Citrix Hypervisor 8.2 Cumulative Update 1 Management API Reference [XAPI]

December 3, 2021

API Basics

This document defines the Citrix Hypervisor 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 (RPCs). There are two supported wire formats, one based upon XML-RPC and one based upon JSON-RPC (v1.0 and v2.0 are both recognised). 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/runtime`: the field is Read Only. Furthermore, its value is automatically computed at runtime. For example, current CPU load and disk IO throughput.
- `RO/constructor`: 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.

Types

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

- `string`: Text strings.
- `int`: 64-bit integers.
- `float`: IEEE double-precision floating-point numbers.
- `bool`: Boolean.
- `datetime`: Date and timestamp.
- `c ref`: Reference to an object of class `c`.
- `t set`: Arbitrary-length set of values of type `t`.
- `(k -> v) map`: Mapping from values of type `k` to values of type `v`.
- `e enum`: Enumeration type with name `e`. Enums are defined in the API reference together with classes that use them.

Note that there are a number of cases where `ref`s are *doubly linked*. For example, 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 lists explicitly 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).

RPCs associated with fields

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

- `get_f (r)`: takes a `ref`, `r` that refers to an object and returns the value of `f`.

Each field, `f`, with qualifier `RW` and whose outermost type is `set` has the following additional RPCs associated with it:

- `add_f(r, v)`: adds a new element `v` to the set. Note that sets cannot contain duplicate values, hence this operation has no action in the case that `v` is already in the set.
- `remove_f(r, v)`: removes element `v` from the set.

Each field, `f`, with qualifier `RW` and whose outermost type is `map` has the following additional RPCs associated with it:

- `add_to_f(r, k, v)`: adds new pair `k -> v` to the mapping stored in `f` in object`r`. Attempting to add a new pair for duplicate key, `k`, fails with a `MAP_DUPLICATE_KEY` error.
- `remove_from_f(r, k)`: removes the pair with key `k` from the mapping stored in `f` in object `r`.

Each field whose outermost type is neither `set` nor `map`, but whose qualifier is `RW` has an RPC accessor associated with it that sets its value:

- `set_f(r, v)`: sets the field `f` on object `r` to value `v`.

RPCs associated with classes

- Most classes have a *constructor* RPC named `create` that takes as parameters all fields marked `RW` and `RO/constructor`. 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_label)` RPC that returns a set of objects of that class that have the specified `name_label`.
- Most classes have a `destroy(r)` RPC that explicitly deletes the persistent object specified by `r` 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.

Apart from 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.

Wire Protocol for Remote API Calls

API calls are sent over a network to a Xen-enabled host using an RPC protocol. Here we describe how the higher-level types used in our API Reference are mapped to primitive RPC types, covering the two supported wire formats XML-RPC and JSON-RPC.

XML-RPC Protocol

We specify the signatures of API functions in the following style:

```
(VM ref set)  VM.get_all()
```

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

- the types `float`, `bool`, `datetime`, and `string` 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`).
- `int` is assumed to be 64-bit in our API and is 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, the value `destroy` of `enum 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 `string 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 -> float) 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.

XML-RPC Return Values and 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 the `Status` is `Success` then the struct contains a second element named `Value`:

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

If the `Status` is `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 rest of the elements are strings representing error parameters relating to that code.

For example, an XML-RPC return value from the `host.get_resident_VMs` function 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>
```

JSON-RPC Protocol

We specify the signatures of API functions in the following style:

```
(VM ref set)  VM.get_all()
```

This specifies that the function with name `VM.get_all` takes no parameters and returns a `set` of `VM ref`. These types are mapped onto JSON-RPC types in the following manner:

- the types `float` and `bool` map directly to the JSON types `number` and `boolean`, while `datetime` and `string` are represented as the JSON `string` type.
- all `ref` types are opaque references, encoded as the JSON `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 JSON `string` type. The string itself is the OSF DCE UUID presentation format (as output by `uuidgen`).
- `int` is assumed to be 64-bit in our API and is encoded as a JSON `number` without decimal point or exponent, preserved as a string.
- values of `enum` types are encoded as the JSON `string` type. For example, the value `destroy` of `enum on_normal_exit`, would be conveyed as:

```
"destroy"
```

- for all our types, `t`, our type `t set` simply maps to the JSON `array` type, so, for example, a value of type `string set` would be transmitted like this:

```
[ "CX8", "PSE36", "FPU" ]
```

- for types `k` and `v`, our type `(k -> v) map` maps onto a JSON object which contains members with name `k` and value `v`. 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 -> float) map` containing the mappings *Mike -> 2.3* and *John -> 1.2* would be represented as:

```
{
  "Mike": 2.3,
  "John": 1.2
}
```

- our `void` type is transmitted as an empty string.

Both versions 1.0 and 2.0 of the JSON-RPC wire format are recognised and, depending on your client library, you can use either of them.

JSON-RPC v1.0

JSON-RPC v1.0 Requests

An API call is represented by sending a single JSON object to the server, which contains the members `method`, `params`, and `id`.

- `method`: A JSON `string` containing the name of the function to be invoked.
- `params`: A JSON `array` of values, which represents the parameters of the function to be invoked.
- `id`: A JSON `string` or `integer` representing the call id. Note that, diverging from the JSON-RPC v1.0 specification the API does not accept *notification* requests (requests without responses), i.e. the id cannot be `null`.

For example, a JSON-RPC v1.0 request to retrieve the resident VMs of a host may look like this:

```
{
  "method": "host.get_resident_VMs",
  "params": [
    "OpaqueRef:74f1a19cd-b660-41e3-a163-10f03e0eae67",
    "OpaqueRef:08c34fc9-f418-4f09-8274-b9cb25cd8550"
  ],
  "id": "xyz"
}
```

In the above example, the first element of the `params` array is the reference of the open session to the host, while the second is the host reference.

JSON-RPC v1.0 Return Values

The return value of a JSON-RPC v1.0 call is a single JSON object containing the members `result`, `error`, and `id`.

- `result`: If the call is successful, it is a JSON value (`string`, `array` etc.) representing the return value of the invoked function. If an error has occurred, it is `null`.
- `error`: If the call is successful, it is `null`. If the call has failed, it a JSON `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.
- `id`: The call id. It is a JSON `string` or `integer` and it is the same id as the request it is responding to.

For example, a JSON-RPC v1.0 return value from the `host.get_resident_VMs` function may look like this:

```
{
  "result": [
```

```
        "OpaqueRef:604f51e7-630f-4412-83fa-b11c6cf008ab",
        "OpaqueRef:670d08f5-cbeb-4336-8420-ccd56390a65f"
    ],
    "error": null,
    "id": "xyz"
}
```

while the return value of the same call made on a logged out session may look like this:

```
{
    "result": null,
    "error": [
        "SESSION_INVALID",
        "OpaqueRef:93f1a23cd-a640-41e3-b163-10f86e0eae67"
    ],
    "id": "xyz"
}
```

JSON-RPC v2.0

JSON-RPC v2.0 Requests

An API call is represented by sending a single JSON object to the server, which contains the members `jsonrpc`, `method`, `params`, and `id`.

- `jsonrpc`: A JSON `string` specifying the version of the JSON-RPC protocol. It is exactly "2.0".
- `method`: A JSON `string` containing the name of the function to be invoked.
- `params`: A JSON `array` of values, which represents the parameters of the function to be invoked. Although the JSON-RPC v2.0 specification allows this member to be ommitted, in practice all API calls accept at least one parameter.
- `id`: A JSON `string` or `integer` representing the call id. Note that, diverging from the JSON-RPC v2.0 specification it cannot be null. Neither can it be ommitted because the API does not accept *notification* requests (requests without responses).

For example, a JSON-RPC v2.0 request to retrieve the VMs resident on a host may may look like this:

```
{
    "jsonrpc": "2.0",
    "method": "host.get_resident_VMs",
    "params": [
      "OpaqueRef:c90cd28f-37ec-4dbf-88e6-f697ccb28b39",
      "OpaqueRef:08c34fc9-f418-4f09-8274-b9cb25cd8550"
    ],
    "id": 3
}
```

As before, the first element of the `parameter` array is the reference of the open session to the host, while the second is the host reference.

JSON-RPC v2.0 Return Values

The return value of a JSON-RPC v2.0 call is a single JSON object containing the members `jsonrpc`, either `result` or `error` depending on the outcome of the call, and `id`.

- `jsonrpc`: A JSON `string` specifying the version of the JSON-RPC protocol. It is exactly "2.0".
- `result`: If the call is successful, it is a JSON value (`string`, `array` etc.) representing the return value of the invoked function. If an error has occurred, it does not exist.
- `error`: If the call is successful, it does not exist. If the call has failed, it is a single structured JSON object (see below).
- `id`: The call id. It is a JSON `string` or `integer` and it is the same id as the request it is responding to.

The `error` object contains the members `code`, `message`, and `data`.

- `code`: The API does not make use of this member and only retains it for compliance with the JSON-RPC v2.0 specification. It is a JSON `integer` which has a non-zero value.
- `message`: A JSON `string` representing an API error code.
- `data`: A JSON array of `string` values representing error parameters relating to the aforementioned API error code.

For example, a JSON-RPC v2.0 return value from the `host.get_resident_VMs` function may look like this:

```
{
  "jsonrpc": "2.0",
  "result": [
      "OpaqueRef:604f51e7-630f-4412-83fa-b11c6cf008ab",
      "OpaqueRef:670d08f5-cbeb-4336-8420-ccd56390a65f"
  ],
  "id": 3
}
```

while the return value of the same call made on a logged out session may look like this:

```
{
  "jsonrpc": "2.0",
  "error": {
      "code": 1,
      "message": "SESSION_INVALID",
      "data": [
          "OpaqueRef:c90cd28f-37ec-4dbf-88e6-f697ccb28b39"
      ]
  },
  "id": 3
}
```

Note on References vs UUIDs

References are opaque types - encoded as XML-RPC and JSON-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 and returns an opaque reference to the server-side object that has that UUID;
- A `get_uuid` function (a regular "field getter" RPC) that takes an opaque reference and returns the UUID of the server-side object that is referenced by it.

Making RPC Calls

Transport Layer

The following transport layers are currently supported:

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

Session Layer

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

```
(session ref) session.login_with_password(string uname, string pwd,
                string version, string originator)
```

where `uname` and `password` refer to your username and password, as defined by the Xen administrator, while `version` and `originator` are optional. The `session ref` returned by `session.login_with_password` is passed to subequent RPC calls as an authentication token. Note that a session reference obtained by a login request to the XML-RPC backend can be used in subsequent requests to the JSON-RPC backend, and vice-versa.

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

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

Synchronous and Asynchronous Invocation

Each method call (apart from methods on the `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 above.

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

Instead of returning its result directly, an asynchronous RPC call returns an identifier of type `task ref` which is subsequently used to track the status of a running asynchronous RPC.

Note that an asychronous call may fail immediately, before a task has even been created. When using the XML-RPC wire protocol, this eventuality is represented by wrapping the returned `task ref` in an XML-RPC struct with a `Status`, `ErrorDescription`, and `Value` fields, exactly as specified above; the `task ref` is provided in the `Value` field if `Status` is set to `Success`. When using the JSON-RPC protocol, the `task ref` is wrapped in a response JSON object as specified above and it is provided by the value of the `result` member of a successful call.

The RPC call

```
(task ref set)  Task.get_all(session ref session_id)
```

returns a set of all task identifiers known to the system. The status (including any returned result and error codes) of these 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.

Example interactive session

This section describes how an interactive session might look, using python XML-RPC and JSON-RPC client libraries.

First, initialise python:

```
$ python2.7
>>>
```

Using the XML-RPC Protocol

Import the library `xmlrpclib` and create a python object referencing the remote server as shown below:

```
>>> import xmlrpclib
>>> xen = xmlrpclib.Server("https://localhost:443")
```

Acquire a session reference by logging in with a username and password; the session reference is returned under the key `Value` in the resulting dictionary (error-handling ommitted for brevity):

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

This is what the call looks like when serialised

```
<?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>
        <param><value><string>version</string></value></param>
        <param><value><string>originator</string></value></param>
    </params>
</methodCall>
```

Next, the user may acquire a list of all the VMs known to the system (note the call takes the session reference as the only parameter):

```
>>> all_vms = xen.VM.get_all(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 reality) and you should treat them as opaque strings. *Templates* are VMs with the `is_a_template` field set to `true`. We can find the subset of template VMs using a command like the following:

```
>>> all_templates = filter(lambda x: xen.VM.get_is_a_template(session, x)['Value'],
                           all_vms)
```

Once a reference to a VM has been acquired, a lifecycle operation may be invoked:

```
>>> xen.VM.start(session, all_templates[0], False, False)
{'Status': 'Failure', 'ErrorDescription': ['VM_IS_TEMPLATE', 'OpaqueRef:X']}
```

In this case the `start` message has been rejected, because the VM is a template, 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.

Rather than querying fields individually, whole *records* may be returned at once. To retrieve the record of a single object as a python dictionary:

```
>>> record = xen.VM.get_record(session, all_templates[0])['Value']
>>> record['power_state']
```

```
'Halted'
>>> record['name_label']
'Windows 10 (64-bit)'
```

To retrieve all the VM records in a single call:

```
>>> records = xen.VM.get_all_records(session)['Value']
>>> records.keys()
['OpaqueRef:1', 'OpaqueRef:2', 'OpaqueRef:3', 'OpaqueRef:4' ]
>>> records['OpaqueRef:1']['name_label']
'Red Hat Enterprise Linux 7'
```

Using the JSON-RPC Protocol

For this example we are making use of the package `python-jsonrpc` due to its simplicity, although other packages can also be used.

First, import the library `pyjsonrpc` and create the object referencing the remote server as follows:

```
>>> import pyjsonrpc
>>> client = pyjsonrpc.HttpClient(url = "https://localhost/jsonrpc:443")
```

Acquire a session reference by logging in with a username and password; the library `pyjsonrpc` returns the response's `result` member, which is the session reference:

```
>>> session = client.call("session.login_with_password",
...                       "user", "passwd", "version", "originator")
```

`pyjsonrpc` uses the JSON-RPC protocol v2.0, so this is what the serialised request looks like:

```
{
  "jsonrpc": "2.0",
  "method": "session.login_with_password",
  "params": ["user", "passwd", "version", "originator"],
  "id": 0
}
```

Next, the user may acquire a list of all the VMs known to the system (note the call takes the session reference as the only parameter):

```
>>> all_vms = client.call("VM.get_all", session)
>>> 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 reality) and you should treat them as opaque strings. *Templates* are VMs with the `is_a_template` field set to `true`. We can find the subset of template VMs using a command like the following:

```
>>> all_templates = filter(
...     lambda x: client.call("VM.get_is_a_template", session, x),
        all_vms)
```

Once a reference to a VM has been acquired, a lifecycle operation may be invoked:

```
>>> from pyjsonrpc import JsonRpcError
>>> try:
...     client.call("VM.start", session, all_templates[0], False, False)
... except JsonRpcError as e:
...     e.message
...     e.data
...
'VM_IS_TEMPLATE'
[ 'OpaqueRef:1', 'start' ]
```

In this case the `start` message has been rejected because the VM is a template, hence an error response has been returned. These high-level errors are returned as structured data, allowing them to be internationalised.

Rather than querying fields individually, whole *records* may be returned at once. To retrieve the record of a single object as a python dictionary:

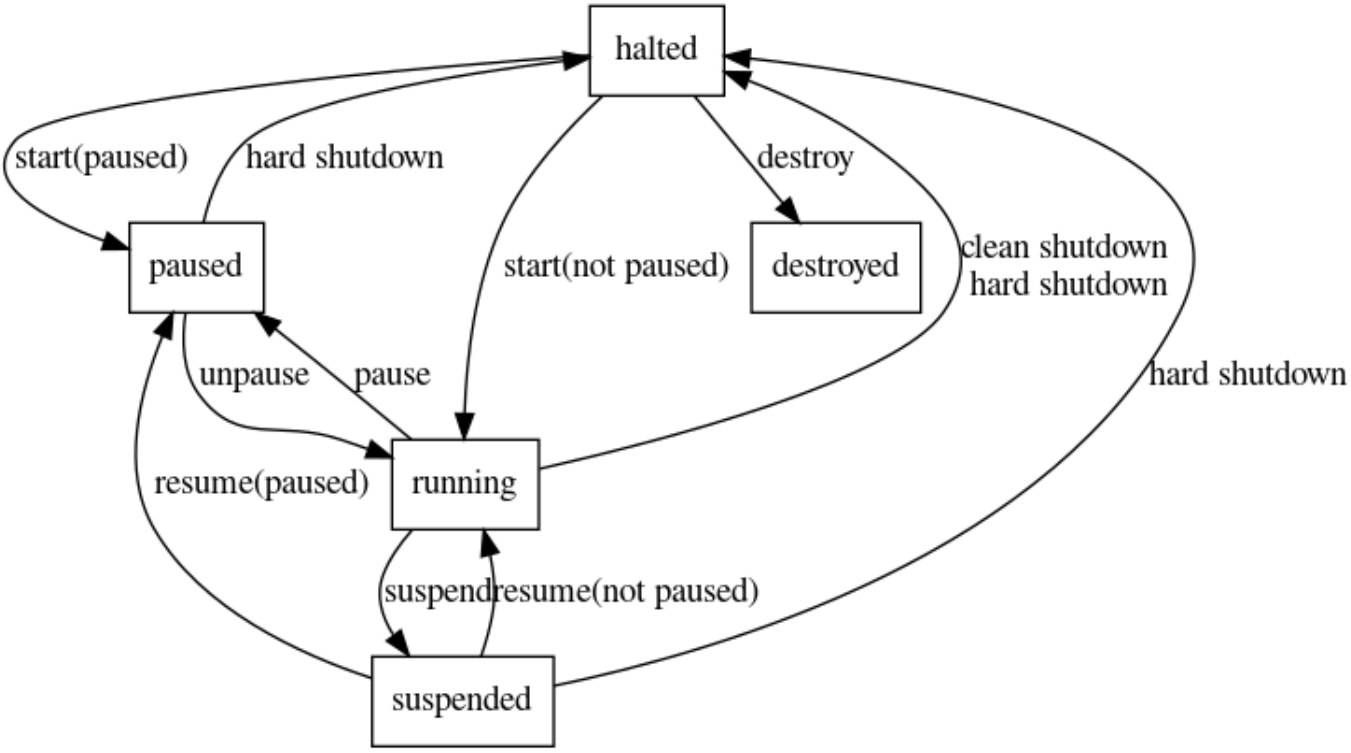
```
>>> record = client.call("VM.get_record", session, all_templates[0])
>>> record['power_state']
'Halted'
>>> record['name_label']
'Windows 10 (64-bit)'
```

To retrieve all the VM records in a single call:

