['VM_BAD_POWER_STATE', 'Halted', 'Running']}
```

In this case the `start` message has been rejected, because the VM is already running, and so an error response has been returned. These high-level errors are returned as structured data (rather than as XML-RPC faults), allowing them to be internationalised. Finally, here are some examples of using accessors for object fields:

```
>>> xen.VM.get_name_label(session, all_vms[3])['Value']  
'SMP'  
>>> xen.VM.get_name_description(session, all_vms[3])['Value']  
'Debian for Xen'
```



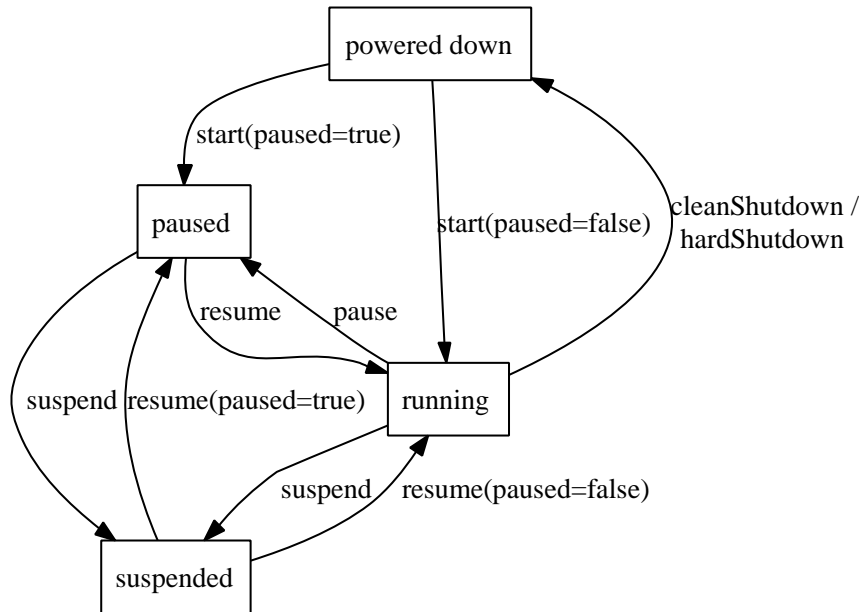


Figure 1.1: VM Lifecycle

## 1.6 VM Lifecycle

Figure 1.1 shows the states that a VM can be in and the API calls that can be used to move the VM between these states.

## 1.7 VM boot parameters

The VM class contains a number of fields that control the way in which the VM is booted. With reference to the fields defined in the VM class (see later in this document), this section outlines the boot options available and the mechanisms provided for controlling them.

VM booting is controlled by setting one of the two mutually exclusive groups: “PV”, and “HVM”. If `HVM.boot_policy` is the empty string, then paravirtual domain building and booting will be used; otherwise the VM will be loaded as an HVM domain, and booted using an emulated BIOS.

When paravirtual booting is in use, the `PV/bootloader` field indicates the bootloader to use. It may be “pygrub”, in which case the platform’s default installation of pygrub will be used, or a full path within the control domain to some other bootloader. The other fields, `PV/kernel`, `PV/ramdisk`, `PV/args` and `PV/bootloader_args` will be passed to the bootloader unmodified, and interpretation of those fields is then specific to the bootloader itself, including the possibility that the bootloader will ignore some or all of those given values. Finally the paths of all bootable disks are added to the bootloader commandline (a disk is bootable if its VBD has the bootable flag set). There may be zero, one or many bootable disks; the bootloader decides which disk (if any) to boot from.

If the bootloader is pygrub, then the `menu.lst` is parsed if present in the guest’s filesystem, otherwise the specified kernel and ramdisk are used, or an autodetected kernel is used if nothing is specified and autodetection is possible. `PV/args` is appended to the kernel command line, no matter which mechanism is used for finding the kernel.

If `PV/bootloader` is empty but `PV/kernel` is specified, then the kernel and ramdisk values will be treated as paths within the control domain. If both `PV/bootloader` and `PV/kernel` are empty, then the behaviour is as if `PV/bootloader` was specified as “pygrub”.

When using HVM booting, `HVM/boot_policy` and `HVM/boot_params` specify the boot handling.

Only one policy is currently defined: “BIOS order”. In this case, HVM/boot\_params should contain one key-value pair “order” = “N” where N is the string that will be passed to QEMU.

## Chapter 2

# API Reference

### 2.1 Classes

The following classes are defined:

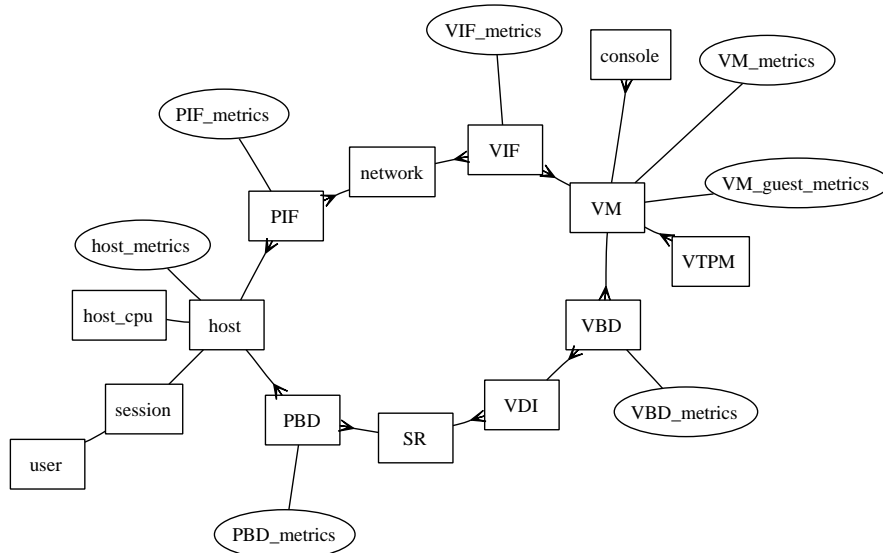
| Name                          | Description  |
|-------------------------------|--|
| <code>session</code>          | A session  |
| <code>task</code>             | A long-running asynchronous task   |
| <code>event</code>            | Asynchronous event registration and handling                                       |
| <code>VM</code>               | A virtual machine (or 'guest')   |
| <code>VM_metrics</code>       | The metrics associated with a VM   |
| <code>VM_guest_metrics</code> | The metrics reported by the guest (as opposed to inferred from outside)            |
| <code>host</code>             | A physical host  |
| <code>host_metrics</code>     | The metrics associated with a host   |
| <code>host_cpu</code>         | A physical CPU   |
| <code>network</code>          | A virtual network  |
| <code>VIF</code>              | A virtual network interface  |
| <code>VIF_metrics</code>      | The metrics associated with a virtual network device                               |
| <code>PIF</code>              | A physical network interface (note separate VLANs are represented as several PIFs) |
| <code>PIF_metrics</code>      | The metrics associated with a physical network interface                           |
| <code>SR</code>               | A storage repository   |
| <code>VDI</code>              | A virtual disk image   |
| <code>VBD</code>              | A virtual block device   |
| <code>VBD_metrics</code>      | The metrics associated with a virtual block device                                 |
| <code>PBD</code>              | The physical block devices through which hosts access SRs                          |
| <code>crashdump</code>        | A VM crashdump   |
| <code>VTPM</code>             | A virtual TPM device   |
| <code>console</code>          | A console  |
| <code>user</code>             | A user of the system   |
| <code>debug</code>            | A basic class for testing  |

### 2.2 Relationships Between Classes

Fields that are bound together are shown in the following table:

| <i>object.field</i> | <i>object.field</i> | <i>relationship</i> |
|---------------------|---------------------|---------------------|
| host.PBDs           | PBD.host            | many-to-one         |
| SR.PBDs             | PBD.SR              | many-to-one         |
| VDI.VBDs            | VBD.VDI             | many-to-one         |
| VDI.crash_dumps     | crashdump.VDI       | many-to-one         |
| VBD.VM              | VM.VBDs             | one-to-many         |
| crashdump.VM        | VM.crash_dumps      | one-to-many         |
| VIF.VM              | VM.VIFs             | one-to-many         |
| VIF.network         | network.VIFs        | one-to-many         |
| PIF.host            | host.PIFs           | one-to-many         |
| PIF.network         | network.PIFs        | one-to-many         |
| SR.VDIs             | VDI.SR              | many-to-one         |
| VTPM.VM             | VM.VTPMs            | one-to-many         |
| console.VM          | VM.consoles         | one-to-many         |
| host.resident_VMs   | VM.resident_on      | many-to-one         |
| host.host_CPUs      | host_cpu.host       | many-to-one         |

The following represents bound fields (as specified above) diagrammatically, using crows-foot notation to specify one-to-one, one-to-many or many-to-many relationships:



### 2.2.1 List of bound fields

## 2.3 Types

### 2.3.1 Primitives

The following primitive types are used to specify methods and fields in the API Reference:

| Type              | Description                                  |
|-------------------|--|
| String            | text strings                                 |
| Int               | 64-bit integers                              |
| Float             | IEEE double-precision floating-point numbers |
| Bool              | boolean                                      |
| DateTime          | date and timestamp                           |
| Ref (object name) | reference to an object of class name         |

### 2.3.2 Higher order types

The following type constructors are used:

| Type        | Description  |
|-------------|--|
| List (t)    | an arbitrary-length list of elements of type t       |
| Map (a → b) | a table mapping values of type a to values of type b |

### 2.3.3 Enumeration types

The following enumeration types are used:

| enum event_operation |                             |
|----------------------|-----------------------------|
| add                  | An object has been created  |
| del                  | An object has been deleted  |
| mod                  | An object has been modified |

| enum console_protocol |  |
|-----------------------|--|
| vt100                 | VT100 terminal                               |
| rfb                   | Remote FrameBuffer protocol (as used in VNC) |
| rdp                   | Remote Desktop Protocol                      |

| enum vdi_type |   |
|---------------|---|
| system        | a disk that may be replaced on upgrade      |
| user          | a disk that is always preserved on upgrade  |
| ephemeral     | a disk that may be reformatted on upgrade   |
| suspend       | a disk that stores a suspend image          |
| crashdump     | a disk that stores VM crashdump information |

| enum vm_power_state |                          |
|---------------------|--------------------------|
| Halted              | Halted                   |
| Paused              | Paused                   |
| Running             | Running                  |
| Suspended           | Suspended                |
| Unknown             | Some other unknown state |

| enum task_allowed_operations |        |
|------------------------------|--------|
| Cancel                       | Cancel |

| enum task_status_type |                                 |
|-----------------------|---------------------------------|
| pending               | task is in progress             |
| success               | task was completed successfully |
| failure               | task has failed                 |
| cancelling            | task is being cancelled         |
| cancelled             | task has been cancelled         |

| enum on_normal_exit |                      |
|---------------------|----------------------|
| destroy             | destroy the VM state |
| restart             | restart the VM       |

| enum on_crash_behaviour |   |
|-------------------------|---|
| destroy                 | destroy the VM state                            |
| coredump_and_destroy    | record a coredump and then destroy the VM state |
| restart                 | restart the VM                                  |
| coredump_and_restart    | record a coredump and then restart the VM       |
| preserve                | leave the crashed VM as-is                      |
| rename_restart          | rename the crashed VM and start a new copy      |

| enum vbd_mode |                            |
|---------------|----------------------------|
| RO            | disk is mounted read-only  |
| RW            | disk is mounted read-write |

| enum vbd_type |                                  |
|---------------|----------------------------------|
| CD            | VBD will appear to guest as CD   |
| Disk          | VBD will appear to guest as disk |

## 2.4 Class: session

### 2.4.1 Fields for class: session

|                         |             |          |  |
|-------------------------|-------------|----------|--|
| Name                    | session     |          |  |
| Description             | A session.  |          |  |
| Quals                   | Field       | Type     | Description                                |
| <i>RO<sub>run</sub></i> | uuid        | string   | unique identifier/object reference         |
| <i>RO<sub>run</sub></i> | this_host   | host ref | Currently connected host                   |
| <i>RO<sub>run</sub></i> | this_user   | user ref | Currently connected user                   |
| <i>RO<sub>run</sub></i> | last_active | int      | Timestamp for last time session was active |

### 2.4.2 RPCs associated with class: session

#### RPC name: login\_with\_password

**Overview:** Attempt to authenticate the user, returning a session\_id if successful.

**Signature:**

```
(session ref) login_with_password (string uname, string pwd)
```

**Arguments:**

| type   | name  | description         |
|--------|-------|---------------------|
| string | uname | Username for login. |
| string | pwd   | Password for login. |

**Return Type:** session ref

ID of newly created session

#### RPC name: logout

**Overview:** Log out of a session.

**Signature:**

```
void logout (session_id s)
```

**Return Type:** void

#### RPC name: get\_uuid

**Overview:** Get the uuid field of the given session.

**Signature:**

```
string get_uuid (session_id s, session ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| session ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `get_this_host`

**Overview:** Get the `this_host` field of the given session.

**Signature:**

```
(host ref) get_this_host (session_id s, session ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| session ref | self | reference to the object |

**Return Type:** `host ref`

value of the field

**RPC name:** `get_this_user`

**Overview:** Get the `this_user` field of the given session.

**Signature:**

```
(user ref) get_this_user (session_id s, session ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| session ref | self | reference to the object |

**Return Type:** `user ref`

value of the field

**RPC name:** `get_last_active`

**Overview:** Get the `last_active` field of the given session.

**Signature:**

```
int get_last_active (session_id s, session ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| session ref | self | reference to the object |

**Return Type:** `int`

value of the field



**RPC name:** `get_by_uuid`**Overview:** Get a reference to the session instance with the specified UUID.**Signature:**`(session ref) get_by_uuid (session_id s, string uuid)`**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** `session ref`

reference to the object

**RPC name:** `get_record`**Overview:** Get a record containing the current state of the given session.**Signature:**`(session record) get_record (session_id s, session ref self)`**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| session ref | self | reference to the object |

**Return Type:** `session record`

all fields from the object

## 2.5 Class: task

### 2.5.1 Fields for class: task

| Name                    | task                                     |                               |  |
|-------------------------|--|-------------------------------|--|
| Description             | <i>A long-running asynchronous task.</i> |                               |  |
| Quals                   | Field                                    | Type                          | Description  |
| <i>RO<sub>run</sub></i> | <code>uuid</code>                        | string                        | unique identifier/object reference   |
| <i>RO<sub>run</sub></i> | <code>name/label</code>                  | string                        | a human-readable name  |
| <i>RO<sub>run</sub></i> | <code>name/description</code>            | string                        | a notes field containing human-readable description  |
| <i>RO<sub>run</sub></i> | <code>status</code>                      | task_status_type              | current status of the task   |
| <i>RO<sub>run</sub></i> | <code>session</code>                     | session ref                   | the session that created the task  |
| <i>RO<sub>run</sub></i> | <code>progress</code>                    | int                           | if the task is still pending, this field contains the estimated percentage complete (0-100). If task has completed (successfully or unsuccessfully) this should be 100.          |
| <i>RO<sub>run</sub></i> | <code>type</code>                        | string                        | if the task has completed successfully, this field contains the type of the encoded result (i.e. name of the class whose reference is in the result field). Undefined otherwise. |
| <i>RO<sub>run</sub></i> | <code>result</code>                      | string                        | if the task has completed successfully, this field contains the result value (either Void or an object reference). Undefined otherwise.  |
| <i>RO<sub>run</sub></i> | <code>error_info</code>                  | string Set                    | if the task has failed, this field contains the set of associated error strings. Undefined otherwise.  |
| <i>RO<sub>run</sub></i> | <code>allowed_operations</code>          | (task_allowed_operations) Set | Operations allowed on this task  |

### 2.5.2 RPCs associated with class: task

#### RPC name: cancel

**Overview:** Cancel this task. If `task.allowed_operations` does not contain `Cancel`, then this will fail with `OPERATION_NOT_ALLOWED`. The task will show the status 'cancelling', and you should continue to check its status until it shows 'cancelled'. There is no guarantee as to the time within which this task will be cancelled.

#### Signature:

```
void cancel (session_id s, task ref task)
```

#### Arguments:

| type     | name | description |
|----------|------|-------------|
| task ref | task | The task    |

**Return Type:** void

**Possible Error Codes:** OPERATION\_NOT\_ALLOWED

**RPC name:** get\_all**Overview:** Return a list of all the tasks known to the system.**Signature:**`((task ref) Set) get_all (session_id s)`**Return Type:** (task ref) Set

references to all objects

**RPC name:** get\_uuid**Overview:** Get the uuid field of the given task.**Signature:**`string get_uuid (session_id s, task ref self)`**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| task ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** get\_name\_label**Overview:** Get the name/label field of the given task.**Signature:**`string get_name_label (session_id s, task ref self)`**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| task ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** get\_name\_description**Overview:** Get the name/description field of the given task.**Signature:**`string get_name_description (session_id s, task ref self)`**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| task ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `get_status`

**Overview:** Get the status field of the given task.

**Signature:**

```
(task_status_type) get_status (session_id s, task ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| task ref | self | reference to the object |

**Return Type:** `task_status_type`

value of the field

**RPC name:** `get_session`

**Overview:** Get the session field of the given task.

**Signature:**

```
(session ref) get_session (session_id s, task ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| task ref | self | reference to the object |

**Return Type:** `session ref`

value of the field

**RPC name:** `get_progress`

**Overview:** Get the progress field of the given task.

**Signature:**

```
int get_progress (session_id s, task ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| task ref | self | reference to the object |

**Return Type:** `int`

value of the field

**RPC name:** `get_type`**Overview:** Get the type field of the given task.**Signature:**

```
string get_type (session_id s, task ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| task ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `get_result`**Overview:** Get the result field of the given task.**Signature:**

```
string get_result (session_id s, task ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| task ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `get_error_info`**Overview:** Get the error\_info field of the given task.**Signature:**

```
((string Set) get_error_info (session_id s, task ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| task ref | self | reference to the object |

**Return Type:** string Set

value of the field

**RPC name:** `get_allowed_operations`**Overview:** Get the allowed\_operations field of the given task.**Signature:**

```
((task_allowed_operations) Set) get_allowed_operations (session_id s, task ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| task ref | self | reference to the object |

**Return Type:** (task\_allowed\_operations) Set  
value of the field

**RPC name:** `get_by_uuid`

**Overview:** Get a reference to the task instance with the specified UUID.

**Signature:**

```
(task ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** task ref  
reference to the object

**RPC name:** `get_record`

**Overview:** Get a record containing the current state of the given task.

**Signature:**

```
(task record) get_record (session_id s, task ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| task ref | self | reference to the object |

**Return Type:** task record  
all fields from the object

**RPC name:** `get_by_name_label`

**Overview:** Get all the task instances with the given label.

**Signature:**

```
((task ref) Set) get_by_name_label (session_id s, string label)
```

**Arguments:**

| type   | name  | description               |
|--------|-------|---------------------------|
| string | label | label of object to return |

**Return Type:** (task ref) Set  
references to objects with match names

## 2.6 Class: event

### 2.6.1 Fields for class: event

|                         |  |                 |   |
|-------------------------|--|-----------------|---|
| Name                    | <b>event</b>   |                 |   |
| Description             | <i>Asynchronous event registration and handling.</i> |                 |   |
| Quals                   | Field  | Type            | Description   |
| <i>RO<sub>ins</sub></i> | id   | int             | An ID, monotonically increasing, and local to the current session |
| <i>RO<sub>ins</sub></i> | timestamp  | datetime        | The time at which the event occurred                              |
| <i>RO<sub>ins</sub></i> | class  | string          | The name of the class of the object that changed                  |
| <i>RO<sub>ins</sub></i> | operation  | event_operation | The operation that was performed                                  |
| <i>RO<sub>ins</sub></i> | ref  | string          | A reference to the object that changed                            |
| <i>RO<sub>ins</sub></i> | obj_uuid   | string          | The uuid of the object that changed                               |