```
>>> records = client.call("VM.get_all_records", session)
>>> records.keys()
['OpaqueRef:1', 'OpaqueRef:2', 'OpaqueRef:3', 'OpaqueRef:4' ]
>>> records['OpaqueRef:1']['name_label']
'Red Hat Enterprise Linux 7'
```

VM Lifecycle

The following diagram shows the states that a VM can be in and the API calls that can be used to move the VM between these states.



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 an empty string, then paravirtual domain building and booting will be used; otherwise the VM will be loaded as a 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` were 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. Optionally `HVM_boot_params` can contain another key-value pair "firmware" with values "bios" or "uefi" (default is "bios" if absent). By default Secure Boot is not enabled, it can be enabled when "uefi" is enabled by setting `VM.platform["secureboot"]` to true.

API Reference - Types and Classes

Classes

The following classes are defined:

Name	Description
auth	Management of remote authentication services
blob	A placeholder for a binary blob
Bond	
Certificate	Description
Cluster	Cluster-wide Cluster metadata
Cluster_host	Cluster member metadata
console	A console
crashdump	**Deprecated**. A VM crashdump
data_source	Data sources for logging in RRDs
DR_task	DR task
event	Asynchronous event registration and handling
Feature	A new piece of functionality
GPU_group	A group of compatible GPUs across the resource pool
host	A physical host
host_cpu	**Deprecated**. A physical CPU
host_crashdump	Represents a host crash dump
host_metrics	The metrics associated with a host
host_patch	**Deprecated**. Represents a patch stored on a server
LVHD	LVHD SR specific operations
message	An message for the attention of the administrator
network	A virtual network
network_sriov	network-sriov which connects logical pif and physical pif
PBD	The physical block devices through which hosts access SRs
PCI	A PCI device
PGPU	A physical GPU (pGPU)
PIF	A physical network interface (note separate VLANs are represented as several PIFs)

Name	Description
PIF_metrics	The metrics associated with a physical network interface
pool	Pool-wide information
pool_patch	**Deprecated**. Pool-wide patches
pool_update	Pool-wide updates to the host software
probe_result	A set of properties that describe one result element of SR.probe. Result elements and properties can change dynamically based on changes to the the SR.probe input-parameters or the target.
PUSB	A physical USB device
PVS_cache_storage	Describes the storage that is available to a PVS site for caching purposes
PVS_proxy	a proxy connects a VM/VIF with a PVS site
PVS_server	individual machine serving provisioning (block) data
PVS_site	machines serving blocks of data for provisioning VMs
role	A set of permissions associated with a subject
SDN_controller	Describes the SDN controller that is to connect with the pool
secret	A secret
session	A session
SM	A storage manager plugin
SR	A storage repository
sr_stat	A set of high-level properties associated with an SR.
subject	A user or group that can log in xapi
task	A long-running asynchronous task
tunnel	A tunnel for network traffic
USB_group	A group of compatible USBs across the resource pool
user	**Deprecated**. A user of the system
VBD	A virtual block device
VBD_metrics	**Removed**. The metrics associated with a virtual block device
VDI	A virtual disk image
vdi_nbd_server_info	Details for connecting to a VDI using the Network Block Device protocol
VGPU	A virtual GPU (vGPU)
VGPU_type	A type of virtual GPU
VIF	A virtual network interface
VIF_metrics	**Removed**. The metrics associated with a virtual network device

Name	Description
VLAN	A VLAN mux/demux
VM	A virtual machine (or 'guest').
VM_appliance	VM appliance
VM_guest_metrics	The metrics reported by the guest (as opposed to inferred from outside)
VM_metrics	The metrics associated with a VM
VMPP	**Removed**. VM Protection Policy
VMSS	VM Snapshot Schedule
VTPM	A virtual TPM device
VUSB	Describes the vusb device

Relationships Between Classes

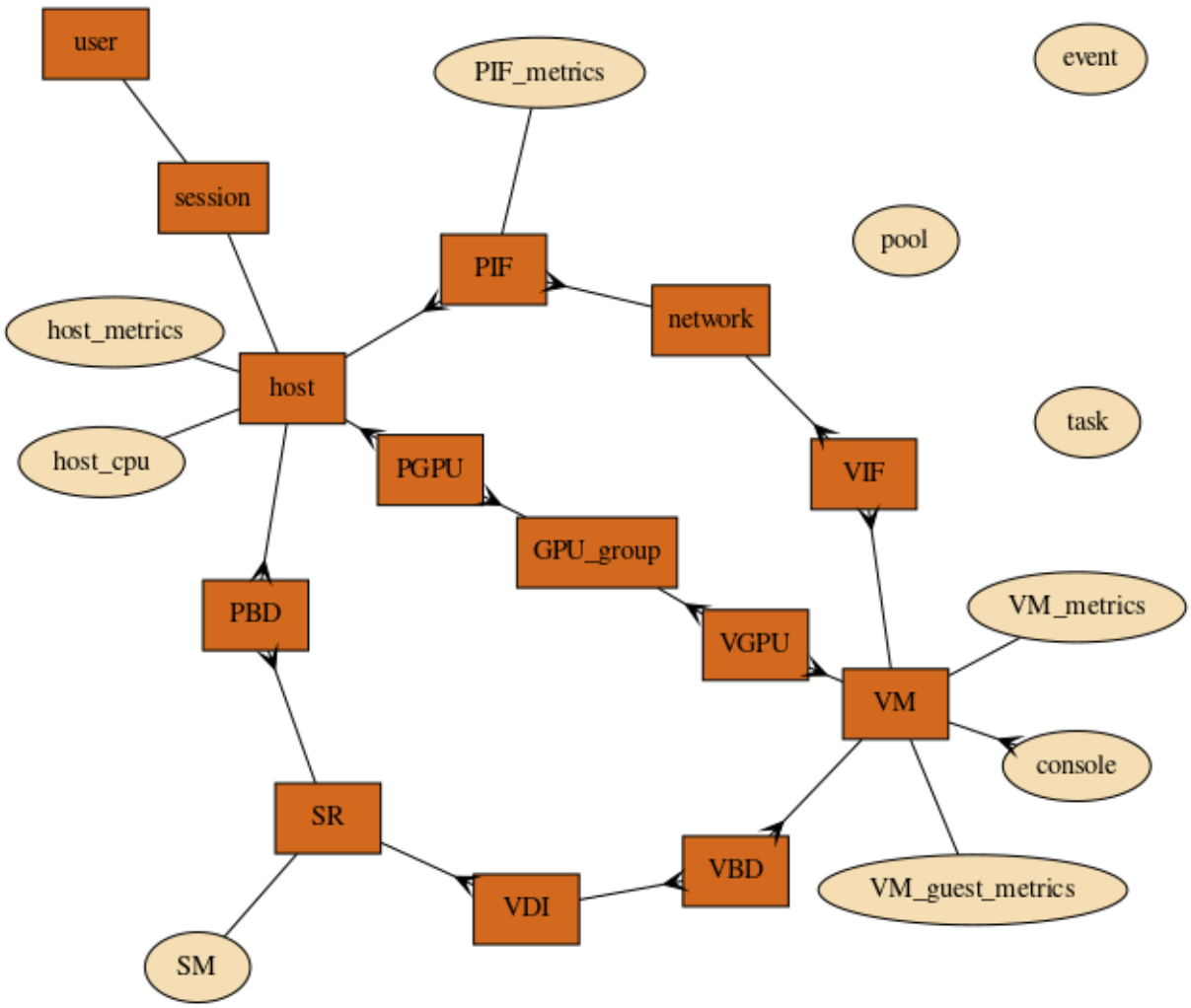
Fields that are bound together are shown in the following table:

object.field	*object.field*	*relationship*
VM.snapshot_of	VM.snapshots	one-to-many
VDI.snapshot_of	VDI.snapshots	one-to-many
VM.parent	VM.children	one-to-many
task.subtask_of	task.subtasks	one-to-many
PIF.bond_slave_of	Bond.slaves	one-to-many
Bond.master	PIF.bond_master_of	one-to-many
VLAN.tagged_PIF	PIF.VLAN_slave_of	one-to-many
tunnel.access_PIF	PIF.tunnel_access_PIF_of	one-to-many
tunnel.transport_PIF	PIF.tunnel_transport_PIF_of	one-to-many
PBD.host	host.PBDs	one-to-many
PBD.SR	SR.PBDs	one-to-many
VBD.VDI	VDI.VBDs	one-to-many
crashdump.VDI	VDI.crash_dumps	one-to-many
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
Cluster_host.cluster	Cluster.cluster_hosts	one-to-many

object.field	*object.field*	*relationship*
PIF.host	host.PIFs	one-to-many
PIF.network	network.PIFs	one-to-many
VDI.SR	SR.VDIs	one-to-many
VTPM.VM	VM.VTPMs	one-to-many
console.VM	VM.consoles	one-to-many
VM.resident_on	host.resident_VMs	one-to-many
host_cpu.host	host.host_CPUs	one-to-many
host_crashdump.host	host.crashdumps	one-to-many
host_patch.host	host.patches	one-to-many
host_patch.pool_patch	pool_patch.host_patches	one-to-many
host.updates	pool_update.hosts	many-to-many
subject.roles	subject.roles	unknown type
role.subroles	role.subroles	many-to-many
VM.protection_policy	VMPP.VMs	one-to-many
VM.snapshot_schedule	VMSS.VMs	one-to-many
VM.appliance	VM_appliance.VMs	one-to-many
PGPU.GPU_group	GPU_group.PGPUs	one-to-many
VGPU.GPU_group	GPU_group.VGPUs	one-to-many
VGPU.type	VGPU_type.VGPUs	one-to-many
VGPU.VM	VM.VGPUs	one-to-many
VGPU.resident_on	PGPU.resident_VGPUs	one-to-many
PGPU.supported_VGPU_types	VGPU_type.supported_on_PGPUs	many-to-many
PGPU.enabled_VGPU_types	VGPU_type.enabled_on_PGPUs	many-to-many
GPU_group.supported_VGPU_types	VGPU_type.supported_on_GPU_groups	many-to-many
GPU_group.enabled_VGPU_types	VGPU_type.enabled_on_GPU_groups	many-to-many
PCI.host	host.PCIs	one-to-many
PGPU.host	host.PGPUs	one-to-many
VDI.metadata_of_pool	pool.metadata_VDIs	one-to-many
SR.introduced_by	DR_task.introduced_SRs	one-to-many
PVS_server.site	PVS_site.servers	one-to-many
PVS_proxy.site	PVS_site.proxies	one-to-many

object.field	*object.field*	*relationship*
PVS_cache_storage.site	PVS_site.cache_storage	one-to-many
PUSB.host	host.PUSBs	one-to-many
PUSB.USB_group	USB_group.PUSBs	one-to-many
VUSB.USB_group	USB_group.VUSBs	one-to-many
VUSB.VM	VM.VUSBs	one-to-many
Feature.host	host.features	one-to-many
network_sriov.physical_PIF	PIF.sriov_physical_PIF_of	one-to-many
network_sriov.logical_PIF	PIF.sriov_logical_PIF_of	one-to-many
Certificate.host	host.certificates	one-to-many

The following figure represents bound fields (as specified above) diagramatically, using crow's foot notation to specify one-to-one, one-to-many or many-to-many relationships:



Types

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

Higher-order types

The following type constructors are used:

Type	Description
c ref	reference to an object of class *c*
t set	a set of elements of type *t*
(*a* -> *b*) map	a table mapping values of type *a* to values of type *b*

Enumeration types

The following enumeration types are used:

`restartHost` This patch requires the host to be restarted once applied.
`restartHVM` This patch requires HVM guests to be restarted once applied.
`restartPV` This patch requires PV guests to be restarted once applied.
`restartXAPI` This patch requires XAPI to be restarted once applied.

`breadth_first` vGPUs of a given type are allocated evenly across supporting pGPUs.
`depth_first` vGPUs of a given type are allocated on supporting pGPUs until they are full.

`active-backup` Active/passive bonding: only one NIC is carrying traffic
`balance-slb` Source-level balancing
`lacp` Link aggregation control protocol

`Host` Host
`Pool` Pool
`PVS_proxy` PVS_proxy

SR	SR
VDI	VDI
VM	VM
VMPP	VMPP
VMSS	VMSS

destroy	completely destroying a cluster host
disable	disabling cluster membership on a particular host
enable	enabling cluster membership on a particular host

add	adding a new member to the cluster
destroy	completely destroying a cluster
disable	disabling any cluster member
enable	enabling any cluster member
remove	removing a member from the cluster

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

hvm	HVM; Fully Virtualised
pv	PV: Paravirtualised
pv_in_pvh	PV inside a PVH container
unspecified	Not specified or unknown domain type

add	An object has been created
del	An object has been deleted
mod	An object has been modified

evacuate	Indicates this host is evacuating
power_on	Indicates this host is in the process of being powered on

provision	Indicates this host is able to provision another VM
reboot	Indicates this host is in the process of rebooting
shutdown	Indicates this host is in the process of shutting itself down
vm_migrate	This host is the migration target of a VM
vm_resume	This host is resuming a VM
vm_start	This host is starting a VM

disable_on_reboot	The host will stop outputting its console to a physical display device on next boot
disabled	This host is not outputting its console to a physical display device
enable_on_reboot	The host will start outputting its console to a physical display device on next boot
enabled	This host is outputting its console to a physical display device

DHCP	Acquire an IP address by DHCP
None	Do not acquire an IP address
Static	Static IP address configuration

Autoconf	Router assigned prefix delegation IPv6 allocation
DHCP	Acquire an IPv6 address by DHCP
None	Do not acquire an IPv6 address
Static	Static IPv6 address configuration

ok	There is no applicable live patch
ok_livepatch_complete	An applicable live patch exists for every required component
ok_livepatch_incomplete	An applicable live patch exists but it is not sufficient

disabled	Treat all VIFs on this network with locking_mode = 'default' as if they have locking_mode = 'disabled'
unlocked	Treat all VIFs on this network with locking_mode = 'default' as if they have locking_mode = 'unlocked'

attaching	Indicates this network is attaching to a VIF or PIF

insecure_nbd	Network Block Device service without integrity or confidentiality: NOT RECOMMENDED
nbd	Network Block Device service using TLS

persist	Standard behaviour.
reset	When a VM containing this VDI is started, the contents of the VDI are reset to the state they were in when this flag was last set.

coredump_and_destroy	record a coredump and then destroy the VM state
coredump_and_restart	record a coredump and then restart the VM
destroy	destroy the VM state
preserve	leave the crashed VM paused
rename_restart	rename the crashed VM and start a new copy
restart	restart the VM

destroy	destroy the VM state
restart	restart the VM

disable_on_reboot	On host reboot dom0 will be blocked from accessing this device
disabled	dom0 cannot access this device
enable_on_reboot	On host reboot dom0 will be allowed to access this device
enabled	dom0 can access this device as normal

disabled	IGMP Snooping is disabled in the corresponding backend bridge.'
enabled	IGMP Snooping is enabled in the corresponding backend bridge.'
unknown	IGMP snooping status is unknown. If this is a VLAN master, then please consult the underlying VLAN slave PIF.

cluster_create	Indicates this pool is in the process of creating a cluster
designate_new_master	Indicates this pool is in the process of changing master

ha_disable	Indicates this pool is in the process of disabling HA
ha_enable	Indicates this pool is in the process of enabling HA

IPv4	Primary address is the IPv4 address
IPv6	Primary address is the IPv6 address

caching	The proxy is currently caching data
incompatible_protocol_version	The PVS protocol in use is not compatible with the PVS proxy
incompatible_write_cache_mode	The PVS device is configured to use an incompatible write-cache mode
initialised	The proxy is setup but has not yet cached anything
stopped	The proxy is not currently running

pssl	Passive ssl connection
ssl	Active ssl connection

healthy	Storage is fully available
recovering	Storage is busy recovering, e.g. rebuilding mirrors.

modprobe	Configure network sriov by modprobe, need reboot
sysfs	Configure network sriov by sysfs, do not need reboot
unknown	Unknown mode

destroy	Destroying the SR
forget	Forgetting about SR
pbd_create	Creating a PBD for this SR
pbd_destroy	Destroying one of this SR's PBDs
plug	Plugging a PBD into this SR
scan	Scanning backends for new or deleted VDIs
unplug	Unplugging a PBD from this SR
update	Refresh the fields on the SR
vdi_clone	Cloneing a VDI

vdi_create	Creating a new VDI
vdi_data_destroy	Deleting the data of the VDI
vdi_destroy	Destroying a VDI
vdi_disable_cbt	Disabling changed block tracking for a VDI
vdi_enable_cbt	Enabling changed block tracking for a VDI
vdi_introduce	Introducing a new VDI
vdi_list_changed_blocks	Exporting a bitmap that shows the changed blocks between two VDIs
vdi_mirror	Mirroring a VDI
vdi_resize	Resizing a VDI
vdi_set_on_boot	Setting the on_boot field of the VDI
vdi_snapshot	Snapshotting a VDI

cancel	refers to the operation "cancel"
destroy	refers to the operation "destroy"

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

no	Known to be false
unspecified	Unknown or unspecified
yes	Known to be true

restartHost	This update requires the host to be restarted once applied.
restartHVM	This update requires HVM guests to be restarted once applied.
restartPV	This update requires PV guests to be restarted once applied.
restartXAPI	This update requires XAPI to be restarted once applied.

RO	only read-only access will be allowed

RW	read-write access will be allowed

attach	Attempting to attach this VBD to a VM
eject	Attempting to eject the media from this VBD
insert	Attempting to insert new media into this VBD
pause	Attempting to pause a block device backend
plug	Attempting to hotplug this VBD
unpause	Attempting to unpause a block device backend
unplug	Attempting to hot unplug this VBD
unplug_force	Attempting to forcibly unplug this VBD

CD	VBD will appear to guest as CD
Disk	VBD will appear to guest as disk
Floppy	VBD will appear as a floppy

blocked	Operations on this VDI are temporarily blocked
clone	Cloning the VDI
copy	Copying the VDI
data_destroy	Deleting the data of the VDI
destroy	Destroying the VDI
disable_cbt	Disabling changed block tracking for a VDI
enable_cbt	Enabling changed block tracking for a VDI
force_unlock	Forcibly unlocking the VDI
forget	Forget about the VDI
generate_config	Generating static configuration
list_changed_blocks	Exporting a bitmap that shows the changed blocks between two VDIs
mirror	Mirroring the VDI
resize	Resizing the VDI
resize_online	Resizing the VDI which may or may not be online
set_on_boot	Setting the on_boot field of the VDI
snapshot	Snapshotting the VDI

update	Refreshing the fields of the VDI

cbt_metadata	Metadata about a snapshot VDI that has been deleted: the set of blocks that changed between some previous version of the disk and the version tracked by the snapshot.
crashdump	a disk that stores VM crashdump information
ephemeral	a disk that may be reformatted on upgrade
ha_statefile	a disk used for HA storage heartbeating
metadata	a disk used for HA Pool metadata
pvs_cache	a disk that stores PVS cache data
redo_log	a disk used for a general metadata redo-log
rrd	a disk that stores SR-level RRDs
suspend	a disk that stores a suspend image
system	a disk that may be replaced on upgrade
user	a disk that is always preserved on upgrade

gvt_g	vGPU using Intel GVT-g
mxgpu	vGPU using AMD MxGPU
nvidia	vGPU using NVIDIA hardware
nvidia_sriov	vGPU using NVIDIA hardware with SR-IOV
passthrough	Pass through an entire physical GPU to a guest

None	Follow the default IPv4 configuration of the guest (this is guest-dependent)
Static	Static IPv4 address configuration

None	Follow the default IPv6 configuration of the guest (this is guest-dependent)
Static	Static IPv6 address configuration

disabled	No traffic is permitted

locked	Only traffic to a specific MAC and a list of IPv4 or IPv6 addresses is permitted
network_default	No specific configuration set - default network policy applies
unlocked	All traffic is permitted

attach	Attempting to attach this VIF to a VM
plug	Attempting to hotplug this VIF
unplug	Attempting to hot unplug this VIF

clean_shutdown	Clean shutdown
hard_shutdown	Hard shutdown
shutdown	Shutdown
start	Start

assert_operation_valid	
awaiting_memory_live	Waiting for the memory settings to change
call_plugin	refers to the operation "call_plugin"
changing_dynamic_range	Changing the memory dynamic range
changing_memory_limits	Changing the memory limits
changing_memory_live	Changing the memory settings
changing_NVRAM	Changing NVRAM for a halted VM.
changing_shadow_memory	Changing the shadow memory for a halted VM.
changing_shadow_memory_live	Changing the shadow memory for a running VM.
changing_static_range	Changing the memory static range
changing_VCPUs	Changing VCPU settings for a halted VM.
changing_VCPUs_live	Changing VCPU settings for a running VM.
checkpoint	refers to the operation "checkpoint"
clean_reboot	refers to the operation "clean_reboot"
clean_shutdown	refers to the operation "clean_shutdown"
clone	refers to the operation "clone"
copy	refers to the operation "copy"
create_template	refers to the operation "create_template"

csvm	refers to the operation "csvm"
data_source_op	Add, remove, query or list data sources
destroy	refers to the act of uninstalling the VM
export	exporting a VM to a network stream
get_boot_record	refers to the operation "get_boot_record"
hard_reboot	refers to the operation "hard_reboot"
hard_shutdown	refers to the operation "hard_shutdown"
import	importing a VM from a network stream
make_into_template	Turning this VM into a template
metadata_export	exporting VM metadata to a network stream
migrate_send	refers to the operation "migrate_send"
pause	refers to the operation "pause"
pool_migrate	refers to the operation "pool_migrate"
power_state_reset	refers to the operation "power_state_reset"
provision	refers to the operation "provision"
query_services	refers to the operation "query_services"
resume	refers to the operation "resume"
resume_on	refers to the operation "resume_on"
revert	refers to the operation "revert"
reverting	Reverting the VM to a previous snapshotted state
send_sysrq	refers to the operation "send_sysrq"
send_trigger	refers to the operation "send_trigger"
shutdown	refers to the operation "shutdown"
snapshot	refers to the operation "snapshot"
snapshot_with_quiesce	refers to the operation "snapshot_with_quiesce"
start	refers to the operation "start"
start_on	refers to the operation "start_on"
suspend	refers to the operation "suspend"
unpause	refers to the operation "unpause"
update_allowed_operations	

`Halted`	VM is offline and not using any resources
`Paused`	All resources have been allocated but the VM itself is paused and its vCPUs are not running
`Running`	Running
`Suspended`	VM state has been saved to disk and it is nolonger running. Note that disks remain in-use while the VM is suspended.

`always_after_backup`	Archive after backup
`daily`	Daily archives
`never`	Never archive
`weekly`	Weekly backups

`cifs`	CIFS target config
`nfs`	NFS target config
`none`	No target config

`daily`	Daily backups
`hourly`	Hourly backups
`weekly`	Weekly backups

`checkpoint`	The backup is a checkpoint
`snapshot`	The backup is a snapshot

`daily`	Daily snapshots
`hourly`	Hourly snapshots
`weekly`	Weekly snapshots

`checkpoint`	The snapshot is a checkpoint
`snapshot`	The snapshot is a disk snapshot
`snapshot_with_quiesce`	Support for VSS has been removed.

`attach`	Attempting to attach this VUSB to a VM
`plug`	Attempting to plug this VUSB into a VM
`unplug`	Attempting to hot unplug this VUSB

Class: auth

Management of remote authentication services

Fields for class: auth

Class auth has no fields.

RPCs associated with class: auth

RPC name: get_group_membership

Overview:

This calls queries the external directory service to obtain the transitively-closed set of groups that the the subject_identifier is member of.

Signature:

```
string set get_group_membership (session ref session_id, string subject_identifier)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	subject_identifier	A string containing the subject_identifier, unique in the external directory service

Minimum Role: read-only

Return Type: `string set`

set of subject_identifiers that provides the group membership of subject_identifier passed as argument, it contains, recursively, all groups a subject_identifier is member of.

RPC name: get_subject_identifier

Overview:

This call queries the external directory service to obtain the subject_identifier as a string from the human-readable subject_name

Signature:

```
string get_subject_identifier (session ref session_id, string subject_name)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	subject_name	The human-readable subject_name, such as a username or a groupname

Minimum Role: read-only

Return Type: `string`

the subject_identifier obtained from the external directory service

RPC name: get_subject_information_from_identifier

Overview:

This call queries the external directory service to obtain the user information (e.g. username, organization etc) from the specified subject_identifier

Signature:

```
(string -> string) map get_subject_information_from_identifier (session ref
session_id, string subject_identifier)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	subject_identifier	A string containing the subject_identifier, unique in the external directory service

Minimum Role: read-only

Return Type: `(string -> string) map`

key-value pairs containing at least a key called subject_name

Class: blob

A placeholder for a binary blob

Fields for class: blob

Field	Type	Qualifier	Description
last_updated	`datetime`	*RO/constructor*	Time at which the data in the blob was last updated
mime_type	`string`	*RO/constructor*	The mime type associated with this object. Defaults to 'application/octet-stream' if the empty string is supplied
name_description	`string`	*RW*	a notes field containing human-readable description
name_label	`string`	*RW*	a human-readable name
public	`bool`	*RW*	True if the blob is publicly accessible
size	`int`	*RO/runtime*	Size of the binary data, in bytes
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: blob

RPC name: create

Overview:

Create a placeholder for a binary blob

Signature:

```
blob ref create (session ref session_id, string mime_type, bool public)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	mime_type	The mime-type of the blob. Defaults to 'application/octet-stream' if the empty string is supplied
`bool`	public	True if the blob should be publicly available

Minimum Role: pool-operator

Return Type: `blob ref`

The reference to the created blob

RPC name: destroy

Overview:

Signature:

```
void destroy (session ref session_id, blob ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`blob ref`	self	The reference of the blob to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the blobs known to the system.

Signature:

```
blob ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `blob ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of blob references to blob records for all blobs known to the system.

Signature:

```
(blob ref -> blob record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(blob ref -> blob record) map`

records of all objects

RPC name: get_by_name_label

Overview:

Get all the blob instances with the given label.

Signature:

```
blob ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `blob ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the blob instance with the specified UUID.

Signature:

```
blob ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `blob ref`

reference to the object

RPC name: get_last_updated

Overview:

Get the last_updated field of the given blob.

Signature:

```
datetime get_last_updated (session ref session_id, blob ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`blob ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_mime_type

Overview:

Get the mime_type field of the given blob.

Signature:

```
string get_mime_type (session ref session_id, blob ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`blob ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given blob.

Signature:

```
string get_name_description (session ref session_id, blob ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`blob ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

Overview:

Get the name/label field of the given blob.

Signature:

```
string get_name_label (session ref session_id, blob ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`blob ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_public

Overview:

Get the public field of the given blob.

Signature:

```
bool get_public (session ref session_id, blob ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`blob ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given blob.

Signature:

```
blob record get_record (session ref session_id, blob ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`blob ref`	self	reference to the object

Minimum Role: read-only

Return Type: `blob record`

all fields from the object

RPC name: get_size

Overview:

Get the size field of the given blob.

Signature:

```
int get_size (session ref session_id, blob ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`blob ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given blob.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`blob ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: set_name_description

Overview:

Set the name/description field of the given blob.

Signature:

```
void set_name_description (session ref session_id, blob ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`blob ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_name_label

Overview:

Set the name/label field of the given blob.

Signature:

```
void set_name_label (session ref session_id, blob ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`blob ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_public

Overview:

Set the public field of the given blob.

Signature:

```
void set_public (session ref session_id, blob ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`blob ref`	self	reference to the object
`bool`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: Bond

Fields for class: Bond

Field	Type	Qualifier	Description
auto_update_mac	bool	*RO/runtime*	true if the MAC was taken from the primary slave when the bond was created, and false if the client specified the MAC
links_up	int	*RO/runtime*	Number of links up in this bond
master	PIF ref	*RO/constructor*	The bonded interface
mode	bond_mode	*RO/runtime*	The algorithm used to distribute traffic among the bonded NICs
other_config	(string -> string) map	*RW*	additional configuration
primary_slave	PIF ref	*RO/runtime*	The PIF of which the IP configuration and MAC were copied to the bond, and which will receive all configuration/VLANs/VIFs on the bond if the bond is destroyed
properties	(string -> string) map	*RO/runtime*	Additional configuration properties specific to the bond mode.
slaves	PIF ref set	*RO/runtime*	The interfaces which are part of this bond
uuid	string	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: Bond

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given Bond.

Signature:

```
void add_to_other_config (session ref session_id, Bond ref self, string key, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
Bond ref	self	reference to the object
string	key	Key to add

type	name	description
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: create

Overview:

Create an interface bond

Signature:

```
Bond ref create (session ref session_id, network ref network, PIF ref set members,
string MAC, bond_mode mode, (string -> string) map properties)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	network	Network to add the bonded PIF to
`PIF ref set`	members	PIFs to add to this bond
`string`	MAC	The MAC address to use on the bond itself. If this parameter is the empty string then the bond will inherit its MAC address from the primary slave.
`bond_mode`	mode	Bonding mode to use for the new bond
`(string -> string) map`	properties	Additional configuration parameters specific to the bond mode

Minimum Role: pool-operator

Return Type: `Bond ref`

The reference of the created Bond object

RPC name: destroy

Overview:

Destroy an interface bond

Signature:

```
void destroy (session ref session_id, Bond ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	Bond to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the Bonds known to the system.

Signature:

```
Bond ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `Bond ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of Bond references to Bond records for all Bonds known to the system.

Signature:

```
(Bond ref -> Bond record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(Bond ref -> Bond record) map`

records of all objects

RPC name: get_auto_update_mac

Overview:

Get the auto_update_mac field of the given Bond.

Signature:

```
bool get_auto_update_mac (session ref session_id, Bond ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_by_uuid

Overview:

Get a reference to the Bond instance with the specified UUID.

Signature:

```
Bond ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `Bond ref`

reference to the object

RPC name: get_links_up

Overview:

Get the links_up field of the given Bond.

Signature:

```
int get_links_up (session ref session_id, Bond ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_master

Overview:

Get the master field of the given Bond.

Signature:

```
PIF ref get_master (session ref session_id, Bond ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF ref`

value of the field

RPC name: get_mode

Overview:

Get the mode field of the given Bond.

Signature:

```
bond_mode get_mode (session ref session_id, Bond ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bond_mode`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given Bond.

Signature:

```
(string -> string) map get_other_config (session ref session_id, Bond ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_primary_slave

Overview:

Get the primary_slave field of the given Bond.

Signature:

```
PIF ref get_primary_slave (session ref session_id, Bond ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`Bond ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF ref`

value of the field

RPC name: get_properties

Overview:

Get the properties field of the given Bond.

Signature:

```
(string -> string) map get_properties (session ref session_id, Bond ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given Bond.

Signature:

```
Bond record get_record (session ref session_id, Bond ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	reference to the object

Minimum Role: read-only

Return Type: `Bond record`

all fields from the object

RPC name: get_slaves

Overview:

Get the slaves field of the given Bond.

Signature:

```
PIF ref set get_slaves (session ref session_id, Bond ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF ref set`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given Bond.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given Bond. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, Bond ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_mode

Overview:

Change the bond mode

Signature:

```
void set_mode (session ref session_id, Bond ref self, bond_mode value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	The bond
`bond_mode`	value	The new bond mode

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given Bond.

Signature:

```
void set_other_config (session ref session_id, Bond ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_property

Overview:

Set the value of a property of the bond

Signature:

```
void set_property (session ref session_id, Bond ref self, string name, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Bond ref`	self	The bond
`string`	name	The property name
`string`	value	The property value

Minimum Role: pool-operator

Return Type: `void`

Class: Certificate

Description

Fields for class: Certificate

Field	Type	Qualifier	Description
fingerprint	`string`	*RO/constructor*	The certificate's fingerprint / hash
host	`host ref`	*RO/constructor*	The host where the certificate is installed
not_after	`datetime`	*RO/constructor*	Date before which the certificate is valid
not_before	`datetime`	*RO/constructor*	Date after which the certificate is valid
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: Certificate

RPC name: get_all

Overview:

Return a list of all the Certificates known to the system.

Signature:

```
Certificate ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `Certificate ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of Certificate references to Certificate records for all Certificates known to the system.

Signature:

```
(Certificate ref -> Certificate record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(Certificate ref -> Certificate record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the Certificate instance with the specified UUID.

Signature:

```
Certificate ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `Certificate ref`

reference to the object

RPC name: get_fingerprint

Overview:

Get the fingerprint field of the given Certificate.

Signature:

```
string get_fingerprint (session ref session_id, Certificate ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Certificate ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_host

Overview:

Get the host field of the given Certificate.

Signature:

```
host ref get_host (session ref session_id, Certificate ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Certificate ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_not_after

Overview:

Get the not_after field of the given Certificate.

Signature:

```
datetime get_not_after (session ref session_id, Certificate ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Certificate ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_not_before

Overview:

Get the not_before field of the given Certificate.

Signature:

```
datetime get_not_before (session ref session_id, Certificate ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Certificate ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given Certificate.

Signature:

```
Certificate record get_record (session ref session_id, Certificate ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Certificate ref`	self	reference to the object

Minimum Role: read-only

Return Type: `Certificate record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given Certificate.

Signature:

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

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`Certificate ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

Class: Cluster

Cluster-wide Cluster metadata

Fields for class: Cluster

Field	Type	Qualifier	Description
allowed_operations	`cluster_operation set`	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
cluster_config	`(string -> string) map`	*RO/constructor*	Contains read-only settings for the cluster, such as timeouts and other options. It can only be set at cluster create time
cluster_hosts	`Cluster_host ref set`	*RO/runtime*	A list of the cluster_host objects associated with the Cluster
cluster_stack	`string`	*RO/constructor*	Simply the string 'corosync'. No other cluster stacks are currently supported
cluster_token	`string`	*RO/constructor*	The secret key used by xapi-clusterd when it talks to itself on other hosts
current_operations	`(string -> cluster_operation) map`	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.
other_config	`(string -> string) map`	*RW*	Additional configuration
pending_forget	`string set`	*RO/runtime*	Internal field used by Host.destroy to store the IP of cluster members marked as permanently dead but not yet removed

Field	Type	Qualifier	Description
pool_auto_join	`bool`	*RO/constructor*	True if automatically joining new pool members to the cluster. This will be `true` in the first release
token_timeout	`float`	*RO/constructor*	The corosync token timeout in seconds
token_timeout_coefficient	`float`	*RO/constructor*	The corosync token timeout coefficient in seconds
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: Cluster

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given Cluster.

Signature:

```
void add_to_other_config (session ref session_id, Cluster ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: create

Overview:

Creates a Cluster object and one Cluster_host object as its first member

Signature:

```
Cluster ref create (session ref session_id, PIF ref PIF, string cluster_stack, bool
pool_auto_join, float token_timeout, float token_timeout_coefficient)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	PIF	The PIF to connect the cluster's first cluster_host to
`string`	cluster_stack	simply the string 'corosync'. No other cluster stacks are currently supported
`bool`	pool_auto_join	true if xapi is automatically joining new pool members to the cluster
`float`	token_timeout	Corosync token timeout in seconds
`float`	token_timeout_coefficient	Corosync token timeout coefficient in seconds

Minimum Role: pool-operator

Return Type: `Cluster ref`

the new Cluster

Possible Error Codes: `INVALID_CLUSTER_STACK`, `INVALID_VALUE`, `PIF_ALLOWS_UNPLUG`, `REQUIRED_PIF_IS_UNPLUGGED`

RPC name: destroy

Overview:

Destroys a Cluster object and the one remaining Cluster_host member

Signature:

```
void destroy (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	the Cluster to destroy

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `CLUSTER_DOES_NOT_HAVE_ONE_NODE`, `CLUSTER_STACK_IN_USE`

RPC name: get_all

Overview:

Return a list of all the Clusters known to the system.

Signature:

```
Cluster ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `Cluster ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of Cluster references to Cluster records for all Clusters known to the system.

Signature:

```
(Cluster ref -> Cluster record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(Cluster ref -> Cluster record) map`

records of all objects

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given Cluster.

Signature:

```
cluster_operation set get_allowed_operations (session ref session_id, Cluster ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `cluster_operation set`

value of the field

RPC name: get_by_uuid

Overview:

Get a reference to the Cluster instance with the specified UUID.

Signature:

```
Cluster ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `Cluster ref`

reference to the object

RPC name: get_cluster_config

Overview:

Get the cluster_config field of the given Cluster.

Signature:

```
(string -> string) map get_cluster_config (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_cluster_hosts

Overview:

Get the cluster_hosts field of the given Cluster.

Signature:

```
Cluster_host ref set get_cluster_hosts (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `Cluster_host ref set`

value of the field

RPC name: get_cluster_stack

Overview:

Get the cluster_stack field of the given Cluster.

Signature:

```
string get_cluster_stack (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_cluster_token

Overview:

Get the cluster_token field of the given Cluster.

Signature:

```
string get_cluster_token (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_current_operations

Overview:

Get the current_operations field of the given Cluster.

Signature:

```
(string -> cluster_operation) map get_current_operations (session ref session_id,
Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> cluster_operation) map`

value of the field

RPC name: get_network

Overview:

Returns the network used by the cluster for inter-host communication, i.e. the network shared by all cluster host PIFs

Signature:

```
network ref get_network (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	the Cluster with the network

Minimum Role: read-only

Return Type: `network ref`

network of cluster

RPC name: get_other_config

Overview:

Get the other_config field of the given Cluster.

Signature:

```
(string -> string) map get_other_config (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_pending_forget

Overview:

Get the pending_forget field of the given Cluster.

Signature:

```
string set get_pending_forget (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_pool_auto_join

Overview:

Get the pool_auto_join field of the given Cluster.

Signature:

```
bool get_pool_auto_join (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given Cluster.

Signature:

```
Cluster record get_record (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `Cluster record`

all fields from the object

RPC name: get_token_timeout

Overview:

Get the token_timeout field of the given Cluster.

Signature:

```
float get_token_timeout (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `float`

value of the field

RPC name: get_token_timeout_coefficient

Overview:

Get the token_timeout_coefficient field of the given Cluster.

Signature:

```
float get_token_timeout_coefficient (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `float`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given Cluster.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: pool_create

Overview:

Attempt to create a Cluster from the entire pool

Signature:

```
Cluster ref pool_create (session ref session_id, network ref network, string
cluster_stack, float token_timeout, float token_timeout_coefficient)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
network ref	network	the single network on which corosync carries out its inter-host communications
string	cluster_stack	simply the string 'corosync'. No other cluster stacks are currently supported
float	token_timeout	Corosync token timeout in seconds
float	token_timeout_coefficient	Corosync token timeout coefficient in seconds

Minimum Role: pool-operator

Return Type: Cluster ref

the new Cluster

RPC name: pool_destroy

Overview:

Attempt to destroy the Cluster_host objects for all hosts in the pool and then destroy the Cluster.

Signature:

```
void pool_destroy (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
Cluster ref	self	The cluster to destroy.

Minimum Role: pool-operator

Return Type: void

Possible Error Codes: CLUSTER_STACK_IN_USE, CLUSTERING_DISABLED, CLUSTER_HOST_IS_LAST

RPC name: pool_force_destroy

Overview:

Attempt to force destroy the Cluster_host objects, and then destroy the Cluster.

Signature:

```
void pool_force_destroy (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	The cluster to force destroy.

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `CLUSTER_FORCE_DESTROY_FAILED`

RPC name: pool_resync

Overview:

Resynchronise the cluster_host objects across the pool. Creates them where they need creating and then plugs them

Signature:

```
void pool_resync (session ref session_id, Cluster ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	The cluster to resync

Minimum Role: pool-operator

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 Cluster. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, Cluster ref self, string key)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given Cluster.

Signature:

```
void set_other_config (session ref session_id, Cluster ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: Cluster_host

Cluster member metadata

Fields for class: Cluster_host

Field	Type	Qualifier	Description
allowed_operations	`cluster_host_operation set`	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
cluster	`Cluster ref`	*RO/constructor*	Reference to the Cluster object

Field	Type	Qualifier	Description
current_operations	(string -> cluster_host_operation) map	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.
enabled	bool	*RO/constructor*	Whether the cluster host believes that clustering should be enabled on this host
host	host ref	*RO/constructor*	Reference to the Host object
joined	bool	*RO/constructor*	Whether the cluster host has joined the cluster
other_config	(string -> string) map	*RO/constructor*	Additional configuration
PIF	PIF ref	*RO/constructor*	Reference to the PIF object
uuid	string	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: Cluster_host

RPC name: create

Overview:

Add a new host to an existing cluster.

Signature:

```
Cluster_host ref create (session ref session_id, Cluster ref cluster, host ref host,
PIF ref pif)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
Cluster ref	cluster	Cluster to join
host ref	host	new cluster member
PIF ref	pif	Network interface to use for communication

Minimum Role: pool-operator

Return Type: Cluster_host ref

the newly created cluster_host object

Possible Error Codes: PIF_NOT_ATTACHED_TO_HOST, NO_CLUSTER_HOSTS_REACHABLE

RPC name: destroy

Overview:

Remove a host from an existing cluster.

Signature:

```
void destroy (session ref session_id, Cluster_host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	the cluster_host to remove from the cluster

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `CLUSTER_STACK_IN_USE`, `CLUSTERING_DISABLED`, `CLUSTER_HOST_IS_LAST`

RPC name: disable

Overview:

Disable cluster membership for an enabled cluster host.

Signature:

```
void disable (session ref session_id, Cluster_host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	the cluster_host to disable

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `CLUSTER_STACK_IN_USE`

RPC name: enable

Overview:

Enable cluster membership for a disabled cluster host.

Signature:

```
void enable (session ref session_id, Cluster_host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	the cluster_host to enable

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `PIF_ALLOWS_UNPLUG`, `REQUIRED_PIF_IS_UNPLUGGED`

RPC name: force_destroy

Overview:

Remove a host from an existing cluster forcefully.

Signature:

```
void force_destroy (session ref session_id, Cluster_host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	the cluster_host to remove from the cluster

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `CLUSTER_STACK_IN_USE`

RPC name: get_all

Overview:

Return a list of all the Cluster_hosts known to the system.

Signature:

```
Cluster_host ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `Cluster_host ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of Cluster_host references to Cluster_host records for all Cluster_hosts known to the system.

Signature:

```
(Cluster_host ref -> Cluster_host record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(Cluster_host ref -> Cluster_host record) map`

records of all objects

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given Cluster_host.

Signature:

```
cluster_host_operation set get_allowed_operations (session ref session_id,
Cluster_host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `cluster_host_operation set`

value of the field

RPC name: get_by_uuid

Overview:

Get a reference to the Cluster_host instance with the specified UUID.

Signature:

```
Cluster_host ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `Cluster_host ref`

reference to the object

RPC name: get_cluster

Overview:

Get the cluster field of the given Cluster_host.

Signature:

```
Cluster ref get_cluster (session ref session_id, Cluster_host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `Cluster ref`

value of the field

RPC name: get_current_operations

Overview:

Get the current_operations field of the given Cluster_host.

Signature:

```
(string -> cluster_host_operation) map get_current_operations (session ref session_id,
Cluster_host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> cluster_host_operation) map`

value of the field

RPC name: get_enabled

Overview:

Get the enabled field of the given Cluster_host.

Signature:

```
bool get_enabled (session ref session_id, Cluster_host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_host

Overview:

Get the host field of the given Cluster_host.

Signature:

```

```

```
host ref get_host (session ref session_id, Cluster_host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_joined

Overview:

Get the joined field of the given Cluster_host.

Signature:

```
bool get_joined (session ref session_id, Cluster_host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given Cluster_host.

Signature:

```
(string -> string) map get_other_config (session ref session_id, Cluster_host ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_PIF

Overview:

Get the PIF field of the given Cluster_host.

Signature:

```
PIF ref get_PIF (session ref session_id, Cluster_host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF ref`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given Cluster_host.

Signature:

```
Cluster_host record get_record (session ref session_id, Cluster_host ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `Cluster_host record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given Cluster_host.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Cluster_host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

Class: console

A console

Fields for class: console

Field	Type	Qualifier	Description
location	`string`	*RO/runtime*	URI for the console service
other_config	`(string -> string) map`	*RW*	additional configuration
protocol	`console_protocol`	*RO/runtime*	the protocol used by this console
uuid	`string`	*RO/runtime*	Unique identifier/object reference
VM	`VM ref`	*RO/runtime*	VM to which this console is attached

RPCs associated with class: console

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 ref session_id, console ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`console ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: vm-admin

Return Type: `void`

RPC name: create

Overview:

Create a new console instance, and return its handle.

Signature:

```
console ref create (session ref session_id, console record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`console record`	args	All constructor arguments

Minimum Role: vm-admin

Return Type: `console ref`

reference to the newly created object

RPC name: destroy

Overview:

Destroy the specified console instance.

Signature:

```
void destroy (session ref session_id, console ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`console ref`	self	reference to the object

Minimum Role: vm-admin

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the consoles known to the system.

Signature:

```
console ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `console ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of console references to console records for all consoles known to the system.

Signature:

```
(console ref -> console record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(console ref -> console record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the console instance with the specified UUID.

Signature:

```
console ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `console ref`

reference to the object

RPC name: get_location

Overview:

Get the location field of the given console.

Signature:

```
string get_location (session ref session_id, console ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`console ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

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 ref session_id, console ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`console ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_protocol

Overview:

Get the protocol field of the given console.

Signature:

```
console_protocol get_protocol (session ref session_id, console ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`console ref`	self	reference to the object

Minimum Role: read-only

Return Type: `console_protocol`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given console.

Signature:

```
console record get_record (session ref session_id, console ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`console ref`	self	reference to the object

Minimum Role: read-only

Return Type: `console record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given console.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`console ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, console ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`console ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref`

value of the field

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 ref session_id, console ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`console ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given console.

Signature:

```
void set_other_config (session ref session_id, console ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`console ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

Class: crashdump

This class is deprecated.

A VM crashdump

Fields for class: crashdump

Field	Type	Qualifier	Description
other_config	`(string -> string) map`	*RW*	**Deprecated**. additional configuration
uuid	`string`	*RO/runtime*	**Deprecated**. Unique identifier/object reference
VDI	`VDI ref`	*RO/constructor*	**Deprecated**. the virtual disk
VM	`VM ref`	*RO/constructor*	**Deprecated**. the virtual machine

RPCs associated with class: crashdump

RPC name: add_to_other_config

This message is deprecated.

Overview:

Add the given key-value pair to the other_config field of the given crashdump.

Signature:

```
void add_to_other_config (session ref session_id, crashdump ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`crashdump ref`	self	reference to the object

type	name	description
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: destroy

This message is deprecated.

Overview:

Destroy the specified crashdump

Signature:

```
void destroy (session ref session_id, crashdump ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`crashdump ref`	self	The crashdump to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

This message is deprecated.

Overview:

Return a list of all the crashdumps known to the system.

Signature:

```
crashdump ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `crashdump ref set`

references to all objects

RPC name: get_all_records

This message is deprecated.

Overview:

Return a map of crashdump references to crashdump records for all crashdumps known to the system.

Signature:

```
(crashdump ref -> crashdump record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(crashdump ref -> crashdump record) map`

records of all objects

RPC name: get_by_uuid

This message is deprecated.

Overview:

Get a reference to the crashdump instance with the specified UUID.

Signature:

```
crashdump ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `crashdump ref`

reference to the object

RPC name: get_other_config

This message is deprecated.

Overview:

Get the other_config field of the given crashdump.

Signature:

```
(string -> string) map get_other_config (session ref session_id, crashdump ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`crashdump ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

This message is deprecated.

Overview:

Get a record containing the current state of the given crashdump.

Signature:

```
crashdump record get_record (session ref session_id, crashdump ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`crashdump ref`	self	reference to the object

Minimum Role: read-only

Return Type: `crashdump record`

all fields from the object

RPC name: get_uuid

This message is deprecated.

Overview:

Get the uuid field of the given crashdump.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`crashdump ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VDI

This message is deprecated.

Overview:

Get the VDI field of the given crashdump.

Signature:

```
VDI ref get_VDI (session ref session_id, crashdump ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`crashdump ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VDI ref`

value of the field

RPC name: get_VM

This message is deprecated.

Overview:

Get the VM field of the given crashdump.

Signature:

```
VM ref get_VM (session ref session_id, crashdump ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`crashdump ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref`

value of the field

RPC name: remove_from_other_config

This message is deprecated.

Overview:

Remove the given key and its corresponding value from the other_config field of the given crashdump. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, crashdump ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`crashdump ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

This message is deprecated.

Overview:

Set the other_config field of the given crashdump.

Signature:

```
void set_other_config (session ref session_id, crashdump ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`crashdump ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: data_source

Data sources for logging in RRDs

Fields for class: data_source

Field	Type	Qualifier	Description
enabled	`bool`	*RO/runtime*	true if the data source is being logged
max	`float`	*RO/runtime*	the maximum value of the data source
min	`float`	*RO/runtime*	the minimum value of the data source
name_description	`string`	*RO/runtime*	a notes field containing human-readable description
name_label	`string`	*RO/runtime*	a human-readable name
standard	`bool`	*RO/runtime*	true if the data source is enabled by default. Non-default data sources cannot be disabled
units	`string`	*RO/runtime*	the units of the value
value	`float`	*RO/runtime*	current value of the data source

RPCs associated with class: data_source

Class data_source has no additional RPCs associated with it.

Class: DR_task

DR task

Fields for class: DR_task

Field	Type	Qualifier	Description

Field	Type	Qualifier	Description
introduced_SRs	`SR ref set`	*RO/runtime*	All SRs introduced by this appliance
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: DR_task

RPC name: create

Overview:

Create a disaster recovery task which will query the supplied list of devices

Signature:

```
DR_task ref create (session ref session_id, string type, (string -> string) map
device_config, string set whitelist)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	type	The SR driver type of the SRs to introduce
`(string -> string) map`	device_config	The device configuration of the SRs to introduce
`string set`	whitelist	The devices to use for disaster recovery

Minimum Role: pool-operator

Return Type: `DR_task ref`

The reference to the created task

RPC name: destroy

Overview:

Destroy the disaster recovery task, detaching and forgetting any SRs introduced which are no longer required

Signature:

```
void destroy (session ref session_id, DR_task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
DR_task ref	self	The disaster recovery task to destroy

Minimum Role: pool-operator

Return Type: void

RPC name: get_all

Overview:

Return a list of all the DR_tasks known to the system.

Signature:

```
DR_task ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: DR_task ref set

references to all objects

RPC name: get_all_records

Overview:

Return a map of DR_task references to DR_task records for all DR_tasks known to the system.

Signature:

```
(DR_task ref -> DR_task record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: (DR_task ref -> DR_task record) map

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the DR_task instance with the specified UUID.

Signature:

```
DR_task ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `DR_task ref`

reference to the object

RPC name: get_introduced_SRs

Overview:

Get the introduced_SRs field of the given DR_task.

Signature:

```
SR ref set get_introduced_SRs (session ref session_id, DR_task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`DR_task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SR ref set`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given DR_task.

Signature:

```
DR_task record get_record (session ref session_id, DR_task ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`DR_task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `DR_task record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given DR_task.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`DR_task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

Class: event

Asynchronous event registration and handling

Fields for class: event

Field	Type	Qualifier	Description
class	`string`	*RO/constructor*	The name of the class of the object that changed
id	`int`	*RO/constructor*	An ID, monotonically increasing, and local to the current session
obj_uuid	`string`	*RO/constructor*	**Deprecated**. The uuid of the object that changed
operation	`event_operation`	*RO/constructor*	The operation that was performed
ref	`string`	*RO/constructor*	A reference to the object that changed

Field	Type	Qualifier	Description
timestamp	`datetime`	*RO/constructor*	**Deprecated**. The time at which the event occurred
snapshot	`<object record>`	*RO/runtime*	The record of the database object that was added, changed or deleted

RPCs associated with class: event

RPC name: from

Overview:

Blocking call which returns a new token and a (possibly empty) batch of events. The returned token can be used in subsequent calls to this function.

Signature:

```
<event batch> from (session ref session_id, string set classes, string token, float
timeout)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string set`	classes	register for events for the indicated classes
`string`	token	A token representing the point from which to generate database events. The empty string represents the beginning.
`float`	timeout	Return after this many seconds if no events match

Minimum Role: read-only

Return Type: an event batch

a structure consisting of a token ('token'), a map of valid references per object type ('valid_ref_counts'), and a set of event records ('events').

Possible Error Codes: `SESSION_NOT_REGISTERED`, `EVENTS_LOST`

RPC name: get_current_id

Overview:

Return the ID of the next event to be generated by the system

Signature:

```
int get_current_id (session ref session_id)
```

Minimum Role: read-only

Return Type: `int`

the event ID

RPC name: inject

Overview:

Injects an artificial event on the given object and returns the corresponding ID in the form of a token, which can be used as a point of reference for database events. For example, to check whether an object has reached the right state before attempting an operation, one can inject an artificial event on the object and wait until the token returned by consecutive event.from calls is lexicographically greater than the one returned by event.inject.

Signature:

```
string inject (session ref session_id, string class, string ref)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	class	class of the object
`string`	ref	A reference to the object that will be changed.

Minimum Role: read-only

Return Type: `string`

the event ID in the form of a token

RPC name: next

This message is deprecated.

Overview:

Blocking call which returns a (possibly empty) batch of events. This method is only recommended for legacy use. New development should use event.from which supercedes this method.

Signature:

```
event record set next (session ref session_id)
```

Minimum Role: read-only

Return Type: `event record set`

A set of events

Possible Error Codes: `SESSION_NOT_REGISTERED`, `EVENTS_LOST`

RPC name: register

This message is deprecated.

Overview:

Registers this session with the event system for a set of given classes. This method is only recommended for legacy use in conjunction with event.next.

Signature:

```
void register (session ref session_id, string set classes)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string set`	classes	the classes for which the session will register with the event system; specifying * as the desired class will register for all classes

Minimum Role: read-only

Return Type: `void`

RPC name: unregister

This message is deprecated.

Overview:

Removes this session's registration with the event system for a set of given classes. This method is only recommended for legacy use in conjunction with event.next.

Signature:

```
void unregister (session ref session_id, string set classes)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`string set`	classes	the classes for which the session's registration with the event system will be removed

Minimum Role: read-only

Return Type: `void`

Class: Feature

A new piece of functionality

Fields for class: Feature

Field	Type	Qualifier	Description
enabled	`bool`	*RO/runtime*	Indicates whether the feature is enabled
experimental	`bool`	*RO/constructor*	Indicates whether the feature is experimental (as opposed to stable and fully supported)
host	`host ref`	*RO/runtime*	The host where this feature is available
name_description	`string`	*RO/constructor*	a notes field containing human-readable description
name_label	`string`	*RO/constructor*	a human-readable name
uuid	`string`	*RO/runtime*	Unique identifier/object reference
version	`string`	*RO/constructor*	The version of this feature

RPCs associated with class: Feature

RPC name: get_all

Overview:

Return a list of all the Features known to the system.

Signature:

```
Feature ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `Feature ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of Feature references to Feature records for all Features known to the system.

Signature:

```
(Feature ref -> Feature record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(Feature ref -> Feature record) map`

records of all objects

RPC name: get_by_name_label

Overview:

Get all the Feature instances with the given label.

Signature:

```
Feature ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `Feature ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the Feature instance with the specified UUID.

Signature:

```
Feature ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `Feature ref`

reference to the object

RPC name: get_enabled

Overview:

Get the enabled field of the given Feature.

Signature:

```
bool get_enabled (session ref session_id, Feature ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Feature ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_experimental

Overview:

Get the experimental field of the given Feature.

Signature:

```
bool get_experimental (session ref session_id, Feature ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`Feature ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_host

Overview:

Get the host field of the given Feature.

Signature:

```
host ref get_host (session ref session_id, Feature ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Feature ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given Feature.

Signature:

```
string get_name_description (session ref session_id, Feature ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Feature ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

Overview:

Get the name/label field of the given Feature.

Signature:

```
string get_name_label (session ref session_id, Feature ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Feature ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given Feature.

Signature:

```
Feature record get_record (session ref session_id, Feature ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Feature ref`	self	reference to the object

Minimum Role: read-only

Return Type: `Feature record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given Feature.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Feature ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_version

Overview:

Get the version field of the given Feature.

Signature:

```
string get_version (session ref session_id, Feature ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`Feature ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

Class: GPU_group

A group of compatible GPUs across the resource pool

Fields for class: GPU_group

Field	Type	Qualifier	Description
allocation_algorithm	`allocation_algorithm`	*RW*	Current allocation of vGPUs to pGPUs for this group
enabled_VGPU_types	`VGPU_type ref set`	*RO/runtime*	vGPU types supported on at least one of the pGPUs in this group
GPU_types	`string set`	*RO/runtime*	List of GPU types (vendor+device ID) that can be in this group
name_description	`string`	*RW*	a notes field containing human-readable description
name_label	`string`	*RW*	a human-readable name
other_config	`(string -> string) map`	*RW*	Additional configuration
PGPUs	`PGPU ref set`	*RO/runtime*	List of pGPUs in the group
supported_VGPU_types	`VGPU_type ref set`	*RO/runtime*	vGPU types supported on at least one of the pGPUs in this group
uuid	`string`	*RO/runtime*	Unique identifier/object reference
VGPUs	`VGPU ref set`	*RO/runtime*	List of vGPUs using the group

RPCs associated with class: GPU_group

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given GPU_group.

Signature:

```
void add_to_other_config (session ref session_id, GPU_group ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object
`string`	key	Key to add

type	name	description
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: create

Overview:

Signature:

```
GPU_group ref create (session ref session_id, string name_label, string
name_description, (string -> string) map other_config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	name_label	
`string`	name_description	
`(string -> string) map`	other_config	

Minimum Role: pool-operator

Return Type: `GPU_group ref`

RPC name: destroy

Overview:

Signature:

```
void destroy (session ref session_id, GPU_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	The GPU group to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the GPU_groups known to the system.

Signature:

```
GPU_group ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `GPU_group ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of GPU_group references to GPU_group records for all GPU_groups known to the system.

Signature:

```
(GPU_group ref -> GPU_group record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(GPU_group ref -> GPU_group record) map`

records of all objects

RPC name: get_allocation_algorithm

Overview:

Get the allocation_algorithm field of the given GPU_group.

Signature:

```
allocation_algorithm get_allocation_algorithm (session ref session_id, GPU_group ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `allocation_algorithm`

value of the field

RPC name: get_by_name_label

Overview:

Get all the GPU_group instances with the given label.

Signature:

```
GPU_group ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `GPU_group ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the GPU_group instance with the specified UUID.

Signature:

```
GPU_group ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `GPU_group ref`

reference to the object

RPC name: get_enabled_VGPU_types

Overview:

Get the enabled_VGPU_types field of the given GPU_group.

Signature:

```
VGPU_type ref set get_enabled_VGPU_types (session ref session_id, GPU_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VGPU_type ref set`

value of the field

RPC name: get_GPU_types

Overview:

Get the GPU_types field of the given GPU_group.

Signature:

```
string set get_GPU_types (session ref session_id, GPU_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given GPU_group.

Signature:

```
string get_name_description (session ref session_id, GPU_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

Overview:

Get the name/label field of the given GPU_group.

Signature:

```
string get_name_label (session ref session_id, GPU_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given GPU_group.

Signature:

```
(string -> string) map get_other_config (session ref session_id, GPU_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_PGPUs

Overview:

Get the PGPUs field of the given GPU_group.

Signature:

```
PGPU ref set get_PGPUs (session ref session_id, GPU_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PGPU ref set`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given GPU_group.

Signature:

```
GPU_group record get_record (session ref session_id, GPU_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `GPU_group record`

all fields from the object

RPC name: get_remaining_capacity

Overview:

Signature:

```
int get_remaining_capacity (session ref session_id, GPU_group ref self, VGPU_type ref
vgpu_type)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	The GPU group to query
`VGPU_type ref`	vgpu_type	The VGPU_type for which the remaining capacity will be calculated

Minimum Role: read-only

Return Type: `int`

The number of VGPUs of the given type which can still be started on the PGPUs in the group

RPC name: get_supported_VGPU_types

Overview:

Get the supported_VGPU_types field of the given GPU_group.

Signature:

```
VGPU_type ref set get_supported_VGPU_types (session ref session_id, GPU_group ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VGPU_type ref set`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given GPU_group.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VGPUs

Overview:

Get the VGPUs field of the given GPU_group.

Signature:

```
VGPU ref set get_VGPUs (session ref session_id, GPU_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`GPU_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VGPU ref set`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given GPU_group. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, GPU_group ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_allocation_algorithm

Overview:

Set the allocation_algorithm field of the given GPU_group.

Signature:

```
void set_allocation_algorithm (session ref session_id, GPU_group ref self,
allocation_algorithm value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`GPU_group ref`	self	reference to the object
`allocation_algorithm`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_name_description

Overview:

Set the name/description field of the given GPU_group.

Signature:

```
void set_name_description (session ref session_id, GPU_group ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_name_label

Overview:

Set the name/label field of the given GPU_group.

Signature:

```
void set_name_label (session ref session_id, GPU_group ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object

type	name	description
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given GPU_group.

Signature:

```
void set_other_config (session ref session_id, GPU_group ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`GPU_group ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: host

A physical host

Fields for class: host

Field	Type	Qualifier	Description
address	`string`	*RW*	The address by which this host can be contacted from any other host in the pool

Field	Type	Qualifier	Description
allowed_operations	host_allowed_operations set	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
API_version_major	int	*RO/runtime*	major version number
API_version_minor	int	*RO/runtime*	minor version number
API_version_vendor	string	*RO/runtime*	identification of vendor
API_version_vendor_implementation	(string -> string) map	*RO/runtime*	details of vendor implementation
bios_strings	(string -> string) map	*RO/runtime*	BIOS strings
blobs	(string -> blob ref) map	*RO/runtime*	Binary blobs associated with this host
capabilities	string set	*RO/constructor*	Xen capabilities
certificates	Certificate ref set	*RO/runtime*	List of certificates installed in the host
chipset_info	(string -> string) map	*RO/runtime*	Information about chipset features
control_domain	VM ref	*RO/runtime*	The control domain (domain 0)
cpu_configuration	(string -> string) map	*RO/runtime*	The CPU configuration on this host. May contain keys such as "nr_nodes", "sockets_per_node", "cores_per_socket", or "threads_per_core"
cpu_info	(string -> string) map	*RO/runtime*	Details about the physical CPUs on this host

Field	Type	Qualifier	Description
crash_dump_sr	SR ref	*RW*	The SR in which VDIs for crash dumps are created
crashdumps	host_crashdump ref set	*RO/runtime*	Set of host crash dumps
current_operations	(string -> host_allowed_operations) map	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.
display	host_display	*RW*	indicates whether the host is configured to output its console to a physical display device
edition	string	*RO/runtime*	Product edition
editions	string set	*RO/runtime*	List of all available product editions
enabled	bool	*RO/runtime*	True if the host is currently enabled
external_auth_configuration	(string -> string) map	*RO/runtime*	configuration specific to external authentication service
external_auth_service_name	string	*RO/runtime*	name of external authentication service configured; empty if none configured.
external_auth_type	string	*RO/runtime*	type of external authentication service configured; empty if none configured.
features	Feature ref set	*RO/runtime*	List of features available on this host

Field	Type	Qualifier	Description
guest_VCPUs_params	(string -> string) map	*RW*	VCPUs params to apply to all resident guests
ha_network_peers	string set	*RO/runtime*	The set of hosts visible via the network from this host
ha_statefiles	string set	*RO/runtime*	The set of statefiles accessible from this host
host_CPUs	host_cpu ref set	*RO/runtime*	The physical CPUs on this host
hostname	string	*RW*	The hostname of this host
iscsi_iqn	string	*RO/constructor*	The initiator IQN for the host
license_params	(string -> string) map	*RO/runtime*	State of the current license
license_server	(string -> string) map	*RW*	Contact information of the license server
local_cache_sr	SR ref	*RO/constructor*	The SR that is used as a local cache
logging	(string -> string) map	*RW*	logging configuration
memory_overhead	int	*RO/runtime*	Virtualization memory overhead (bytes).
metrics	host_metrics ref	*RO/runtime*	metrics associated with this host
multipathing	bool	*RO/constructor*	Specifies whether multipathing is enabled
name_description	string	*RW*	a notes field containing human-readable description
name_label	string	*RW*	a human-readable name
other_config	(string -> string) map	*RW*	additional configuration

Field	Type	Qualifier	Description
patches	`host_patch ref set`	*RO/runtime*	**Deprecated**. Set of host patches
PBDs	`PBD ref set`	*RO/runtime*	physical blockdevices
PCIs	`PCI ref set`	*RO/runtime*	List of PCI devices in the host
PGPUs	`PGPU ref set`	*RO/runtime*	List of physical GPUs in the host
PIFs	`PIF ref set`	*RO/runtime*	physical network interfaces
power_on_config	`(string -> string) map`	*RO/runtime*	The power on config
power_on_mode	`string`	*RO/runtime*	The power on mode
PUSBs	`PUSB ref set`	*RO/runtime*	List of physical USBs in the host
resident_VMs	`VM ref set`	*RO/runtime*	list of VMs currently resident on host
sched_policy	`string`	*RO/runtime*	Scheduler policy currently in force on this host
software_version	`(string -> string) map`	*RO/constructor*	version strings
ssl_legacy	`bool`	*RO/constructor*	**Deprecated**. Allow SSLv3 protocol and ciphersuites as used by older server versions. This controls both incoming and outgoing connections. When this is set to a different value, the host immediately restarts its SSL/TLS listening service; typically this takes less than a second but existing connections to it will be broken. API login sessions will remain valid.