### 2.6.2 RPCs associated with class: event

#### RPC name: register

**Overview:** Registers this session with the event system. Specifying the empty list will register for all classes.

#### Signature:

```
void register (session_id s, string Set classes)
```

#### Arguments:

| type       | name    | description                                   |
|------------|---------|---|
| string Set | classes | register for events for the indicated classes |

**Return Type:** void

#### RPC name: unregister

**Overview:** Unregisters this session with the event system.

#### Signature:

```
void unregister (session_id s, string Set classes)
```

#### Arguments:

| type       | name    | description  |
|------------|---------|--|
| string Set | classes | remove this session's registration for the indicated classes |

**Return Type:** void

#### RPC name: next

**Overview:** Blocking call which returns a (possibly empty) batch of events.

#### Signature:

((event record) Set) next (session\_id s)

**Return Type:** (event record) Set  
the batch of events

**Possible Error Codes:** SESSION\_NOT\_REGISTERED



## 2.7 Class: VM

### 2.7.1 Fields for class: VM

| Name                    | VM                     |  |   |
|-------------------------|------------------------|--|---|
| Quals                   | Field                  | Type                                   | Description   |
|                         | Description            | <i>A virtual machine (or 'guest').</i> |   |
| <i>RO<sub>run</sub></i> | uuid                   | string                                 | unique identifier/object reference  |
| <i>RO<sub>run</sub></i> | power_state            | vm_power_state                         | Current power state of the machine  |
| <i>RW</i>               | name/label             | string                                 | a human-readable name   |
| <i>RW</i>               | name/description       | string                                 | a notes field containing human-readable description   |
| <i>RW</i>               | user_version           | int                                    | a user version number for this machine  |
| <i>RW</i>               | is_a_template          | bool                                   | true if this is a template. Template VMs can never be started, they are used only for cloning other VMs |
| <i>RW</i>               | auto_power_on          | bool                                   | true if this VM should be started automatically after host boot   |
| <i>RO<sub>run</sub></i> | suspend_VDI            | VDI ref                                | The VDI that a suspend image is stored on. (Only has meaning if VM is currently suspended)              |
| <i>RO<sub>run</sub></i> | resident_on            | host ref                               | the host the VM is currently resident on  |
| <i>RW</i>               | memory/static_max      | int                                    | Statically-set (i.e. absolute) maximum (bytes)  |
| <i>RW</i>               | memory/dynamic_max     | int                                    | Dynamic maximum (bytes)   |
| <i>RW</i>               | memory/dynamic_min     | int                                    | Dynamic minimum (bytes)   |
| <i>RW</i>               | memory/static_min      | int                                    | Statically-set (i.e. absolute) minimum (bytes)  |
| <i>RW</i>               | VCPUs/params           | (string → string) Map                  | configuration parameters for the selected VCPU policy   |
| <i>RW</i>               | VCPUs/max              | int                                    | Max number of VCPUs   |
| <i>RW</i>               | VCPUs/at_startup       | int                                    | Boot number of VCPUs  |
| <i>RW</i>               | actions/after_shutdown | on_normal_exit                         | action to take after the guest has shutdown itself  |
| <i>RW</i>               | actions/after_reboot   | on_normal_exit                         | action to take after the guest has rebooted itself  |
| <i>RW</i>               | actions/after_crash    | on_crash_behaviour                     | action to take if the guest crashes   |
| <i>RO<sub>run</sub></i> | consoles               | (console ref) Set                      | virtual console devices   |
| <i>RO<sub>run</sub></i> | VIFs                   | (VIF ref) Set                          | virtual network interfaces  |
| <i>RO<sub>run</sub></i> | VBDs                   | (VBD ref) Set                          | virtual block devices   |
| <i>RO<sub>run</sub></i> | crash_dumps            | (crashdump ref) Set                    | crash dumps associated with this VM   |
| <i>RO<sub>run</sub></i> | VTPMs                  | (VTPM ref) Set                         | virtual TPMs  |
| <i>RW</i>               | PV/bootloader          | string                                 | name of or path to bootloader   |
| <i>RW</i>               | PV/kernel              | string                                 | path to the kernel  |
| <i>RW</i>               | PV/ramdisk             | string                                 | path to the initrd  |
| <i>RW</i>               | PV/args                | string                                 | kernel command-line arguments   |
| <i>RW</i>               | PV/bootloader_args     | string                                 | miscellaneous arguments for the bootloader  |
| <i>RW</i>               | HVM/boot_policy        | string                                 | HVM boot policy   |
| <i>RW</i>               | HVM/boot_params        | (string → string) Map                  | HVM boot params   |
| <i>RW</i>               | platform               | (string → string) Map                  | platform-specific configuration   |

|                         |                                |                       |  |
|-------------------------|--------------------------------|-----------------------|--|
| <i>RW</i>               | <code>PCI_bus</code>           | string                | PCI bus path for pass-through devices                          |
| <i>RW</i>               | <code>other_config</code>      | (string → string) Map | additional configuration                                       |
| <i>RO<sub>run</sub></i> | <code>domid</code>             | int                   | domain ID (if available, -1 otherwise)                         |
| <i>RO<sub>run</sub></i> | <code>is_control_domain</code> | bool                  | true if this is a control domain (domain 0 or a driver domain) |
| <i>RO<sub>run</sub></i> | <code>metrics</code>           | VM_metrics ref        | metrics associated with this VM                                |
| <i>RO<sub>run</sub></i> | <code>guest_metrics</code>     | VM_guest_metrics ref  | metrics associated with the running guest                      |

## 2.7.2 RPCs associated with class: VM

### RPC name: clone

**Overview:** Clones the specified VM, making a new VM. Clone automatically exploits the capabilities of the underlying storage repository in which the VM's disk images are stored (e.g. Copy on Write). This function can only be called when the VM is in the Halted State.

**Signature:**

```
(VM ref) clone (session_id s, VM ref vm, string new_name)
```

**Arguments:**

| type   | name     | description               |
|--------|----------|---------------------------|
| VM ref | vm       | The VM to be cloned       |
| string | new_name | The name of the cloned VM |

**Return Type:** VM ref

The ID of the newly created VM.

**Possible Error Codes:** VM\_BAD\_POWER\_STATE

### RPC name: start

**Overview:** Start the specified VM. This function can only be called with the VM is in the Halted State.

**Signature:**

```
void start (session_id s, VM ref vm, bool start_paused)
```

**Arguments:**

| type   | name         | description                                    |
|--------|--------------|--|
| VM ref | vm           | The VM to start                                |
| bool   | start_paused | Instantiate VM in paused state if set to true. |

**Return Type:** void

**Possible Error Codes:** VM\_BAD\_POWER\_STATE, VM\_HVM\_REQUIRED

### RPC name: pause

**Overview:** Pause the specified VM. This can only be called when the specified VM is in the Running state.

**Signature:**

```
void pause (session_id s, VM ref vm)
```

**Arguments:**

| type   | name | description     |
|--------|------|-----------------|
| VM ref | vm   | The VM to pause |

**Return Type:** void**Possible Error Codes:** VM\_BAD\_POWER\_STATE**RPC name:** unpause

**Overview:** Resume the specified VM. This can only be called when the specified VM is in the Paused state.

**Signature:**

```
void unpause (session_id s, VM ref vm)
```

**Arguments:**

| type   | name | description       |
|--------|------|-------------------|
| VM ref | vm   | The VM to unpause |

**Return Type:** void**Possible Error Codes:** VM\_BAD\_POWER\_STATE**RPC name:** clean\_shutdown

**Overview:** Attempt to cleanly shutdown the specified VM. (Note: this may not be supported—e.g. if a guest agent is not installed).

Once shutdown has been completed perform poweroff action specified in guest configuration.

This can only be called when the specified VM is in the Running state.

**Signature:**

```
void clean_shutdown (session_id s, VM ref vm)
```

**Arguments:**

| type   | name | description        |
|--------|------|--------------------|
| VM ref | vm   | The VM to shutdown |

**Return Type:** void**Possible Error Codes:** VM\_BAD\_POWER\_STATE**RPC name:** clean\_reboot

**Overview:** Attempt to cleanly shutdown the specified VM (Note: this may not be supported—e.g. if a guest agent is not installed).

Once shutdown has been completed perform reboot action specified in guest configuration.

This can only be called when the specified VM is in the Running state.

**Signature:**

```
void clean_reboot (session_id s, VM ref vm)
```

**Arguments:**

| type   | name | description        |
|--------|------|--------------------|
| VM ref | vm   | The VM to shutdown |

**Return Type:** void

**Possible Error Codes:** VM\_BAD\_POWER\_STATE

**RPC name:** hard\_shutdown

**Overview:** Stop executing the specified VM without attempting a clean shutdown. Then perform poweroff action specified in VM configuration.

**Signature:**

```
void hard_shutdown (session_id s, VM ref vm)
```

**Arguments:**

| type   | name | description       |
|--------|------|-------------------|
| VM ref | vm   | The VM to destroy |

**Return Type:** void

**Possible Error Codes:** VM\_BAD\_POWER\_STATE

**RPC name:** hard\_reboot

**Overview:** Stop executing the specified VM without attempting a clean shutdown. Then perform reboot action specified in VM configuration.

**Signature:**

```
void hard_reboot (session_id s, VM ref vm)
```

**Arguments:**

| type   | name | description      |
|--------|------|------------------|
| VM ref | vm   | The VM to reboot |

**Return Type:** void

**RPC name:** suspend

**Overview:** Suspend the specified VM to disk. This can only be called when the specified VM is in the Running state.

**Signature:**

```
void suspend (session_id s, VM ref vm)
```

**Arguments:**

| type   | name | description       |
|--------|------|-------------------|
| VM ref | vm   | The VM to suspend |

**Return Type:** void

**Possible Error Codes:** VM\_BAD\_POWER\_STATE

**RPC name:** resume

**Overview:** Awaken the specified VM and resume it. This can only be called when the specified VM is in the Suspended state.

**Signature:**

```
void resume (session_id s, VM ref vm, bool start_paused)
```

**Arguments:**

| type   | name         | description                               |
|--------|--------------|---|
| VM ref | vm           | The VM to resume                          |
| bool   | start_paused | Resume VM in paused state if set to true. |

**Return Type:** void

**Possible Error Codes:** VM\_BAD\_POWER\_STATE

**RPC name:** set\_VCPUs\_number\_live

**Overview:** Set this VM's VCPUs/at\_startup value, and set the same value on the VM, if running.

**Signature:**

```
void set_VCPUs_number_live (session_id s, VM ref self, int nvcpu)
```

**Arguments:**

| type   | name  | description         |
|--------|-------|---------------------|
| VM ref | self  | The VM              |
| int    | nvcpu | The number of VCPUs |

**Return Type:** void

**RPC name:** add\_to\_VCPUs\_params\_live

**Overview:** Add the given key-value pair to VM.VCPUs\_params, and apply that value on the running VM.

**Signature:**

```
void add_to_VCPUs_params_live (session_id s, VM ref self, string key, string value)
```

**Arguments:**

| type   | name  | description |
|--------|-------|-------------|
| VM ref | self  | The VM      |
| string | key   | The key     |
| string | value | The value   |

**Return Type:** void

**RPC name:** `set_memory_dynamic_max_live`**Overview:** Set `memory_dynamic_max` in database and on running VM.**Signature:**

```
void set_memory_dynamic_max_live (session_id s, VM ref self, int max)
```

**Arguments:**

| type   | name | description                               |
|--------|------|---|
| VM ref | self | The VM                                    |
| int    | max  | The <code>memory_dynamic_max</code> value |

**Return Type:** void**RPC name:** `set_memory_dynamic_min_live`**Overview:** Set `memory_dynamic_min` in database and on running VM.**Signature:**

```
void set_memory_dynamic_min_live (session_id s, VM ref self, int min)
```

**Arguments:**

| type   | name | description                               |
|--------|------|---|
| VM ref | self | The VM                                    |
| int    | min  | The <code>memory_dynamic_min</code> value |

**Return Type:** void**RPC name:** `send_sysrq`**Overview:** Send the given key as a `sysrq` to this VM. The key is specified as a single character (a String of length 1). This can only be called when the specified VM is in the Running state.**Signature:**

```
void send_sysrq (session_id s, VM ref vm, string key)
```

**Arguments:**

| type   | name | description     |
|--------|------|-----------------|
| VM ref | vm   | The VM          |
| string | key  | The key to send |

**Return Type:** void**Possible Error Codes:** `VM_BAD_POWER_STATE`**RPC name:** `send_trigger`**Overview:** Send the named trigger to this VM. This can only be called when the specified VM is in the Running state.**Signature:**

```
void send_trigger (session_id s, VM ref vm, string trigger)
```

**Arguments:**

| type   | name    | description         |
|--------|---------|---------------------|
| VM ref | vm      | The VM              |
| string | trigger | The trigger to send |

**Return Type:** void**Possible Error Codes:** VM\_BAD\_POWER\_STATE**RPC name:** migrate

**Overview:** Migrate the VM to another host. This can only be called when the specified VM is in the Running state.

**Signature:**

```
void migrate (session_id s, VM ref vm, string dest, bool live, (string -> string) Map options)
```

**Arguments:**

| type                  | name    | description          |
|-----------------------|---------|----------------------|
| VM ref                | vm      | The VM               |
| string                | dest    | The destination host |
| bool                  | live    | Live migration       |
| (string → string) Map | options | Other parameters     |

**Return Type:** void**Possible Error Codes:** VM\_BAD\_POWER\_STATE**RPC name:** get\_all

**Overview:** Return a list of all the VMs known to the system.

**Signature:**

```
((VM ref) Set) get_all (session_id s)
```

**Return Type:** (VM ref) Set

A list of all the IDs of all the VMs

**RPC name:** get\_uuid

**Overview:** Get the uuid field of the given VM.

**Signature:**

```
string get_uuid (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** get\_power\_state

**Overview:** Get the power.state field of the given VM.

**Signature:**

```
(vm_power_state) get_power_state (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** vm\_power\_state  
value of the field

**RPC name:** get\_name\_label

**Overview:** Get the name/label field of the given VM.

**Signature:**

```
string get_name_label (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** set\_name\_label

**Overview:** Set the name/label field of the given VM.

**Signature:**

```
void set_name_label (session_id s, VM ref self, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | value | New value to set        |

**Return Type:** void



**RPC name:** `get_name_description`**Overview:** Get the name/description field of the given VM.**Signature:**

```
string get_name_description (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `set_name_description`**Overview:** Set the name/description field of the given VM.**Signature:**

```
void set_name_description (session_id s, VM ref self, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | value | New value to set        |

**Return Type:** void**RPC name:** `get_user_version`**Overview:** Get the user\_version field of the given VM.**Signature:**

```
int get_user_version (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** int

value of the field

**RPC name:** `set_user_version`**Overview:** Set the user\_version field of the given VM.**Signature:**

```
void set_user_version (session_id s, VM ref self, int value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| int    | value | New value to set        |

**Return Type:** void**RPC name:** get\_is\_a\_template**Overview:** Get the is\_a\_template field of the given VM.**Signature:**

```
bool get_is_a_template (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** bool

value of the field

**RPC name:** set\_is\_a\_template**Overview:** Set the is\_a\_template field of the given VM.**Signature:**

```
void set_is_a_template (session_id s, VM ref self, bool value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| bool   | value | New value to set        |

**Return Type:** void**RPC name:** get\_auto\_power\_on**Overview:** Get the auto\_power\_on field of the given VM.**Signature:**

```
bool get_auto_power_on (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** bool

value of the field

**RPC name:** set\_auto\_power\_on**Overview:** Set the auto\_power\_on field of the given VM.**Signature:**

```
void set_auto_power_on (session_id s, VM ref self, bool value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| bool   | value | New value to set        |

**Return Type:** void**RPC name:** get\_suspend\_VDI**Overview:** Get the suspend\_VDI field of the given VM.**Signature:**

```
(VDI ref) get_suspend_VDI (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** VDI ref

value of the field

**RPC name:** get\_resident\_on**Overview:** Get the resident\_on field of the given VM.**Signature:**

```
(host ref) get_resident_on (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** host ref

value of the field

**RPC name:** get\_memory\_static\_max**Overview:** Get the memory/static\_max field of the given VM.**Signature:**

```
int get_memory_static_max (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** int  
value of the field

**RPC name:** set\_memory\_static\_max

**Overview:** Set the memory/static\_max field of the given VM.

**Signature:**

```
void set_memory_static_max (session_id s, VM ref self, int value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| int    | value | New value to set        |

**Return Type:** void

**RPC name:** get\_memory\_dynamic\_max

**Overview:** Get the memory/dynamic\_max field of the given VM.

**Signature:**

```
int get_memory_dynamic_max (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** int  
value of the field

**RPC name:** set\_memory\_dynamic\_max

**Overview:** Set the memory/dynamic\_max field of the given VM.

**Signature:**

```
void set_memory_dynamic_max (session_id s, VM ref self, int value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| int    | value | New value to set        |

**Return Type:** void

**RPC name:** `get_memory_dynamic_min`**Overview:** Get the `memory/dynamic_min` field of the given VM.**Signature:**

```
int get_memory_dynamic_min (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** `int`

value of the field

**RPC name:** `set_memory_dynamic_min`**Overview:** Set the `memory/dynamic_min` field of the given VM.**Signature:**

```
void set_memory_dynamic_min (session_id s, VM ref self, int value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| int    | value | New value to set        |

**Return Type:** `void`**RPC name:** `get_memory_static_min`**Overview:** Get the `memory/static_min` field of the given VM.**Signature:**

```
int get_memory_static_min (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** `int`

value of the field

**RPC name:** `set_memory_static_min`**Overview:** Set the `memory/static_min` field of the given VM.**Signature:**

```
void set_memory_static_min (session_id s, VM ref self, int value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| int    | value | New value to set        |

**Return Type:** void

**RPC name:** get\_VCPUs\_params

**Overview:** Get the VCPUs/params field of the given VM.

**Signature:**

```
((string -> string) Map) get_VCPUs_params (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name:** set\_VCPUs\_params

**Overview:** Set the VCPUs/params field of the given VM.

**Signature:**

```
void set_VCPUs_params (session_id s, VM ref self, (string -> string) Map value)
```

**Arguments:**

| type                  | name  | description             |
|-----------------------|-------|-------------------------|
| VM ref                | self  | reference to the object |
| (string → string) Map | value | New value to set        |

**Return Type:** void

**RPC name:** add\_to\_VCPUs\_params

**Overview:** Add the given key-value pair to the VCPUs/params field of the given VM.

**Signature:**

```
void add_to_VCPUs_params (session_id s, VM ref self, string key, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | key   | Key to add              |
| string | value | Value to add            |

**Return Type:** void

**RPC name: remove\_from\_VCPUs\_params**

**Overview:** Remove the given key and its corresponding value from the VCPUs/params field of the given VM. If the key is not in that Map, then do nothing.