Field	Type	Qualifier	Description
supported_bootloaders	`string set`	*RO/runtime*	a list of the bootloaders installed on the machine
suspend_image_sr	`SR ref`	*RW*	The SR in which VDIs for suspend images are created
tags	`string set`	*RW*	user-specified tags for categorization purposes
uefi_certificates	`string`	*RO/constructor*	The UEFI certificates allowing Secure Boot
updates	`pool_update ref set`	*RO/runtime*	Set of updates
updates_requiring_reboot	`pool_update ref set`	*RO/runtime*	List of updates which require reboot
uuid	`string`	*RO/runtime*	Unique identifier/object reference
virtual_hardware_platform_versions	`int set`	*RO/runtime*	The set of versions of the virtual hardware platform that the host can offer to its guests

RPCs associated with class: host

RPC name: add_tags

Overview:

Add the given value to the tags field of the given host. If the value is already in that Set, then do nothing.

Signature:

```
void add_tags (session ref session_id, host ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

type	name	description
`string`	value	New value to add

Minimum Role: vm-operator

Return Type: `void`

RPC name: add_to_guest_VCPUs_params

Overview:

Add the given key-value pair to the guest_VCPUs_params field of the given host.

Signature:

```
void add_to_guest_VCPUs_params (session ref session_id, host ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: add_to_license_server

Overview:

Add the given key-value pair to the license_server field of the given host.

Signature:

```
void add_to_license_server (session ref session_id, host ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

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

Minimum Role: pool-operator

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 ref session_id, host ref self, string key, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

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 ref session_id, host ref self, string key, string
value)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: apply_edition

Overview:

Change to another edition, or reactivate the current edition after a license has expired. This may be subject to the successful checkout of an appropriate license.

Signature:

```
void apply_edition (session ref session_id, host ref host, string edition, bool force)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host
`string`	edition	The requested edition
`bool`	force	Update the license params even if the apply call fails

Minimum Role: pool-operator

Return Type: `void`

RPC name: assert_can_evacuate

Overview:

Check this host can be evacuated.

Signature:

```
void assert_can_evacuate (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host to evacuate

Minimum Role: pool-operator

Return Type: `void`

RPC name: backup_rrds

Overview:

This causes the RRDs to be backed up to the master

Signature:

```
void backup_rrds (session ref session_id, host ref host, float delay)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	Schedule a backup of the RRDs of this host
`float`	delay	Delay in seconds from when the call is received to perform the backup

Minimum Role: pool-admin

Return Type: `void`

RPC name: bugreport_upload

Overview:

Run xen-bugtool --yestoall and upload the output to support

Signature:

```
void bugreport_upload (session ref session_id, host ref host, string url, (string ->
string) map options)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host on which to run xen-bugtool

type	name	description
`string`	url	The URL to upload to
`(string -> string) map`	options	Extra configuration operations

Minimum Role: pool-operator

Return Type: `void`

RPC name: call_extension

Overview:

Call an API extension on this host

Signature:

```
string call_extension (session ref session_id, host ref host, string call)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host
`string`	call	Rpc call for the extension

Minimum Role: pool-admin

Return Type: `string`

Result from the extension

RPC name: call_plugin

Overview:

Call an API plugin on this host

Signature:

```
string call_plugin (session ref session_id, host ref host, string plugin, string fn,
(string -> string) map args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`host ref`	host	The host
`string`	plugin	The name of the plugin
`string`	fn	The name of the function within the plugin
`(string -> string) map`	args	Arguments for the function

Minimum Role: pool-admin

Return Type: `string`

Result from the plugin

RPC name: compute_free_memory

Overview:

Computes the amount of free memory on the host.

Signature:

```
int compute_free_memory (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host to send the request to

Minimum Role: read-only

Return Type: `int`

the amount of free memory on the host.

RPC name: compute_memory_overhead

Overview:

Computes the virtualization memory overhead of a host.

Signature:

```
int compute_memory_overhead (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host for which to compute the memory overhead

Minimum Role: read-only

Return Type: `int`

the virtualization memory overhead of the host.

RPC name: create_new_blob

Overview:

Create a placeholder for a named binary blob of data that is associated with this host

Signature:

```
blob ref create_new_blob (session ref session_id, host ref host, string name, string
mime_type, bool public)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host
`string`	name	The name associated with the blob
`string`	mime_type	The mime type for the data. Empty string translates to application/octet-stream
`bool`	public	True if the blob should be publicly available

Minimum Role: pool-operator

Return Type: `blob ref`

The reference of the blob, needed for populating its data

RPC name: declare_dead

Overview:

Declare that a host is dead. This is a dangerous operation, and should only be called if the administrator is absolutely sure the host is definitely dead

Signature:

```
void declare_dead (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The Host to declare is dead

Minimum Role: pool-operator

Return Type: `void`

RPC name: destroy

Overview:

Destroy specified host record in database

Signature:

```
void destroy (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	The host record to remove

Minimum Role: pool-operator

Return Type: `void`

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 ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The Host to disable

Minimum Role: pool-operator

Return Type: `void`

RPC name: disable_display

Overview:

Disable console output to the physical display device next time this host boots

Signature:

```
host_display disable_display (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host

Minimum Role: pool-operator

Return Type: `host_display`

This host's physical display usage

RPC name: disable_external_auth

Overview:

This call disables external authentication on the local host

Signature:

```
void disable_external_auth (session ref session_id, host ref host, (string -> string)
map config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host whose external authentication should be disabled
`(string -> string) map`	config	Optional parameters as a list of key-values containing the configuration data

Minimum Role: pool-admin

Return Type: `void`

RPC name: disable_local_storage_caching

Overview:

Disable the use of a local SR for caching purposes

Signature:

```
void disable_local_storage_caching (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host

Minimum Role: pool-operator

Return Type: `void`

RPC name: dmesg

Overview:

Get the host xen dmesg.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The Host to query

Minimum Role: pool-operator

Return Type: `string`

dmesg string

RPC name: dmesg_clear

Overview:

Get the host xen dmesg, and clear the buffer.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The Host to query

Minimum Role: pool-operator

Return Type: `string`

dmesg string

RPC name: emergency_ha_disable

Overview:

This call disables HA on the local host. This should only be used with extreme care.

Signature:

```
void emergency_ha_disable (session ref session_id, bool soft)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`bool`	soft	Disable HA temporarily, revert upon host reboot or further changes, idempotent

Minimum Role: pool-operator

Return Type: `void`

RPC name: emergency_reset_server_certificate

Overview:

Delete the current TLS server certificate and replace by a new, self-signed one. This should only be used with extreme care.

Signature:

```

```

```
void emergency_reset_server_certificate (session ref session_id)
```

Return Type: `void`

RPC name: enable

Overview:

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

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The Host to enable

Minimum Role: pool-operator

Return Type: `void`

RPC name: enable_display

Overview:

Enable console output to the physical display device next time this host boots

Signature:

```
host_display enable_display (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host

Minimum Role: pool-operator

Return Type: `host_display`

This host's physical display usage

RPC name: enable_external_auth

Overview:

This call enables external authentication on a host

Signature:

```
void enable_external_auth (session ref session_id, host ref host, (string -> string)
map config, string service_name, string auth_type)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host whose external authentication should be enabled
`(string -> string) map`	config	A list of key-values containing the configuration data
`string`	service_name	The name of the service
`string`	auth_type	The type of authentication (e.g. AD for Active Directory)

Minimum Role: pool-admin

Return Type: `void`

RPC name: enable_local_storage_caching

Overview:

Enable the use of a local SR for caching purposes

Signature:

```
void enable_local_storage_caching (session ref session_id, host ref host, SR ref sr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host
`SR ref`	sr	The SR to use as a local cache

Minimum Role: pool-operator

Return Type: `void`

RPC name: evacuate

Overview:

Migrate all VMs off of this host, where possible.

Signature:

```
void evacuate (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host to evacuate

Minimum Role: pool-operator

Return Type: `void`

RPC name: forget_data_source_archives

Overview:

Forget the recorded statistics related to the specified data source

Signature:

```
void forget_data_source_archives (session ref session_id, host ref host, string
data_source)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host
`string`	data_source	The data source whose archives are to be forgotten

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_address

Overview:

Get the address field of the given host.

Signature:

```
string get_address (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_all

Overview:

Return a list of all the hosts known to the system.

Signature:

```
host ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `host ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of host references to host records for all hosts known to the system.

Signature:

```
(host ref -> host record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(host ref -> host record) map`

records of all objects

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given host.

Signature:

```
host_allowed_operations set get_allowed_operations (session ref session_id, host ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host_allowed_operations set`

value of the field

RPC name: get_API_version_major

Overview:

Get the API_version/major field of the given host.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id,
host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_bios_strings

Overview:

Get the bios_strings field of the given host.

Signature:

```
(string -> string) map get_bios_strings (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_blobs

Overview:

Get the blobs field of the given host.

Signature:

```
(string -> blob ref) map get_blobs (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> blob ref) map`

value of the field

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 ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `host ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the host instance with the specified UUID.

Signature:

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

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `host ref`

reference to the object

RPC name: get_capabilities

Overview:

Get the capabilities field of the given host.

Signature:

```
string set get_capabilities (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_certificates

Overview:

Get the certificates field of the given host.

Signature:

```
Certificate ref set get_certificates (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `Certificate ref set`

value of the field

RPC name: get_chipset_info

Overview:

Get the chipset_info field of the given host.

Signature:

```
(string -> string) map get_chipset_info (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_control_domain

Overview:

Get the control_domain field of the given host.

Signature:

```
VM ref get_control_domain (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref`

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 ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_cpu_info

Overview:

Get the cpu_info field of the given host.

Signature:

```
(string -> string) map get_cpu_info (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

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 ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SR ref`

value of the field

RPC name: get_crashdumps

Overview:

Get the crashdumps field of the given host.

Signature:

```
host_crashdump ref set get_crashdumps (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host_crashdump ref set`

value of the field

RPC name: get_current_operations

Overview:

Get the current_operations field of the given host.

Signature:

```
(string -> host_allowed_operations) map get_current_operations (session ref
session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> host_allowed_operations) map`

value of the field

RPC name: get_data_sources

Overview:

Signature:

```
data_source record set get_data_sources (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host to interrogate

Minimum Role: read-only

Return Type: `data_source record set`

A set of data sources

RPC name: get_display

Overview:

Get the display field of the given host.

Signature:

```
host_display get_display (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host_display`

value of the field

RPC name: get_edition

Overview:

Get the edition field of the given host.

Signature:

```
string get_edition (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_editions

Overview:

Get the editions field of the given host.

Signature:

```
string set get_editions (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_enabled

Overview:

Get the enabled field of the given host.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_external_auth_configuration

Overview:

Get the external_auth_configuration field of the given host.

Signature:

```
(string -> string) map get_external_auth_configuration (session ref session_id, host
ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_external_auth_service_name

Overview:

Get the external_auth_service_name field of the given host.

Signature:

```
string get_external_auth_service_name (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_external_auth_type

Overview:

Get the external_auth_type field of the given host.

Signature:

```
string get_external_auth_type (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_features

Overview:

Get the features field of the given host.

Signature:

```
Feature ref set get_features (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `Feature ref set`

value of the field

RPC name: get_guest_VCPUs_params

Overview:

Get the guest_VCPUs_params field of the given host.

Signature:

```
(string -> string) map get_guest_VCPUs_params (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_ha_network_peers

Overview:

Get the ha_network_peers field of the given host.

Signature:

```
string set get_ha_network_peers (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_ha_statefiles

Overview:

Get the ha_statefiles field of the given host.

Signature:

```
string set get_ha_statefiles (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string 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 ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host_cpu ref set`

value of the field

RPC name: get_hostname

Overview:

Get the hostname field of the given host.

Signature:

```
string get_hostname (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_iscsi_iqn

Overview:

Get the iscsi_iqn field of the given host.

Signature:

```
string get_iscsi_iqn (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_license_params

Overview:

Get the license_params field of the given host.

Signature:

```
(string -> string) map get_license_params (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_license_server

Overview:

Get the license_server field of the given host.

Signature:

```
(string -> string) map get_license_server (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_local_cache_sr

Overview:

Get the local_cache_sr field of the given host.

Signature:

```
SR ref get_local_cache_sr (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SR ref`

value of the field

RPC name: get_log

Overview:

Get the host's log file

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The Host to query

Minimum Role: read-only

Return Type: `string`

The contents of the host's primary log file

RPC name: get_logging

Overview:

Get the logging field of the given host.

Signature:

```
(string -> string) map get_logging (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_management_interface

Overview:

Returns the management interface for the specified host

Signature:

```
PIF ref get_management_interface (session ref session_id, host ref host)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	Which host's management interface is required

Minimum Role: pool-operator

Return Type: `PIF ref`

The management interface for the host

RPC name: get_memory_overhead

Overview:

Get the memory/overhead field of the given host.

Signature:

```
int get_memory_overhead (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_metrics

Overview:

Get the metrics field of the given host.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host_metrics ref`

value of the field

RPC name: get_multipathing

Overview:

Get the multipathing field of the given host.

Signature:

```
bool get_multipathing (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given host.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

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 ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_patches

This message is deprecated.

Overview:

Get the patches field of the given host.

Signature:

```
host_patch ref set get_patches (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host_patch ref set`

value of the field

RPC name: get_PBDs

Overview:

Get the PBDs field of the given host.

Signature:

```
PBD ref set get_PBDs (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PBD ref set`

value of the field

RPC name: get_PCIs

Overview:

Get the PCIs field of the given host.

Signature:

```
PCI ref set get_PCIs (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PCI ref set`

value of the field

RPC name: get_PGPUs

Overview:

Get the PGPUs field of the given host.

Signature:

```
PGPU ref set get_PGPUs (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PGPU ref set`

value of the field

RPC name: get_PIFs

Overview:

Get the PIFs field of the given host.

Signature:

```
PIF ref set get_PIFs (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF ref set`

value of the field

RPC name: get_power_on_config

Overview:

Get the power_on_config field of the given host.

Signature:

```
(string -> string) map get_power_on_config (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_power_on_mode

Overview:

Get the power_on_mode field of the given host.

Signature:

```
string get_power_on_mode (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_PUSBs

Overview:

Get the PUSBs field of the given host.

Signature:

```
PUSB ref set get_PUSBs (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PUSB ref set`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given host.

Signature:

```
host record get_record (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host record`

all fields from the object

RPC name: get_resident_VMs

Overview:

Get the resident_VMs field of the given host.

Signature:

```
VM ref set get_resident_VMs (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref set`

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 ref session_id, host ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_server_certificate

Overview:

Get the installed server public TLS certificate.

Signature:

```
string get_server_certificate (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host

Minimum Role: read-only

Return Type: `string`

The installed server public TLS certificate, in PEM form.

RPC name: get_server_localtime

Overview:

This call queries the host's clock for the current time in the host's local timezone

Signature:

```
datetime get_server_localtime (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`host ref`	host	The host whose clock should be queried

Minimum Role: read-only

Return Type: `datetime`

The current local time

RPC name: get_servertime

Overview:

This call queries the host's clock for the current time

Signature:

```
datetime get_servertime (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host whose clock should be queried

Minimum Role: read-only

Return Type: `datetime`

The current time

RPC name: get_software_version

Overview:

Get the software_version field of the given host.

Signature:

```
(string -> string) map get_software_version (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_ssl_legacy

This message is deprecated.

Overview:

Get the ssl_legacy field of the given host.

Signature:

```
bool get_ssl_legacy (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

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 ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string 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 ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SR ref`

value of the field

RPC name: get_system_status_capabilities

Overview:

Signature:

```
string get_system_status_capabilities (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host to interrogate

Minimum Role: read-only

Return Type: `string`

An XML fragment containing the system status capabilities.

RPC name: get_tags

Overview:

Get the tags field of the given host.

Signature:

```
string set get_tags (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_uefi_certificates

Overview:

Get the uefi_certificates field of the given host.

Signature:

```
string get_uefi_certificates (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_uncooperative_resident_VMs

This message is deprecated.

Overview:

Return a set of VMs which are not co-operating with the host's memory control system

Signature:

```
VM ref set get_uncooperative_resident_VMs (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	The host to query

Minimum Role: read-only

Return Type: `VM ref set`

VMs which are not co-operating

RPC name: get_updates

Overview:

Get the updates field of the given host.

Signature:

```
pool_update ref set get_updates (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `pool_update ref set`

value of the field

RPC name: get_updates_requiring_reboot

Overview:

Get the updates_requiring_reboot field of the given host.

Signature:

```
pool_update ref set get_updates_requiring_reboot (session ref session_id, host ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `pool_update ref set`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given host.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_virtual_hardware_platform_versions

Overview:

Get the virtual_hardware_platform_versions field of the given host.

Signature:

```
int set get_virtual_hardware_platform_versions (session ref session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int set`

value of the field

RPC name: get_vms_which_prevent_evacuation

Overview:

Return a set of VMs which prevent the host being evacuated, with per-VM error codes

Signature:

```
(VM ref -> string set) map get_vms_which_prevent_evacuation (session ref session_id,
host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	The host to query

Minimum Role: read-only

Return Type: `(VM ref -> string set) map`

VMs which block evacuation together with reasons

RPC name: has_extension

Overview:

Return true if the extension is available on the host

Signature:

```
bool has_extension (session ref session_id, host ref host, string name)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host
`string`	name	The name of the API call

Minimum Role: pool-admin

Return Type: `bool`

True if the extension exists, false otherwise

RPC name: install_server_certificate

Overview:

Install the TLS server certificate.

Signature:

```
void install_server_certificate (session ref session_id, host ref host, string
certificate, string private_key, string certificate_chain)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host
`string`	certificate	The server certificate, in PEM form
`string`	private_key	The unencrypted private key used to sign the certificate, in PKCS#8 form
`string`	certificate_chain	The certificate chain, in PEM form

Minimum Role: pool-admin

Return Type: `void`

RPC name: license_add

Overview:

Apply a new license to a host

Signature:

```
void license_add (session ref session_id, host ref host, string contents)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host to upload the license to
`string`	contents	The contents of the license file, base64 encoded

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `LICENSE_PROCESSING_ERROR`

RPC name: license_apply

This message is removed.

Overview:

Apply a new license to a host

Signature:

```
void license_apply (session ref session_id, host ref host, string contents)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host to upload the license to
`string`	contents	The contents of the license file, base64 encoded

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `LICENSE_PROCESSING_ERROR`

RPC name: license_remove

Overview:

Remove any license file from the specified host, and switch that host to the unlicensed edition

Signature:

```
void license_remove (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host from which any license will be removed

Minimum Role: pool-operator

Return Type: `void`

RPC name: list_methods

Overview:

List all supported methods

Signature:

```
string set list_methods (session ref session_id)
```

Minimum Role: read-only

Return Type: `string set`

The name of every supported method.

RPC name: local_management_reconfigure

Overview:

Reconfigure the management network interface. Should only be used if Host.management_reconfigure is impossible because the network configuration is broken.

Signature:

```
void local_management_reconfigure (session ref session_id, string interface)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	interface	name of the interface to use as a management interface

Minimum Role: pool-operator

Return Type: `void`

RPC name: management_disable

Overview:

Disable the management network interface

Signature:

```
void management_disable (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `void`

RPC name: management_reconfigure

Overview:

Reconfigure the management network interface

Signature:

```
void management_reconfigure (session ref session_id, PIF ref pif)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	pif	reference to a PIF object corresponding to the management interface

Minimum Role: pool-operator

Return Type: `void`

RPC name: migrate_receive

Overview:

Prepare to receive a VM, returning a token which can be passed to VM.migrate.

Signature:

```
(string -> string) map migrate_receive (session ref session_id, host ref host, network
ref network, (string -> string) map options)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`host ref`	host	The target host
`network ref`	network	The network through which migration traffic should be received.
`(string -> string) map`	options	Extra configuration operations

Minimum Role: vm-power-admin

Return Type: `(string -> string) map`

A value which should be passed to VM.migrate

RPC name: power_on

Overview:

Attempt to power-on the host (if the capability exists).

Signature:

```
void power_on (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The Host to power on

Minimum Role: pool-operator

Return Type: `void`

RPC name: query_data_source

Overview:

Query the latest value of the specified data source

Signature:

```
float query_data_source (session ref session_id, host ref host, string data_source)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`host ref`	host	The host
`string`	data_source	The data source to query

Minimum Role: read-only

Return Type: `float`

The latest value, averaged over the last 5 seconds

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 ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The Host to reboot

Minimum Role: pool-operator

Return Type: `void`

RPC name: record_data_source

Overview:

Start recording the specified data source

Signature:

```
void record_data_source (session ref session_id, host ref host, string data_source)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host

type	name	description
`string`	data_source	The data source to record

Minimum Role: pool-operator

Return Type: `void`

RPC name: refresh_pack_info

This message is deprecated.

Overview:

Refresh the list of installed Supplemental Packs.

Signature:

```
void refresh_pack_info (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The Host to modify

Minimum Role: pool-operator

Return Type: `void`

RPC name: remove_from_guest_VCPUs_params

Overview:

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

Signature:

```
void remove_from_guest_VCPUs_params (session ref session_id, host ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

type	name	description
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: remove_from_license_server

Overview:

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

Signature:

```
void remove_from_license_server (session ref session_id, host ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

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 ref session_id, host ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

type	name	description
`string`	key	Key to remove

Minimum Role: pool-operator

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 ref session_id, host ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: remove_tags

Overview:

Remove the given value from the tags field of the given host. If the value is not in that Set, then do nothing.

Signature:

```
void remove_tags (session ref session_id, host ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`string`	value	Value to remove

Minimum Role: vm-operator

Return Type: `void`

RPC name: reset_cpu_features

This message is removed.

Overview:

Remove the feature mask, such that after a reboot all features of the CPU are enabled.

Signature:

```
void reset_cpu_features (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host

Minimum Role: pool-operator

Return Type: `void`

RPC name: restart_agent

Overview:

Restarts the agent after a 10 second pause. WARNING: this is a dangerous operation. Any operations in progress will be aborted, and unrecoverable data loss may occur. The caller is responsible for ensuring that there are no operations in progress when this method is called.

Signature:

```
void restart_agent (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The Host on which you want to restart the agent

Minimum Role: pool-operator

Return Type: `void`

RPC name: retrieve_wlb_evacuate_recommendations

Overview:

Retrieves recommended host migrations to perform when evacuating the host from the wlb server. If a VM cannot be migrated from the host the reason is listed instead of a recommendation.

Signature:

```
(VM ref -> string set) map retrieve_wlb_evacuate_recommendations (session ref
session_id, host ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	The host to query

Minimum Role: read-only

Return Type: `(VM ref -> string set) map`

VMs and the reasons why they would block evacuation, or their target host recommended by the wlb server

RPC name: send_debug_keys

Overview:

Inject the given string as debugging keys into Xen

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host
`string`	keys	The keys to send

Minimum Role: pool-admin

Return Type: `void`

RPC name: set_address

Overview:

Set the address field of the given host.

Signature:

```
void set_address (session ref session_id, host ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_cpu_features

This message is removed.

Overview:

Set the CPU features to be used after a reboot, if the given features string is valid.

Signature:

```
void set_cpu_features (session ref session_id, host ref host, string features)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host
`string`	features	The features string (32 hexadecimal digits)

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_crash_dump_sr

Overview:

Set the crash_dump_sr field of the given host.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`SR ref`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_display

Overview:

Set the display field of the given host.

Signature:

```
void set_display (session ref session_id, host ref self, host_display value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`host_display`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_guest_VCPUs_params

Overview:

Set the guest_VCPUs_params field of the given host.

Signature:

```
void set_guest_VCPUs_params (session ref session_id, host ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_hostname

Overview:

Set the hostname field of the given host.

Signature:

```
void set_hostname (session ref session_id, host ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_hostname_live

Overview:

Sets the host name to the specified string. Both the API and lower-level system hostname are changed immediately.

Signature:

```
void set_hostname_live (session ref session_id, host ref host, string hostname)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host whose host name to set
`string`	hostname	The new host name

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `HOST_NAME_INVALID`

RPC name: set_iscsi_iqn

Overview:

Sets the initiator IQN for the host

Signature:

```
void set_iscsi_iqn (session ref session_id, host ref host, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host
`string`	value	The value to which the IQN should be set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_license_server

Overview:

Set the license_server field of the given host.

Signature:

```
void set_license_server (session ref session_id, host ref self, (string -> string) map
value)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_logging

Overview:

Set the logging field of the given host.

Signature:

```
void set_logging (session ref session_id, host ref self, (string -> string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_multipathing

Overview:

Specifies whether multipathing is enabled

Signature:

```
void set_multipathing (session ref session_id, host ref host, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`host ref`	host	The host
`bool`	value	Whether multipathing should be enabled

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_name_description

Overview:

Set the name/description field of the given host.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_name_label

Overview:

Set the name/label field of the given host.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object

type	name	description
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given host.

Signature:

```
void set_other_config (session ref session_id, host ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_power_on_mode

Overview:

Set the power-on-mode, host, user and password

Signature:

```
void set_power_on_mode (session ref session_id, host ref self, string power_on_mode,
(string -> string) map power_on_config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	The host

type	name	description
`string`	power_on_mode	power-on-mode can be empty, wake-on-lan, DRAC or other
`(string -> string) map`	power_on_config	Power on config

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_ssl_legacy

Overview:

Enable/disable SSLv3 for interoperability with older server versions. When this is set to a different value, the host immediately restarts its SSL/TLS listening service; typically this takes less than a second but existing connections to it will be broken. API login sessions will remain valid.

Signature:

```
void set_ssl_legacy (session ref session_id, host ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	The host
`bool`	value	True to allow SSLv3 and ciphersuites as used in old XenServer versions

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_suspend_image_sr

Overview:

Set the suspend_image_sr field of the given host.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

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

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_tags

Overview:

Set the tags field of the given host.

Signature:

```
void set_tags (session ref session_id, host ref self, string set value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	self	reference to the object
`string set`	value	New value to set

Minimum Role: vm-operator

Return Type: `void`

RPC name: set_uefi_certificates

Overview:

Sets the UEFI certificates on a host

Signature:

```
void set_uefi_certificates (session ref session_id, host ref host, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host

type	name	description
`string`	value	The certificates to apply to a host

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 ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The Host to shutdown

Minimum Role: pool-operator

Return Type: `void`

RPC name: shutdown_agent

Overview:

Shuts the agent down after a 10 second pause. WARNING: this is a dangerous operation. Any operations in progress will be aborted, and unrecoverable data loss may occur. The caller is responsible for ensuring that there are no operations in progress when this method is called.

Signature:

```
void shutdown_agent (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `void`

RPC name: sync_data

Overview:

This causes the synchronisation of the non-database data (messages, RRDs and so on) stored on the master to be synchronised with the host

Signature:

```
void sync_data (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host to whom the data should be sent

Minimum Role: pool-admin

Return Type: `void`

RPC name: syslog_reconfigure

Overview:

Re-configure syslog logging

Signature:

```
void syslog_reconfigure (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	Tell the host to reread its Host.logging parameters and reconfigure itself accordingly

Minimum Role: pool-operator

Return Type: `void`

Class: host_cpu

This class is deprecated.

A physical CPU

Fields for class: host_cpu

Field	Type	Qualifier	Description
family	`int`	*RO/runtime*	**Deprecated**. the family (number) of the physical CPU

Field	Type	Qualifier	Description
features	`string`	*RO/runtime*	**Deprecated**. the physical CPU feature bitmap
flags	`string`	*RO/runtime*	**Deprecated**. the flags of the physical CPU (a decoded version of the features field)
host	`host ref`	*RO/runtime*	**Deprecated**. the host the CPU is in
model	`int`	*RO/runtime*	**Deprecated**. the model number of the physical CPU
modelname	`string`	*RO/runtime*	**Deprecated**. the model name of the physical CPU
number	`int`	*RO/runtime*	**Deprecated**. the number of the physical CPU within the host
other_config	`(string -> string) map`	*RW*	**Deprecated**. additional configuration
speed	`int`	*RO/runtime*	**Deprecated**. the speed of the physical CPU
stepping	`string`	*RO/runtime*	**Deprecated**. the stepping of the physical CPU
utilisation	`float`	*RO/runtime*	**Deprecated**. the current CPU utilisation
uuid	`string`	*RO/runtime*	**Deprecated**. Unique identifier/object reference
vendor	`string`	*RO/runtime*	**Deprecated**. the vendor of the physical CPU

RPCs associated with class: host_cpu

RPC name: add_to_other_config

This message is deprecated.