**Signature:**

```
void remove_from_VCPUs_params (session_id s, VM ref self, string key)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |
| string | key  | Key to remove           |

**Return Type:** void

**RPC name: get\_VCPUs\_max**

**Overview:** Get the VCPUs/max field of the given VM.

**Signature:**

```
int get_VCPUs_max (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** int

value of the field

**RPC name: set\_VCPUs\_max**

**Overview:** Set the VCPUs/max field of the given VM.

**Signature:**

```
void set_VCPUs_max (session_id s, VM ref self, int value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| int    | value | New value to set        |

**Return Type:** void

**RPC name: get\_VCPUs\_at\_startup**

**Overview:** Get the VCPUs/at\_startup field of the given VM.

**Signature:**

```
int get_VCPUs_at_startup (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** int  
value of the field

**RPC name:** set\_VCPUs\_at\_startup

**Overview:** Set the VCPUs/at\_startup field of the given VM.

**Signature:**

```
void set_VCPUs_at_startup (session_id s, VM ref self, int value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| int    | value | New value to set        |

**Return Type:** void

**RPC name:** get\_actions\_after\_shutdown

**Overview:** Get the actions/after\_shutdown field of the given VM.

**Signature:**

```
(on_normal_exit) get_actions_after_shutdown (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** on\_normal\_exit  
value of the field

**RPC name:** set\_actions\_after\_shutdown

**Overview:** Set the actions/after\_shutdown field of the given VM.

**Signature:**

```
void set_actions_after_shutdown (session_id s, VM ref self, on_normal_exit value)
```

**Arguments:**

| type           | name  | description             |
|----------------|-------|-------------------------|
| VM ref         | self  | reference to the object |
| on_normal_exit | value | New value to set        |

**Return Type:** void



**RPC name:** `get_actions_after_reboot`**Overview:** Get the `actions/after_reboot` field of the given VM.**Signature:**

```
(on_normal_exit) get_actions_after_reboot (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** `on_normal_exit`

value of the field

**RPC name:** `set_actions_after_reboot`**Overview:** Set the `actions/after_reboot` field of the given VM.**Signature:**

```
void set_actions_after_reboot (session_id s, VM ref self, on_normal_exit value)
```

**Arguments:**

| type                        | name  | description             |
|-----------------------------|-------|-------------------------|
| VM ref                      | self  | reference to the object |
| <code>on_normal_exit</code> | value | New value to set        |

**Return Type:** `void`**RPC name:** `get_actions_after_crash`**Overview:** Get the `actions/after_crash` field of the given VM.**Signature:**

```
(on_crash_behaviour) get_actions_after_crash (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** `on_crash_behaviour`

value of the field

**RPC name:** `set_actions_after_crash`**Overview:** Set the `actions/after_crash` field of the given VM.**Signature:**

```
void set_actions_after_crash (session_id s, VM ref self, on_crash_behaviour value)
```

**Arguments:**

| type               | name  | description             |
|--------------------|-------|-------------------------|
| VM ref             | self  | reference to the object |
| on_crash_behaviour | value | New value to set        |

**Return Type:** void**RPC name:** get\_consoles**Overview:** Get the consoles field of the given VM.**Signature:**

```
((console ref) Set) get_consoles (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** (console ref) Set  
value of the field**RPC name:** get\_VIFs**Overview:** Get the VIFs field of the given VM.**Signature:**

```
((VIF ref) Set) get_VIFs (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** (VIF ref) Set  
value of the field**RPC name:** get\_VBDs**Overview:** Get the VBDs field of the given VM.**Signature:**

```
((VBD ref) Set) get_VBDs (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** (VBD ref) Set  
value of the field

**RPC name:** `get_crash_dumps`**Overview:** Get the `crash_dumps` field of the given VM.**Signature:**

```
((crashdump ref) Set) get_crash_dumps (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** (crashdump ref) Set

value of the field

**RPC name:** `get_VTPMs`**Overview:** Get the `VTPMs` field of the given VM.**Signature:**

```
((VTPM ref) Set) get_VTPMs (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** (VTPM ref) Set

value of the field

**RPC name:** `get_PV_bootloader`**Overview:** Get the `PV/bootloader` field of the given VM.**Signature:**

```
string get_PV_bootloader (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `set_PV_bootloader`**Overview:** Set the `PV/bootloader` field of the given VM.**Signature:**

```
void set_PV_bootloader (session_id s, VM ref self, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | value | New value to set        |

**Return Type:** void**RPC name:** get\_PV\_kernel**Overview:** Get the PV/kernel field of the given VM.**Signature:**

```
string get_PV_kernel (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** set\_PV\_kernel**Overview:** Set the PV/kernel field of the given VM.**Signature:**

```
void set_PV_kernel (session_id s, VM ref self, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | value | New value to set        |

**Return Type:** void**RPC name:** get\_PV\_ramdisk**Overview:** Get the PV/ramdisk field of the given VM.**Signature:**

```
string get_PV_ramdisk (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name: set\_PV\_ramdisk****Overview:** Set the PV/ramdisk field of the given VM.**Signature:**

```
void set_PV_ramdisk (session_id s, VM ref self, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | value | New value to set        |

**Return Type:** void**RPC name: get\_PV\_args****Overview:** Get the PV/args field of the given VM.**Signature:**

```
string get_PV_args (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name: set\_PV\_args****Overview:** Set the PV/args field of the given VM.**Signature:**

```
void set_PV_args (session_id s, VM ref self, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | value | New value to set        |

**Return Type:** void**RPC name: get\_PV\_bootloader\_args****Overview:** Get the PV/bootloader\_args field of the given VM.**Signature:**

```
string get_PV_bootloader_args (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** set\_PV\_bootloader\_args

**Overview:** Set the PV/bootloader\_args field of the given VM.

**Signature:**

```
void set_PV_bootloader_args (session_id s, VM ref self, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | value | New value to set        |

**Return Type:** void

**RPC name:** get\_HVM\_boot\_policy

**Overview:** Get the HVM/boot\_policy field of the given VM.

**Signature:**

```
string get_HVM_boot_policy (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** set\_HVM\_boot\_policy

**Overview:** Set the HVM/boot\_policy field of the given VM.

**Signature:**

```
void set_HVM_boot_policy (session_id s, VM ref self, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | value | New value to set        |

**Return Type:** void

**RPC name: get\_HVM\_boot\_params****Overview:** Get the HVM/boot\_params field of the given VM.**Signature:**

```
((string -> string) Map) get_HVM_boot_params (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** (string → string) Map

value of the field

**RPC name: set\_HVM\_boot\_params****Overview:** Set the HVM/boot\_params field of the given VM.**Signature:**

```
void set_HVM_boot_params (session_id s, VM ref self, (string -> string) Map value)
```

**Arguments:**

| type                  | name  | description             |
|-----------------------|-------|-------------------------|
| VM ref                | self  | reference to the object |
| (string → string) Map | value | New value to set        |

**Return Type:** void**RPC name: add\_to\_HVM\_boot\_params****Overview:** Add the given key-value pair to the HVM/boot\_params field of the given VM.**Signature:**

```
void add_to_HVM_boot_params (session_id s, VM ref self, string key, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | key   | Key to add              |
| string | value | Value to add            |

**Return Type:** void**RPC name: remove\_from\_HVM\_boot\_params****Overview:** Remove the given key and its corresponding value from the HVM/boot\_params field of the given VM. If the key is not in that Map, then do nothing.**Signature:**

```
void remove_from_HVM_boot_params (session_id s, VM ref self, string key)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |
| string | key  | Key to remove           |

**Return Type:** void

**RPC name:** get\_platform

**Overview:** Get the platform field of the given VM.

**Signature:**

```
((string -> string) Map) get_platform (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name:** set\_platform

**Overview:** Set the platform field of the given VM.

**Signature:**

```
void set_platform (session_id s, VM ref self, (string -> string) Map value)
```

**Arguments:**

| type                  | name  | description             |
|-----------------------|-------|-------------------------|
| VM ref                | self  | reference to the object |
| (string → string) Map | value | New value to set        |

**Return Type:** void

**RPC name:** add\_to\_platform

**Overview:** Add the given key-value pair to the platform field of the given VM.

**Signature:**

```
void add_to_platform (session_id s, VM ref self, string key, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | key   | Key to add              |
| string | value | Value to add            |

**Return Type:** void



**RPC name: remove\_from\_platform**

**Overview:** Remove the given key and its corresponding value from the platform field of the given VM. If the key is not in that Map, then do nothing.

**Signature:**

```
void remove_from_platform (session_id s, VM ref self, string key)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |
| string | key  | Key to remove           |

**Return Type:** void

**RPC name: get\_PCI\_bus**

**Overview:** Get the PCI\_bus field of the given VM.

**Signature:**

```
string get_PCI_bus (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name: set\_PCI\_bus**

**Overview:** Set the PCI\_bus field of the given VM.

**Signature:**

```
void set_PCI_bus (session_id s, VM ref self, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | value | New value to set        |

**Return Type:** void

**RPC name: get\_other\_config**

**Overview:** Get the other\_config field of the given VM.

**Signature:**

```
((string -> string) Map) get_other_config (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name: set\_other\_config**

**Overview:** Set the other\_config field of the given VM.

**Signature:**

```
void set_other_config (session_id s, VM ref self, (string -> string) Map value)
```

**Arguments:**

| type                  | name  | description             |
|-----------------------|-------|-------------------------|
| VM ref                | self  | reference to the object |
| (string → string) Map | value | New value to set        |

**Return Type:** void

**RPC name: add\_to\_other\_config**

**Overview:** Add the given key-value pair to the other\_config field of the given VM.

**Signature:**

```
void add_to_other_config (session_id s, VM ref self, string key, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| VM ref | self  | reference to the object |
| string | key   | Key to add              |
| string | value | Value to add            |

**Return Type:** void

**RPC name: remove\_from\_other\_config**

**Overview:** Remove the given key and its corresponding value from the other\_config field of the given VM. If the key is not in that Map, then do nothing.

**Signature:**

```
void remove_from_other_config (session_id s, VM ref self, string key)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |
| string | key  | Key to remove           |

**Return Type:** void

**RPC name:** get\_domid

**Overview:** Get the domid field of the given VM.

**Signature:**

```
int get_domid (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** int

value of the field

**RPC name:** get\_is\_control\_domain

**Overview:** Get the is\_control\_domain field of the given VM.

**Signature:**

```
bool get_is_control_domain (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** bool

value of the field

**RPC name:** get\_metrics

**Overview:** Get the metrics field of the given VM.

**Signature:**

```
(VM_metrics ref) get_metrics (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** VM\_metrics ref

value of the field

**RPC name:** `get_guest_metrics`**Overview:** Get the `guest_metrics` field of the given VM.**Signature:**

```
(VM_guest_metrics ref) get_guest_metrics (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** `VM_guest_metrics ref`

value of the field

**RPC name:** `create`**Overview:** Create a new VM instance, and return its handle.**Signature:**

```
(VM ref) create (session_id s, VM record args)
```

**Arguments:**

| type      | name | description               |
|-----------|------|---------------------------|
| VM record | args | All constructor arguments |

**Return Type:** `VM ref`

reference to the newly created object

**RPC name:** `destroy`**Overview:** Destroy the specified VM. The VM is completely removed from the system. This function can only be called when the VM is in the Halted State.**Signature:**

```
void destroy (session_id s, VM ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** `void`**RPC name:** `get_by_uuid`**Overview:** Get a reference to the VM instance with the specified UUID.**Signature:**

```
(VM ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** VM ref  
reference to the object

**RPC name:** get\_record

**Overview:** Get a record containing the current state of the given VM.

**Signature:**

(VM record) get\_record (session\_id s, VM ref self)

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| VM ref | self | reference to the object |

**Return Type:** VM record  
all fields from the object

**RPC name:** get\_by\_name\_label

**Overview:** Get all the VM instances with the given label.

**Signature:**

((VM ref) Set) get\_by\_name\_label (session\_id s, string label)

**Arguments:**

| type   | name  | description               |
|--------|-------|---------------------------|
| string | label | label of object to return |

**Return Type:** (VM ref) Set  
references to objects with match names

## 2.8 Class: VM\_metrics

### 2.8.1 Fields for class: VM\_metrics

| Name                    | VM_metrics                               |                        |   |
|-------------------------|--|------------------------|---|
| Description             | <i>The metrics associated with a VM.</i> |                        |   |
| Quals                   | Field                                    | Type                   | Description                                     |
| <i>RO<sub>run</sub></i> | uuid                                     | string                 | unique identifier/object reference              |
| <i>RO<sub>run</sub></i> | memory/actual                            | int                    | Guest's actual memory (bytes)                   |
| <i>RO<sub>run</sub></i> | VCPUs/number                             | int                    | Current number of VCPUs                         |
| <i>RO<sub>run</sub></i> | VCPUs/utilisation                        | (int → float) Map      | Utilisation for all of guest's current VCPUs    |
| <i>RO<sub>run</sub></i> | VCPUs/CPU                                | (int → int) Map        | VCPU to PCPU map                                |
| <i>RO<sub>run</sub></i> | VCPUs/params                             | (string → string) Map  | The live equivalent to VM.VCPUs_params          |
| <i>RO<sub>run</sub></i> | VCPUs/flags                              | (int → string Set) Map | CPU flags (blocked,online,running)              |
| <i>RO<sub>run</sub></i> | state                                    | string Set             | The state of the guest, eg blocked, dying etc   |
| <i>RO<sub>run</sub></i> | start_time                               | datetime               | Time at which this VM was last booted           |
| <i>RO<sub>run</sub></i> | last_updated                             | datetime               | Time at which this information was last updated |

### 2.8.2 RPCs associated with class: VM\_metrics

#### RPC name: get\_all

**Overview:** Return a list of all the VM\_metrics instances known to the system.

**Signature:**

```
((VM_metrics ref) Set) get_all (session_id s)
```

**Return Type:** (VM\_metrics ref) Set  
references to all objects

#### RPC name: get\_uuid

**Overview:** Get the uuid field of the given VM\_metrics.

**Signature:**

```
string get_uuid (session_id s, VM_metrics ref self)
```

**Arguments:**

| type           | name | description             |
|----------------|------|-------------------------|
| VM_metrics ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** `get_memory_actual`**Overview:** Get the memory/actual field of the given VM\_metrics.**Signature:**

```
int get_memory_actual (session_id s, VM_metrics ref self)
```

**Arguments:**

| type           | name | description             |
|----------------|------|-------------------------|
| VM_metrics ref | self | reference to the object |

**Return Type:** int

value of the field

**RPC name:** `get_VCPUs_number`**Overview:** Get the VCPUs/number field of the given VM\_metrics.**Signature:**

```
int get_VCPUs_number (session_id s, VM_metrics ref self)
```

**Arguments:**

| type           | name | description             |
|----------------|------|-------------------------|
| VM_metrics ref | self | reference to the object |

**Return Type:** int

value of the field

**RPC name:** `get_VCPUs_utilisation`**Overview:** Get the VCPUs/utilisation field of the given VM\_metrics.**Signature:**

```
((int -> float) Map) get_VCPUs_utilisation (session_id s, VM_metrics ref self)
```

**Arguments:**

| type           | name | description             |
|----------------|------|-------------------------|
| VM_metrics ref | self | reference to the object |

**Return Type:** (int → float) Map

value of the field

**RPC name:** `get_VCPUs_CPU`**Overview:** Get the VCPUs/CPU field of the given VM\_metrics.**Signature:**

```
((int -> int) Map) get_VCPUs_CPU (session_id s, VM_metrics ref self)
```

**Arguments:**

| type           | name | description             |
|----------------|------|-------------------------|
| VM_metrics ref | self | reference to the object |

**Return Type:** (int → int) Map  
value of the field

**RPC name:** get\_VCPUs\_params

**Overview:** Get the VCPUs/params field of the given VM\_metrics.

**Signature:**

```
((string -> string) Map) get_VCPUs_params (session_id s, VM_metrics ref self)
```

**Arguments:**

| type           | name | description             |
|----------------|------|-------------------------|
| VM_metrics ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name:** get\_VCPUs\_flags

**Overview:** Get the VCPUs/flags field of the given VM\_metrics.

**Signature:**

```
((int -> string Set) Map) get_VCPUs_flags (session_id s, VM_metrics ref self)
```

**Arguments:**

| type           | name | description             |
|----------------|------|-------------------------|
| VM_metrics ref | self | reference to the object |

**Return Type:** (int → string Set) Map  
value of the field

**RPC name:** get\_state

**Overview:** Get the state field of the given VM\_metrics.

**Signature:**

```
(string Set) get_state (session_id s, VM_metrics ref self)
```

**Arguments:**

| type           | name | description             |
|----------------|------|-------------------------|
| VM_metrics ref | self | reference to the object |

**Return Type:** string Set  
value of the field



**RPC name:** `get_start_time`

**Overview:** Get the `start_time` field of the given `VM_metrics`.

**Signature:**

```
datetime get_start_time (session_id s, VM_metrics ref self)
```

**Arguments:**

| type                        | name              | description             |
|-----------------------------|-------------------|-------------------------|
| <code>VM_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `datetime`

value of the field

**RPC name:** `get_last_updated`

**Overview:** Get the `last_updated` field of the given `VM_metrics`.

**Signature:**

```
datetime get_last_updated (session_id s, VM_metrics ref self)
```

**Arguments:**

| type                        | name              | description             |
|-----------------------------|-------------------|-------------------------|
| <code>VM_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `datetime`

value of the field

**RPC name:** `get_by_uuid`

**Overview:** Get a reference to the `VM_metrics` instance with the specified UUID.

**Signature:**

```
(VM_metrics ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type                | name              | description              |
|---------------------|-------------------|--------------------------|
| <code>string</code> | <code>uuid</code> | UUID of object to return |

**Return Type:** `VM_metrics ref`

reference to the object

**RPC name:** `get_record`

**Overview:** Get a record containing the current state of the given `VM_metrics`.

**Signature:**

```
(VM_metrics record) get_record (session_id s, VM_metrics ref self)
```

**Arguments:**

| <b>type</b>    | <b>name</b> | <b>description</b>      |
|----------------|-------------|-------------------------|
| VM_metrics ref | self        | reference to the object |

**Return Type:** VM\_metrics record  
all fields from the object

## 2.9 Class: VM\_guest\_metrics

### 2.9.1 Fields for class: VM\_guest\_metrics

| Name         | VM_guest_metrics  |                       |   |
|--------------|---|-----------------------|---|
| Description  | <i>The metrics reported by the guest (as opposed to inferred from outside).</i> |                       |   |
| Quals        | Field   | Type                  | Description                                     |
| <i>ROrun</i> | uuid  | string                | unique identifier/object reference              |
| <i>ROrun</i> | os_version  | (string → string) Map | version of the OS                               |
| <i>ROrun</i> | PV_drivers_version  | (string → string) Map | version of the PV drivers                       |
| <i>ROrun</i> | memory  | (string → string) Map | free/used/total memory                          |
| <i>ROrun</i> | disks   | (string → string) Map | disk configuration/free space                   |
| <i>ROrun</i> | networks  | (string → string) Map | network configuration                           |
| <i>ROrun</i> | other   | (string → string) Map | anything else                                   |
| <i>ROrun</i> | last_updated  | datetime              | Time at which this information was last updated |

### 2.9.2 RPCs associated with class: VM\_guest\_metrics

**RPC name:** get\_all

**Overview:** Return a list of all the VM\_guest\_metrics instances known to the system.

**Signature:**

```
((VM_guest_metrics ref) Set) get_all (session_id s)
```

**Return Type:** (VM\_guest\_metrics ref) Set  
references to all objects

**RPC name:** get\_uuid

**Overview:** Get the uuid field of the given VM\_guest\_metrics.

**Signature:**

```
string get_uuid (session_id s, VM_guest_metrics ref self)
```

**Arguments:**

| type                 | name | description             |
|----------------------|------|-------------------------|
| VM_guest_metrics ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** get\_os\_version

**Overview:** Get the os\_version field of the given VM\_guest\_metrics.

**Signature:**

```
((string -> string) Map) get_os_version (session_id s, VM_guest_metrics ref self)
```

**Arguments:**

| type                 | name | description             |
|----------------------|------|-------------------------|
| VM_guest_metrics ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name:** get\_PV\_drivers\_version

**Overview:** Get the PV\_drivers\_version field of the given VM\_guest\_metrics.

**Signature:**

```
((string -> string) Map) get_PV_drivers_version (session_id s, VM_guest_metrics ref self)
```

**Arguments:**

| type                 | name | description             |
|----------------------|------|-------------------------|
| VM_guest_metrics ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name:** get\_memory

**Overview:** Get the memory field of the given VM\_guest\_metrics.

**Signature:**

```
((string -> string) Map) get_memory (session_id s, VM_guest_metrics ref self)
```

**Arguments:**

| type                 | name | description             |
|----------------------|------|-------------------------|
| VM_guest_metrics ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name:** get\_disks

**Overview:** Get the disks field of the given VM\_guest\_metrics.

**Signature:**

```
((string -> string) Map) get_disks (session_id s, VM_guest_metrics ref self)
```

**Arguments:**

| type                 | name | description             |
|----------------------|------|-------------------------|
| VM_guest_metrics ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name:** `get_networks`**Overview:** Get the networks field of the given VM\_guest\_metrics.**Signature:**

```
((string -> string) Map) get_networks (session_id s, VM_guest_metrics ref self)
```

**Arguments:**

| type                 | name | description             |
|----------------------|------|-------------------------|
| VM_guest_metrics ref | self | reference to the object |

**Return Type:** (string → string) Map

value of the field

**RPC name:** `get_other`**Overview:** Get the other field of the given VM\_guest\_metrics.**Signature:**

```
((string -> string) Map) get_other (session_id s, VM_guest_metrics ref self)
```

**Arguments:**

| type                 | name | description             |
|----------------------|------|-------------------------|
| VM_guest_metrics ref | self | reference to the object |

**Return Type:** (string → string) Map

value of the field

**RPC name:** `get_last_updated`**Overview:** Get the last\_updated field of the given VM\_guest\_metrics.**Signature:**

```
datetime get_last_updated (session_id s, VM_guest_metrics ref self)
```

**Arguments:**

| type                 | name | description             |
|----------------------|------|-------------------------|
| VM_guest_metrics ref | self | reference to the object |

**Return Type:** datetime

value of the field

**RPC name:** `get_by_uuid`**Overview:** Get a reference to the VM\_guest\_metrics instance with the specified UUID.**Signature:**

```
(VM_guest_metrics ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** VM\_guest\_metrics ref  
reference to the object

**RPC name:** get\_record

**Overview:** Get a record containing the current state of the given VM\_guest\_metrics.

**Signature:**

(VM\_guest\_metrics record) get\_record (session\_id s, VM\_guest\_metrics ref self)

**Arguments:**

| type                 | name | description             |
|----------------------|------|-------------------------|
| VM_guest_metrics ref | self | reference to the object |

**Return Type:** VM\_guest\_metrics record  
all fields from the object

## 2.10 Class: host

### 2.10.1 Fields for class: host

| Name                    | host                              |                       |  |
|-------------------------|-----------------------------------|-----------------------|--|
| Description             | <i>A physical host.</i>           |                       |  |
| Quals                   | Field                             | Type                  | Description  |
| <i>RO<sub>run</sub></i> | uuid                              | string                | unique identifier/object reference   |
| <i>RW</i>               | name/label                        | string                | a human-readable name  |
| <i>RW</i>               | name/description                  | string                | a notes field containing human-readable description  |
| <i>RO<sub>run</sub></i> | API_version/major                 | int                   | major version number   |
| <i>RO<sub>run</sub></i> | API_version/minor                 | int                   | minor version number   |
| <i>RO<sub>run</sub></i> | API_version/vendor                | string                | identification of vendor   |
| <i>RO<sub>run</sub></i> | API_version/vendor_implementation | (string → string) Map | details of vendor implementation   |
| <i>RO<sub>run</sub></i> | enabled                           | bool                  | True if the host is currently enabled  |
| <i>RO<sub>run</sub></i> | software_version                  | (string → string) Map | version strings  |
| <i>RW</i>               | other_config                      | (string → string) Map | additional configuration   |
| <i>RO<sub>run</sub></i> | capabilities                      | string Set            | Xen capabilities   |
| <i>RO<sub>run</sub></i> | cpu_configuration                 | (string → string) Map | The CPU configuration on this host. May contain keys such as “nr_nodes”, “sockets_per_node”, “cores_per_socket”, or “threads_per_core” |
| <i>RO<sub>run</sub></i> | sched_policy                      | string                | Scheduler policy currently in force on this host   |
| <i>RO<sub>run</sub></i> | supported_bootloaders             | string Set            | a list of the bootloaders installed on the machine   |
| <i>RO<sub>run</sub></i> | resident_VMs                      | (VM ref) Set          | list of VMs currently resident on host   |
| <i>RW</i>               | logging                           | (string → string) Map | logging configuration  |
| <i>RO<sub>run</sub></i> | PIFs                              | (PIF ref) Set         | physical network interfaces  |
| <i>RW</i>               | suspend_image_sr                  | SR ref                | The SR in which VDIs for suspend images are created  |
| <i>RW</i>               | crash_dump_sr                     | SR ref                | The SR in which VDIs for crash dumps are created   |
| <i>RO<sub>run</sub></i> | PBDs                              | (PBD ref) Set         | physical blockdevices  |
| <i>RO<sub>run</sub></i> | host_CPUs                         | (host_cpu ref) Set    | The physical CPUs on this host   |
| <i>RO<sub>run</sub></i> | metrics                           | host_metrics ref      | metrics associated with this host  |

### 2.10.2 RPCs associated with class: host

#### RPC name: disable

**Overview:** Puts the host into a state in which no new VMs can be started. Currently active VMs on the host continue to execute.

#### Signature:

```
void disable (session_id s, host ref host)
```

#### Arguments:

| type     | name | description         |
|----------|------|---------------------|
| host ref | host | The Host to disable |

**Return Type:** void

**RPC name: enable**

**Overview:** Puts the host into a state in which new VMs can be started.

**Signature:**

```
void enable (session_id s, host ref host)
```

**Arguments:**

| type     | name | description        |
|----------|------|--------------------|
| host ref | host | The Host to enable |

**Return Type:** void

**RPC name: shutdown**

**Overview:** Shutdown the host. (This function can only be called if there are no currently running VMs on the host and it is disabled.).

**Signature:**

```
void shutdown (session_id s, host ref host)
```

**Arguments:**

| type     | name | description          |
|----------|------|----------------------|
| host ref | host | The Host to shutdown |

**Return Type:** void

**RPC name: reboot**

**Overview:** Reboot the host. (This function can only be called if there are no currently running VMs on the host and it is disabled.).

**Signature:**

```
void reboot (session_id s, host ref host)
```

**Arguments:**

| type     | name | description        |
|----------|------|--------------------|
| host ref | host | The Host to reboot |

**Return Type:** void

**RPC name: dmesg**

**Overview:** Get the host xen dmesg.

**Signature:**

```
string dmesg (session_id s, host ref host)
```



**Arguments:**

| type     | name | description       |
|----------|------|-------------------|
| host ref | host | The Host to query |

**Return Type:** string  
dmesg string

**RPC name:** dmesg\_clear

**Overview:** Get the host xen dmesg, and clear the buffer.

**Signature:**

```
string dmesg_clear (session_id s, host ref host)
```

**Arguments:**

| type     | name | description       |
|----------|------|-------------------|
| host ref | host | The Host to query |

**Return Type:** string  
dmesg string

**RPC name:** get\_log

**Overview:** Get the host's log file.

**Signature:**

```
string get_log (session_id s, host ref host)
```

**Arguments:**

| type     | name | description       |
|----------|------|-------------------|
| host ref | host | The Host to query |

**Return Type:** string  
The contents of the host's primary log file

**RPC name:** send\_debug\_keys

**Overview:** Inject the given string as debugging keys into Xen.

**Signature:**

```
void send_debug_keys (session_id s, host ref host, string keys)
```

**Arguments:**

| type     | name | description      |
|----------|------|------------------|
| host ref | host | The host         |
| string   | keys | The keys to send |

**Return Type:** void

**RPC name: list\_methods****Overview:** List all supported methods.**Signature:**`(string Set) list_methods (session_id s)`**Return Type:** string Set

The name of every supported method.

**RPC name: get\_all****Overview:** Return a list of all the hosts known to the system.**Signature:**`((host ref) Set) get_all (session_id s)`**Return Type:** (host ref) Set

A list of all the IDs of all the hosts

**RPC name: get\_uuid****Overview:** Get the uuid field of the given host.**Signature:**`string get_uuid (session_id s, host ref self)`**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name: get\_name\_label****Overview:** Get the name/label field of the given host.**Signature:**`string get_name_label (session_id s, host ref self)`**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name: set\_name\_label****Overview:** Set the name/label field of the given host.**Signature:**

```
void set_name_label (session_id s, host ref self, string value)
```

**Arguments:**

| type     | name  | description             |
|----------|-------|-------------------------|
| host ref | self  | reference to the object |
| string   | value | New value to set        |

**Return Type:** void**RPC name: get\_name\_description****Overview:** Get the name/description field of the given host.**Signature:**

```
string get_name_description (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name: set\_name\_description****Overview:** Set the name/description field of the given host.**Signature:**

```
void set_name_description (session_id s, host ref self, string value)
```

**Arguments:**

| type     | name  | description             |
|----------|-------|-------------------------|
| host ref | self  | reference to the object |
| string   | value | New value to set        |

**Return Type:** void**RPC name: get\_API\_version\_major****Overview:** Get the API\_version/major field of the given host.**Signature:**

```
int get_API_version_major (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** int  
value of the field

**RPC name:** get\_API\_version\_minor

**Overview:** Get the API\_version/minor field of the given host.

**Signature:**

```
int get_API_version_minor (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** int  
value of the field

**RPC name:** get\_API\_version\_vendor

**Overview:** Get the API\_version/vendor field of the given host.

**Signature:**

```
string get_API_version_vendor (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** get\_API\_version\_vendor\_implementation

**Overview:** Get the API\_version/vendor\_implementation field of the given host.

**Signature:**

```
((string -> string) Map) get_API_version_vendor_implementation (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name:** `get_enabled`**Overview:** Get the `enabled` field of the given host.**Signature:**

```
bool get_enabled (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** `bool`

value of the field

**RPC name:** `get_software_version`**Overview:** Get the `software_version` field of the given host.**Signature:**

```
((string -> string) Map) get_software_version (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** `(string → string) Map`

value of the field

**RPC name:** `get_other_config`**Overview:** Get the `other_config` field of the given host.**Signature:**

```
((string -> string) Map) get_other_config (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** `(string → string) Map`

value of the field

**RPC name:** `set_other_config`**Overview:** Set the `other_config` field of the given host.**Signature:**

```
void set_other_config (session_id s, host ref self, (string -> string) Map value)
```

**Arguments:**

| type                  | name  | description             |
|-----------------------|-------|-------------------------|
| host ref              | self  | reference to the object |
| (string → string) Map | value | New value to set        |

**Return Type:** void**RPC name:** add\_to\_other\_config**Overview:** Add the given key-value pair to the other\_config field of the given host.**Signature:**

```
void add_to_other_config (session_id s, host ref self, string key, string value)
```

**Arguments:**

| type     | name  | description             |
|----------|-------|-------------------------|
| host ref | self  | reference to the object |
| string   | key   | Key to add              |
| string   | value | Value to add            |

**Return Type:** void**RPC name:** remove\_from\_other\_config**Overview:** Remove the given key and its corresponding value from the other\_config field of the given host. If the key is not in that Map, then do nothing.**Signature:**

```
void remove_from_other_config (session_id s, host ref self, string key)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |
| string   | key  | Key to remove           |

**Return Type:** void**RPC name:** get\_capabilities**Overview:** Get the capabilities field of the given host.**Signature:**

```
(string Set) get_capabilities (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** string Set

value of the field

**RPC name:** `get_cpu_configuration`

**Overview:** Get the `cpu_configuration` field of the given host.

**Signature:**

```
((string -> string) Map) get_cpu_configuration (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** `(string → string) Map`  
value of the field

**RPC name:** `get_sched_policy`

**Overview:** Get the `sched_policy` field of the given host.

**Signature:**

```
string get_sched_policy (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** `string`  
value of the field

**RPC name:** `get_supported_bootloaders`

**Overview:** Get the `supported_bootloaders` field of the given host.

**Signature:**

```
(string Set) get_supported_bootloaders (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** `string Set`  
value of the field

**RPC name:** `get_resident_VMs`**Overview:** Get the `resident_VMs` field of the given host.**Signature:**

```
((VM ref) Set) get_resident_VMs (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** `(VM ref) Set`

value of the field

**RPC name:** `get_logging`**Overview:** Get the `logging` field of the given host.**Signature:**

```
((string -> string) Map) get_logging (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** `(string → string) Map`

value of the field

**RPC name:** `set_logging`**Overview:** Set the `logging` field of the given host.**Signature:**

```
void set_logging (session_id s, host ref self, (string -> string) Map value)
```

**Arguments:**

| type                               | name  | description             |
|------------------------------------|-------|-------------------------|
| host ref                           | self  | reference to the object |
| <code>(string → string) Map</code> | value | New value to set        |

**Return Type:** `void`**RPC name:** `add_to_logging`**Overview:** Add the given key-value pair to the `logging` field of the given host.**Signature:**

```
void add_to_logging (session_id s, host ref self, string key, string value)
```



**Arguments:**

| type     | name  | description             |
|----------|-------|-------------------------|
| host ref | self  | reference to the object |
| string   | key   | Key to add              |
| string   | value | Value to add            |

**Return Type:** void**RPC name:** `remove_from_logging`

**Overview:** Remove the given key and its corresponding value from the logging field of the given host. If the key is not in that Map, then do nothing.

**Signature:**

```
void remove_from_logging (session_id s, host ref self, string key)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |
| string   | key  | Key to remove           |

**Return Type:** void**RPC name:** `get_PIFs`

**Overview:** Get the PIFs field of the given host.

**Signature:**

```
((PIF ref) Set) get_PIFs (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** (PIF ref) Set  
value of the field

**RPC name:** `get_suspend_image_sr`

**Overview:** Get the `suspend_image_sr` field of the given host.

**Signature:**

```
(SR ref) get_suspend_image_sr (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** SR ref

value of the field

**RPC name:** `set_suspend_image_sr`

**Overview:** Set the `suspend_image_sr` field of the given host.

**Signature:**

```
void set_suspend_image_sr (session_id s, host ref self, SR ref value)
```

**Arguments:**

| type     | name  | description             |
|----------|-------|-------------------------|
| host ref | self  | reference to the object |
| SR ref   | value | New value to set        |

**Return Type:** void

**RPC name:** `get_crash_dump_sr`

**Overview:** Get the `crash_dump_sr` field of the given host.

**Signature:**

```
(SR ref) get_crash_dump_sr (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** SR ref  
value of the field

**RPC name:** `set_crash_dump_sr`

**Overview:** Set the `crash_dump_sr` field of the given host.

**Signature:**

```
void set_crash_dump_sr (session_id s, host ref self, SR ref value)
```

**Arguments:**

| type     | name  | description             |
|----------|-------|-------------------------|
| host ref | self  | reference to the object |
| SR ref   | value | New value to set        |

**Return Type:** void

**RPC name:** get\_PBDs**Overview:** Get the PBDs field of the given host.**Signature:**

```
((PBD ref) Set) get_PBDs (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** (PBD ref) Set

value of the field

**RPC name:** get\_host\_CPUs**Overview:** Get the host\_CPUs field of the given host.**Signature:**

```
((host_cpu ref) Set) get_host_CPUs (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** (host\_cpu ref) Set

value of the field

**RPC name:** get\_metrics**Overview:** Get the metrics field of the given host.**Signature:**

```
(host_metrics ref) get_metrics (session_id s, host ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** host\_metrics ref

value of the field

**RPC name:** get\_by\_uuid**Overview:** Get a reference to the host instance with the specified UUID.**Signature:**

```
(host ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** host ref  
reference to the object

**RPC name:** get\_record

**Overview:** Get a record containing the current state of the given host.

**Signature:**

(host record) get\_record (session\_id s, host ref self)

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| host ref | self | reference to the object |

**Return Type:** host record  
all fields from the object

**RPC name:** get\_by\_name\_label

**Overview:** Get all the host instances with the given label.

**Signature:**

((host ref) Set) get\_by\_name\_label (session\_id s, string label)

**Arguments:**

| type   | name  | description               |
|--------|-------|---------------------------|
| string | label | label of object to return |

**Return Type:** (host ref) Set  
references to objects with match names

## 2.11 Class: host\_metrics

### 2.11.1 Fields for class: host\_metrics

|               |  |          |   |
|---------------|--|----------|---|
| Name          | <b>host_metrics</b>                        |          |   |
| Description   | <i>The metrics associated with a host.</i> |          |   |
| Quals         | Field                                      | Type     | Description                                     |
| <i>RO_run</i> | <code>uuid</code>                          | string   | unique identifier/object reference              |
| <i>RO_run</i> | <code>memory/total</code>                  | int      | Host's total memory (bytes)                     |
| <i>RO_run</i> | <code>memory/free</code>                   | int      | Host's free memory (bytes)                      |
| <i>RO_run</i> | <code>last_updated</code>                  | datetime | Time at which this information was last updated |

### 2.11.2 RPCs associated with class: host\_metrics

#### RPC name: get\_all

**Overview:** Return a list of all the host\_metrics instances known to the system.

**Signature:**

```
((host_metrics ref) Set) get_all (session_id s)
```

**Return Type:** (host\_metrics ref) Set

references to all objects

#### RPC name: get\_uuid

**Overview:** Get the uuid field of the given host\_metrics.

**Signature:**

```
string get_uuid (session_id s, host_metrics ref self)
```

**Arguments:**

| type             | name | description             |
|------------------|------|-------------------------|
| host_metrics ref | self | reference to the object |

**Return Type:** string

value of the field

#### RPC name: get\_memory\_total

**Overview:** Get the memory/total field of the given host\_metrics.

**Signature:**

```
int get_memory_total (session_id s, host_metrics ref self)
```

**Arguments:**

| type             | name | description             |
|------------------|------|-------------------------|
| host_metrics ref | self | reference to the object |

**Return Type:** int

value of the field

**RPC name:** `get_memory_free`

**Overview:** Get the `memory_free` field of the given `host_metrics`.

**Signature:**

```
int get_memory_free (session_id s, host_metrics ref self)
```

**Arguments:**

| type                          | name              | description             |
|-------------------------------|-------------------|-------------------------|
| <code>host_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `int`

value of the field

**RPC name:** `get_last_updated`

**Overview:** Get the `last_updated` field of the given `host_metrics`.

**Signature:**

```
datetime get_last_updated (session_id s, host_metrics ref self)
```

**Arguments:**

| type                          | name              | description             |
|-------------------------------|-------------------|-------------------------|
| <code>host_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `datetime`

value of the field

**RPC name:** `get_by_uuid`

**Overview:** Get a reference to the `host_metrics` instance with the specified UUID.

**Signature:**

```
(host_metrics ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type                | name              | description              |
|---------------------|-------------------|--------------------------|
| <code>string</code> | <code>uuid</code> | UUID of object to return |