Overview:

Add the given key-value pair to the other_config field of the given host_cpu.

Signature:

```
void add_to_other_config (session ref session_id, host_cpu ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

This message is deprecated.

Overview:

Return a list of all the host_cpus known to the system.

Signature:

```
host_cpu ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `host_cpu ref set`

references to all objects

RPC name: get_all_records

This message is deprecated.

Overview:

Return a map of host_cpu references to host_cpu records for all host_cpus known to the system.

Signature:

```
(host_cpu ref -> host_cpu record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(host_cpu ref -> host_cpu record) map`

records of all objects

RPC name: get_by_uuid

This message is deprecated.

Overview:

Get a reference to the host_cpu instance with the specified UUID.

Signature:

```
host_cpu ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `host_cpu ref`

reference to the object

RPC name: get_family

This message is deprecated.

Overview:

Get the family field of the given host_cpu.

Signature:

```
int get_family (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_features

This message is deprecated.

Overview:

Get the features field of the given host_cpu.

Signature:

```
string get_features (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_flags

This message is deprecated.

Overview:

Get the flags field of the given host_cpu.

Signature:

```
string get_flags (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_host

This message is deprecated.

Overview:

Get the host field of the given host_cpu.

Signature:

```
host ref get_host (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_model

This message is deprecated.

Overview:

Get the model field of the given host_cpu.

Signature:

```
int get_model (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_modelname

This message is deprecated.

Overview:

Get the modelname field of the given host_cpu.

Signature:

```
string get_modelname (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_number

This message is deprecated.

Overview:

Get the number field of the given host_cpu.

Signature:

```
int get_number (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_other_config

This message is deprecated.

Overview:

Get the other_config field of the given host_cpu.

Signature:

```
(string -> string) map get_other_config (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

This message is deprecated.

Overview:

Get a record containing the current state of the given host_cpu.

Signature:

```
host_cpu record get_record (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host_cpu record`

all fields from the object

RPC name: get_speed

This message is deprecated.

Overview:

Get the speed field of the given host_cpu.

Signature:

```
int get_speed (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_stepping

This message is deprecated.

Overview:

Get the stepping field of the given host_cpu.

Signature:

```
string get_stepping (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_utilisation

This message is deprecated.

Overview:

Get the utilisation field of the given host_cpu.

Signature:

```
float get_utilisation (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `float`

value of the field

RPC name: get_uuid

This message is deprecated.

Overview:

Get the uuid field of the given host_cpu.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_vendor

This message is deprecated.

Overview:

Get the vendor field of the given host_cpu.

Signature:

```
string get_vendor (session ref session_id, host_cpu ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

This message is deprecated.

Overview:

Remove the given key and its corresponding value from the other_config field of the given host_cpu. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, host_cpu ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

This message is deprecated.

Overview:

Set the other_config field of the given host_cpu.

Signature:

```
void set_other_config (session ref session_id, host_cpu ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_cpu ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: host_crashdump

Represents a host crash dump

Fields for class: host_crashdump

Field	Type	Qualifier	Description
host	`host ref`	*RO/constructor*	Host the crashdump relates to
other_config	`(string -> string) map`	*RW*	additional configuration
size	`int`	*RO/runtime*	Size of the crashdump
timestamp	`datetime`	*RO/runtime*	Time the crash happened
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: host_crashdump

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given host_crashdump.

Signature:

```
void add_to_other_config (session ref session_id, host_crashdump ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_crashdump ref`	self	reference to the object
`string`	key	Key to add

type	name	description
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: destroy

Overview:

Destroy specified host crash dump, removing it from the disk.

Signature:

```
void destroy (session ref session_id, host_crashdump ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_crashdump ref`	self	The host crashdump to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the host_crashdumps known to the system.

Signature:

```
host_crashdump ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `host_crashdump ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of host_crashdump references to host_crashdump records for all host_crashdumps known to the system.

Signature:

```
(host_crashdump ref -> host_crashdump record) map get_all_records (session ref
session_id)
```

Minimum Role: read-only

Return Type: `(host_crashdump ref -> host_crashdump record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the host_crashdump instance with the specified UUID.

Signature:

```
host_crashdump ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `host_crashdump ref`

reference to the object

RPC name: get_host

Overview:

Get the host field of the given host_crashdump.

Signature:

```
host ref get_host (session ref session_id, host_crashdump ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_crashdump ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given host_crashdump.

Signature:

```
(string -> string) map get_other_config (session ref session_id, host_crashdump ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_crashdump ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given host_crashdump.

Signature:

```
host_crashdump record get_record (session ref session_id, host_crashdump ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`host_crashdump ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host_crashdump record`

all fields from the object

RPC name: get_size

Overview:

Get the size field of the given host_crashdump.

Signature:

```
int get_size (session ref session_id, host_crashdump ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_crashdump ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_timestamp

Overview:

Get the timestamp field of the given host_crashdump.

Signature:

```
datetime get_timestamp (session ref session_id, host_crashdump ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_crashdump ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given host_crashdump.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_crashdump ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given host_crashdump. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, host_crashdump ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_crashdump ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given host_crashdump.

Signature:

```
void set_other_config (session ref session_id, host_crashdump ref self, (string ->
string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_crashdump ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: upload

Overview:

Upload the specified host crash dump to a specified URL

Signature:

```
void upload (session ref session_id, host_crashdump ref self, string url, (string ->
string) map options)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_crashdump ref`	self	The host crashdump to upload
`string`	url	The URL to upload to
`(string -> string) map`	options	Extra configuration operations

Minimum Role: pool-operator

Return Type: `void`

Class: host_metrics

The metrics associated with a host

Fields for class: host_metrics

Field	Type	Qualifier	Description
last_updated	`datetime`	*RO/runtime*	Time at which this information was last updated
live	`bool`	*RO/runtime*	Pool master thinks this host is live
memory_free	`int`	*RO/runtime*	**Removed**. Free host memory (bytes)
memory_total	`int`	*RO/runtime*	Total host memory (bytes)
other_config	`(string -> string) map`	*RW*	additional configuration
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: host_metrics

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given host_metrics.

Signature:

```
void add_to_other_config (session ref session_id, host_metrics ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_metrics ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

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 ref session_id)
```

Minimum Role: read-only

Return Type: `host_metrics ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of host_metrics references to host_metrics records for all host_metrics instances known to the system.

Signature:

```
(host_metrics ref -> host_metrics record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(host_metrics ref -> host_metrics record) map`

records of all objects

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 ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `host_metrics ref`

reference to the object

RPC name: get_last_updated

Overview:

Get the last_updated field of the given host_metrics.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_live

Overview:

Get the live field of the given host_metrics.

Signature:

```
bool get_live (session ref session_id, host_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_memory_free

This message is removed.

Overview:

Get the memory/free field of the given host_metrics.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

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 ref session_id, host_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given host_metrics.

Signature:

```
(string -> string) map get_other_config (session ref session_id, host_metrics ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given host_metrics.

Signature:

```
host_metrics record get_record (session ref session_id, host_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host_metrics record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given host_metrics.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given host_metrics. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, host_metrics ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_metrics ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given host_metrics.

Signature:

```

```

```
void set_other_config (session ref session_id, host_metrics ref self, (string ->
string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_metrics ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: host_patch

This class is deprecated.

Represents a patch stored on a server

Fields for class: host_patch

Field	Type	Qualifier	Description
applied	`bool`	*RO/runtime*	**Deprecated**. True if the patch has been applied
host	`host ref`	*RO/constructor*	**Deprecated**. Host the patch relates to
name_description	`string`	*RO/constructor*	**Deprecated**. a notes field containing human-readable description
name_label	`string`	*RO/constructor*	**Deprecated**. a human-readable name
other_config	`(string -> string) map`	*RW*	**Deprecated**. additional configuration
pool_patch	`pool_patch ref`	*RO/constructor*	**Deprecated**. The patch applied
size	`int`	*RO/runtime*	**Deprecated**. Size of the patch
timestamp_applied	`datetime`	*RO/runtime*	**Deprecated**. Time the patch was applied
uuid	`string`	*RO/runtime*	**Deprecated**. Unique identifier/object reference
version	`string`	*RO/constructor*	**Deprecated**. Patch version number

RPCs associated with class: host_patch

RPC name: add_to_other_config

This message is deprecated.

Overview:

Add the given key-value pair to the other_config field of the given host_patch.

Signature:

```
void add_to_other_config (session ref session_id, host_patch ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: apply

This message is deprecated.

Overview:

Apply the selected patch and return its output

Signature:

```
string apply (session ref session_id, host_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	The patch to apply

Minimum Role: pool-operator

Return Type: `string`

the output of the patch application process

RPC name: destroy

This message is deprecated.

Overview:

Destroy the specified host patch, removing it from the disk. This does NOT reverse the patch

Signature:

```
void destroy (session ref session_id, host_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	The patch to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

This message is deprecated.

Overview:

Return a list of all the host_patchs known to the system.

Signature:

```
host_patch ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `host_patch ref set`

references to all objects

RPC name: get_all_records

This message is deprecated.

Overview:

Return a map of host_patch references to host_patch records for all host_patchs known to the system.

Signature:

```
```

```
(host_patch ref -> host_patch record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(host_patch ref -> host_patch record) map`

records of all objects

RPC name: get_applied

This message is deprecated.

Overview:

Get the applied field of the given host_patch.

Signature:

```
bool get_applied (session ref session_id, host_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_by_name_label

This message is deprecated.

Overview:

Get all the host_patch instances with the given label.

Signature:

```
host_patch ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `host_patch ref set`

references to objects with matching names

RPC name: get_by_uuid

This message is deprecated.

Overview:

Get a reference to the host_patch instance with the specified UUID.

Signature:

```
host_patch ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `host_patch ref`

reference to the object

RPC name: get_host

This message is deprecated.

Overview:

Get the host field of the given host_patch.

Signature:

```
host ref get_host (session ref session_id, host_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_name_description

This message is deprecated.

Overview:

Get the name/description field of the given host_patch.

Signature:

```
string get_name_description (session ref session_id, host_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

This message is deprecated.

Overview:

Get the name/label field of the given host_patch.

Signature:

```
string get_name_label (session ref session_id, host_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_other_config

This message is deprecated.

Overview:

Get the other_config field of the given host_patch.

Signature:

```
(string -> string) map get_other_config (session ref session_id, host_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_pool_patch

This message is deprecated.

Overview:

Get the pool_patch field of the given host_patch.

Signature:

```
pool_patch ref get_pool_patch (session ref session_id, host_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `pool_patch ref`

value of the field

RPC name: get_record

This message is deprecated.

Overview:

Get a record containing the current state of the given host_patch.

Signature:

```
host_patch record get_record (session ref session_id, host_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host_patch record`

all fields from the object

RPC name: get_size

This message is deprecated.

Overview:

Get the size field of the given host_patch.

Signature:

```
int get_size (session ref session_id, host_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_timestamp_applied

This message is deprecated.

Overview:

Get the timestamp_applied field of the given host_patch.

Signature:

```
datetime get_timestamp_applied (session ref session_id, host_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_uuid

This message is deprecated.

Overview:

Get the uuid field of the given host_patch.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_version

This message is deprecated.

Overview:

Get the version field of the given host_patch.

Signature:

```
string get_version (session ref session_id, host_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

This message is deprecated.

Overview:

Remove the given key and its corresponding value from the other_config field of the given host_patch. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, host_patch ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

This message is deprecated.

Overview:

Set the other_config field of the given host_patch.

Signature:

```
void set_other_config (session ref session_id, host_patch ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host_patch ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: LVHD

LVHD SR specific operations

Fields for class: LVHD

Field	Type	Qualifier	Description
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: LVHD

RPC name: enable_thin_provisioning

Overview:

Upgrades an LVHD SR to enable thin-provisioning. Future VDIs created in this SR will be thinly-provisioned, although existing VDIs will be left alone. Note that the SR must be attached to the SRmaster for upgrade to work.

Signature:

```
string enable_thin_provisioning (session ref session_id, host ref host, SR ref SR, int
initial_allocation, int allocation_quantum)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The LVHD Host to upgrade to being thin-provisioned.
`SR ref`	SR	The LVHD SR to upgrade to being thin-provisioned.
`int`	initial_allocation	The initial amount of space to allocate to a newly-created VDI in bytes
`int`	allocation_quantum	The amount of space to allocate to a VDI when it needs to be enlarged in bytes

Minimum Role: pool-admin

Return Type: `string`

Message from LVHD.enable_thin_provisioning extension

RPC name: get_by_uuid

Overview:

Get a reference to the LVHD instance with the specified UUID.

Signature:

```
LVHD ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `LVHD ref`

reference to the object

RPC name: get_record

Overview:

Get a record containing the current state of the given LVHD.

Signature:

```
LVHD record get_record (session ref session_id, LVHD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`LVHD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `LVHD record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given LVHD.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`LVHD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

Class: message

An message for the attention of the administrator

Fields for class: message

Field	Type	Qualifier	Description
body	`string`	*RO/runtime*	The body of the message
cls	`cls`	*RO/runtime*	The class of the object this message is associated with
name	`string`	*RO/runtime*	The name of the message
obj_uuid	`string`	*RO/runtime*	The uuid of the object this message is associated with
priority	`int`	*RO/runtime*	The message priority, 0 being low priority
timestamp	`datetime`	*RO/runtime*	The time at which the message was created
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: message

RPC name: create

Overview:

Signature:

```
message ref create (session ref session_id, string name, int priority, cls cls, string
obj_uuid, string body)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	name	The name of the message
`int`	priority	The priority of the message
`cls`	cls	The class of object this message is associated with
`string`	obj_uuid	The uuid of the object this message is associated with
`string`	body	The body of the message

Minimum Role: pool-operator

Return Type: `message ref`

The reference of the created message

RPC name: destroy

Overview:

Signature:

```
void destroy (session ref session_id, message ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`message ref`	self	The reference of the message to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get

Overview:

Signature:

```
(message ref -> message record) map get (session ref session_id, cls cls, string
obj_uuid, datetime since)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`cls`	cls	The class of object
`string`	obj_uuid	The uuid of the object
`datetime`	since	The cutoff time

Minimum Role: read-only

Return Type: `(message ref -> message record) map`

The relevant messages

RPC name: get_all

Overview:

Signature:

```
message ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `message ref set`

The references to the messages

RPC name: get_all_records

Overview:

Signature:

```
(message ref -> message record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(message ref -> message record) map`

The messages

RPC name: get_all_records_where

Overview:

Signature:

```
(message ref -> message record) map get_all_records_where (session ref session_id,
string expr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	expr	The expression to match (not currently used)

Minimum Role: read-only

Return Type: `(message ref -> message record) map`

The messages

RPC name: get_by_uuid

Overview:

Signature:

```
message ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	The uuid of the message

Minimum Role: read-only

Return Type: `message ref`

The message reference

RPC name: get_record

Overview:

Signature:

```
message record get_record (session ref session_id, message ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`message ref`	self	The reference to the message

Minimum Role: read-only

Return Type: `message record`

The message record

RPC name: get_since

Overview:

Signature:

```
(message ref -> message record) map get_since (session ref session_id, datetime since)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`datetime`	since	The cutoff time

Minimum Role: read-only

Return Type: `(message ref -> message record) map`

The relevant messages

Class: network

A virtual network

Fields for class: network

Field	Type	Qualifier	Description
allowed_operations	`network_operations set`	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
assigned_ips	`(VIF ref -> string) map`	*RO/runtime*	The IP addresses assigned to VIFs on networks that have active xapi-managed DHCP
blobs	`(string -> blob ref) map`	*RO/runtime*	Binary blobs associated with this network
bridge	`string`	*RO/constructor*	name of the bridge corresponding to this network on the local host
current_operations	`(string -> network_operations) map`	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.
default_locking_mode	`network_default_locking_mode`	*RO/runtime*	The network will use this value to determine the behaviour of all VIFs where locking_mode = default
managed	`bool`	*RO/constructor*	true if the bridge is managed by xapi
MTU	`int`	*RW*	MTU in octets
name_description	`string`	*RW*	a notes field containing human-readable description
name_label	`string`	*RW*	a human-readable name

Field	Type	Qualifier	Description
other_config	`(string -> string) map`	*RW*	additional configuration
PIFs	`PIF ref set`	*RO/runtime*	list of connected pifs
purpose	`network_purpose set`	*RO/runtime*	Set of purposes for which the server will use this network
tags	`string set`	*RW*	user-specified tags for categorization purposes
uuid	`string`	*RO/runtime*	Unique identifier/object reference
VIFs	`VIF ref set`	*RO/runtime*	list of connected vifs

RPCs associated with class: network

RPC name: add_purpose

Overview:

Give a network a new purpose (if not present already)

Signature:

```
void add_purpose (session ref session_id, network ref self, network_purpose value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	The network
`network_purpose`	value	The purpose to add

Minimum Role: pool-admin

Return Type: `void`

Possible Error Codes: `NETWORK_INCOMPATIBLE_PURPOSES`

RPC name: add_tags

Overview:

Add the given value to the tags field of the given network. If the value is already in that Set, then do nothing.

Signature:

```
```

```
void add_tags (session ref session_id, network ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
network ref	self	reference to the object
string	value	New value to add

Minimum Role: vm-operator

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 ref session_id, network ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
network ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: pool-operator

Return Type: void

RPC name: create

Overview:

Create a new network instance, and return its handle.

Signature:

```
network ref create (session ref session_id, network record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network record`	args	All constructor arguments

Minimum Role: vm-admin

Return Type: `network ref`

reference to the newly created object

RPC name: create_new_blob

Overview:

Create a placeholder for a named binary blob of data that is associated with this pool

Signature:

```
blob ref create_new_blob (session ref session_id, network ref network, string name,
string mime_type, bool public)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	network	The network
`string`	name	The name associated with the blob
`string`	mime_type	The mime type for the data. Empty string translates to application/octet-stream
`bool`	public	True if the blob should be publicly available

Minimum Role: pool-operator

Return Type: `blob ref`

The reference of the blob, needed for populating its data

RPC name: destroy

Overview:

Destroy the specified network instance.

Signature:

```
void destroy (session ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: vm-admin

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the networks known to the system.

Signature:

```
network ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `network ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of network references to network records for all networks known to the system.

Signature:

```
(network ref -> network record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(network ref -> network record) map`

records of all objects

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given network.

Signature:

```
network_operations set get_allowed_operations (session ref session_id, network ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `network_operations set`

value of the field

RPC name: get_assigned_ips

Overview:

Get the assigned_ips field of the given network.

Signature:

```
(VIF ref -> string) map get_assigned_ips (session ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(VIF ref -> string) map`

value of the field

RPC name: get_blobs

Overview:

Get the blobs field of the given network.

Signature:

```
(string -> blob ref) map get_blobs (session ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> blob ref) map`

value of the field

RPC name: get_bridge

Overview:

Get the bridge field of the given network.

Signature:

```
string get_bridge (session ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

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 ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `network ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the network instance with the specified UUID.

Signature:

```
network ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `network ref`

reference to the object

RPC name: get_current_operations

Overview:

Get the current_operations field of the given network.

Signature:

```
(string -> network_operations) map get_current_operations (session ref session_id,
network ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> network_operations) map`

value of the field

RPC name: get_default_locking_mode

Overview:

Get the default_locking_mode field of the given network.

Signature:

```
network_default_locking_mode get_default_locking_mode (session ref session_id, network
ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `network_default_locking_mode`

value of the field

RPC name: get_managed

Overview:

Get the managed field of the given network.

Signature:

```
bool get_managed (session ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_MTU

Overview:

Get the MTU field of the given network.

Signature:

```
int get_MTU (session ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given network.

Signature:

```
string get_name_description (session ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

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 ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_PIFs

Overview:

Get the PIFs field of the given network.

Signature:

```
PIF ref set get_PIFs (session ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF ref set`

value of the field

RPC name: get_purpose

Overview:

Get the purpose field of the given network.

Signature:

```
network_purpose set get_purpose (session ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `network_purpose set`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given network.

Signature:

```
network record get_record (session ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `network record`

all fields from the object

RPC name: get_tags

Overview:

Get the tags field of the given network.

Signature:

```
string set get_tags (session ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given network.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VIFs

Overview:

Get the VIFs field of the given network.

Signature:

```
VIF ref set get_VIFs (session ref session_id, network ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VIF ref set`

value of the field

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 ref session_id, network ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: remove_purpose

Overview:

Remove a purpose from a network (if present)

Signature:

```
void remove_purpose (session ref session_id, network ref self, network_purpose value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	self	The network
`network_purpose`	value	The purpose to remove

Minimum Role: pool-admin

Return Type: `void`

RPC name: remove_tags

Overview:

Remove the given value from the tags field of the given network. If the value is not in that Set, then do nothing.

Signature:

```
void remove_tags (session ref session_id, network ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
network ref	self	reference to the object
string	value	Value to remove

Minimum Role: vm-operator

Return Type: void

RPC name: set_default_locking_mode

Overview:

Set the default locking mode for VIFs attached to this network

Signature:

```
void set_default_locking_mode (session ref session_id, network ref network,
network_default_locking_mode value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
network ref	network	The network
network_default_locking_mode	value	The default locking mode for VIFs attached to this network.

Minimum Role: pool-operator

Return Type: void

RPC name: set_MTU

Overview:

Set the MTU field of the given network.

Signature:

```
void set_MTU (session ref session_id, network ref self, int value)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
network ref	self	reference to the object
int	value	New value to set

Minimum Role: vm-admin

Return Type: void

RPC name: set_name_description

Overview:

Set the name/description field of the given network.

Signature:

```
void set_name_description (session ref session_id, network ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
network ref	self	reference to the object
string	value	New value to set

Minimum Role: pool-operator

Return Type: void

RPC name: set_name_label

Overview:

Set the name/label field of the given network.

Signature:

```
void set_name_label (session ref session_id, network ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
network ref	self	reference to the object
string	value	New value to set

Minimum Role: pool-operator

Return Type: void

RPC name: set_other_config

Overview:

Set the other_config field of the given network.

Signature:

```
void set_other_config (session ref session_id, network ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
network ref	self	reference to the object
(string -> string) map	value	New value to set

Minimum Role: pool-operator

Return Type: void

RPC name: set_tags

Overview:

Set the tags field of the given network.

Signature:

```
void set_tags (session ref session_id, network ref self, string set value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
network ref	self	reference to the object

type	name	description
`string set`	value	New value to set

Minimum Role: vm-operator

Return Type: `void`

Class: network_sriov

network-sriov which connects logical pif and physical pif

Fields for class: network_sriov

Field	Type	Qualifier	Description
configuration_mode	`sriov_configuration_mode`	*RO/runtime*	The mode for configure network sriov
logical_PIF	`PIF ref`	*RO/constructor*	The logical PIF to connect to the SR-IOV network after enable SR-IOV on the physical PIF
physical_PIF	`PIF ref`	*RO/constructor*	The PIF that has SR-IOV enabled
requires_reboot	`bool`	*RO/runtime*	Indicates whether the host need to be rebooted before SR-IOV is enabled on the physical PIF
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: network_sriov

RPC name: create

Overview:

Enable SR-IOV on the specific PIF. It will create a network-sriov based on the specific PIF and automatically create a logical PIF to connect the specific network.

Signature:

```
network_sriov ref create (session ref session_id, PIF ref pif, network ref network)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	pif	PIF on which to enable SR-IOV
`network ref`	network	Network to connect SR-IOV virtual functions with VM VIFs

Minimum Role: pool-operator

Return Type: `network_sriov ref`

The reference of the created network_sriov object

RPC name: destroy

Overview:

Disable SR-IOV on the specific PIF. It will destroy the network-sriov and the logical PIF accordingly.

Signature:

```
void destroy (session ref session_id, network_sriov ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network_sriov ref`	self	SRIOV to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the network_sriovs known to the system.

Signature:

```
network_sriov ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `network_sriov ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of network_sriov references to network_sriov records for all network_sriovs known to the system.

Signature:

```
(network_sriov ref -> network_sriov record) map get_all_records (session ref
session_id)
```

Minimum Role: read-only

Return Type: `(network_sriov ref -> network_sriov record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the network_sriov instance with the specified UUID.

Signature:

```
network_sriov ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `network_sriov ref`

reference to the object

RPC name: get_configuration_mode

Overview:

Get the configuration_mode field of the given network_sriov.

Signature:

```
sriov_configuration_mode get_configuration_mode (session ref session_id, network_sriov
ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
network_sriov ref	self	reference to the object

Minimum Role: read-only

Return Type: sriov_configuration_mode

value of the field

RPC name: get_logical_PIF

Overview:

Get the logical_PIF field of the given network_sriov.

Signature:

```
PIF ref get_logical_PIF (session ref session_id, network_sriov ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
network_sriov ref	self	reference to the object

Minimum Role: read-only

Return Type: PIF ref

value of the field

RPC name: get_physical_PIF

Overview:

Get the physical_PIF field of the given network_sriov.

Signature:

```
PIF ref get_physical_PIF (session ref session_id, network_sriov ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
network_sriov ref	self	reference to the object

Minimum Role: read-only

Return Type: `PIF ref`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given network_sriov.

Signature:

```
network_sriov record get_record (session ref session_id, network_sriov ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network_sriov ref`	self	reference to the object

Minimum Role: read-only

Return Type: `network_sriov record`

all fields from the object

RPC name: get_remaining_capacity

Overview:

Get the number of free SR-IOV VFs on the associated PIF

Signature:

```
int get_remaining_capacity (session ref session_id, network_sriov ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network_sriov ref`	self	the NETWORK_SRIOV object

Minimum Role: read-only

Return Type: `int`

The number of free SR-IOV VFs on the associated PIF

RPC name: get_requires_reboot

Overview:

Get the requires_reboot field of the given network_sriov.

Signature:

```
bool get_requires_reboot (session ref session_id, network_sriov ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network_sriov ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given network_sriov.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network_sriov ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

Class: PBD

The physical block devices through which hosts access SRs

Fields for class: PBD

Field	Type	Qualifier	Description
currently_attached	`bool`	*RO/runtime*	is the SR currently attached on this host?
device_config	`(string -> string) map`	*RO/constructor*	a config string to string map that is provided to the host's SR-backend-driver
host	`host ref`	*RO/constructor*	physical machine on which the pbd is available
other_config	`(string -> string) map`	*RW*	additional configuration
SR	`SR ref`	*RO/constructor*	the storage repository that the pbd realises
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: PBD

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given PBD.

Signature:

```
void add_to_other_config (session ref session_id, PBD ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PBD ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: create

Overview:

Create a new PBD instance, and return its handle.

Signature:

```
PBD ref create (session ref session_id, PBD record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PBD record`	args	All constructor arguments

Minimum Role: pool-operator

Return Type: `PBD ref`

reference to the newly created object

RPC name: destroy

Overview:

Destroy the specified PBD instance.

Signature:

```
void destroy (session ref session_id, PBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PBD ref`	self	reference to the object

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the PBDs known to the system.

Signature:

```
PBD ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `PBD ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of PBD references to PBD records for all PBDs known to the system.

Signature:

```
(PBD ref -> PBD record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(PBD ref -> PBD record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the PBD instance with the specified UUID.

Signature:

```
PBD ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `PBD ref`

reference to the object

RPC name: get_currently_attached

Overview:

Get the currently_attached field of the given PBD.

Signature:

```
bool get_currently_attached (session ref session_id, PBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

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 ref session_id, PBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_host

Overview:

Get the host field of the given PBD.

Signature:

```
host ref get_host (session ref session_id, PBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given PBD.

Signature:

```
(string -> string) map get_other_config (session ref session_id, PBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given PBD.

Signature:

```
PBD record get_record (session ref session_id, PBD ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`PBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PBD record`

all fields from the object

RPC name: get_SR

Overview:

Get the SR field of the given PBD.

Signature:

```
SR ref get_SR (session ref session_id, PBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SR ref`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given PBD.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
PBD ref	self	reference to the object

Minimum Role: read-only

Return Type: string

value of the field

RPC name: plug

Overview:

Activate the specified PBD, causing the referenced SR to be attached and scanned

Signature:

```
void plug (session ref session_id, PBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PBD ref	self	The PBD to activate

Minimum Role: pool-operator

Return Type: void

Possible Error Codes: SR_UNKNOWN_DRIVER

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given PBD. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, PBD ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PBD ref	self	reference to the object

type	name	description
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_device_config

Overview:

Sets the PBD's device_config field

Signature:

```
void set_device_config (session ref session_id, PBD ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PBD ref`	self	The PBD to modify
`(string -> string) map`	value	The new value of the PBD's device_config

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given PBD.

Signature:

```
void set_other_config (session ref session_id, PBD ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PBD ref`	self	reference to the object

type	name	description
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: unplug

Overview:

Deactivate the specified PBD, causing the referenced SR to be detached and nolonger scanned

Signature:

```
void unplug (session ref session_id, PBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PBD ref`	self	The PBD to deactivate

Minimum Role: pool-operator

Return Type: `void`

Class: PCI

A PCI device

Fields for class: PCI

Field	Type	Qualifier	Description
class_name	`string`	*RO/constructor*	PCI class name
dependencies	`PCI ref set`	*RO/runtime*	List of dependent PCI devices
device_name	`string`	*RO/constructor*	Device name
driver_name	`string`	*RO/constructor*	Driver name
host	`host ref`	*RO/constructor*	Physical machine that owns the PCI device
other_config	`(string -> string) map`	*RW*	Additional configuration
pci_id	`string`	*RO/constructor*	PCI ID of the physical device
subsystem_device_name	`string`	*RO/constructor*	Subsystem device name

Field	Type	Qualifier	Description
subsystem_vendor_name	`string`	*RO/constructor*	Subsystem vendor name
uuid	`string`	*RO/runtime*	Unique identifier/object reference
vendor_name	`string`	*RO/constructor*	Vendor name

RPCs associated with class: PCI

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given PCI.

Signature:

```
void add_to_other_config (session ref session_id, PCI ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the PCIs known to the system.

Signature:

```
PCI ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `PCI ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of PCI references to PCI records for all PCIs known to the system.

Signature:

```
(PCI ref -> PCI record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(PCI ref -> PCI record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the PCI instance with the specified UUID.

Signature:

```
PCI ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `PCI ref`

reference to the object

RPC name: get_class_name

Overview:

Get the class_name field of the given PCI.

Signature:

```
string get_class_name (session ref session_id, PCI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_dependencies

Overview:

Get the dependencies field of the given PCI.

Signature:

```
PCI ref set get_dependencies (session ref session_id, PCI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PCI ref set`

value of the field

RPC name: get_device_name

Overview:

Get the device_name field of the given PCI.

Signature:

```
string get_device_name (session ref session_id, PCI ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_driver_name

Overview:

Get the driver_name field of the given PCI.

Signature:

```
string get_driver_name (session ref session_id, PCI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_host

Overview:

Get the host field of the given PCI.

Signature:

```
host ref get_host (session ref session_id, PCI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`PCI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given PCI.

Signature:

```
(string -> string) map get_other_config (session ref session_id, PCI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_pci_id

Overview:

Get the pci_id field of the given PCI.

Signature:

```
string get_pci_id (session ref session_id, PCI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given PCI.

Signature:

```
PCI record get_record (session ref session_id, PCI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PCI record`

all fields from the object

RPC name: get_subsystem_device_name

Overview:

Get the subsystem_device_name field of the given PCI.

Signature:

```
string get_subsystem_device_name (session ref session_id, PCI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_subsystem_vendor_name

Overview:

Get the subsystem_vendor_name field of the given PCI.

Signature:

```
string get_subsystem_vendor_name (session ref session_id, PCI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given PCI.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_vendor_name

Overview:

Get the vendor_name field of the given PCI.

Signature:

```
string get_vendor_name (session ref session_id, PCI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given PCI. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, PCI ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given PCI.

Signature:

```
void set_other_config (session ref session_id, PCI ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PCI ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: PGPU

A physical GPU (pGPU)

Fields for class: PGPU

Field	Type	Qualifier	Description
compatibility_metadata	`(string -> string) map`	*RO/runtime*	PGPU metadata to determine whether a VGPU can migrate between two PGPUs
dom0_access	`pgpu_dom0_access`	*RO/runtime*	The accessibility of this device from dom0
enabled_VGPU_types	`VGPU_type ref set`	*RO/runtime*	List of VGPU types which have been enabled for this PGPU
GPU_group	`GPU_group ref`	*RO/constructor*	GPU group the pGPU is contained in
host	`host ref`	*RO/runtime*	Host that owns the GPU
is_system_display_device	`bool`	*RO/runtime*	Is this device the system display device
other_config	`(string -> string) map`	*RW*	Additional configuration
PCI	`PCI ref`	*RO/constructor*	Link to underlying PCI device
resident_VGPUs	`VGPU ref set`	*RO/runtime*	List of VGPUs running on this PGPU

Field	Type	Qualifier	Description
supported_VGPU_max_capacities	`(VGPU_type ref -> int) map`	*RO/runtime*	A map relating each VGPU type supported on this GPU to the maximum number of VGPUs of that type which can run simultaneously on this GPU
supported_VGPU_types	`VGPU_type ref set`	*RO/runtime*	List of VGPU types supported by the underlying hardware
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: PGPU

RPC name: add_enabled_VGPU_types

Overview:

Signature:

```
void add_enabled_VGPU_types (session ref session_id, PGPU ref self, VGPU_type ref
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	The PGPU to which we are adding an enabled VGPU type
`VGPU_type ref`	value	The VGPU type to enable

Minimum Role: pool-operator

Return Type: `void`

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given PGPU.

Signature:

```
void add_to_other_config (session ref session_id, PGPU ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: disable_dom0_access

Overview:

Signature:

```
pgpu_dom0_access disable_dom0_access (session ref session_id, PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	The PGPU to which dom0 will be denied access

Minimum Role: pool-operator

Return Type: `pgpu_dom0_access`

The accessibility of this PGPU from dom0

RPC name: enable_dom0_access

Overview:

Signature:

```
pgpu_dom0_access enable_dom0_access (session ref session_id, PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`PGPU ref`	self	The PGPU to which dom0 will be granted access

Minimum Role: pool-operator

Return Type: `pgpu_dom0_access`

The accessibility of this PGPU from dom0

RPC name: get_all

Overview:

Return a list of all the PGPUs known to the system.

Signature:

```
PGPU ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `PGPU ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of PGPU references to PGPU records for all PGPUs known to the system.

Signature:

```
(PGPU ref -> PGPU record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(PGPU ref -> PGPU record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the PGPU instance with the specified UUID.

Signature:

```
PGPU ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `PGPU ref`

reference to the object

RPC name: get_compatibility_metadata

Overview:

Get the compatibility_metadata field of the given PGPU.

Signature:

```
(string -> string) map get_compatibility_metadata (session ref session_id, PGPU ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_dom0_access

Overview:

Get the dom0_access field of the given PGPU.

Signature:

```
pgpu_dom0_access get_dom0_access (session ref session_id, PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `pgpu_dom0_access`

value of the field

RPC name: get_enabled_VGPU_types

Overview:

Get the enabled_VGPU_types field of the given PGPU.

Signature:

```
VGPU_type ref set get_enabled_VGPU_types (session ref session_id, PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VGPU_type ref set`

value of the field

RPC name: get_GPU_group

Overview:

Get the GPU_group field of the given PGPU.

Signature:

```
GPU_group ref get_GPU_group (session ref session_id, PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `GPU_group ref`

value of the field

RPC name: get_host

Overview:

Get the host field of the given PGPU.

Signature:

```
host ref get_host (session ref session_id, PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_is_system_display_device

Overview:

Get the is_system_display_device field of the given PGPU.

Signature:

```
bool get_is_system_display_device (session ref session_id, PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given PGPU.

Signature:

```
(string -> string) map get_other_config (session ref session_id, PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_PCI

Overview:

Get the PCI field of the given PGPU.

Signature:

```
PCI ref get_PCI (session ref session_id, PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PCI ref`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given PGPU.

Signature:

```
PGPU record get_record (session ref session_id, PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PGPU record`

all fields from the object

RPC name: get_remaining_capacity

Overview:

Signature:

```
int get_remaining_capacity (session ref session_id, PGPU ref self, VGPU_type ref
vgpu_type)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	The PGPU to query
`VGPU_type ref`	vgpu_type	The VGPU type for which we want to find the number of VGPUs which can still be started on this PGPU

Minimum Role: read-only

Return Type: `int`

The number of VGPUs of the specified type which can still be started on this PGPU

RPC name: get_resident_VGPUs

Overview:

Get the resident_VGPUs field of the given PGPU.

Signature:

```
VGPU ref set get_resident_VGPUs (session ref session_id, PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VGPU ref set`

value of the field

RPC name: get_supported_VGPU_max_capacities

Overview:

Get the supported_VGPU_max_capacities field of the given PGPU.

Signature:

```
(VGPU_type ref -> int) map get_supported_VGPU_max_capacities (session ref session_id,
PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(VGPU_type ref -> int) map`

value of the field

RPC name: get_supported_VGPU_types

Overview:

Get the supported_VGPU_types field of the given PGPU.

Signature:

```
VGPU_type ref set get_supported_VGPU_types (session ref session_id, PGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VGPU_type ref set`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given PGPU.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_enabled_VGPU_types

Overview:

Signature:

```

```

```
void remove_enabled_VGPU_types (session ref session_id, PGPU ref self, VGPU_type ref
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	The PGPU from which we are removing an enabled VGPU type
`VGPU_type ref`	value	The VGPU type to disable

Minimum Role: pool-operator

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 PGPU. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, PGPU ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_enabled_VGPU_types

Overview:

Signature:

```
void set_enabled_VGPU_types (session ref session_id, PGPU ref self, VGPU_type ref set
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	The PGPU on which we are enabling a set of VGPU types
`VGPU_type ref set`	value	The VGPU types to enable

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_GPU_group

Overview:

Signature:

```
void set_GPU_group (session ref session_id, PGPU ref self, GPU_group ref value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PGPU ref`	self	The PGPU to move to a new group
`GPU_group ref`	value	The group to which the PGPU will be moved

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given PGPU.

Signature:

```
void set_other_config (session ref session_id, PGPU ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`PGPU ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: PIF

A physical network interface (note separate VLANs are represented as several PIFs)

Fields for class: PIF

Field	Type	Qualifier	Description
bond_master_of	`Bond ref set`	*RO/runtime*	Indicates this PIF represents the results of a bond
bond_slave_of	`Bond ref`	*RO/runtime*	Indicates which bond this interface is part of
capabilities	`string set`	*RO/runtime*	Additional capabilities on the interface.
currently_attached	`bool`	*RO/runtime*	true if this interface is online
device	`string`	*RO/constructor*	machine-readable name of the interface (e.g. eth0)
disallow_unplug	`bool`	*RO/runtime*	Prevent this PIF from being unplugged; set this to notify the management tool-stack that the PIF has a special use and should not be unplugged under any circumstances (e.g. because you're running storage traffic over it)
DNS	`string`	*RO/runtime*	Comma separated list of the IP addresses of the DNS servers to use
gateway	`string`	*RO/runtime*	IP gateway
host	`host ref`	*RO/constructor*	physical machine to which this pif is connected
igmp_snooping_status	`pif_igmp_status`	*RO/runtime*	The IGMP snooping status of the corresponding network bridge
IP	`string`	*RO/runtime*	IP address

Field	Type	Qualifier	Description
ip_configuration_mode	ip_configuration_mode	*RO/runtime*	Sets if and how this interface gets an IP address
IPv6	string set	*RO/runtime*	IPv6 address
ipv6_configuration_mode	ipv6_configuration_mode	*RO/runtime*	Sets if and how this interface gets an IPv6 address
ipv6_gateway	string	*RO/runtime*	IPv6 gateway
MAC	string	*RO/constructor*	ethernet MAC address of physical interface
managed	bool	*RO/constructor*	Indicates whether the interface is managed by xapi. If it is not, then xapi will not configure the interface, the commands PIF.plug/unplug/reconfigure_ip(v6) cannot be used, nor can the interface be bonded or have VLANs based on top through xapi.
management	bool	*RO/runtime*	Indicates whether the control software is listening for connections on this interface
metrics	PIF_metrics ref	*RO/runtime*	metrics associated with this PIF
MTU	int	*RO/constructor*	MTU in octets
netmask	string	*RO/runtime*	IP netmask
network	network ref	*RO/constructor*	virtual network to which this pif is connected
other_config	(string -> string) map	*RW*	Additional configuration
PCI	PCI ref	*RO/runtime*	Link to underlying PCI device
physical	bool	*RO/runtime*	true if this represents a physical network interface
primary_address_type	primary_address_type	*RO/runtime*	Which protocol should define the primary address of this interface
properties	(string -> string) map	*RO/runtime*	Additional configuration properties for the interface.
sriov_logical_PIF_of	network_sriov ref set	*RO/runtime*	Indicates which network_sriov this interface is logical of
sriov_physical_PIF_of	network_sriov ref set	*RO/runtime*	Indicates which network_sriov this interface is physical of

Field	Type	Qualifier	Description
tunnel_access_PIF_of	`tunnel ref set`	*RO/runtime*	Indicates to which tunnel this PIF gives access
tunnel_transport_PIF_of	`tunnel ref set`	*RO/runtime*	Indicates to which tunnel this PIF provides transport
uuid	`string`	*RO/runtime*	Unique identifier/object reference
VLAN	`int`	*RO/constructor*	VLAN tag for all traffic passing through this interface
VLAN_master_of	`VLAN ref`	*RO/runtime*	Indicates wich VLAN this interface receives untagged traffic from
VLAN_slave_of	`VLAN ref set`	*RO/runtime*	Indicates which VLANs this interface transmits tagged traffic to

RPCs associated with class: PIF

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given PIF.

Signature:

```
void add_to_other_config (session ref session_id, PIF ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: create_VLAN

This message is deprecated.

Overview:

Create a VLAN interface from an existing physical interface. This call is deprecated: use VLAN.create instead

Signature:

```
PIF ref create_VLAN (session ref session_id, string device, network ref network, host ref host, int VLAN)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	device	physical interface on which to create 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

Minimum Role: pool-operator

Return Type: `PIF ref`

The reference of the created PIF object

Possible Error Codes: `VLAN_TAG_INVALID`

RPC name: db_forget

Overview:

Destroy a PIF database record.

Signature:

```
void db_forget (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	The ref of the PIF whose database record should be destroyed

Minimum Role: pool-operator

Return Type: `void`

RPC name: db_introduce

Overview:

Create a new PIF record in the database only

Signature:

```
PIF ref db_introduce (session ref session_id, string device, network ref network, host
ref host, string MAC, int MTU, int VLAN, bool physical, ip_configuration_mode
ip_configuration_mode, string IP, string netmask, string gateway, string DNS, Bond ref
bond_slave_of, VLAN ref VLAN_master_of, bool management, (string -> string) map
other_config, bool disallow_unplug, ipv6_configuration_mode ipv6_configuration_mode,
string set IPv6, string ipv6_gateway, primary_address_type primary_address_type, bool
managed, (string -> string) map properties)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	device	
`network ref`	network	
`host ref`	host	
`string`	MAC	
`int`	MTU	
`int`	VLAN	
`bool`	physical	
`ip_configuration_mode`	ip_configuration_mode	
`string`	IP	
`string`	netmask	
`string`	gateway	
`string`	DNS	
`Bond ref`	bond_slave_of	
`VLAN ref`	VLAN_master_of	
`bool`	management	
`(string -> string) map`	other_config	
`bool`	disallow_unplug	
`ipv6_configuration_mode`	ipv6_configuration_mode	
`string set`	IPv6	
`string`	ipv6_gateway	

type	name	description
`primary_address_type`	primary_address_type	
`bool`	managed	
`(string -> string) map`	properties	

Minimum Role: pool-operator

Return Type: `PIF ref`

The ref of the newly created PIF record.

RPC name: destroy

This message is deprecated.

Overview:

Destroy the PIF object (provided it is a VLAN interface). This call is deprecated: use VLAN.destroy or Bond.destroy instead

Signature:

```
void destroy (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	the PIF object to destroy

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `PIF_IS_PHYSICAL`

RPC name: forget

Overview:

Destroy the PIF object matching a particular network interface

Signature:

```
void forget (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	The PIF object to destroy

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `PIF_TUNNEL_STILL_EXISTS`, `CLUSTERING_ENABLED`

RPC name: get_all

Overview:

Return a list of all the PIFs known to the system.

Signature:

```
PIF ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `PIF ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of PIF references to PIF records for all PIFs known to the system.

Signature:

```
(PIF ref -> PIF record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(PIF ref -> PIF record) map`

records of all objects

RPC name: get_bond_master_of

Overview:

Get the bond_master_of field of the given PIF.

Signature:

```
Bond ref set get_bond_master_of (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `Bond ref set`

value of the field

RPC name: get_bond_slave_of

Overview:

Get the bond_slave_of field of the given PIF.

Signature:

```
Bond ref get_bond_slave_of (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `Bond 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 ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `PIF ref`

reference to the object

RPC name: get_capabilities

Overview:

Get the capabilities field of the given PIF.

Signature:

```
string set get_capabilities (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_currently_attached

Overview:

Get the currently_attached field of the given PIF.

Signature:

```
bool get_currently_attached (session ref session_id, PIF ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_device

Overview:

Get the device field of the given PIF.

Signature:

```
string get_device (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_disallow_unplug

Overview:

Get the disallow_unplug field of the given PIF.

Signature:

```
bool get_disallow_unplug (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_DNS

Overview:

Get the DNS field of the given PIF.

Signature:

```
string get_DNS (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_gateway

Overview:

Get the gateway field of the given PIF.

Signature:

```
string get_gateway (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_host

Overview:

Get the host field of the given PIF.

Signature:

```
host ref get_host (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_igmp_snooping_status

Overview:

Get the igmp_snooping_status field of the given PIF.

Signature:

```
pif_igmp_status get_igmp_snooping_status (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `pif_igmp_status`

value of the field

RPC name: get_IP

Overview:

Get the IP field of the given PIF.

Signature:

```
string get_IP (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_ip_configuration_mode

Overview:

Get the ip_configuration_mode field of the given PIF.

Signature:

```
ip_configuration_mode get_ip_configuration_mode (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `ip_configuration_mode`

value of the field

RPC name: get_IPv6

Overview:

Get the IPv6 field of the given PIF.

Signature:

```
string set get_IPv6 (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_ipv6_configuration_mode

Overview:

Get the ipv6_configuration_mode field of the given PIF.

Signature:

```
ipv6_configuration_mode get_ipv6_configuration_mode (session ref session_id, PIF ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `ipv6_configuration_mode`

value of the field

RPC name: get_ipv6_gateway

Overview:

Get the ipv6_gateway field of the given PIF.

Signature:

```
string get_ipv6_gateway (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_MAC

Overview:

Get the MAC field of the given PIF.

Signature:

```
string get_MAC (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_managed

Overview:

Get the managed field of the given PIF.

Signature:

```
bool get_managed (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_management

Overview:

Get the management field of the given PIF.

Signature:

```
bool get_management (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_metrics

Overview:

Get the metrics field of the given PIF.

Signature:

```
PIF_metrics ref get_metrics (session ref session_id, PIF ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF_metrics ref`

value of the field

RPC name: get_MTU

Overview:

Get the MTU field of the given PIF.

Signature:

```
int get_MTU (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_netmask

Overview:

Get the netmask field of the given PIF.

Signature:

```
string get_netmask (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_network

Overview:

Get the network field of the given PIF.

Signature:

```
network ref get_network (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `network ref`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given PIF.

Signature:

```
(string -> string) map get_other_config (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_PCI

Overview:

Get the PCI field of the given PIF.

Signature:

```
PCI ref get_PCI (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PCI ref`

value of the field

RPC name: get_physical

Overview:

Get the physical field of the given PIF.

Signature:

```
bool get_physical (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_primary_address_type

Overview:

Get the primary_address_type field of the given PIF.

Signature:

```
primary_address_type get_primary_address_type (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `primary_address_type`

value of the field

RPC name: get_properties

Overview:

Get the properties field of the given PIF.

Signature:

```
(string -> string) map get_properties (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given PIF.

Signature:

```
PIF record get_record (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF record`

all fields from the object

RPC name: get_sriov_logical_PIF_of

Overview:

Get the sriov_logical_PIF_of field of the given PIF.

Signature:

```
network_sriov ref set get_sriov_logical_PIF_of (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `network_sriov ref set`

value of the field

RPC name: get_sriov_physical_PIF_of

Overview:

Get the sriov_physical_PIF_of field of the given PIF.

Signature:

```
network_sriov ref set get_sriov_physical_PIF_of (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `network_sriov ref set`

value of the field

RPC name: get_tunnel_access_PIF_of

Overview:

Get the tunnel_access_PIF_of field of the given PIF.

Signature:

```
tunnel ref set get_tunnel_access_PIF_of (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `tunnel ref set`

value of the field

RPC name: get_tunnel_transport_PIF_of

Overview:

Get the tunnel_transport_PIF_of field of the given PIF.

Signature:

```
tunnel ref set get_tunnel_transport_PIF_of (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `tunnel ref set`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given PIF.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VLAN

Overview:

Get the VLAN field of the given PIF.

Signature:

```
int get_VLAN (session ref session_id, PIF ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_VLAN_master_of

Overview:

Get the VLAN_master_of field of the given PIF.

Signature:

```
VLAN ref get_VLAN_master_of (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VLAN ref`

value of the field

RPC name: get_VLAN_slave_of

Overview:

Get the VLAN_slave_of field of the given PIF.

Signature:

```
VLAN ref set get_VLAN_slave_of (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`PIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VLAN ref set`

value of the field

RPC name: introduce

Overview:

Create a PIF object matching a particular network interface

Signature:

```
PIF ref introduce (session ref session_id, host ref host, string MAC, string device,
bool managed)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host on which the interface exists
`string`	MAC	The MAC address of the interface
`string`	device	The device name to use for the interface
`bool`	managed	Indicates whether the interface is managed by xapi (defaults to "true")

Minimum Role: pool-operator

Return Type: `PIF ref`

The reference of the created PIF object

RPC name: plug

Overview:

Attempt to bring up a physical interface

Signature:

```
void plug (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PIF ref	self	the PIF object to plug

Minimum Role: pool-operator

Return Type: void

Possible Error Codes: TRANSPORT_PIF_NOT_CONFIGURED

RPC name: reconfigure_ip

Overview:

Reconfigure the IP address settings for this interface

Signature:

```
void reconfigure_ip (session ref session_id, PIF ref self, ip_configuration_mode mode,
string IP, string netmask, string gateway, string DNS)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PIF ref	self	the PIF object to reconfigure
ip_configuration_mode	mode	whether to use dynamic/static/no-assignment
string	IP	the new IP address
string	netmask	the new netmask
string	gateway	the new gateway
string	DNS	the new DNS settings

Minimum Role: pool-operator

Return Type: void

Possible Error Codes: CLUSTERING_ENABLED

RPC name: reconfigure_ipv6

Overview:

Reconfigure the IPv6 address settings for this interface

Signature:

```
void reconfigure_ipv6 (session ref session_id, PIF ref self, ipv6_configuration_mode
mode, string IPv6, string gateway, string DNS)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	the PIF object to reconfigure
`ipv6_configuration_mode`	mode	whether to use dynamic/static/no-assignment
`string`	IPv6	the new IPv6 address (in <addr>/<prefix length> format)
`string`	gateway	the new gateway
`string`	DNS	the new DNS settings

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `CLUSTERING_ENABLED`

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given PIF. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, PIF ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: scan

Overview:

Scan for physical interfaces on a host and create PIF objects to represent them

Signature:

```
void scan (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host on which to scan

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_disallow_unplug

Overview:

Set whether unplugging the PIF is allowed

Signature:

```
void set_disallow_unplug (session ref session_id, PIF ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	Reference to the object
`bool`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `OTHER_OPERATION_IN_PROGRESS`, `CLUSTERING_ENABLED`

RPC name: set_other_config

Overview:

Set the other_config field of the given PIF.

Signature:

```
```

```
void set_other_config (session ref session_id, PIF ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_primary_address_type

Overview:

Change the primary address type used by this PIF

Signature:

```
void set_primary_address_type (session ref session_id, PIF ref self,
primary_address_type primary_address_type)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	self	the PIF object to reconfigure
`primary_address_type`	primary_address_type	Whether to prefer IPv4 or IPv6 connections

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_property

Overview:

Set the value of a property of the PIF

Signature:

```
void set_property (session ref session_id, PIF ref self, string name, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PIF ref	self	The PIF
string	name	The property name
string	value	The property value

Minimum Role: pool-operator

Return Type: void

RPC name: unplug

Overview:

Attempt to bring down a physical interface

Signature:

```
void unplug (session ref session_id, PIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PIF ref	self	the PIF object to unplug

Minimum Role: pool-operator

Return Type: void

Possible Error Codes: HA_OPERATION_WOULD_BREAK_FAILOVER_PLAN, VIF_IN_USE, PIF_DOES_NOT_ALLOW_UNPLUG, PIF_HAS_FCOE_SR_IN_USE

Class: PIF_metrics

The metrics associated with a physical network interface

Fields for class: PIF_metrics

Field	Type	Qualifier	Description
carrier	bool	*RO/runtime*	Report if the PIF got a carrier or not
device_id	string	*RO/runtime*	Report device ID
device_name	string	*RO/runtime*	Report device name

Field	Type	Qualifier	Description
duplex	`bool`	*RO/runtime*	Full duplex capability of the link (if available)
io_read_kbs	`float`	*RO/runtime*	**Removed**. Read bandwidth (KiB/s)
io_write_kbs	`float`	*RO/runtime*	**Removed**. Write bandwidth (KiB/s)
last_updated	`datetime`	*RO/runtime*	Time at which this information was last updated
other_config	`(string -> string) map`	*RW*	additional configuration
pci_bus_path	`string`	*RO/runtime*	PCI bus path of the pif (if available)
speed	`int`	*RO/runtime*	Speed of the link (if available)
uuid	`string`	*RO/runtime*	Unique identifier/object reference
vendor_id	`string`	*RO/runtime*	Report vendor ID
vendor_name	`string`	*RO/runtime*	Report vendor name

RPCs associated with class: PIF_metrics

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given PIF_metrics.

Signature:

```
void add_to_other_config (session ref session_id, PIF_metrics ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

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 ref session_id)
```

Minimum Role: read-only

Return Type: `PIF_metrics ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of PIF_metrics references to PIF_metrics records for all PIF_metrics instances known to the system.

Signature:

```
(PIF_metrics ref -> PIF_metrics record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(PIF_metrics ref -> PIF_metrics record) map`

records of all objects

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 ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `PIF_metrics ref`

reference to the object

RPC name: get_carrier

Overview:

Get the carrier field of the given PIF_metrics.

Signature:

```
bool get_carrier (session ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_device_id

Overview:

Get the device_id field of the given PIF_metrics.

Signature:

```
string get_device_id (session ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_device_name

Overview:

Get the device_name field of the given PIF_metrics.

Signature:

```
string get_device_name (session ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_duplex

Overview:

Get the duplex field of the given PIF_metrics.

Signature:

```
bool get_duplex (session ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_io_read_kbs

This message is removed.

Overview:

Get the io/read_kbs field of the given PIF_metrics.

Signature:

```
float get_io_read_kbs (session ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `float`

value of the field

RPC name: get_io_write_kbs

This message is removed.

Overview:

Get the io/write_kbs field of the given PIF_metrics.

Signature:

```
float get_io_write_kbs (session ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given PIF_metrics.

Signature:

```
(string -> string) map get_other_config (session ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_pci_bus_path

Overview:

Get the pci_bus_path field of the given PIF_metrics.