**Return Type:** `host_metrics ref`

reference to the object

**RPC name:** `get_record`

**Overview:** Get a record containing the current state of the given `host_metrics`.

**Signature:**

```
(host_metrics record) get_record (session_id s, host_metrics ref self)
```

**Arguments:**

| type                          | name              | description             |
|-------------------------------|-------------------|-------------------------|
| <code>host_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `host_metrics record`

all fields from the object

## 2.12 Class: host\_cpu

### 2.12.1 Fields for class: host\_cpu

| Name          | host_cpu              |          |   |
|---------------|-----------------------|----------|---|
| Description   | <i>A physical CPU</i> |          |   |
| Quals         | Field                 | Type     | Description   |
| <i>RO_run</i> | uuid                  | string   | unique identifier/object reference                                      |
| <i>RO_run</i> | host                  | host ref | the host the CPU is in  |
| <i>RO_run</i> | number                | int      | the number of the physical CPU within the host                          |
| <i>RO_run</i> | vendor                | string   | the vendor of the physical CPU  |
| <i>RO_run</i> | speed                 | int      | the speed of the physical CPU   |
| <i>RO_run</i> | modelname             | string   | the model name of the physical CPU                                      |
| <i>RO_run</i> | stepping              | string   | the stepping of the physical CPU  |
| <i>RO_run</i> | flags                 | string   | the flags of the physical CPU (a decoded version of the features field) |
| <i>RO_run</i> | features              | string   | the physical CPU feature bitmap   |
| <i>RO_run</i> | utilisation           | float    | the current CPU utilisation   |

### 2.12.2 RPCs associated with class: host\_cpu

#### RPC name: get\_all

**Overview:** Return a list of all the host\_cpus known to the system.

**Signature:**

```
((host_cpu ref) Set) get_all (session_id s)
```

**Return Type:** (host\_cpu ref) Set  
references to all objects

#### RPC name: get\_uuid

**Overview:** Get the uuid field of the given host\_cpu.

**Signature:**

```
string get_uuid (session_id s, host_cpu ref self)
```

**Arguments:**

| type         | name | description             |
|--------------|------|-------------------------|
| host_cpu ref | self | reference to the object |

**Return Type:** string  
value of the field

#### RPC name: get\_host

**Overview:** Get the host field of the given host\_cpu.

**Signature:**



```
(host ref) get_host (session_id s, host_cpu ref self)
```

**Arguments:**

| type         | name | description             |
|--------------|------|-------------------------|
| host_cpu ref | self | reference to the object |

**Return Type:** host ref  
value of the field

**RPC name:** get\_number

**Overview:** Get the number field of the given host\_cpu.

**Signature:**

```
int get_number (session_id s, host_cpu ref self)
```

**Arguments:**

| type         | name | description             |
|--------------|------|-------------------------|
| host_cpu ref | self | reference to the object |

**Return Type:** int  
value of the field

**RPC name:** get\_vendor

**Overview:** Get the vendor field of the given host\_cpu.

**Signature:**

```
string get_vendor (session_id s, host_cpu ref self)
```

**Arguments:**

| type         | name | description             |
|--------------|------|-------------------------|
| host_cpu ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** get\_speed

**Overview:** Get the speed field of the given host\_cpu.

**Signature:**

```
int get_speed (session_id s, host_cpu ref self)
```

**Arguments:**

| type         | name | description             |
|--------------|------|-------------------------|
| host_cpu ref | self | reference to the object |

**Return Type:** int

value of the field

**RPC name:** `get_modelname`

**Overview:** Get the `modelname` field of the given `host_cpu`.

**Signature:**

```
string get_modelname (session_id s, host_cpu ref self)
```

**Arguments:**

| type                      | name              | description             |
|---------------------------|-------------------|-------------------------|
| <code>host_cpu ref</code> | <code>self</code> | reference to the object |

**Return Type:** `string`

value of the field

**RPC name:** `get_stepping`

**Overview:** Get the `stepping` field of the given `host_cpu`.

**Signature:**

```
string get_stepping (session_id s, host_cpu ref self)
```

**Arguments:**

| type                      | name              | description             |
|---------------------------|-------------------|-------------------------|
| <code>host_cpu ref</code> | <code>self</code> | reference to the object |

**Return Type:** `string`

value of the field

**RPC name:** `get_flags`

**Overview:** Get the `flags` field of the given `host_cpu`.

**Signature:**

```
string get_flags (session_id s, host_cpu ref self)
```

**Arguments:**

| type                      | name              | description             |
|---------------------------|-------------------|-------------------------|
| <code>host_cpu ref</code> | <code>self</code> | reference to the object |

**Return Type:** `string`

value of the field

**RPC name:** `get_features`

**Overview:** Get the features field of the given `host_cpu`.

**Signature:**

```
string get_features (session_id s, host_cpu ref self)
```

**Arguments:**

| type         | name | description             |
|--------------|------|-------------------------|
| host_cpu ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `get_utilisation`

**Overview:** Get the utilisation field of the given `host_cpu`.

**Signature:**

```
float get_utilisation (session_id s, host_cpu ref self)
```

**Arguments:**

| type         | name | description             |
|--------------|------|-------------------------|
| host_cpu ref | self | reference to the object |

**Return Type:** float

value of the field

**RPC name:** `get_by_uuid`

**Overview:** Get a reference to the `host_cpu` instance with the specified UUID.

**Signature:**

```
(host_cpu ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** host\_cpu ref

reference to the object

**RPC name:** `get_record`

**Overview:** Get a record containing the current state of the given `host_cpu`.

**Signature:**

```
(host_cpu record) get_record (session_id s, host_cpu ref self)
```

**Arguments:**

| <b>type</b>  | <b>name</b> | <b>description</b>      |
|--------------|-------------|-------------------------|
| host_cpu ref | self        | reference to the object |

**Return Type:** host\_cpu record  
all fields from the object

## 2.13 Class: network

### 2.13.1 Fields for class: network

|                         |                           |                       |   |
|-------------------------|---------------------------|-----------------------|---|
| Name                    | <b>network</b>            |                       |   |
| Description             | <i>A virtual network.</i> |                       |   |
| Quals                   | Field                     | Type                  | Description   |
| <i>RO<sub>run</sub></i> | uuid                      | string                | unique identifier/object reference                  |
| <i>RW</i>               | name/label                | string                | a human-readable name                               |
| <i>RW</i>               | name/description          | string                | a notes field containing human-readable description |
| <i>RO<sub>run</sub></i> | VIFs                      | (VIF ref) Set         | list of connected vifs                              |
| <i>RO<sub>run</sub></i> | PIFs                      | (PIF ref) Set         | list of connected pifs                              |
| <i>RW</i>               | other_config              | (string → string) Map | additional configuration                            |

### 2.13.2 RPCs associated with class: network

#### RPC name: get\_all

**Overview:** Return a list of all the networks known to the system

**Signature:**

```
((network ref) Set) get_all (session_id s)
```

**Return Type:** (network ref) Set

A list of all the IDs of all the networks

#### RPC name: get\_uuid

**Overview:** Get the uuid field of the given network.

**Signature:**

```
string get_uuid (session_id s, network ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| network ref | self | reference to the object |

**Return Type:** string

value of the field

#### RPC name: get\_name\_label

**Overview:** Get the name/label field of the given network.

**Signature:**

```
string get_name_label (session_id s, network ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| network ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** set\_name\_label

**Overview:** Set the name/label field of the given network.

**Signature:**

```
void set_name_label (session_id s, network ref self, string value)
```

**Arguments:**

| type        | name  | description             |
|-------------|-------|-------------------------|
| network ref | self  | reference to the object |
| string      | value | New value to set        |

**Return Type:** void

**RPC name:** get\_name\_description

**Overview:** Get the name/description field of the given network.

**Signature:**

```
string get_name_description (session_id s, network ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| network ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** set\_name\_description

**Overview:** Set the name/description field of the given network.

**Signature:**

```
void set_name_description (session_id s, network ref self, string value)
```

**Arguments:**

| type        | name  | description             |
|-------------|-------|-------------------------|
| network ref | self  | reference to the object |
| string      | value | New value to set        |

**Return Type:** void

**RPC name:** `get_VIFs`**Overview:** Get the VIFs field of the given network.**Signature:**`((VIF ref) Set) get_VIFs (session_id s, network ref self)`**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| network ref | self | reference to the object |

**Return Type:** `(VIF ref) Set`

value of the field

**RPC name:** `get_PIFs`**Overview:** Get the PIFs field of the given network.**Signature:**`((PIF ref) Set) get_PIFs (session_id s, network ref self)`**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| network ref | self | reference to the object |

**Return Type:** `(PIF ref) Set`

value of the field

**RPC name:** `get_other_config`**Overview:** Get the other\_config field of the given network.**Signature:**`((string -> string) Map) get_other_config (session_id s, network ref self)`**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| network ref | self | reference to the object |

**Return Type:** `(string -> string) Map`

value of the field

**RPC name:** `set_other_config`**Overview:** Set the other\_config field of the given network.**Signature:**`void set_other_config (session_id s, network ref self, (string -> string) Map value)`

**Arguments:**

| type                  | name  | description             |
|-----------------------|-------|-------------------------|
| network ref           | self  | reference to the object |
| (string → string) Map | value | New value to set        |

**Return Type:** void**RPC name:** add\_to\_other\_config**Overview:** Add the given key-value pair to the other\_config field of the given network.**Signature:**

```
void add_to_other_config (session_id s, network ref self, string key, string value)
```

**Arguments:**

| type        | name  | description             |
|-------------|-------|-------------------------|
| network ref | self  | reference to the object |
| string      | key   | Key to add              |
| string      | value | Value to add            |

**Return Type:** void**RPC name:** remove\_from\_other\_config**Overview:** Remove the given key and its corresponding value from the other\_config field of the given network. If the key is not in that Map, then do nothing.**Signature:**

```
void remove_from_other_config (session_id s, network ref self, string key)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| network ref | self | reference to the object |
| string      | key  | Key to remove           |

**Return Type:** void**RPC name:** create**Overview:** Create a new network instance, and return its handle.**Signature:**

```
(network ref) create (session_id s, network record args)
```

**Arguments:**

| type           | name | description               |
|----------------|------|---------------------------|
| network record | args | All constructor arguments |

**Return Type:** network ref



reference to the newly created object

**RPC name:** `destroy`

**Overview:** Destroy the specified network instance.

**Signature:**

```
void destroy (session_id s, network ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| network ref | self | reference to the object |

**Return Type:** void

**RPC name:** `get_by_uuid`

**Overview:** Get a reference to the network instance with the specified UUID.

**Signature:**

```
(network ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** network ref

reference to the object

**RPC name:** `get_record`

**Overview:** Get a record containing the current state of the given network.

**Signature:**

```
(network record) get_record (session_id s, network ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| network ref | self | reference to the object |

**Return Type:** network record

all fields from the object

**RPC name:** `get_by_name_label`

**Overview:** Get all the network instances with the given label.

**Signature:**

```
((network ref) Set) get_by_name_label (session_id s, string label)
```

**Arguments:**

| type   | name  | description               |
|--------|-------|---------------------------|
| string | label | label of object to return |

**Return Type:** (network ref) Set  
references to objects with match names

## 2.14 Class: VIF

### 2.14.1 Fields for class: VIF

| Name                    | VIF                                 |                       |   |
|-------------------------|-------------------------------------|-----------------------|---|
| Description             | <i>A virtual network interface.</i> |                       |   |
| Quals                   | Field                               | Type                  | Description   |
| <i>RO<sub>run</sub></i> | uuid                                | string                | unique identifier/object reference  |
| <i>RW</i>               | device                              | string                | name of network device as exposed to guest e.g. eth0                                      |
| <i>RO<sub>ins</sub></i> | network                             | network ref           | virtual network to which this vif is connected  |
| <i>RO<sub>ins</sub></i> | VM                                  | VM ref                | virtual machine to which this vif is connected  |
| <i>RW</i>               | MAC                                 | string                | ethernet MAC address of virtual interface, as exposed to guest                            |
| <i>RW</i>               | MTU                                 | int                   | MTU in octets   |
| <i>RO<sub>run</sub></i> | currently_attached                  | bool                  | is the device currently attached (erased on reboot)                                       |
| <i>RO<sub>run</sub></i> | status_code                         | int                   | error/success code associated with last attach-operation (erased on reboot)               |
| <i>RO<sub>run</sub></i> | status_detail                       | string                | error/success information associated with last attach-operation status (erased on reboot) |
| <i>RO<sub>run</sub></i> | runtime_properties                  | (string → string) Map | Device runtime properties   |
| <i>RW</i>               | qos/algorithm.type                  | string                | QoS algorithm to use  |
| <i>RW</i>               | qos/algorithm.params                | (string → string) Map | parameters for chosen QoS algorithm   |
| <i>RO<sub>run</sub></i> | qos/supported.algorithms            | string Set            | supported QoS algorithms for this VIF   |
| <i>RO<sub>run</sub></i> | metrics                             | VIF_metrics ref       | metrics associated with this VIF  |

### 2.14.2 RPCs associated with class: VIF

#### RPC name: plug

**Overview:** Hotplug the specified VIF, dynamically attaching it to the running VM.

#### Signature:

```
void plug (session_id s, VIF ref self)
```

#### Arguments:

| type    | name | description        |
|---------|------|--------------------|
| VIF ref | self | The VIF to hotplug |

**Return Type:** void

#### RPC name: unplug

**Overview:** Hot-unplug the specified VIF, dynamically unattaching it from the running VM.

#### Signature:

```
void unplug (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description           |
|---------|------|-----------------------|
| VIF ref | self | The VIF to hot-unplug |

**Return Type:** void

**RPC name:** get\_all

**Overview:** Return a list of all the VIFs known to the system.

**Signature:**

```
((VIF ref) Set) get_all (session_id s)
```

**Return Type:** (VIF ref) Set  
references to all objects

**RPC name:** get\_uuid

**Overview:** Get the uuid field of the given VIF.

**Signature:**

```
string get_uuid (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** get\_device

**Overview:** Get the device field of the given VIF.

**Signature:**

```
string get_device (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** set\_device**Overview:** Set the device field of the given VIF.**Signature:**

```
void set_device (session_id s, VIF ref self, string value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VIF ref | self  | reference to the object |
| string  | value | New value to set        |

**Return Type:** void**RPC name:** get\_network**Overview:** Get the network field of the given VIF.**Signature:**

```
(network ref) get_network (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** network ref  
value of the field**RPC name:** get\_VM**Overview:** Get the VM field of the given VIF.**Signature:**

```
(VM ref) get_VM (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** VM ref  
value of the field**RPC name:** get\_MAC**Overview:** Get the MAC field of the given VIF.**Signature:**

```
string get_MAC (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** set\_MAC

**Overview:** Set the MAC field of the given VIF.

**Signature:**

```
void set_MAC (session_id s, VIF ref self, string value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VIF ref | self  | reference to the object |
| string  | value | New value to set        |

**Return Type:** void

**RPC name:** get\_MTU

**Overview:** Get the MTU field of the given VIF.

**Signature:**

```
int get_MTU (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** int  
value of the field

**RPC name:** set\_MTU

**Overview:** Set the MTU field of the given VIF.

**Signature:**

```
void set_MTU (session_id s, VIF ref self, int value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VIF ref | self  | reference to the object |
| int     | value | New value to set        |

**Return Type:** void

**RPC name:** `get_currently_attached`**Overview:** Get the `currently_attached` field of the given VIF.**Signature:**

```
bool get_currently_attached (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** `bool`

value of the field

**RPC name:** `get_status_code`**Overview:** Get the `status_code` field of the given VIF.**Signature:**

```
int get_status_code (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** `int`

value of the field

**RPC name:** `get_status_detail`**Overview:** Get the `status_detail` field of the given VIF.**Signature:**

```
string get_status_detail (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** `string`

value of the field

**RPC name:** `get_runtime_properties`**Overview:** Get the `runtime_properties` field of the given VIF.**Signature:**

```
((string -> string) Map) get_runtime_properties (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name: get\_qos\_algorithm\_type**

**Overview:** Get the qos/algorithm\_type field of the given VIF.

**Signature:**

```
string get_qos_algorithm_type (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name: set\_qos\_algorithm\_type**

**Overview:** Set the qos/algorithm\_type field of the given VIF.

**Signature:**

```
void set_qos_algorithm_type (session_id s, VIF ref self, string value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VIF ref | self  | reference to the object |
| string  | value | New value to set        |

**Return Type:** void

**RPC name: get\_qos\_algorithm\_params**

**Overview:** Get the qos/algorithm\_params field of the given VIF.

**Signature:**

```
((string -> string) Map) get_qos_algorithm_params (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field



**RPC name: set\_qos\_algorithm\_params****Overview:** Set the qos/algorithm\_params field of the given VIF.**Signature:**

```
void set_qos_algorithm_params (session_id s, VIF ref self, (string -> string) Map value)
```

**Arguments:**

| type                  | name  | description             |
|-----------------------|-------|-------------------------|
| VIF ref               | self  | reference to the object |
| (string → string) Map | value | New value to set        |

**Return Type:** void**RPC name: add\_to\_qos\_algorithm\_params****Overview:** Add the given key-value pair to the qos/algorithm\_params field of the given VIF.**Signature:**

```
void add_to_qos_algorithm_params (session_id s, VIF ref self, string key, string value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VIF ref | self  | reference to the object |
| string  | key   | Key to add              |
| string  | value | Value to add            |

**Return Type:** void**RPC name: remove\_from\_qos\_algorithm\_params****Overview:** Remove the given key and its corresponding value from the qos/algorithm\_params field of the given VIF. If the key is not in that Map, then do nothing.**Signature:**

```
void remove_from_qos_algorithm_params (session_id s, VIF ref self, string key)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |
| string  | key  | Key to remove           |

**Return Type:** void**RPC name: get\_qos\_supported\_algorithms****Overview:** Get the qos/supported\_algorithms field of the given VIF.**Signature:**

```
(string Set) get_qos_supported_algorithms (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** string Set  
value of the field

**RPC name:** get\_metrics

**Overview:** Get the metrics field of the given VIF.

**Signature:**

```
(VIF_metrics ref) get_metrics (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** VIF\_metrics ref  
value of the field

**RPC name:** create

**Overview:** Create a new VIF instance, and return its handle.

**Signature:**

```
(VIF ref) create (session_id s, VIF record args)
```

**Arguments:**

| type       | name | description               |
|------------|------|---------------------------|
| VIF record | args | All constructor arguments |

**Return Type:** VIF ref  
reference to the newly created object

**RPC name:** destroy

**Overview:** Destroy the specified VIF instance.

**Signature:**

```
void destroy (session_id s, VIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** void

**RPC name:** `get_by_uuid`

**Overview:** Get a reference to the VIF instance with the specified UUID.

**Signature:**

(VIF ref) `get_by_uuid (session_id s, string uuid)`

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** VIF ref

reference to the object

**RPC name:** `get_record`

**Overview:** Get a record containing the current state of the given VIF.

**Signature:**

(VIF record) `get_record (session_id s, VIF ref self)`

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VIF ref | self | reference to the object |

**Return Type:** VIF record

all fields from the object

## 2.15 Class: VIF\_metrics

### 2.15.1 Fields for class: VIF\_metrics

|               |  |          |   |
|---------------|--|----------|---|
| Name          | <b>VIF_metrics</b>   |          |   |
| Description   | <i>The metrics associated with a virtual network device.</i> |          |   |
| Quals         | Field  | Type     | Description                                     |
| <i>RO_run</i> | <code>uuid</code>  | string   | unique identifier/object reference              |
| <i>RO_run</i> | <code>io/read_kbs</code>                                     | float    | Read bandwidth (KiB/s)                          |
| <i>RO_run</i> | <code>io/write_kbs</code>                                    | float    | Write bandwidth (KiB/s)                         |
| <i>RO_run</i> | <code>last_updated</code>                                    | datetime | Time at which this information was last updated |

### 2.15.2 RPCs associated with class: VIF\_metrics

**RPC name:** `get_all`

**Overview:** Return a list of all the VIF\_metrics instances known to the system.

**Signature:**

```
((VIF_metrics ref) Set) get_all (session_id s)
```

**Return Type:** (VIF\_metrics ref) Set

references to all objects

**RPC name:** `get_uuid`

**Overview:** Get the uuid field of the given VIF\_metrics.

**Signature:**

```
string get_uuid (session_id s, VIF_metrics ref self)
```

**Arguments:**

| type            | name | description             |
|-----------------|------|-------------------------|
| VIF_metrics ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `get_io_read_kbs`