Signature:

```
string get_pci_bus_path (session ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given PIF_metrics.

Signature:

```
PIF_metrics record get_record (session ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF_metrics record`

all fields from the object

RPC name: get_speed

Overview:

Get the speed field of the given PIF_metrics.

Signature:

```
int get_speed (session ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given PIF_metrics.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_vendor_id

Overview:

Get the vendor_id field of the given PIF_metrics.

Signature:

```
string get_vendor_id (session ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_vendor_name

Overview:

Get the vendor_name field of the given PIF_metrics.

Signature:

```
string get_vendor_name (session ref session_id, PIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given PIF_metrics. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, PIF_metrics ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
PIF_metrics ref	self	reference to the object
string	key	Key to remove

Minimum Role: pool-operator

Return Type: void

RPC name: set_other_config

Overview:

Set the other_config field of the given PIF_metrics.

Signature:

```
void set_other_config (session ref session_id, PIF_metrics ref self, (string ->
string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PIF_metrics ref	self	reference to the object
(string -> string) map	value	New value to set

Minimum Role: pool-operator

Return Type: void

Class: pool

Pool-wide information

Fields for class: pool

Field	Type	Qualifier	Description
allowed_operations	pool_allowed_operations set	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
blobs	(string -> blob ref) map	*RO/runtime*	Binary blobs associated with this pool

Field	Type	Qualifier	Description
cpu_info	`(string -> string) map`	*RO/runtime*	Details about the physical CPUs on the pool
crash_dump_SR	`SR ref`	*RW*	The SR in which VDIs for crash dumps are created
current_operations	`(string -> pool_allowed_operations) map`	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.
default_SR	`SR ref`	*RW*	Default SR for VDIs
guest_agent_config	`(string -> string) map`	*RO/runtime*	Pool-wide guest agent configuration information
gui_config	`(string -> string) map`	*RW*	gui-specific configuration for pool
ha_allow_overcommit	`bool`	*RW*	If set to false then operations which would cause the Pool to become overcommitted will be blocked.
ha_cluster_stack	`string`	*RO/runtime*	The HA cluster stack that is currently in use. Only valid when HA is enabled.
ha_configuration	`(string -> string) map`	*RO/runtime*	The current HA configuration
ha_enabled	`bool`	*RO/runtime*	true if HA is enabled on the pool, false otherwise
ha_host_failures_to_tolerate	`int`	*RO/runtime*	Number of host failures to tolerate before the Pool is declared to be overcommitted
ha_overcommitted	`bool`	*RO/runtime*	True if the Pool is considered to be overcommitted i.e. if there exist insufficient physical resources to tolerate the configured number of host failures
ha_plan_exists_for	`int`	*RO/runtime*	Number of future host failures we have managed to find a plan for. Once this reaches zero any future host failures will cause the failure of protected VMs.

Field	Type	Qualifier	Description
ha_statefiles	`string set`	*RO/runtime*	HA statefile VDIs in use
health_check_config	`(string -> string) map`	*RW*	Configuration for the automatic health check feature
igmp_snooping_enabled	`bool`	*RO/runtime*	true if IGMP snooping is enabled in the pool, false otherwise.
is_psr_pending	`bool`	*RW*	True if either a PSR is running or we are waiting for a PSR to be re-run
live_patching_disabled	`bool`	*RW*	The pool-wide flag to show if the live patching feauture is disabled or not.
master	`host ref`	*RO/runtime*	The host that is pool master
metadata_VDIs	`VDI ref set`	*RO/runtime*	The set of currently known metadata VDIs for this pool
name_description	`string`	*RW*	Description
name_label	`string`	*RW*	Short name
other_config	`(string -> string) map`	*RW*	additional configuration
policy_no_vendor_device	`bool`	*RW*	The pool-wide policy for clients on whether to use the vendor device or not on newly created VMs. This field will also be consulted if the 'has_vendor_device' field is not specified in the VM.create call.
redo_log_enabled	`bool`	*RO/runtime*	true a redo-log is to be used other than when HA is enabled, false otherwise
redo_log_vdi	`VDI ref`	*RO/runtime*	indicates the VDI to use for the redo-log other than when HA is enabled
restrictions	`(string -> string) map`	*RO/runtime*	Pool-wide restrictions currently in effect
suspend_image_SR	`SR ref`	*RW*	The SR in which VDIs for suspend images are created
tags	`string set`	*RW*	user-specified tags for categorization purposes

Field	Type	Qualifier	Description
uefi_certificates	`string`	*RW*	The UEFI certificates allowing Secure Boot
uuid	`string`	*RO/runtime*	Unique identifier/object reference
vswitch_controller	`string`	*RO/runtime*	**Deprecated**. address of the vswitch controller
wlb_enabled	`bool`	*RW*	true if workload balancing is enabled on the pool, false otherwise
wlb_url	`string`	*RO/runtime*	Url for the configured workload balancing host
wlb_username	`string`	*RO/runtime*	Username for accessing the workload balancing host
wlb_verify_cert	`bool`	*RW*	true if communication with the WLB server should enforce TLS certificate verification.

RPCs associated with class: pool

RPC name: add_tags

Overview:

Add the given value to the tags field of the given pool. If the value is already in that Set, then do nothing.

Signature:

```
void add_tags (session ref session_id, pool ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`string`	value	New value to add

Minimum Role: vm-operator

Return Type: `void`

RPC name: add_to_guest_agent_config

Overview:

Add a key-value pair to the pool-wide guest agent configuration

Signature:

```
void add_to_guest_agent_config (session ref session_id, pool ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	The pool
`string`	key	The key to add
`string`	value	The value to add

Minimum Role: pool-admin

Return Type: `void`

RPC name: add_to_gui_config

Overview:

Add the given key-value pair to the gui_config field of the given pool.

Signature:

```
void add_to_gui_config (session ref session_id, pool ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: vm-operator

Return Type: `void`

RPC name: add_to_health_check_config

Overview:

Add the given key-value pair to the health_check_config field of the given pool.

Signature:

```
void add_to_health_check_config (session ref session_id, pool ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given pool.

Signature:

```
void add_to_other_config (session ref session_id, pool ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: apply_edition

Overview:

Apply an edition to all hosts in the pool

Signature:

```
void apply_edition (session ref session_id, pool ref self, string edition)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	Reference to the pool
`string`	edition	The requested edition

Minimum Role: pool-operator

Return Type: `void`

RPC name: certificate_install

Overview:

Install a TLS CA certificate, pool-wide.

Signature:

```
void certificate_install (session ref session_id, string name, string cert)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	name	A name to give the certificate
`string`	cert	The certificate

Minimum Role: pool-operator

Return Type: `void`

RPC name: certificate_list

Overview:

List the names of all installed TLS CA certificates.

Signature:

```
```

```
string set certificate_list (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `string set`

All installed certificates

RPC name: certificate_sync

Overview:

Copy the TLS CA certificates and CRLs of the master to all slaves.

Signature:

```
void certificate_sync (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `void`

RPC name: certificate_uninstall

Overview:

Remove a pool-wide TLS CA certificate.

Signature:

```
void certificate_uninstall (session ref session_id, string name)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	name	The certificate name

Minimum Role: pool-operator

Return Type: `void`

RPC name: create_new_blob

Overview:

Create a placeholder for a named binary blob of data that is associated with this pool

Signature:

```
blob ref create_new_blob (session ref session_id, pool ref pool, string name, string
mime_type, bool public)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	pool	The pool
`string`	name	The name associated with the blob
`string`	mime_type	The mime type for the data. Empty string translates to application/octet-stream
`bool`	public	True if the blob should be publicly available

Minimum Role: pool-operator

Return Type: `blob ref`

The reference of the blob, needed for populating its data

RPC name: create_VLAN

Overview:

Create PIFs, mapping a network to the same physical interface/VLAN on each host. This call is deprecated: use Pool.create_VLAN_from_PIF instead.

Signature:

```
PIF ref set create_VLAN (session ref session_id, string device, network ref network,
int VLAN)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	device	physical interface on which to create the VLAN interface
`network ref`	network	network to which this interface should be connected
`int`	VLAN	VLAN tag for the new interface

Minimum Role: pool-operator

Return Type: `PIF ref set`

The references of the created PIF objects

Possible Error Codes: `VLAN_TAG_INVALID`

RPC name: create_VLAN_from_PIF

Overview:

Create a pool-wide VLAN by taking the PIF.

Signature:

```
PIF ref set create_VLAN_from_PIF (session ref session_id, PIF ref pif, network ref
network, int VLAN)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	pif	physical interface on any particular host, that identifies the PIF on which to create the (pool-wide) VLAN interface
`network ref`	network	network to which this interface should be connected
`int`	VLAN	VLAN tag for the new interface

Minimum Role: pool-operator

Return Type: `PIF ref set`

The references of the created PIF objects

Possible Error Codes: `VLAN_TAG_INVALID`

RPC name: crl_install

Overview:

Install a TLS Certificate Revocation List, pool-wide.

Signature:

```
void crl_install (session ref session_id, string name, string cert)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`string`	name	A name to give the CRL
`string`	cert	The CRL

Minimum Role: pool-operator

Return Type: `void`

RPC name: crl_list

Overview:

List the names of all installed TLS Certificate Revocation Lists.

Signature:

```
string set crl_list (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `string set`

The names of all installed CRLs

RPC name: crl_uninstall

Overview:

Remove a pool-wide TLS Certificate Revocation List.

Signature:

```
void crl_uninstall (session ref session_id, string name)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	name	The CRL name

Minimum Role: pool-operator

Return Type: `void`

RPC name: deconfigure_wlb

Overview:

Permanently deconfigures workload balancing monitoring on this pool

Signature:

```
void deconfigure_wlb (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `void`

RPC name: designate_new_master

Overview:

Perform an orderly handover of the role of master to the referenced host.

Signature:

```
void designate_new_master (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host who should become the new master

Minimum Role: pool-operator

Return Type: `void`

RPC name: detect_nonhomogeneous_external_auth

Overview:

This call asynchronously detects if the external authentication configuration in any slave is different from that in the master and raises appropriate alerts

Signature:

```
void detect_nonhomogeneous_external_auth (session ref session_id, pool ref pool)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	pool	The pool where to detect non-homogeneous external authentication configuration

Minimum Role: pool-operator

Return Type: `void`

RPC name: disable_external_auth

Overview:

This call disables external authentication on all the hosts of the pool

Signature:

```
void disable_external_auth (session ref session_id, pool ref pool, (string -> string)
map config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	pool	The pool whose external authentication should be disabled
`(string -> string) map`	config	Optional parameters as a list of key-values containing the configuration data

Minimum Role: pool-admin

Return Type: `void`

RPC name: disable_ha

Overview:

Turn off High Availability mode

Signature:

```
void disable_ha (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `void`

RPC name: disable_local_storage_caching

Overview:

This call disables pool-wide local storage caching

Signature:

```
void disable_local_storage_caching (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	Reference to the pool

Minimum Role: pool-operator

Return Type: `void`

RPC name: disable_redo_log

Overview:

Disable the redo log if in use, unless HA is enabled.

Signature:

```
void disable_redo_log (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `void`

RPC name: disable_ssl_legacy

This message is deprecated.

Overview:

Sets ssl_legacy false on each host, pool-master last. See Host.ssl_legacy and Host.set_ssl_legacy.

Signature:

```
void disable_ssl_legacy (session ref session_id, pool ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	(ignored)

Minimum Role: pool-operator

Return Type: `void`

RPC name: eject

Overview:

Instruct a pool master to eject a host from the pool

Signature:

```
void eject (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host to eject

Minimum Role: pool-operator

Return Type: `void`

RPC name: emergency_reset_master

Overview:

Instruct a slave already in a pool that the master has changed

Signature:

```
void emergency_reset_master (session ref session_id, string master_address)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	master_address	The hostname of the master

Minimum Role: pool-operator

Return Type: `void`

RPC name: emergency_transition_to_master

Overview:

Instruct host that's currently a slave to transition to being master

Signature:

```
void emergency_transition_to_master (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `void`

RPC name: enable_external_auth

Overview:

This call enables external authentication on all the hosts of the pool

Signature:

```
void enable_external_auth (session ref session_id, pool ref pool, (string -> string)
map config, string service_name, string auth_type)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	pool	The pool whose external authentication should be enabled
`(string -> string) map`	config	A list of key-values containing the configuration data
`string`	service_name	The name of the service
`string`	auth_type	The type of authentication (e.g. AD for Active Directory)

Minimum Role: pool-admin

Return Type: `void`

RPC name: enable_ha

Overview:

Turn on High Availability mode

Signature:

```
void enable_ha (session ref session_id, SR ref set heartbeat_srs, (string -> string)
map configuration)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref set`	heartbeat_srs	Set of SRs to use for storage heartbeating
`(string -> string) map`	configuration	Detailed HA configuration to apply

Minimum Role: pool-operator

Return Type: `void`

RPC name: enable_local_storage_caching

Overview:

This call attempts to enable pool-wide local storage caching

Signature:

```
void enable_local_storage_caching (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	Reference to the pool

Minimum Role: pool-operator

Return Type: `void`

RPC name: enable_redo_log

Overview:

Enable the redo log on the given SR and start using it, unless HA is enabled.

Signature:

```
void enable_redo_log (session ref session_id, SR ref sr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
SR ref	sr	SR to hold the redo log.

Minimum Role: pool-operator

Return Type: `void`

RPC name: enable_ssl_legacy

This message is removed.

Overview:

Sets ssl_legacy true on each host, pool-master last. See Host.ssl_legacy and Host.set_ssl_legacy.

Signature:

```
void enable_ssl_legacy (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
pool ref	self	(ignored)

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the pools known to the system.

Signature:

```
pool ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `pool ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of pool references to pool records for all pools known to the system.

Signature:

```
(pool ref -> pool record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(pool ref -> pool record) map`

records of all objects

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given pool.

Signature:

```
pool_allowed_operations set get_allowed_operations (session ref session_id, pool ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `pool_allowed_operations set`

value of the field

RPC name: get_blobs

Overview:

Get the blobs field of the given pool.

Signature:

```
(string -> blob ref) map get_blobs (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> blob ref) map`

value of the field

RPC name: get_by_uuid

Overview:

Get a reference to the pool instance with the specified UUID.

Signature:

```
pool ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `pool ref`

reference to the object

RPC name: get_cpu_info

Overview:

Get the cpu_info field of the given pool.

Signature:

```
(string -> string) map get_cpu_info (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_crash_dump_SR

Overview:

Get the crash_dump_SR field of the given pool.

Signature:

```
SR ref get_crash_dump_SR (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SR ref`

value of the field

RPC name: get_current_operations

Overview:

Get the current_operations field of the given pool.

Signature:

```
(string -> pool_allowed_operations) map get_current_operations (session ref
session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> pool_allowed_operations) map`

value of the field

RPC name: get_default_SR

Overview:

Get the default_SR field of the given pool.

Signature:

```
SR ref get_default_SR (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SR ref`

value of the field

RPC name: get_guest_agent_config

Overview:

Get the guest_agent_config field of the given pool.

Signature:

```
(string -> string) map get_guest_agent_config (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_gui_config

Overview:

Get the gui_config field of the given pool.

Signature:

```
(string -> string) map get_gui_config (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_ha_allow_overcommit

Overview:

Get the ha_allow_overcommit field of the given pool.

Signature:

```
bool get_ha_allow_overcommit (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_ha_cluster_stack

Overview:

Get the ha_cluster_stack field of the given pool.

Signature:

```
string get_ha_cluster_stack (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_ha_configuration

Overview:

Get the ha_configuration field of the given pool.

Signature:

```
(string -> string) map get_ha_configuration (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_ha_enabled

Overview:

Get the ha_enabled field of the given pool.

Signature:

```
bool get_ha_enabled (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_ha_host_failures_to_tolerate

Overview:

Get the ha_host_failures_to_tolerate field of the given pool.

Signature:

```
int get_ha_host_failures_to_tolerate (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_ha_overcommitted

Overview:

Get the ha_overcommitted field of the given pool.

Signature:

```
bool get_ha_overcommitted (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_ha_plan_exists_for

Overview:

Get the ha_plan_exists_for field of the given pool.

Signature:

```
int get_ha_plan_exists_for (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_ha_statefiles

Overview:

Get the ha_statefiles field of the given pool.

Signature:

```
string set get_ha_statefiles (session ref session_id, pool ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_health_check_config

Overview:

Get the health_check_config field of the given pool.

Signature:

```
(string -> string) map get_health_check_config (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_igmp_snooping_enabled

Overview:

Get the igmp_snooping_enabled field of the given pool.

Signature:

```
bool get_igmp_snooping_enabled (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_is_psr_pending

Overview:

Get the is_psr_pending field of the given pool.

Signature:

```
bool get_is_psr_pending (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_license_state

Overview:

This call returns the license state for the pool

Signature:

```
(string -> string) map get_license_state (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	Reference to the pool

Minimum Role: read-only

Return Type: `(string -> string) map`

The pool's license state

RPC name: get_live_patching_disabled

Overview:

Get the live_patching_disabled field of the given pool.

Signature:

```
bool get_live_patching_disabled (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_master

Overview:

Get the master field of the given pool.

Signature:

```
host ref get_master (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_metadata_VDIs

Overview:

Get the metadata_VDIs field of the given pool.

Signature:

```
VDI ref set get_metadata_VDIs (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VDI ref set`

value of the field

RPC name: get_name_description

Overview:

Get the name_description field of the given pool.

Signature:

```
string get_name_description (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

Overview:

Get the name_label field of the given pool.

Signature:

```
string get_name_label (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given pool.

Signature:

```
(string -> string) map get_other_config (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_policy_no_vendor_device

Overview:

Get the policy_no_vendor_device field of the given pool.

Signature:

```
bool get_policy_no_vendor_device (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given pool.

Signature:

```
pool record get_record (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `pool record`

all fields from the object

RPC name: get_redo_log_enabled

Overview:

Get the redo_log_enabled field of the given pool.

Signature:

```
bool get_redo_log_enabled (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_redo_log_vdi

Overview:

Get the redo_log_vdi field of the given pool.

Signature:

```
VDI ref get_redo_log_vdi (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VDI ref`

value of the field

RPC name: get_restrictions

Overview:

Get the restrictions field of the given pool.

Signature:

```
(string -> string) map get_restrictions (session ref session_id, pool ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_suspend_image_SR

Overview:

Get the suspend_image_SR field of the given pool.

Signature:

```
SR ref get_suspend_image_SR (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SR ref`

value of the field

RPC name: get_tags

Overview:

Get the tags field of the given pool.

Signature:

```
string set get_tags (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_uefi_certificates

Overview:

Get the uefi_certificates field of the given pool.

Signature:

```
string get_uefi_certificates (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given pool.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_vswitch_controller

This message is deprecated.

Overview:

Get the vswitch_controller field of the given pool.

Signature:

```
string get_vswitch_controller (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_wlb_enabled

Overview:

Get the wlb_enabled field of the given pool.

Signature:

```
bool get_wlb_enabled (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_wlb_url

Overview:

Get the wlb_url field of the given pool.

Signature:

```
string get_wlb_url (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_wlb_username

Overview:

Get the wlb_username field of the given pool.

Signature:

```
string get_wlb_username (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_wlb_verify_cert

Overview:

Get the wlb_verify_cert field of the given pool.

Signature:

```
bool get_wlb_verify_cert (session ref session_id, pool ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: ha_compute_hypothetical_max_host_failures_to_tolerate

Overview:

Returns the maximum number of host failures we could tolerate before we would be unable to restart the provided VMs

Signature:

```
int ha_compute_hypothetical_max_host_failures_to_tolerate (session ref session_id, (VM
ref -> string) map configuration)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`(VM ref -> string) map`	configuration	Map of protected VM reference to restart priority

Minimum Role: read-only

Return Type: `int`

maximum value for ha_host_failures_to_tolerate given provided configuration

RPC name: ha_compute_max_host_failures_to_tolerate

Overview:

Returns the maximum number of host failures we could tolerate before we would be unable to restart configured VMs

Signature:

```
int ha_compute_max_host_failures_to_tolerate (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `int`

maximum value for ha_host_failures_to_tolerate given current configuration

RPC name: ha_compute_vm_failover_plan

Overview:

Return a VM failover plan assuming a given subset of hosts fail

Signature:

```
(VM ref -> (string -> string) map) map ha_compute_vm_failover_plan (session ref
session_id, host ref set failed_hosts, VM ref set failed_vms)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref set`	failed_hosts	The set of hosts to assume have failed
`VM ref set`	failed_vms	The set of VMs to restart

Minimum Role: pool-operator

Return Type: `(VM ref -> (string -> string) map) map`

VM failover plan: a map of VM to host to restart the host on

RPC name: ha_failover_plan_exists

Overview:

Returns true if a VM failover plan exists for up to 'n' host failures

Signature:

```
bool ha_failover_plan_exists (session ref session_id, int n)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`int`	n	The number of host failures to plan for

Minimum Role: pool-operator

Return Type: `bool`

true if a failover plan exists for the supplied number of host failures

RPC name: ha_prevent_restarts_for

Overview:

When this call returns the VM restart logic will not run for the requested number of seconds. If the argument is zero then the restart thread is immediately unblocked

Signature:

```
void ha_prevent_restarts_for (session ref session_id, int seconds)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`int`	seconds	The number of seconds to block the restart thread for

Minimum Role: pool-operator

Return Type: `void`

RPC name: has_extension

Overview:

Return true if the extension is available on the pool

Signature:

```
bool has_extension (session ref session_id, pool ref self, string name)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	The pool
`string`	name	The name of the API call

Minimum Role: pool-admin

Return Type: `bool`

True if the extension exists, false otherwise

RPC name: initialize_wlb

Overview:

Initializes workload balancing monitoring on this pool with the specified wlb server

Signature:

```
void initialize_wlb (session ref session_id, string wlb_url, string wlb_username,
string wlb_password, string xenserver_username, string xenserver_password)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	wlb_url	The ip address and port to use when accessing the wlb server
`string`	wlb_username	The username used to authenticate with the wlb server
`string`	wlb_password	The password used to authenticate with the wlb server
`string`	xenserver_username	The username used by the wlb server to authenticate with the xenserver
`string`	xenserver_password	The password used by the wlb server to authenticate with the xenserver

Minimum Role: pool-operator

Return Type: `void`

RPC name: join

Overview:

Instruct host to join a new pool

Signature:

```
void join (session ref session_id, string master_address, string master_username,
string master_password)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	master_address	The hostname of the master of the pool to join
`string`	master_username	The username of the master (for initial authentication)
`string`	master_password	The password for the master (for initial authentication)

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `JOINING_HOST_CANNOT_CONTAIN_SHARED_SRS`

RPC name: join_force

Overview:

Instruct host to join a new pool

Signature:

```
void join_force (session ref session_id, string master_address, string
master_username, string master_password)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	master_address	The hostname of the master of the pool to join
`string`	master_username	The username of the master (for initial authentication)
`string`	master_password	The password for the master (for initial authentication)

Minimum Role: pool-operator

Return Type: `void`

RPC name: management_reconfigure

Overview:

Reconfigure the management network interface for all Hosts in the Pool

Signature:

```
void management_reconfigure (session ref session_id, network ref network)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`network ref`	network	The network

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `HA_IS_ENABLED`, `PIF_NOT_PRESENT`, `CANNOT_PLUG_BOND_SLAVE`, `PIF_INCOMPATIBLE_PRIMARY_ADDRESS_TYPE`, `PIF_HAS_NO_NETWORK_CONFIGURATION`, `PIF_HAS_NO_V6_NETWORK_CONFIGURATION`

RPC name: recover_slaves

Overview:

Instruct a pool master, M, to try and contact its slaves and, if slaves are in emergency mode, reset their master address to M.

Signature:

```
host ref set recover_slaves (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `host ref set`

list of hosts whose master address were successfully reset

RPC name: remove_from_guest_agent_config

Overview:

Remove a key-value pair from the pool-wide guest agent configuration

Signature:

```
void remove_from_guest_agent_config (session ref session_id, pool ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	The pool
`string`	key	The key to remove

Minimum Role: pool-admin

Return Type: `void`

RPC name: remove_from_gui_config

Overview:

Remove the given key and its corresponding value from the gui_config field of the given pool. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_gui_config (session ref session_id, pool ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: vm-operator

Return Type: `void`

RPC name: remove_from_health_check_config

Overview:

Remove the given key and its corresponding value from the health_check_config field of the given pool. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_health_check_config (session ref session_id, pool ref self, string
key)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

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 pool. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, pool ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: remove_tags

Overview:

Remove the given value from the tags field of the given pool. If the value is not in that Set, then do nothing.

Signature:

```
void remove_tags (session ref session_id, pool ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`pool ref`	self	reference to the object
`string`	value	Value to remove

Minimum Role: vm-operator

Return Type: `void`

RPC name: retrieve_wlb_configuration

Overview:

Retrieves the pool optimization criteria from the workload balancing server

Signature:

```
(string -> string) map retrieve_wlb_configuration (session ref session_id)
```

Minimum Role: read-only

Return Type: `(string -> string) map`

The configuration used in optimizing this pool

RPC name: retrieve_wlb_recommendations

Overview:

Retrieves vm migrate recommendations for the pool from the workload balancing server

Signature:

```
(VM ref -> string set) map retrieve_wlb_recommendations (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VM ref -> string set) map`

The list of vm migration recommendations

RPC name: rotate_secret

Overview:

Signature:

```
void rotate_secret (session ref session_id)
```

Minimum Role: pool-admin

Return Type: `void`

Possible Error Codes: `INTERNAL_ERROR`, `HOST_IS_SLAVE`, `CANNOT_CONTACT_HOST`, `HA_IS_ENABLED`, `NOT_SUPPORTED_DURING_UPGRADE`

RPC name: send_test_post

Overview:

Send the given body to the given host and port, using HTTPS, and print the response. This is used for debugging the SSL layer.

Signature:

```
string send_test_post (session ref session_id, string host, int port, string body)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	host	
`int`	port	
`string`	body	

Minimum Role: pool-admin

Return Type: `string`

The response

RPC name: send_wlb_configuration

Overview:

Sets the pool optimization criteria for the workload balancing server

Signature:

```
void send_wlb_configuration (session ref session_id, (string -> string) map config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`(string -> string) map`	config	The configuration to use in optimizing this pool

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_crash_dump_SR

Overview:

Set the crash_dump_SR field of the given pool.

Signature:

```
void set_crash_dump_SR (session ref session_id, pool ref self, SR ref value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`SR ref`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_default_SR

Overview:

Set the default_SR field of the given pool.

Signature:

```
void set_default_SR (session ref session_id, pool ref self, SR ref value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`SR ref`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_gui_config

Overview:

Set the gui_config field of the given pool.

Signature:

```
void set_gui_config (session ref session_id, pool ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-operator

Return Type: `void`

RPC name: set_ha_allow_overcommit

Overview:

Set the ha_allow_overcommit field of the given pool.

Signature:

```
void set_ha_allow_overcommit (session ref session_id, pool ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`bool`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_ha_host_failures_to_tolerate

Overview:

Set the maximum number of host failures to consider in the HA VM restart planner

Signature:

```
void set_ha_host_failures_to_tolerate (session ref session_id, pool ref self, int
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	The pool
`int`	value	New number of host failures to consider

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_health_check_config

Overview:

Set the health_check_config field of the given pool.

Signature:

```
void set_health_check_config (session ref session_id, pool ref self, (string ->
string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_igmp_snooping_enabled

Overview:

Enable or disable IGMP Snooping on the pool.

Signature:

```
void set_igmp_snooping_enabled (session ref session_id, pool ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	The pool
`bool`	value	Enable or disable IGMP Snooping on the pool

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_is_psr_pending

Overview:

Set the is_psr_pending field of the given pool.

Signature:

```
void set_is_psr_pending (session ref session_id, pool ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`bool`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_live_patching_disabled

Overview:

Set the live_patching_disabled field of the given pool.

Signature:

```
void set_live_patching_disabled (session ref session_id, pool ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`bool`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_name_description

Overview:

Set the name_description field of the given pool.

Signature:

```
void set_name_description (session ref session_id, pool ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_name_label

Overview:

Set the name_label field of the given pool.