**Overview:** Get the io/read\_kbs field of the given VIF\_metrics.

**Signature:**

```
float get_io_read_kbs (session_id s, VIF_metrics ref self)
```

**Arguments:**

| type            | name | description             |
|-----------------|------|-------------------------|
| VIF_metrics ref | self | reference to the object |

**Return Type:** float

value of the field

**RPC name:** `get_io_write_kbs`

**Overview:** Get the `io/write_kbs` field of the given `VIF_metrics`.

**Signature:**

```
float get_io_write_kbs (session_id s, VIF_metrics ref self)
```

**Arguments:**

| type                         | name              | description             |
|------------------------------|-------------------|-------------------------|
| <code>VIF_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `float`

value of the field

**RPC name:** `get_last_updated`

**Overview:** Get the `last_updated` field of the given `VIF_metrics`.

**Signature:**

```
datetime get_last_updated (session_id s, VIF_metrics ref self)
```

**Arguments:**

| type                         | name              | description             |
|------------------------------|-------------------|-------------------------|
| <code>VIF_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `datetime`

value of the field

**RPC name:** `get_by_uuid`

**Overview:** Get a reference to the `VIF_metrics` instance with the specified UUID.

**Signature:**

```
(VIF_metrics ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type                | name              | description              |
|---------------------|-------------------|--------------------------|
| <code>string</code> | <code>uuid</code> | UUID of object to return |

**Return Type:** `VIF_metrics ref`

reference to the object

**RPC name:** `get_record`

**Overview:** Get a record containing the current state of the given `VIF_metrics`.

**Signature:**

```
(VIF_metrics record) get_record (session_id s, VIF_metrics ref self)
```

**Arguments:**

| type                         | name              | description             |
|------------------------------|-------------------|-------------------------|
| <code>VIF_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `VIF_metrics record`

all fields from the object

## 2.16 Class: PIF

### 2.16.1 Fields for class: PIF

| Name                    | PIF  |                 |   |
|-------------------------|--|-----------------|---|
| Description             | <i>A physical network interface (note separate VLANs are represented as several PIFs).</i> |                 |   |
| Quals                   | Field  | Type            | Description   |
| <i>RO<sub>run</sub></i> | uuid   | string          | unique identifier/object reference                      |
| <i>RW</i>               | device   | string          | machine-readable name of the interface (e.g. eth0)      |
| <i>RO<sub>ins</sub></i> | network  | network ref     | virtual network to which this pif is connected          |
| <i>RO<sub>ins</sub></i> | host   | host ref        | physical machine to which this pif is connected         |
| <i>RW</i>               | MAC  | string          | ethernet MAC address of physical interface              |
| <i>RW</i>               | MTU  | int             | MTU in octets   |
| <i>RW</i>               | VLAN   | int             | VLAN tag for all traffic passing through this interface |
| <i>RO<sub>run</sub></i> | metrics  | PIF_metrics ref | metrics associated with this PIF                        |

### 2.16.2 RPCs associated with class: PIF

#### RPC name: create\_VLAN

**Overview:** Create a VLAN interface from an existing physical interface.

**Signature:**

(PIF ref) create\_VLAN (session\_id s, string device, network ref network, host ref host, int VLAN)

**Arguments:**

| type        | name    | description   |
|-------------|---------|---|
| string      | device  | physical interface on which to crate the VLAN interface |
| network ref | network | network to which this interface should be connected     |
| host ref    | host    | physical machine to which this PIF is connected         |
| int         | VLAN    | VLAN tag for the new interface                          |

**Return Type:** PIF ref

The reference of the created PIF object

**Possible Error Codes:** VLAN\_TAG\_INVALID

#### RPC name: destroy

**Overview:** Destroy the interface (provided it is a synthetic interface like a VLAN; fail if it is a physical interface).

**Signature:**

void destroy (session\_id s, PIF ref self)

**Arguments:**

| type    | name | description               |
|---------|------|---------------------------|
| PIF ref | self | the PIF object to destroy |

**Return Type:** void**Possible Error Codes:** PIF\_IS\_PHYSICAL**RPC name:** get\_all**Overview:** Return a list of all the PIFs known to the system.**Signature:**

```
((PIF ref) Set) get_all (session_id s)
```

**Return Type:** (PIF ref) Set  
references to all objects**RPC name:** get\_uuid**Overview:** Get the uuid field of the given PIF.**Signature:**

```
string get_uuid (session_id s, PIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PIF ref | self | reference to the object |

**Return Type:** string  
value of the field**RPC name:** get\_device**Overview:** Get the device field of the given PIF.**Signature:**

```
string get_device (session_id s, PIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PIF ref | self | reference to the object |

**Return Type:** string  
value of the field



**RPC name:** set\_device**Overview:** Set the device field of the given PIF.**Signature:**

```
void set_device (session_id s, PIF ref self, string value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| PIF ref | self  | reference to the object |
| string  | value | New value to set        |

**Return Type:** void**RPC name:** get\_network**Overview:** Get the network field of the given PIF.**Signature:**

```
(network ref) get_network (session_id s, PIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PIF ref | self | reference to the object |

**Return Type:** network ref  
value of the field**RPC name:** get\_host**Overview:** Get the host field of the given PIF.**Signature:**

```
(host ref) get_host (session_id s, PIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PIF ref | self | reference to the object |

**Return Type:** host ref  
value of the field**RPC name:** get\_MAC**Overview:** Get the MAC field of the given PIF.**Signature:**

```
string get_MAC (session_id s, PIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PIF ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** set\_MAC

**Overview:** Set the MAC field of the given PIF.

**Signature:**

```
void set_MAC (session_id s, PIF ref self, string value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| PIF ref | self  | reference to the object |
| string  | value | New value to set        |

**Return Type:** void

**RPC name:** get\_MTU

**Overview:** Get the MTU field of the given PIF.

**Signature:**

```
int get_MTU (session_id s, PIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PIF ref | self | reference to the object |

**Return Type:** int  
value of the field

**RPC name:** set\_MTU

**Overview:** Set the MTU field of the given PIF.

**Signature:**

```
void set_MTU (session_id s, PIF ref self, int value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| PIF ref | self  | reference to the object |
| int     | value | New value to set        |

**Return Type:** void

**RPC name:** get\_VLAN**Overview:** Get the VLAN field of the given PIF.**Signature:**

```
int get_VLAN (session_id s, PIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PIF ref | self | reference to the object |

**Return Type:** int

value of the field

**RPC name:** set\_VLAN**Overview:** Set the VLAN field of the given PIF.**Signature:**

```
void set_VLAN (session_id s, PIF ref self, int value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| PIF ref | self  | reference to the object |
| int     | value | New value to set        |

**Return Type:** void**RPC name:** get\_metrics**Overview:** Get the metrics field of the given PIF.**Signature:**

```
(PIF_metrics ref) get_metrics (session_id s, PIF ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PIF ref | self | reference to the object |

**Return Type:** PIF\_metrics ref

value of the field

**RPC name:** get\_by\_uuid**Overview:** Get a reference to the PIF instance with the specified UUID.**Signature:**

```
(PIF ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** PIF ref  
reference to the object

**RPC name:** get\_record

**Overview:** Get a record containing the current state of the given PIF.

**Signature:**

(PIF record) get\_record (session\_id s, PIF ref self)

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PIF ref | self | reference to the object |

**Return Type:** PIF record  
all fields from the object

## 2.17 Class: PIF\_metrics

### 2.17.1 Fields for class: PIF\_metrics

|               |  |          |   |
|---------------|--|----------|---|
| Name          | <b>PIF_metrics</b>   |          |   |
| Description   | <i>The metrics associated with a physical network interface.</i> |          |   |
| Quals         | Field  | Type     | Description                                     |
| <i>RO_run</i> | <code>uuid</code>  | string   | unique identifier/object reference              |
| <i>RO_run</i> | <code>io/read_kbs</code>   | float    | Read bandwidth (KiB/s)                          |
| <i>RO_run</i> | <code>io/write_kbs</code>  | float    | Write bandwidth (KiB/s)                         |
| <i>RO_run</i> | <code>last_updated</code>  | datetime | Time at which this information was last updated |

### 2.17.2 RPCs associated with class: PIF\_metrics

**RPC name:** `get_all`

**Overview:** Return a list of all the PIF\_metrics instances known to the system.

**Signature:**

```
((PIF_metrics ref) Set) get_all (session_id s)
```

**Return Type:** (PIF\_metrics ref) Set

references to all objects

**RPC name:** `get_uuid`

**Overview:** Get the uuid field of the given PIF\_metrics.

**Signature:**

```
string get_uuid (session_id s, PIF_metrics ref self)
```

**Arguments:**

| type            | name | description             |
|-----------------|------|-------------------------|
| PIF_metrics ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `get_io_read_kbs`

**Overview:** Get the io/read\_kbs field of the given PIF\_metrics.

**Signature:**

```
float get_io_read_kbs (session_id s, PIF_metrics ref self)
```

**Arguments:**

| type            | name | description             |
|-----------------|------|-------------------------|
| PIF_metrics ref | self | reference to the object |

**Return Type:** float

value of the field

**RPC name:** `get_io_write_kbs`

**Overview:** Get the `io/write_kbs` field of the given `PIF_metrics`.

**Signature:**

```
float get_io_write_kbs (session_id s, PIF_metrics ref self)
```

**Arguments:**

| type                         | name              | description             |
|------------------------------|-------------------|-------------------------|
| <code>PIF_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `float`

value of the field

**RPC name:** `get_last_updated`

**Overview:** Get the `last_updated` field of the given `PIF_metrics`.

**Signature:**

```
datetime get_last_updated (session_id s, PIF_metrics ref self)
```

**Arguments:**

| type                         | name              | description             |
|------------------------------|-------------------|-------------------------|
| <code>PIF_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `datetime`

value of the field

**RPC name:** `get_by_uuid`

**Overview:** Get a reference to the `PIF_metrics` instance with the specified UUID.

**Signature:**

```
(PIF_metrics ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type                | name              | description              |
|---------------------|-------------------|--------------------------|
| <code>string</code> | <code>uuid</code> | UUID of object to return |

**Return Type:** `PIF_metrics ref`

reference to the object

**RPC name:** `get_record`

**Overview:** Get a record containing the current state of the given `PIF_metrics`.

**Signature:**

```
(PIF_metrics record) get_record (session_id s, PIF_metrics ref self)
```

**Arguments:**

| type                         | name              | description             |
|------------------------------|-------------------|-------------------------|
| <code>PIF_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `PIF_metrics record`

all fields from the object

## 2.18 Class: SR

### 2.18.1 Fields for class: SR

| Name                    | SR                                |               |  |
|-------------------------|-----------------------------------|---------------|--|
| Description             | <i>A storage repository.</i>      |               |  |
| Quals                   | Field                             | Type          | Description  |
| <i>RO<sub>run</sub></i> | <code>uuid</code>                 | string        | unique identifier/object reference   |
| <i>RW</i>               | <code>name/label</code>           | string        | a human-readable name  |
| <i>RW</i>               | <code>name/description</code>     | string        | a notes field containing human-readable description  |
| <i>RO<sub>run</sub></i> | <code>VDIs</code>                 | (VDI ref) Set | managed virtual disks  |
| <i>RO<sub>run</sub></i> | <code>PBDs</code>                 | (PBD ref) Set | physical blockdevices  |
| <i>RO<sub>run</sub></i> | <code>virtual_allocation</code>   | int           | sum of <code>virtual_sizes</code> of all VDIs in this storage repository (in bytes)  |
| <i>RO<sub>run</sub></i> | <code>physical_utilisation</code> | int           | physical space currently utilised on this storage repository (in bytes). Note that for sparse disk formats, <code>physical_utilisation</code> may be less than <code>virtual_allocation</code> |
| <i>RO<sub>ins</sub></i> | <code>physical_size</code>        | int           | total physical size of the repository (in bytes)   |
| <i>RO<sub>ins</sub></i> | <code>type</code>                 | string        | type of the storage repository   |
| <i>RO<sub>ins</sub></i> | <code>content_type</code>         | string        | the type of the SR's content, if required (e.g. ISOs)  |

### 2.18.2 RPCs associated with class: SR

**RPC name:** `get_supported_types`

**Overview:** Return a set of all the SR types supported by the system.

**Signature:**

```
(string Set) get_supported_types (session_id s)
```

**Return Type:** string Set

the supported SR types

**RPC name:** `get_all`

**Overview:** Return a list of all the SRs known to the system.

**Signature:**

```
((SR ref) Set) get_all (session_id s)
```

**Return Type:** (SR ref) Set

references to all objects



**RPC name:** `get_uuid`**Overview:** Get the uuid field of the given SR.**Signature:**

```
string get_uuid (session_id s, SR ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| SR ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `get_name_label`**Overview:** Get the name/label field of the given SR.**Signature:**

```
string get_name_label (session_id s, SR ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| SR ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `set_name_label`**Overview:** Set the name/label field of the given SR.**Signature:**

```
void set_name_label (session_id s, SR ref self, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| SR ref | self  | reference to the object |
| string | value | New value to set        |

**Return Type:** void**RPC name:** `get_name_description`**Overview:** Get the name/description field of the given SR.**Signature:**

```
string get_name_description (session_id s, SR ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| SR ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** set\_name\_description

**Overview:** Set the name/description field of the given SR.

**Signature:**

```
void set_name_description (session_id s, SR ref self, string value)
```

**Arguments:**

| type   | name  | description             |
|--------|-------|-------------------------|
| SR ref | self  | reference to the object |
| string | value | New value to set        |

**Return Type:** void

**RPC name:** get\_VDI

**Overview:** Get the VDIs field of the given SR.

**Signature:**

```
((VDI ref) Set) get_VDI (session_id s, SR ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| SR ref | self | reference to the object |

**Return Type:** (VDI ref) Set  
value of the field

**RPC name:** get\_PBD

**Overview:** Get the PBDs field of the given SR.

**Signature:**

```
((PBD ref) Set) get_PBD (session_id s, SR ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| SR ref | self | reference to the object |

**Return Type:** (PBD ref) Set  
value of the field

**RPC name: get\_virtual\_allocation****Overview:** Get the virtual\_allocation field of the given SR.**Signature:**

```
int get_virtual_allocation (session_id s, SR ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| SR ref | self | reference to the object |

**Return Type:** int

value of the field

**RPC name: get\_physical\_utilisation****Overview:** Get the physical\_utilisation field of the given SR.**Signature:**

```
int get_physical_utilisation (session_id s, SR ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| SR ref | self | reference to the object |

**Return Type:** int

value of the field

**RPC name: get\_physical\_size****Overview:** Get the physical\_size field of the given SR.**Signature:**

```
int get_physical_size (session_id s, SR ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| SR ref | self | reference to the object |

**Return Type:** int

value of the field

**RPC name: get\_type****Overview:** Get the type field of the given SR.**Signature:**

```
string get_type (session_id s, SR ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| SR ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** get\_content\_type

**Overview:** Get the content\_type field of the given SR.

**Signature:**

```
string get_content_type (session_id s, SR ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| SR ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** get\_by\_uuid

**Overview:** Get a reference to the SR instance with the specified UUID.

**Signature:**

```
(SR ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** SR ref  
reference to the object

**RPC name:** get\_record

**Overview:** Get a record containing the current state of the given SR.

**Signature:**

```
(SR record) get_record (session_id s, SR ref self)
```

**Arguments:**

| type   | name | description             |
|--------|------|-------------------------|
| SR ref | self | reference to the object |

**Return Type:** SR record  
all fields from the object

**RPC name:** `get_by_name_label`

**Overview:** Get all the SR instances with the given label.

**Signature:**

`((SR ref) Set) get_by_name_label (session_id s, string label)`

**Arguments:**

| type   | name  | description               |
|--------|-------|---------------------------|
| string | label | label of object to return |

**Return Type:** `(SR ref) Set`

references to objects with match names

## 2.19 Class: VDI

### 2.19.1 Fields for class: VDI

| Name                    | VDI                               |                       |  |
|-------------------------|-----------------------------------|-----------------------|--|
| Description             | <i>A virtual disk image.</i>      |                       |  |
| Quals                   | Field                             | Type                  | Description  |
| <i>RO<sub>run</sub></i> | <code>uuid</code>                 | string                | unique identifier/object reference   |
| <i>RW</i>               | <code>name/label</code>           | string                | a human-readable name  |
| <i>RW</i>               | <code>name/description</code>     | string                | a notes field containing human-readable description  |
| <i>RO<sub>ins</sub></i> | <code>SR</code>                   | SR ref                | storage repository in which the VDI resides  |
| <i>RO<sub>run</sub></i> | <code>VBDs</code>                 | (VBD ref) Set         | list of vbds that refer to this disk   |
| <i>RO<sub>run</sub></i> | <code>crash_dumps</code>          | (crashdump ref) Set   | list of crash dumps that refer to this disk  |
| <i>RW</i>               | <code>virtual_size</code>         | int                   | size of disk as presented to the guest (in bytes). Note that, depending on storage backend type, requested size may not be respected exactly |
| <i>RO<sub>run</sub></i> | <code>physical_utilisation</code> | int                   | amount of physical space that the disk image is currently taking up on the storage repository (in bytes)                                     |
| <i>RO<sub>ins</sub></i> | <code>type</code>                 | <code>vdi_type</code> | type of the VDI  |
| <i>RW</i>               | <code>sharable</code>             | bool                  | true if this disk may be shared  |
| <i>RW</i>               | <code>read_only</code>            | bool                  | true if this disk may ONLY be mounted read-only  |
| <i>RW</i>               | <code>other_config</code>         | (string → string) Map | additional configuration   |

### 2.19.2 RPCs associated with class: VDI

#### RPC name: `get_all`

**Overview:** Return a list of all the VDIs known to the system.

**Signature:**

```
((VDI ref) Set) get_all (session_id s)
```

**Return Type:** (VDI ref) Set

references to all objects

#### RPC name: `get_uuid`

**Overview:** Get the uuid field of the given VDI.

**Signature:**

```
string get_uuid (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `get_name_label`

**Overview:** Get the name/label field of the given VDI.

**Signature:**

```
string get_name_label (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `set_name_label`

**Overview:** Set the name/label field of the given VDI.

**Signature:**

```
void set_name_label (session_id s, VDI ref self, string value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VDI ref | self  | reference to the object |
| string  | value | New value to set        |

**Return Type:** void

**RPC name:** `get_name_description`

**Overview:** Get the name/description field of the given VDI.

**Signature:**

```
string get_name_description (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name: set\_name\_description****Overview:** Set the name/description field of the given VDI.**Signature:**

```
void set_name_description (session_id s, VDI ref self, string value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VDI ref | self  | reference to the object |
| string  | value | New value to set        |

**Return Type:** void**RPC name: get\_SR****Overview:** Get the SR field of the given VDI.**Signature:**

```
(SR ref) get_SR (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** SR ref  
value of the field**RPC name: get\_VBDs****Overview:** Get the VBDs field of the given VDI.**Signature:**

```
((VBD ref) Set) get_VBDs (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** (VBD ref) Set  
value of the field**RPC name: get\_crash\_dumps****Overview:** Get the crash\_dumps field of the given VDI.**Signature:**

```
((crashdump ref) Set) get_crash_dumps (session_id s, VDI ref self)
```



**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** (crashdump ref) Set  
value of the field

**RPC name:** `get_virtual_size`

**Overview:** Get the `virtual_size` field of the given VDI.

**Signature:**

```
int get_virtual_size (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** int  
value of the field

**RPC name:** `set_virtual_size`

**Overview:** Set the `virtual_size` field of the given VDI.

**Signature:**

```
void set_virtual_size (session_id s, VDI ref self, int value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VDI ref | self  | reference to the object |
| int     | value | New value to set        |

**Return Type:** void

**RPC name:** `get_physical_utilisation`

**Overview:** Get the `physical_utilisation` field of the given VDI.

**Signature:**

```
int get_physical_utilisation (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** int  
value of the field

**RPC name:** `get_type`**Overview:** Get the type field of the given VDI.**Signature:**

```
(vdi_type) get_type (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** `vdi_type`

value of the field

**RPC name:** `get_sharable`**Overview:** Get the sharable field of the given VDI.**Signature:**

```
bool get_sharable (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** `bool`

value of the field

**RPC name:** `set_sharable`**Overview:** Set the sharable field of the given VDI.**Signature:**

```
void set_sharable (session_id s, VDI ref self, bool value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VDI ref | self  | reference to the object |
| bool    | value | New value to set        |

**Return Type:** `void`**RPC name:** `get_read_only`**Overview:** Get the read\_only field of the given VDI.**Signature:**

```
bool get_read_only (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** bool  
value of the field

**RPC name:** set\_read\_only

**Overview:** Set the read\_only field of the given VDI.

**Signature:**

```
void set_read_only (session_id s, VDI ref self, bool value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VDI ref | self  | reference to the object |
| bool    | value | New value to set        |

**Return Type:** void

**RPC name:** get\_other\_config

**Overview:** Get the other\_config field of the given VDI.

**Signature:**

```
((string -> string) Map) get_other_config (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name:** set\_other\_config

**Overview:** Set the other\_config field of the given VDI.

**Signature:**

```
void set_other_config (session_id s, VDI ref self, (string -> string) Map value)
```

**Arguments:**

| type                  | name  | description             |
|-----------------------|-------|-------------------------|
| VDI ref               | self  | reference to the object |
| (string → string) Map | value | New value to set        |

**Return Type:** void

**RPC name: add\_to\_other\_config**

**Overview:** Add the given key-value pair to the other\_config field of the given VDI.

**Signature:**

```
void add_to_other_config (session_id s, VDI ref self, string key, string value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VDI ref | self  | reference to the object |
| string  | key   | Key to add              |
| string  | value | Value to add            |

**Return Type:** void

**RPC name: remove\_from\_other\_config**

**Overview:** Remove the given key and its corresponding value from the other\_config field of the given VDI. If the key is not in that Map, then do nothing.

**Signature:**

```
void remove_from_other_config (session_id s, VDI ref self, string key)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |
| string  | key  | Key to remove           |

**Return Type:** void

**RPC name: create**

**Overview:** Create a new VDI instance, and return its handle.

**Signature:**

```
(VDI ref) create (session_id s, VDI record args)
```

**Arguments:**

| type       | name | description               |
|------------|------|---------------------------|
| VDI record | args | All constructor arguments |

**Return Type:** VDI ref

reference to the newly created object

**RPC name: destroy**

**Overview:** Destroy the specified VDI instance.

**Signature:**

```
void destroy (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** void

**RPC name:** `get_by_uuid`

**Overview:** Get a reference to the VDI instance with the specified UUID.

**Signature:**

```
(VDI ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** VDI ref  
reference to the object

**RPC name:** `get_record`

**Overview:** Get a record containing the current state of the given VDI.

**Signature:**

```
(VDI record) get_record (session_id s, VDI ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VDI ref | self | reference to the object |

**Return Type:** VDI record  
all fields from the object

**RPC name:** `get_by_name_label`

**Overview:** Get all the VDI instances with the given label.

**Signature:**

```
((VDI ref) Set) get_by_name_label (session_id s, string label)
```

**Arguments:**

| type   | name  | description               |
|--------|-------|---------------------------|
| string | label | label of object to return |

**Return Type:** (VDI ref) Set  
references to objects with match names

## 2.20 Class: VBD

### 2.20.1 Fields for class: VBD

| Name                    | VBD                            |                       |   |
|-------------------------|--------------------------------|-----------------------|---|
| Description             | <i>A virtual block device.</i> |                       |   |
| Quals                   | Field                          | Type                  | Description   |
| <i>RO<sub>run</sub></i> | uuid                           | string                | unique identifier/object reference  |
| <i>RO<sub>ins</sub></i> | VM                             | VM ref                | the virtual machine   |
| <i>RO<sub>ins</sub></i> | VDI                            | VDI ref               | the virtual disk  |
| <i>RW</i>               | device                         | string                | device seen by the guest e.g. hda1  |
| <i>RW</i>               | bootable                       | bool                  | true if this VBD is bootable  |
| <i>RW</i>               | mode                           | vbd_mode              | the mode the VBD should be mounted with   |
| <i>RW</i>               | type                           | vbd_type              | how the VBD will appear to the guest (e.g. disk or CD)                                    |
| <i>RO<sub>run</sub></i> | currently_attached             | bool                  | is the device currently attached (erased on reboot)                                       |
| <i>RO<sub>run</sub></i> | status_code                    | int                   | error/success code associated with last attach-operation (erased on reboot)               |
| <i>RO<sub>run</sub></i> | status_detail                  | string                | error/success information associated with last attach-operation status (erased on reboot) |
| <i>RO<sub>run</sub></i> | runtime_properties             | (string → string) Map | Device runtime properties   |
| <i>RW</i>               | qos/algorithm.type             | string                | QoS algorithm to use  |
| <i>RW</i>               | qos/algorithm.params           | (string → string) Map | parameters for chosen QoS algorithm   |
| <i>RO<sub>run</sub></i> | qos/supported.algorithms       | string Set            | supported QoS algorithms for this VBD   |
| <i>RO<sub>run</sub></i> | metrics                        | VBD_metrics ref       | metrics associated with this VBD  |

### 2.20.2 RPCs associated with class: VBD

#### RPC name: media\_change

**Overview:** Change the media in the device for CDROM-like devices only. For other devices, detach the VBD and attach a new one.

#### Signature:

```
void media_change (session_id s, VBD ref vbd, VDI ref vdi)
```

#### Arguments:

| type    | name | description                                |
|---------|------|--|
| VBD ref | vbd  | The vbd representing the CDROM-like device |
| VDI ref | vdi  | The new VDI to 'insert'                    |

**Return Type:** void

#### RPC name: plug

**Overview:** Hotplug the specified VBD, dynamically attaching it to the running VM.

#### Signature:

```
void plug (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description        |
|---------|------|--------------------|
| VBD ref | self | The VBD to hotplug |

**Return Type:** void**RPC name:** unplug**Overview:** Hot-unplug the specified VBD, dynamically unattaching it from the running VM.**Signature:**

```
void unplug (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description           |
|---------|------|-----------------------|
| VBD ref | self | The VBD to hot-unplug |

**Return Type:** void**RPC name:** get\_all**Overview:** Return a list of all the VBDs known to the system.**Signature:**

```
((VBD ref) Set) get_all (session_id s)
```

**Return Type:** (VBD ref) Set  
references to all objects**RPC name:** get\_uuid**Overview:** Get the uuid field of the given VBD.**Signature:**

```
string get_uuid (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** get\_VM**Overview:** Get the VM field of the given VBD.**Signature:**`(VM ref) get_VM (session_id s, VBD ref self)`**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** VM ref

value of the field

**RPC name:** get\_VDI**Overview:** Get the VDI field of the given VBD.**Signature:**`(VDI ref) get_VDI (session_id s, VBD ref self)`**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** VDI ref

value of the field

**RPC name:** get\_device**Overview:** Get the device field of the given VBD.**Signature:**`string get_device (session_id s, VBD ref self)`**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** set\_device**Overview:** Set the device field of the given VBD.**Signature:**`void set_device (session_id s, VBD ref self, string value)`



**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VBD ref | self  | reference to the object |
| string  | value | New value to set        |

**Return Type:** void**RPC name:** get\_bootable**Overview:** Get the bootable field of the given VBD.**Signature:**

```
bool get_bootable (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** bool

value of the field

**RPC name:** set\_bootable**Overview:** Set the bootable field of the given VBD.**Signature:**

```
void set_bootable (session_id s, VBD ref self, bool value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VBD ref | self  | reference to the object |
| bool    | value | New value to set        |

**Return Type:** void**RPC name:** get\_mode**Overview:** Get the mode field of the given VBD.**Signature:**

```
(vbd_mode) get_mode (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** vbd\_mode

value of the field

**RPC name: set\_mode****Overview:** Set the mode field of the given VBD.**Signature:**

```
void set_mode (session_id s, VBD ref self, vbd_mode value)
```

**Arguments:**

| type     | name  | description             |
|----------|-------|-------------------------|
| VBD ref  | self  | reference to the object |
| vbd_mode | value | New value to set        |

**Return Type:** void**RPC name: get\_type****Overview:** Get the type field of the given VBD.**Signature:**

```
(vbd_type) get_type (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** vbd\_type

value of the field

**RPC name: set\_type****Overview:** Set the type field of the given VBD.**Signature:**

```
void set_type (session_id s, VBD ref self, vbd_type value)
```

**Arguments:**

| type     | name  | description             |
|----------|-------|-------------------------|
| VBD ref  | self  | reference to the object |
| vbd_type | value | New value to set        |

**Return Type:** void**RPC name: get\_currently\_attached****Overview:** Get the currently\_attached field of the given VBD.**Signature:**

```
bool get_currently_attached (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** bool  
value of the field

**RPC name:** `get_status_code`

**Overview:** Get the `status_code` field of the given VBD.

**Signature:**

```
int get_status_code (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** int  
value of the field

**RPC name:** `get_status_detail`

**Overview:** Get the `status_detail` field of the given VBD.

**Signature:**

```
string get_status_detail (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** `get_runtime_properties`

**Overview:** Get the `runtime_properties` field of the given VBD.

**Signature:**

```
((string -> string) Map) get_runtime_properties (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name:** `get_qos_algorithm_type`**Overview:** Get the `qos/algorithm_type` field of the given VBD.**Signature:**

```
string get_qos_algorithm_type (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** `string`

value of the field

**RPC name:** `set_qos_algorithm_type`**Overview:** Set the `qos/algorithm_type` field of the given VBD.**Signature:**

```
void set_qos_algorithm_type (session_id s, VBD ref self, string value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VBD ref | self  | reference to the object |
| string  | value | New value to set        |

**Return Type:** `void`**RPC name:** `get_qos_algorithm_params`**Overview:** Get the `qos/algorithm_params` field of the given VBD.**Signature:**

```
((string -> string) Map) get_qos_algorithm_params (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** `(string → string) Map`

value of the field

**RPC name:** `set_qos_algorithm_params`**Overview:** Set the `qos/algorithm_params` field of the given VBD.**Signature:**

```
void set_qos_algorithm_params (session_id s, VBD ref self, (string -> string) Map value)
```

**Arguments:**

| type                  | name  | description             |
|-----------------------|-------|-------------------------|
| VBD ref               | self  | reference to the object |
| (string → string) Map | value | New value to set        |

**Return Type:** void**RPC name:** add\_to\_qos\_algorithm\_params**Overview:** Add the given key-value pair to the qos/algorithm\_params field of the given VBD.**Signature:**

```
void add_to_qos_algorithm_params (session_id s, VBD ref self, string key, string value)
```

**Arguments:**

| type    | name  | description             |
|---------|-------|-------------------------|
| VBD ref | self  | reference to the object |
| string  | key   | Key to add              |
| string  | value | Value to add            |

**Return Type:** void**RPC name:** remove\_from\_qos\_algorithm\_params**Overview:** Remove the given key and its corresponding value from the qos/algorithm\_params field of the given VBD. If the key is not in that Map, then do nothing.**Signature:**

```
void remove_from_qos_algorithm_params (session_id s, VBD ref self, string key)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |
| string  | key  | Key to remove           |

**Return Type:** void**RPC name:** get\_qos\_supported\_algorithms**Overview:** Get the qos/supported\_algorithms field of the given VBD.**Signature:**

```
(string Set) get_qos_supported_algorithms (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** string Set

value of the field

**RPC name:** `get_metrics`

**Overview:** Get the metrics field of the given VBD.

**Signature:**

```
(VBD_metrics ref) get_metrics (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** `VBD_metrics ref`

value of the field

**RPC name:** `create`

**Overview:** Create a new VBD instance, and return its handle.

**Signature:**

```
(VBD ref) create (session_id s, VBD record args)
```

**Arguments:**

| type       | name | description               |
|------------|------|---------------------------|
| VBD record | args | All constructor arguments |

**Return Type:** `VBD ref`

reference to the newly created object

**RPC name:** `destroy`

**Overview:** Destroy the specified VBD instance.

**Signature:**

```
void destroy (session_id s, VBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** `void`

**RPC name:** `get_by_uuid`

**Overview:** Get a reference to the VBD instance with the specified UUID.

**Signature:**

`(VBD ref) get_by_uuid (session_id s, string uuid)`

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** `VBD ref`

reference to the object

**RPC name:** `get_record`

**Overview:** Get a record containing the current state of the given VBD.

**Signature:**

`(VBD record) get_record (session_id s, VBD ref self)`

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| VBD ref | self | reference to the object |

**Return Type:** `VBD record`

all fields from the object

## 2.21 Class: VBD\_metrics

### 2.21.1 Fields for class: VBD\_metrics

| Name          | VBD_metrics  |          |   |
|---------------|--|----------|---|
| Description   | <i>The metrics associated with a virtual block device.</i> |          |   |
| Quals         | Field  | Type     | Description                                     |
| <i>RO_run</i> | <code>uuid</code>  | string   | unique identifier/object reference              |
| <i>RO_run</i> | <code>io/read_kbs</code>                                   | float    | Read bandwidth (KiB/s)                          |
| <i>RO_run</i> | <code>io/write_kbs</code>                                  | float    | Write bandwidth (KiB/s)                         |
| <i>RO_run</i> | <code>last_updated</code>                                  | datetime | Time at which this information was last updated |

### 2.21.2 RPCs associated with class: VBD\_metrics

**RPC name:** `get_all`

**Overview:** Return a list of all the VBD\_metrics instances known to the system.

**Signature:**

```
((VBD_metrics ref) Set) get_all (session_id s)
```

**Return Type:** (VBD\_metrics ref) Set

references to all objects

**RPC name:** `get_uuid`

**Overview:** Get the uuid field of the given VBD\_metrics.

**Signature:**

```
string get_uuid (session_id s, VBD_metrics ref self)
```

**Arguments:**

| type            | name | description             |
|-----------------|------|-------------------------|
| VBD_metrics ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `get_io_read_kbs`

**Overview:** Get the io/read\_kbs field of the given VBD\_metrics.

**Signature:**

```
float get_io_read_kbs (session_id s, VBD_metrics ref self)
```

**Arguments:**

| type            | name | description             |
|-----------------|------|-------------------------|
| VBD_metrics ref | self | reference to the object |

**Return Type:** float



value of the field

**RPC name:** `get_io_write_kbs`

**Overview:** Get the `io/write_kbs` field of the given `VBD_metrics`.

**Signature:**

```
float get_io_write_kbs (session_id s, VBD_metrics ref self)
```

**Arguments:**

| type                         | name              | description             |
|------------------------------|-------------------|-------------------------|
| <code>VBD_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `float`

value of the field

**RPC name:** `get_last_updated`

**Overview:** Get the `last_updated` field of the given `VBD_metrics`.

**Signature:**

```
datetime get_last_updated (session_id s, VBD_metrics ref self)
```

**Arguments:**

| type                         | name              | description             |
|------------------------------|-------------------|-------------------------|
| <code>VBD_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `datetime`

value of the field

**RPC name:** `get_by_uuid`

**Overview:** Get a reference to the `VBD_metrics` instance with the specified UUID.

**Signature:**

```
(VBD_metrics ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type                | name              | description              |
|---------------------|-------------------|--------------------------|
| <code>string</code> | <code>uuid</code> | UUID of object to return |

**Return Type:** `VBD_metrics ref`

reference to the object

**RPC name:** `get_record`

**Overview:** Get a record containing the current state of the given `VBD_metrics`.

**Signature:**

```
(VBD_metrics record) get_record (session_id s, VBD_metrics ref self)
```

**Arguments:**

| type                         | name              | description             |
|------------------------------|-------------------|-------------------------|
| <code>VBD_metrics ref</code> | <code>self</code> | reference to the object |

**Return Type:** `VBD_metrics record`

all fields from the object

## 2.22 Class: PBD

### 2.22.1 Fields for class: PBD

| Name                    | PBD   |                       |  |
|-------------------------|---|-----------------------|--|
| Description             | <i>The physical block devices through which hosts access SRs.</i> |                       |  |
| Quals                   | Field   | Type                  | Description  |
| <i>RO<sub>run</sub></i> | uuid  | string                | unique identifier/object reference   |
| <i>RO<sub>ins</sub></i> | host  | host ref              | physical machine on which the pbd is available                                 |
| <i>RO<sub>ins</sub></i> | SR  | SR ref                | the storage repository that the pbd realises                                   |
| <i>RO<sub>ins</sub></i> | device_config   | (string → string) Map | a config string to string map that is provided to the host's SR-backend-driver |
| <i>RO<sub>run</sub></i> | currently_attached  | bool                  | is the SR currently attached on this host?                                     |

### 2.22.2 RPCs associated with class: PBD

#### RPC name: get\_all

**Overview:** Return a list of all the PBDs known to the system.

**Signature:**

```
((PBD ref) Set) get_all (session_id s)
```

**Return Type:** (PBD ref) Set

references to all objects

#### RPC name: get\_uuid

**Overview:** Get the uuid field of the given PBD.

**Signature:**

```
string get_uuid (session_id s, PBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PBD ref | self | reference to the object |

**Return Type:** string

value of the field

#### RPC name: get\_host

**Overview:** Get the host field of the given PBD.

**Signature:**

```
(host ref) get_host (session_id s, PBD ref self)
```

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PBD ref | self | reference to the object |

**Return Type:** host ref  
value of the field

**RPC name:** get\_SR

**Overview:** Get the SR field of the given PBD.

**Signature:**

(SR ref) get\_SR (session\_id s, PBD ref self)

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PBD ref | self | reference to the object |

**Return Type:** SR ref  
value of the field

**RPC name:** get\_device\_config

**Overview:** Get the device\_config field of the given PBD.

**Signature:**

((string -> string) Map) get\_device\_config (session\_id s, PBD ref self)

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PBD ref | self | reference to the object |

**Return Type:** (string → string) Map  
value of the field

**RPC name:** get\_currently\_attached

**Overview:** Get the currently\_attached field of the given PBD.

**Signature:**

bool get\_currently\_attached (session\_id s, PBD ref self)

**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PBD ref | self | reference to the object |

**Return Type:** bool  
value of the field

**RPC name: create****Overview:** Create a new PBD instance, and return its handle.**Signature:**`(PBD ref) create (session_id s, PBD record args)`**Arguments:**

| type       | name | description               |
|------------|------|---------------------------|
| PBD record | args | All constructor arguments |

**Return Type:** PBD ref

reference to the newly created object

**RPC name: destroy****Overview:** Destroy the specified PBD instance.**Signature:**`void destroy (session_id s, PBD ref self)`**Arguments:**

| type    | name | description             |
|---------|------|-------------------------|
| PBD ref | self | reference to the object |

**Return Type:** void**RPC name: get\_by\_uuid****Overview:** Get a reference to the PBD instance with the specified UUID.**Signature:**`(PBD ref) get_by_uuid (session_id s, string uuid)`**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** PBD ref

reference to the object

**RPC name: get\_record****Overview:** Get a record containing the current state of the given PBD.**Signature:**`(PBD record) get_record (session_id s, PBD ref self)`

**Arguments:**

| <b>type</b> | <b>name</b> | <b>description</b>      |
|-------------|-------------|-------------------------|
| PBD ref     | self        | reference to the object |

**Return Type:** PBD record

all fields from the object

## 2.23 Class: crashdump

### 2.23.1 Fields for class: crashdump

|                         |                  |         |                                    |
|-------------------------|------------------|---------|------------------------------------|
| Name                    | <b>crashdump</b> |         |                                    |
| Description             | A VM crashdump.  |         |                                    |
| Quals                   | Field            | Type    | Description                        |
| <i>RO<sub>run</sub></i> | uuid             | string  | unique identifier/object reference |
| <i>RO<sub>ins</sub></i> | VM               | VM ref  | the virtual machine                |
| <i>RO<sub>ins</sub></i> | VDI              | VDI ref | the virtual disk                   |

### 2.23.2 RPCs associated with class: crashdump

**RPC name:** destroy

**Overview:** Destroy the specified crashdump.

**Signature:**

```
void destroy (session_id s, crashdump ref self)
```

**Arguments:**

| type          | name | description              |
|---------------|------|--------------------------|
| crashdump ref | self | The crashdump to destroy |

**Return Type:** void

**RPC name:** get\_all

**Overview:** Return a list of all the crashdumps known to the system.

**Signature:**

```
((crashdump ref) Set) get_all (session_id s)
```

**Return Type:** (crashdump ref) Set  
references to all objects

**RPC name:** get\_uuid

**Overview:** Get the uuid field of the given crashdump.

**Signature:**

```
string get_uuid (session_id s, crashdump ref self)
```

**Arguments:**

| type          | name | description             |
|---------------|------|-------------------------|
| crashdump ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** get\_VM**Overview:** Get the VM field of the given crashdump.**Signature:**

(VM ref) get\_VM (session\_id s, crashdump ref self)

**Arguments:**

| type          | name | description             |
|---------------|------|-------------------------|
| crashdump ref | self | reference to the object |

**Return Type:** VM ref

value of the field

**RPC name:** get\_VDI**Overview:** Get the VDI field of the given crashdump.**Signature:**

(VDI ref) get\_VDI (session\_id s, crashdump ref self)

**Arguments:**

| type          | name | description             |
|---------------|------|-------------------------|
| crashdump ref | self | reference to the object |

**Return Type:** VDI ref

value of the field

**RPC name:** get\_by\_uuid**Overview:** Get a reference to the crashdump instance with the specified UUID.**Signature:**

(crashdump ref) get\_by\_uuid (session\_id s, string uuid)

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** crashdump ref

reference to the object

**RPC name:** get\_record**Overview:** Get a record containing the current state of the given crashdump.**Signature:**

(crashdump record) get\_record (session\_id s, crashdump ref self)



**Arguments:**

| type          | name | description             |
|---------------|------|-------------------------|
| crashdump ref | self | reference to the object |

**Return Type:** crashdump record  
all fields from the object

## 2.24 Class: VTPM

### 2.24.1 Fields for class: VTPM

|                         |                              |        |   |
|-------------------------|------------------------------|--------|---|
| Name                    | <b>VTPM</b>                  |        |   |
| Description             | <i>A virtual TPM device.</i> |        |   |
| Quals                   | Field                        | Type   | Description                             |
| <i>RO<sub>run</sub></i> | uuid                         | string | unique identifier/object reference      |
| <i>RO<sub>ins</sub></i> | VM                           | VM ref | the virtual machine                     |
| <i>RO<sub>ins</sub></i> | backend                      | VM ref | the domain where the backend is located |

### 2.24.2 RPCs associated with class: VTPM

#### RPC name: get\_uuid

**Overview:** Get the uuid field of the given VTPM.

**Signature:**

```
string get_uuid (session_id s, VTPM ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| VTPM ref | self | reference to the object |

**Return Type:** string

value of the field

#### RPC name: get\_VM

**Overview:** Get the VM field of the given VTPM.

**Signature:**

```
(VM ref) get_VM (session_id s, VTPM ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| VTPM ref | self | reference to the object |

**Return Type:** VM ref

value of the field

#### RPC name: get\_backend

**Overview:** Get the backend field of the given VTPM.

**Signature:**

```
(VM ref) get_backend (session_id s, VTPM ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| VTPM ref | self | reference to the object |

**Return Type:** VM ref  
value of the field

**RPC name:** create

**Overview:** Create a new VTPM instance, and return its handle.

**Signature:**

```
(VTPM ref) create (session_id s, VTPM record args)
```

**Arguments:**

| type        | name | description               |
|-------------|------|---------------------------|
| VTPM record | args | All constructor arguments |

**Return Type:** VTPM ref  
reference to the newly created object

**RPC name:** destroy

**Overview:** Destroy the specified VTPM instance.

**Signature:**

```
void destroy (session_id s, VTPM ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| VTPM ref | self | reference to the object |

**Return Type:** void

**RPC name:** get\_by\_uuid

**Overview:** Get a reference to the VTPM instance with the specified UUID.

**Signature:**

```
(VTPM ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** VTPM ref  
reference to the object

**RPC name:** `get_record`

**Overview:** Get a record containing the current state of the given VTPM.

**Signature:**

`(VTPM record) get_record (session_id s, VTPM ref self)`

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| VTPM ref | self | reference to the object |

**Return Type:** VTPM record

all fields from the object

## 2.25 Class: console

### 2.25.1 Fields for class: console

|                         |                     |                       |                                      |
|-------------------------|---------------------|-----------------------|--------------------------------------|
| Name                    | <b>console</b>      |                       |                                      |
| Description             | <i>A console.</i>   |                       |                                      |
| Quals                   | Field               | Type                  | Description                          |
| <i>RO<sub>run</sub></i> | <b>uuid</b>         | string                | unique identifier/object reference   |
| <i>RO<sub>run</sub></i> | <b>protocol</b>     | console_protocol      | the protocol used by this console    |
| <i>RO<sub>run</sub></i> | <b>location</b>     | string                | URI for the console service          |
| <i>RO<sub>run</sub></i> | <b>VM</b>           | VM ref                | VM to which this console is attached |
| <i>RW</i>               | <b>other_config</b> | (string → string) Map | additional configuration             |

### 2.25.2 RPCs associated with class: console

#### RPC name: get\_all

**Overview:** Return a list of all the consoles known to the system.

**Signature:**

```
((console ref) Set) get_all (session_id s)
```

**Return Type:** (console ref) Set

references to all objects

#### RPC name: get\_uuid

**Overview:** Get the uuid field of the given console.

**Signature:**

```
string get_uuid (session_id s, console ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| console ref | self | reference to the object |

**Return Type:** string

value of the field

#### RPC name: get\_protocol

**Overview:** Get the protocol field of the given console.

**Signature:**

```
(console_protocol) get_protocol (session_id s, console ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| console ref | self | reference to the object |

**Return Type:** console\_protocol

value of the field

**RPC name:** `get_location`

**Overview:** Get the location field of the given console.

**Signature:**

```
string get_location (session_id s, console ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| console ref | self | reference to the object |

**Return Type:** string

value of the field

**RPC name:** `get_VM`

**Overview:** Get the VM field of the given console.

**Signature:**

```
(VM ref) get_VM (session_id s, console ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| console ref | self | reference to the object |

**Return Type:** VM ref

value of the field

**RPC name:** `get_other_config`

**Overview:** Get the other\_config field of the given console.

**Signature:**

```
((string -> string) Map) get_other_config (session_id s, console ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| console ref | self | reference to the object |

**Return Type:** (string → string) Map

value of the field

**RPC name: set\_other\_config****Overview:** Set the other\_config field of the given console.**Signature:**

```
void set_other_config (session_id s, console ref self, (string -> string) Map value)
```

**Arguments:**

| type                  | name  | description             |
|-----------------------|-------|-------------------------|
| console ref           | self  | reference to the object |
| (string → string) Map | value | New value to set        |

**Return Type:** void**RPC name: add\_to\_other\_config****Overview:** Add the given key-value pair to the other\_config field of the given console.**Signature:**

```
void add_to_other_config (session_id s, console ref self, string key, string value)
```

**Arguments:**

| type        | name  | description             |
|-------------|-------|-------------------------|
| console ref | self  | reference to the object |
| string      | key   | Key to add              |
| string      | value | Value to add            |

**Return Type:** void**RPC name: remove\_from\_other\_config****Overview:** Remove the given key and its corresponding value from the other\_config field of the given console. If the key is not in that Map, then do nothing.**Signature:**

```
void remove_from_other_config (session_id s, console ref self, string key)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| console ref | self | reference to the object |
| string      | key  | Key to remove           |

**Return Type:** void**RPC name: create****Overview:** Create a new console instance, and return its handle.**Signature:**

```
(console ref) create (session_id s, console record args)
```

**Arguments:**

| type           | name | description               |
|----------------|------|---------------------------|
| console record | args | All constructor arguments |

**Return Type:** console ref  
reference to the newly created object

**RPC name:** destroy

**Overview:** Destroy the specified console instance.

**Signature:**

```
void destroy (session_id s, console ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| console ref | self | reference to the object |

**Return Type:** void

**RPC name:** get\_by\_uuid

**Overview:** Get a reference to the console instance with the specified UUID.

**Signature:**

```
(console ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** console ref  
reference to the object

**RPC name:** get\_record

**Overview:** Get a record containing the current state of the given console.

**Signature:**

```
(console record) get_record (session_id s, console ref self)
```

**Arguments:**

| type        | name | description             |
|-------------|------|-------------------------|
| console ref | self | reference to the object |

**Return Type:** console record  
all fields from the object



## 2.26 Class: user

### 2.26.1 Fields for class: user

|                         |                              |        |                                    |
|-------------------------|------------------------------|--------|------------------------------------|
| Name                    | <b>user</b>                  |        |                                    |
| Description             | <i>A user of the system.</i> |        |                                    |
| Quals                   | Field                        | Type   | Description                        |
| <i>RO<sub>run</sub></i> | <b>uuid</b>                  | string | unique identifier/object reference |
| <i>RO<sub>ins</sub></i> | <b>short_name</b>            | string | short name (e.g. userid)           |
| <i>RW</i>               | <b>fullname</b>              | string | full name                          |

### 2.26.2 RPCs associated with class: user

#### RPC name: get\_uuid

**Overview:** Get the uuid field of the given user.

**Signature:**

```
string get_uuid (session_id s, user ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| user ref | self | reference to the object |

**Return Type:** string

value of the field

#### RPC name: get\_short\_name

**Overview:** Get the short\_name field of the given user.

**Signature:**

```
string get_short_name (session_id s, user ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| user ref | self | reference to the object |

**Return Type:** string

value of the field

#### RPC name: get\_fullname

**Overview:** Get the fullname field of the given user.

**Signature:**

```
string get_fullname (session_id s, user ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| user ref | self | reference to the object |

**Return Type:** string  
value of the field

**RPC name:** set\_fullname

**Overview:** Set the fullname field of the given user.

**Signature:**

```
void set_fullname (session_id s, user ref self, string value)
```

**Arguments:**

| type     | name  | description             |
|----------|-------|-------------------------|
| user ref | self  | reference to the object |
| string   | value | New value to set        |

**Return Type:** void

**RPC name:** create

**Overview:** Create a new user instance, and return its handle.

**Signature:**

```
(user ref) create (session_id s, user record args)
```

**Arguments:**

| type        | name | description               |
|-------------|------|---------------------------|
| user record | args | All constructor arguments |

**Return Type:** user ref  
reference to the newly created object

**RPC name:** destroy

**Overview:** Destroy the specified user instance.

**Signature:**

```
void destroy (session_id s, user ref self)
```

**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| user ref | self | reference to the object |

**Return Type:** void

**RPC name:** `get_by_uuid`**Overview:** Get a reference to the user instance with the specified UUID.**Signature:**`(user ref) get_by_uuid (session_id s, string uuid)`**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** `user ref`

reference to the object

**RPC name:** `get_record`**Overview:** Get a record containing the current state of the given user.**Signature:**`(user record) get_record (session_id s, user ref self)`**Arguments:**

| type     | name | description             |
|----------|------|-------------------------|
| user ref | self | reference to the object |

**Return Type:** `user record`

all fields from the object

## 2.27 Class: debug

### 2.27.1 Fields for class: debug

Class debug has no fields.

### 2.27.2 RPCs associated with class: debug

**RPC name:** `get_all`

**Overview:** Return a list of all the debug records known to the system

**Signature:**

```
((debug ref) Set) get_all (session_id s)
```

**Return Type:** (debug ref) Set

A list of all the IDs of all the debug records

**RPC name:** `return_failure`

**Overview:** Return an API 'successful' failure.

**Signature:**

```
void return_failure (session_id s)
```

**Return Type:** void

**RPC name:** `create`

**Overview:** Create a new debug instance, and return its handle.

**Signature:**

```
(debug ref) create (session_id s, debug record args)
```

**Arguments:**

| type         | name | description               |
|--------------|------|---------------------------|
| debug record | args | All constructor arguments |

**Return Type:** debug ref

reference to the newly created object

**RPC name:** `destroy`

**Overview:** Destroy the specified debug instance.

**Signature:**

```
void destroy (session_id s, debug ref self)
```

**Arguments:**

| type      | name | description             |
|-----------|------|-------------------------|
| debug ref | self | reference to the object |

**Return Type:** void

**RPC name:** get\_by\_uuid

**Overview:** Get a reference to the debug instance with the specified UUID.

**Signature:**

```
(debug ref) get_by_uuid (session_id s, string uuid)
```

**Arguments:**

| type   | name | description              |
|--------|------|--------------------------|
| string | uuid | UUID of object to return |

**Return Type:** debug ref  
reference to the object

**RPC name:** get\_record

**Overview:** Get a record containing the current state of the given debug.

**Signature:**

```
(debug record) get_record (session_id s, debug ref self)
```

**Arguments:**

| type      | name | description             |
|-----------|------|-------------------------|
| debug ref | self | reference to the object |

**Return Type:** debug record  
all fields from the object

## 2.28 Error Handling

When a low-level transport error occurs, or a request is malformed at the HTTP or XML-RPC level, the server may send an XML-RPC Fault response, or the client may simulate the same. The client must be prepared to handle these errors, though they may be treated as fatal. On the wire, these are transmitted in a form similar to this:

```
<methodResponse>
  <fault>
    <value>
      <struct>
        <member>
          <name>faultCode</name>
          <value><int>-1</int></value>
        </member>
        <member>
          <name>faultString</name>
          <value><string>Malformed request</string></value>
        </member>
      </struct>
    </value>
  </fault>
</methodResponse>
```

All other failures are reported with a more structured error response, to allow better automatic response to failures, proper internationalisation of any error message, and easier debugging. On the wire, these are transmitted like this:

```
<struct>
  <member>
    <name>Status</name>
    <value>Failure</value>
  </member>
  <member>
    <name>ErrorDescription</name>
    <value>
      <array>
        <data>
          <value>MAP_DUPLICATE_KEY</value>
          <value>Customer</value>
          <value>eSpeil Inc.</value>
          <value>eSpeil Incorporated</value>
        </data>
      </array>
    </value>
  </member>
</struct>
```

Note that `ErrorDescription` value is an array of string values. The first element of the array is an error code; the remainder of the array are strings representing error parameters relating to that code. In this case, the client has attempted to add the mapping `Customer` → `eSpeil Incorporated` to a Map, but it already contains the mapping `Customer` → `eSpeil Inc.`, and so the request has failed.

Each possible error code is documented in the following section.

### 2.28.1 Error Codes

#### **HANDLE\_INVALID**

You gave an invalid handle. The object may have recently been deleted. The class parameter gives the type of reference given, and the handle parameter echoes the bad value given.

**Signature:**

`HANDLE_INVALID(class, handle)`

---

#### **INTERNAL\_ERROR**

The server failed to handle your request, due to an internal error. The given message may give details useful for debugging the problem.

**Signature:**

`INTERNAL_ERROR(message)`

---

#### **MAP\_DUPLICATE\_KEY**

You tried to add a key-value pair to a map, but that key is already there. The key, current value, and the new value that you tried to set are all echoed.

**Signature:**

`MAP_DUPLICATE_KEY(key, current value, new value)`

---

#### **MESSAGE\_METHOD\_UNKNOWN**

You tried to call a method that does not exist. The method name that you used is echoed.

**Signature:**

`MESSAGE_METHOD_UNKNOWN(method)`

---

#### **MESSAGE\_PARAMETER\_COUNT\_MISMATCH**

You tried to call a method with the incorrect number of parameters. The fully-qualified method name that you used, and the number of received and expected parameters are returned.

**Signature:**

`MESSAGE_PARAMETER_COUNT_MISMATCH(method, expected, received)`

---

**NETWORK\_ALREADY\_CONNECTED**

You tried to create a PIF, but the network you tried to attach it to is already attached to some other PIF, and so the creation failed.

**Signature:**

NETWORK\_ALREADY\_CONNECTED(network, connected PIF)

---

**OPERATION\_NOT\_ALLOWED**

You attempted an operation that was not allowed.

No parameters.

---

**PIF\_IS\_PHYSICAL**

You tried to destroy a PIF, but it represents an aspect of the physical host configuration, and so cannot be destroyed. The parameter echoes the PIF handle you gave.

**Signature:**

PIF\_IS\_PHYSICAL(PIF)

---

**SESSION\_AUTHENTICATION\_FAILED**

The credentials given by the user are incorrect, so access has been denied, and you have not been issued a session handle.

No parameters.

---

**SESSION\_INVALID**

You gave an invalid session handle. It may have been invalidated by a server restart, or timed out. You should get a new session handle, using one of the session.login\_ calls. This error does not invalidate the current connection. The handle parameter echoes the bad value given.

**Signature:**

SESSION\_INVALID(handle)

---

**SESSION\_NOT\_REGISTERED**

This session is not registered to receive events. You must call event.register before event.next. The session handle you are using is echoed.

**Signature:**

SESSION\_NOT\_REGISTERED(handle)

---



**VALUE\_NOT\_SUPPORTED**

You attempted to set a value that is not supported by this implementation. The fully-qualified field name and the value that you tried to set are returned. Also returned is a developer-only diagnostic reason.

**Signature:**

```
VALUE_NOT_SUPPORTED(field, value, reason)
```

---

**VLAN\_TAG\_INVALID**

You tried to create a VLAN, but the tag you gave was invalid – it must be between 0 and 4095. The parameter echoes the VLAN tag you gave.

**Signature:**

```
VLAN_TAG_INVALID(VLAN)
```

---

**VM\_BAD\_POWER\_STATE**

You attempted an operation on a VM that was not in an appropriate power state at the time; for example, you attempted to start a VM that was already running. The parameters returned are the VM's handle, and the expected and actual VM state at the time of the call.

**Signature:**

```
VM_BAD_POWER_STATE(vm, expected, actual)
```

---

**VM\_HVM\_REQUIRED**

HVM is required for this operation

**Signature:**

```
VM_HVM_REQUIRED(vm)
```

---

## Chapter 3

# GNU Free Documentation License

Version 1.2, November 2002  
Copyright ©2000,2001,2002 Free Software Foundation, Inc.

51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

### Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

## 1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "**Document**", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "**you**". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "**Modified Version**" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "**Secondary Section**" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The **”Invariant Sections”** are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The **”Cover Texts”** are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A **”Transparent”** copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not **”Transparent”** is called **”Opaque”**.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The **”Title Page”** means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, **”Title Page”** means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

A section **”Entitled XYZ”** means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as **”Acknowledgements”**, **”Dedications”**, **”Endorsements”**, or **”History”**.) To **”Preserve the Title”** of such a section when you modify the Document means that it remains a section **”Entitled XYZ”** according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

## 2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

## 3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts,

you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages. If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

## 4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.

- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles. You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

## 5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

## 6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

## 7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

## 8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

## 9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

## 10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

## **ADDENDUM: How to use this License for your documents**

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

Copyright ©YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.