Signature:

```
void set_name_label (session ref session_id, pool ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given pool.

Signature:

```
void set_other_config (session ref session_id, pool ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_policy_no_vendor_device

Overview:

Set the policy_no_vendor_device field of the given pool.

Signature:

```
void set_policy_no_vendor_device (session ref session_id, pool ref self, bool value)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`bool`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_suspend_image_SR

Overview:

Set the suspend_image_SR field of the given pool.

Signature:

```
void set_suspend_image_SR (session ref session_id, pool ref self, SR ref value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`SR ref`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_tags

Overview:

Set the tags field of the given pool.

Signature:

```
void set_tags (session ref session_id, pool ref self, string set value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`pool ref`	self	reference to the object
`string set`	value	New value to set

Minimum Role: vm-operator

Return Type: `void`

RPC name: set_uefi_certificates

Overview:

Set the uefi_certificates field of the given pool.

Signature:

```
void set_uefi_certificates (session ref session_id, pool ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_vswitch_controller

This message is deprecated.

Overview:

Set the IP address of the vswitch controller.

Signature:

```
void set_vswitch_controller (session ref session_id, string address)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`string`	address	IP address of the vswitch controller.

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_wlb_enabled

Overview:

Set the wlb_enabled field of the given pool.

Signature:

```
void set_wlb_enabled (session ref session_id, pool ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`bool`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_wlb_verify_cert

Overview:

Set the wlb_verify_cert field of the given pool.

Signature:

```
void set_wlb_verify_cert (session ref session_id, pool ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	reference to the object
`bool`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: sync_database

Overview:

Forcibly synchronise the database now

Signature:

```
void sync_database (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `void`

RPC name: test_archive_target

Overview:

This call tests if a location is valid

Signature:

```
string test_archive_target (session ref session_id, pool ref self, (string -> string)
map config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool ref`	self	Reference to the pool
`(string -> string) map`	config	Location config settings to test

Minimum Role: pool-operator

Return Type: `string`

An XMLRPC result

Class: pool_patch

This class is deprecated.

Pool-wide patches

Fields for class: pool_patch

Field	Type	Qualifier	Description
after_apply_guidance	after_apply_guidance set	*RO/runtime*	**Deprecated**. What the client should do after this patch has been applied.
host_patches	host_patch ref set	*RO/runtime*	**Deprecated**. This hosts this patch is applied to.
name_description	string	*RO/constructor*	**Deprecated**. a notes field containing human-readable description
name_label	string	*RO/constructor*	**Deprecated**. a human-readable name
other_config	(string -> string) map	*RW*	**Deprecated**. additional configuration
pool_applied	bool	*RO/runtime*	**Deprecated**. This patch should be applied across the entire pool
pool_update	pool_update ref	*RO/constructor*	**Deprecated**. A reference to the associated pool_update object
size	int	*RO/runtime*	**Deprecated**. Size of the patch
uuid	string	*RO/runtime*	**Deprecated**. Unique identifier/object reference
version	string	*RO/constructor*	**Deprecated**. Patch version number

RPCs associated with class: pool_patch

RPC name: add_to_other_config

This message is deprecated.

Overview:

Add the given key-value pair to the other_config field of the given pool_patch.

Signature:

```
void add_to_other_config (session ref session_id, pool_patch ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
pool_patch ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: apply

This message is deprecated.

Overview:

Apply the selected patch to a host and return its output

Signature:

```
string apply (session ref session_id, pool_patch ref self, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	The patch to apply
`host ref`	host	The host to apply the patch too

Minimum Role: pool-operator

Return Type: `string`

the output of the patch application process

RPC name: clean

This message is deprecated.

Overview:

Removes the patch's files from the server

Signature:

```
void clean (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	The patch to clean up

Minimum Role: pool-operator

Return Type: `void`

RPC name: clean_on_host

This message is deprecated.

Overview:

Removes the patch's files from the specified host

Signature:

```
void clean_on_host (session ref session_id, pool_patch ref self, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	The patch to clean up
`host ref`	host	The host on which to clean the patch

Minimum Role: pool-operator

Return Type: `void`

RPC name: destroy

This message is deprecated.

Overview:

Removes the patch's files from all hosts in the pool, and removes the database entries. Only works on unapplied patches.

Signature:

```
void destroy (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	The patch to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_after_apply_guidance

This message is deprecated.

Overview:

Get the after_apply_guidance field of the given pool_patch.

Signature:

```
after_apply_guidance set get_after_apply_guidance (session ref session_id, pool_patch
ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `after_apply_guidance set`

value of the field

RPC name: get_all

This message is deprecated.

Overview:

Return a list of all the pool_patchs known to the system.

Signature:

```
pool_patch ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `pool_patch ref set`

references to all objects

RPC name: get_all_records

This message is deprecated.

Overview:

Return a map of pool_patch references to pool_patch records for all pool_patchs known to the system.

Signature:

```
(pool_patch ref -> pool_patch record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(pool_patch ref -> pool_patch record) map`

records of all objects

RPC name: get_by_name_label

This message is deprecated.

Overview:

Get all the pool_patch instances with the given label.

Signature:

```
pool_patch ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `pool_patch ref set`

references to objects with matching names

RPC name: get_by_uuid

This message is deprecated.

Overview:

Get a reference to the pool_patch instance with the specified UUID.

Signature:

```
pool_patch ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `pool_patch ref`

reference to the object

RPC name: get_host_patches

This message is deprecated.

Overview:

Get the host_patches field of the given pool_patch.

Signature:

```
host_patch ref set get_host_patches (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host_patch ref set`

value of the field

RPC name: get_name_description

This message is deprecated.

Overview:

Get the name/description field of the given pool_patch.

Signature:

```
string get_name_description (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

This message is deprecated.

Overview:

Get the name/label field of the given pool_patch.

Signature:

```
string get_name_label (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_other_config

This message is deprecated.

Overview:

Get the other_config field of the given pool_patch.

Signature:

```
(string -> string) map get_other_config (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_pool_applied

This message is deprecated.

Overview:

Get the pool_applied field of the given pool_patch.

Signature:

```
bool get_pool_applied (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_pool_update

This message is deprecated.

Overview:

Get the pool_update field of the given pool_patch.

Signature:

```
pool_update ref get_pool_update (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `pool_update ref`

value of the field

RPC name: get_record

This message is deprecated.

Overview:

Get a record containing the current state of the given pool_patch.

Signature:

```
pool_patch record get_record (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `pool_patch record`

all fields from the object

RPC name: get_size

This message is deprecated.

Overview:

Get the size field of the given pool_patch.

Signature:

```
int get_size (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_uuid

This message is deprecated.

Overview:

Get the uuid field of the given pool_patch.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_version

This message is deprecated.

Overview:

Get the version field of the given pool_patch.

Signature:

```
string get_version (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: pool_apply

This message is deprecated.

Overview:

Apply the selected patch to all hosts in the pool and return a map of host_ref -> patch output

Signature:

```
void pool_apply (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	The patch to apply

Minimum Role: pool-operator

Return Type: `void`

RPC name: pool_clean

This message is deprecated.

Overview:

Removes the patch's files from all hosts in the pool, but does not remove the database entries

Signature:

```
void pool_clean (session ref session_id, pool_patch ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	The patch to clean up

Minimum Role: pool-operator

Return Type: `void`

RPC name: precheck

This message is deprecated.

Overview:

Execute the precheck stage of the selected patch on a host and return its output

Signature:

```
string precheck (session ref session_id, pool_patch ref self, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	The patch whose prechecks will be run
`host ref`	host	The host to run the prechecks on

Minimum Role: pool-operator

Return Type: `string`

the output of the patch prechecks

RPC name: remove_from_other_config

This message is deprecated.

Overview:

Remove the given key and its corresponding value from the other_config field of the given pool_patch. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, pool_patch ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

This message is deprecated.

Overview:

Set the other_config field of the given pool_patch.

Signature:

```
void set_other_config (session ref session_id, pool_patch ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_patch ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: pool_update

Pool-wide updates to the host software

Fields for class: pool_update

Field	Type	Qualifier	Description
after_apply_guidance	`update_after_apply_guidance set`	*RO/constructor*	What the client should do after this update has been applied.

Field	Type	Qualifier	Description
enforce_homogeneity	`bool`	*RO/constructor*	Flag - if true, all hosts in a pool must apply this update
hosts	`host ref set`	*RO/runtime*	The hosts that have applied this update.
installation_size	`int`	*RO/constructor*	Size of the update in bytes
key	`string`	*RO/constructor*	GPG key of the update
name_description	`string`	*RO/constructor*	a notes field containing human-readable description
name_label	`string`	*RO/constructor*	a human-readable name
other_config	`(string -> string) map`	*RW*	additional configuration
uuid	`string`	*RO/runtime*	Unique identifier/object reference
vdi	`VDI ref`	*RO/constructor*	VDI the update was uploaded to
version	`string`	*RO/constructor*	Update version number

RPCs associated with class: pool_update

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given pool_update.

Signature:

```
void add_to_other_config (session ref session_id, pool_update ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: apply

Overview:

Apply the selected update to a host

Signature:

```
void apply (session ref session_id, pool_update ref self, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	The update to apply
`host ref`	host	The host to apply the update to.

Minimum Role: pool-operator

Return Type: `void`

RPC name: destroy

Overview:

Removes the database entry. Only works on unapplied update.

Signature:

```
void destroy (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	The update to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_after_apply_guidance

Overview:

Get the after_apply_guidance field of the given pool_update.

Signature:

```
update_after_apply_guidance set get_after_apply_guidance (session ref session_id,
pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object

Minimum Role: read-only

Return Type: `update_after_apply_guidance set`

value of the field

RPC name: get_all

Overview:

Return a list of all the pool_updates known to the system.

Signature:

```
pool_update ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `pool_update ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of pool_update references to pool_update records for all pool_updates known to the system.

Signature:

```
(pool_update ref -> pool_update record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(pool_update ref -> pool_update record) map`

records of all objects

RPC name: get_by_name_label

Overview:

Get all the pool_update instances with the given label.

Signature:

```
pool_update ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `pool_update ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the pool_update instance with the specified UUID.

Signature:

```
pool_update ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `pool_update ref`

reference to the object

RPC name: get_enforce_homogeneity

Overview:

Get the enforce_homogeneity field of the given pool_update.

Signature:

```
bool get_enforce_homogeneity (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_hosts

Overview:

Get the hosts field of the given pool_update.

Signature:

```
host ref set get_hosts (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref set`

value of the field

RPC name: get_installation_size

Overview:

Get the installation_size field of the given pool_update.

Signature:

```
int get_installation_size (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_key

Overview:

Get the key field of the given pool_update.

Signature:

```
string get_key (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given pool_update.

Signature:

```
string get_name_description (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

Overview:

Get the name/label field of the given pool_update.

Signature:

```
string get_name_label (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given pool_update.

Signature:

```
(string -> string) map get_other_config (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given pool_update.

Signature:

```
pool_update record get_record (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object

Minimum Role: read-only

Return Type: `pool_update record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given pool_update.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`pool_update ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_vdi

Overview:

Get the vdi field of the given pool_update.

Signature:

```
VDI ref get_vdi (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VDI ref`

value of the field

RPC name: get_version

Overview:

Get the version field of the given pool_update.

Signature:

```
string get_version (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: introduce

Overview:

Introduce update VDI

Signature:

```
pool_update ref introduce (session ref session_id, VDI ref vdi)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	vdi	The VDI which contains a software update.

Minimum Role: pool-operator

Return Type: `pool_update ref`

the introduced pool update

RPC name: pool_apply

Overview:

Apply the selected update to all hosts in the pool

Signature:

```
void pool_apply (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	The update to apply

Minimum Role: pool-operator

Return Type: `void`

RPC name: pool_clean

Overview:

Removes the update's files from all hosts in the pool, but does not revert the update

Signature:

```
void pool_clean (session ref session_id, pool_update ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	The update to clean up

Minimum Role: pool-operator

Return Type: `void`

RPC name: precheck

Overview:

Execute the precheck stage of the selected update on a host

Signature:

```
livepatch_status precheck (session ref session_id, pool_update ref self, host ref
host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	The update whose prechecks will be run
`host ref`	host	The host to run the prechecks on.

Minimum Role: pool-operator

Return Type: `livepatch_status`

The precheck pool update

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given pool_update. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, pool_update ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given pool_update.

Signature:

```
void set_other_config (session ref session_id, pool_update ref self, (string ->
string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`pool_update ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: probe_result

A set of properties that describe one result element of SR.probe. Result elements and properties can change dynamically based on changes to the the SR.probe input-parameters or the target.

Fields for class: probe_result

Field	Type	Qualifier	Description
complete	bool	*RO/runtime*	True if this configuration is complete and can be used to call SR.create. False if it requires further iterative calls to SR.probe, to potentially narrow down on a configuration that can be used.
configuration	(string -> string) map	*RO/runtime*	Plugin-specific configuration which describes where and how to locate the storage repository. This may include the physical block device name, a remote NFS server and path or an RBD storage pool.
extra_info	(string -> string) map	*RO/runtime*	Additional plugin-specific information about this configuration, that might be of use for an API user. This can for example include the LUN or the WWPN.
sr	sr_stat record option	*RO/runtime*	Existing SR found for this configuration

RPCs associated with class: probe_result

Class probe_result has no additional RPCs associated with it.

Class: PUSB

A physical USB device

Fields for class: PUSB

Field	Type	Qualifier	Description
description	string	*RO/constructor*	USB device description
host	host ref	*RO/constructor*	Physical machine that owns the USB device
other_config	(string -> string) map	*RW*	additional configuration
passthrough_enabled	bool	*RO/runtime*	enabled for passthrough
path	string	*RO/constructor*	port path of USB device
product_desc	string	*RO/constructor*	product description of the USB device
product_id	string	*RO/constructor*	product id of the USB device
serial	string	*RO/constructor*	serial of the USB device
speed	float	*RO/constructor*	USB device speed
USB_group	USB_group ref	*RO/constructor*	USB group the PUSB is contained in

Field	Type	Qualifier	Description
uuid	`string`	*RO/runtime*	Unique identifier/object reference
vendor_desc	`string`	*RO/constructor*	vendor description of the USB device
vendor_id	`string`	*RO/constructor*	vendor id of the USB device
version	`string`	*RO/constructor*	USB device version

RPCs associated with class: PUSB

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given PUSB.

Signature:

```
void add_to_other_config (session ref session_id, PUSB ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-admin

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the PUSBs known to the system.

Signature:

```
PUSB ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `PUSB ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of PUSB references to PUSB records for all PUSBs known to the system.

Signature:

```
(PUSB ref -> PUSB record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(PUSB ref -> PUSB record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the PUSB instance with the specified UUID.

Signature:

```
PUSB ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `PUSB ref`

reference to the object

RPC name: get_description

Overview:

Get the description field of the given PUSB.

Signature:

```
string get_description (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_host

Overview:

Get the host field of the given PUSB.

Signature:

```
host ref get_host (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given PUSB.

Signature:

```
(string -> string) map get_other_config (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_passthrough_enabled

Overview:

Get the passthrough_enabled field of the given PUSB.

Signature:

```
bool get_passthrough_enabled (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_path

Overview:

Get the path field of the given PUSB.

Signature:

```
string get_path (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_product_desc

Overview:

Get the product_desc field of the given PUSB.

Signature:

```
string get_product_desc (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_product_id

Overview:

Get the product_id field of the given PUSB.

Signature:

```
string get_product_id (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given PUSB.

Signature:

```
PUSB record get_record (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PUSB record`

all fields from the object

RPC name: get_serial

Overview:

Get the serial field of the given PUSB.

Signature:

```
string get_serial (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_speed

Overview:

Get the speed field of the given PUSB.

Signature:

```
float get_speed (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `float`

value of the field

RPC name: get_USB_group

Overview:

Get the USB_group field of the given PUSB.

Signature:

```
USB_group ref get_USB_group (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `USB_group ref`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given PUSB.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_vendor_desc

Overview:

Get the vendor_desc field of the given PUSB.

Signature:

```
string get_vendor_desc (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_vendor_id

Overview:

Get the vendor_id field of the given PUSB.

Signature:

```
string get_vendor_id (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_version

Overview:

Get the version field of the given PUSB.

Signature:

```
string get_version (session ref session_id, PUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given PUSB. If the key is not in that Map, then do nothing.

Signature:

```

```

```
void remove_from_other_config (session ref session_id, PUSB ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PUSB ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-admin

Return Type: `void`

RPC name: scan

Overview:

Signature:

```
void scan (session ref session_id, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host

Minimum Role: pool-admin

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given PUSB.

Signature:

```
void set_other_config (session ref session_id, PUSB ref self, (string -> string) map
value)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
PUSB ref	self	reference to the object
(string -> string) map	value	New value to set

Minimum Role: pool-admin

Return Type: void

RPC name: set_passthrough_enabled

Overview:

Signature:

```
void set_passthrough_enabled (session ref session_id, PUSB ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PUSB ref	self	this PUSB
bool	value	passthrough is enabled when true and disabled with false

Minimum Role: pool-admin

Return Type: void

Class: PVS_cache_storage

Describes the storage that is available to a PVS site for caching purposes

Fields for class: PVS_cache_storage

Field	Type	Qualifier	Description
host	host ref	*RO/constructor*	The host on which this object defines PVS cache storage
site	PVS_site ref	*RO/constructor*	The PVS_site for which this object defines the storage
size	int	*RO/constructor*	The size of the cache VDI (in bytes)
SR	SR ref	*RO/constructor*	SR providing storage for the PVS cache
uuid	string	*RO/runtime*	Unique identifier/object reference
VDI	VDI ref	*RO/runtime*	The VDI used for caching

RPCs associated with class: PVS_cache_storage

RPC name: create

Overview:

Create a new PVS_cache_storage instance, and return its handle.

Signature:

```
PVS_cache_storage ref create (session ref session_id, PVS_cache_storage record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_cache_storage record`	args	All constructor arguments

Minimum Role: pool-operator

Return Type: `PVS_cache_storage ref`

reference to the newly created object

RPC name: destroy

Overview:

Destroy the specified PVS_cache_storage instance.

Signature:

```
void destroy (session ref session_id, PVS_cache_storage ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_cache_storage ref`	self	reference to the object

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the PVS_cache_storages known to the system.

Signature:

```
PVS_cache_storage ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `PVS_cache_storage ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of PVS_cache_storage references to PVS_cache_storage records for all PVS_cache_storages known to the system.

Signature:

```
(PVS_cache_storage ref -> PVS_cache_storage record) map get_all_records (session ref
session_id)
```

Minimum Role: read-only

Return Type: `(PVS_cache_storage ref -> PVS_cache_storage record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the PVS_cache_storage instance with the specified UUID.

Signature:

```
PVS_cache_storage ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `PVS_cache_storage ref`

reference to the object

RPC name: get_host

Overview:

Get the host field of the given PVS_cache_storage.

Signature:

```
host ref get_host (session ref session_id, PVS_cache_storage ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PVS_cache_storage ref	self	reference to the object

Minimum Role: read-only

Return Type: host ref

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given PVS_cache_storage.

Signature:

```
PVS_cache_storage record get_record (session ref session_id, PVS_cache_storage ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PVS_cache_storage ref	self	reference to the object

Minimum Role: read-only

Return Type: PVS_cache_storage record

all fields from the object

RPC name: get_site

Overview:

Get the site field of the given PVS_cache_storage.

Signature:

```
PVS_site ref get_site (session ref session_id, PVS_cache_storage ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_cache_storage ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PVS_site ref`

value of the field

RPC name: get_size

Overview:

Get the size field of the given PVS_cache_storage.

Signature:

```
int get_size (session ref session_id, PVS_cache_storage ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_cache_storage ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_SR

Overview:

Get the SR field of the given PVS_cache_storage.

Signature:

```
SR ref get_SR (session ref session_id, PVS_cache_storage ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_cache_storage ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SR ref`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given PVS_cache_storage.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_cache_storage ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VDI

Overview:

Get the VDI field of the given PVS_cache_storage.

Signature:

```
VDI ref get_VDI (session ref session_id, PVS_cache_storage ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_cache_storage ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VDI ref`

value of the field

Class: PVS_proxy

a proxy connects a VM/VIF with a PVS site

Fields for class: PVS_proxy

Field	Type	Qualifier	Description
currently_attached	`bool`	*RO/runtime*	true = VM is currently proxied
site	`PVS_site ref`	*RO/constructor*	PVS site this proxy is part of
status	`pvs_proxy_status`	*RO/runtime*	The run-time status of the proxy
uuid	`string`	*RO/runtime*	Unique identifier/object reference
VIF	`VIF ref`	*RO/constructor*	VIF of the VM using the proxy

RPCs associated with class: PVS_proxy

RPC name: create

Overview:

Configure a VM/VIF to use a PVS proxy

Signature:

```
PVS_proxy ref create (session ref session_id, PVS_site ref site, VIF ref VIF)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_site ref`	site	PVS site that we proxy for
`VIF ref`	VIF	VIF for the VM that needs to be proxied

Minimum Role: pool-operator

Return Type: `PVS_proxy ref`

the new PVS proxy

RPC name: destroy

Overview:

remove (or switch off) a PVS proxy for this VM

Signature:

```
void destroy (session ref session_id, PVS_proxy ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_proxy ref`	self	this PVS proxy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the PVS_proxys known to the system.

Signature:

```
PVS_proxy ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `PVS_proxy ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of PVS_proxy references to PVS_proxy records for all PVS_proxys known to the system.

Signature:

```
(PVS_proxy ref -> PVS_proxy record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(PVS_proxy ref -> PVS_proxy record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the PVS_proxy instance with the specified UUID.

Signature:

```
PVS_proxy ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `PVS_proxy ref`

reference to the object

RPC name: get_currently_attached

Overview:

Get the currently_attached field of the given PVS_proxy.

Signature:

```
bool get_currently_attached (session ref session_id, PVS_proxy ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_proxy ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given PVS_proxy.

Signature:

```
PVS_proxy record get_record (session ref session_id, PVS_proxy ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_proxy ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PVS_proxy record`

all fields from the object

RPC name: get_site

Overview:

Get the site field of the given PVS_proxy.

Signature:

```
PVS_site ref get_site (session ref session_id, PVS_proxy ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_proxy ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PVS_site ref`

value of the field

RPC name: get_status

Overview:

Get the status field of the given PVS_proxy.

Signature:

```
pvs_proxy_status get_status (session ref session_id, PVS_proxy ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_proxy ref`	self	reference to the object

Minimum Role: read-only

Return Type: `pvs_proxy_status`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given PVS_proxy.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_proxy ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VIF

Overview:

Get the VIF field of the given PVS_proxy.

Signature:

```
VIF ref get_VIF (session ref session_id, PVS_proxy ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_proxy ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VIF ref`

value of the field

Class: PVS_server

individual machine serving provisioning (block) data

Fields for class: PVS_server

Field	Type	Qualifier	Description
addresses	`string set`	*RO/constructor*	IPv4 addresses of this server
first_port	`int`	*RO/constructor*	First UDP port accepted by this server
last_port	`int`	*RO/constructor*	Last UDP port accepted by this server
site	`PVS_site ref`	*RO/constructor*	PVS site this server is part of
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: PVS_server

RPC name: forget

Overview:

forget a PVS server

Signature:

```
void forget (session ref session_id, PVS_server ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PVS_server ref	self	this PVS server

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_addresses

Overview:

Get the addresses field of the given PVS_server.

Signature:

```
string set get_addresses (session ref session_id, PVS_server ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PVS_server ref	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_all

Overview:

Return a list of all the PVS_servers known to the system.

Signature:

```
PVS_server ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `PVS_server ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of PVS_server references to PVS_server records for all PVS_servers known to the system.

Signature:

```
(PVS_server ref -> PVS_server record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(PVS_server ref -> PVS_server record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the PVS_server instance with the specified UUID.

Signature:

```
PVS_server ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `PVS_server ref`

reference to the object

RPC name: get_first_port

Overview:

Get the first_port field of the given PVS_server.

Signature:

```
int get_first_port (session ref session_id, PVS_server ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_server ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_last_port

Overview:

Get the last_port field of the given PVS_server.

Signature:

```
int get_last_port (session ref session_id, PVS_server ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_server ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given PVS_server.

Signature:

```
PVS_server record get_record (session ref session_id, PVS_server ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`PVS_server ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PVS_server record`

all fields from the object

RPC name: get_site

Overview:

Get the site field of the given PVS_server.

Signature:

```
PVS_site ref get_site (session ref session_id, PVS_server ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_server ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PVS_site ref`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given PVS_server.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_server ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: introduce

Overview:

introduce new PVS server

Signature:

```
PVS_server ref introduce (session ref session_id, string set addresses, int
first_port, int last_port, PVS_site ref site)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string set`	addresses	IPv4 addresses of the server
`int`	first_port	first UDP port accepted by this server
`int`	last_port	last UDP port accepted by this server
`PVS_site ref`	site	PVS site this server is a part of

Minimum Role: pool-operator

Return Type: `PVS_server ref`

the new PVS server

Class: PVS_site

machines serving blocks of data for provisioning VMs

Fields for class: PVS_site

Field	Type	Qualifier	Description
cache_storage	`PVS_cache_storage ref set`	*RO/runtime*	The SR used by PVS proxy for the cache
name_description	`string`	*RW*	a notes field containing human-readable description
name_label	`string`	*RW*	a human-readable name
proxies	`PVS_proxy ref set`	*RO/runtime*	The set of proxies associated with the site

Field	Type	Qualifier	Description
PVS_uuid	`string`	*RO/constructor*	Unique identifier of the PVS site, as configured in PVS
servers	`PVS_server ref set`	*RO/runtime*	The set of PVS servers in the site
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: PVS_site

RPC name: forget

Overview:

Remove a site's meta data

Signature:

```
void forget (session ref session_id, PVS_site ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_site ref`	self	this PVS site

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `PVS_SITE_CONTAINS_RUNNING_PROXIES`, `PVS_SITE_CONTAINS_SERVERS`

RPC name: get_all

Overview:

Return a list of all the PVS_sites known to the system.

Signature:

```
PVS_site ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `PVS_site ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of PVS_site references to PVS_site records for all PVS_sites known to the system.

Signature:

```
(PVS_site ref -> PVS_site record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(PVS_site ref -> PVS_site record) map`

records of all objects

RPC name: get_by_name_label

Overview:

Get all the PVS_site instances with the given label.

Signature:

```
PVS_site ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `PVS_site ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the PVS_site instance with the specified UUID.

Signature:

```
PVS_site ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `PVS_site ref`

reference to the object

RPC name: get_cache_storage

Overview:

Get the cache_storage field of the given PVS_site.

Signature:

```
PVS_cache_storage ref set get_cache_storage (session ref session_id, PVS_site ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_site ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PVS_cache_storage ref set`

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given PVS_site.

Signature:

```
string get_name_description (session ref session_id, PVS_site ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`PVS_site ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

Overview:

Get the name/label field of the given PVS_site.

Signature:

```
string get_name_label (session ref session_id, PVS_site ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_site ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_proxies

Overview:

Get the proxies field of the given PVS_site.

Signature:

```
PVS_proxy ref set get_proxies (session ref session_id, PVS_site ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_site ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PVS_proxy ref set`

value of the field

RPC name: get_PVS_uuid

Overview:

Get the PVS_uuid field of the given PVS_site.

Signature:

```
string get_PVS_uuid (session ref session_id, PVS_site ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_site ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given PVS_site.

Signature:

```
PVS_site record get_record (session ref session_id, PVS_site ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_site ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PVS_site record`

all fields from the object

RPC name: get_servers

Overview:

Get the servers field of the given PVS_site.

Signature:

```
PVS_server ref set get_servers (session ref session_id, PVS_site ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_site ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PVS_server ref set`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given PVS_site.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_site ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: introduce

Overview:

Introduce new PVS site

Signature:

```
PVS_site ref introduce (session ref session_id, string name_label, string
name_description, string PVS_uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	name_label	name of the PVS site
`string`	name_description	description of the PVS site
`string`	PVS_uuid	unique identifier of the PVS site

Minimum Role: pool-operator

Return Type: `PVS_site ref`

the new PVS site

RPC name: set_name_description

Overview:

Set the name/description field of the given PVS_site.

Signature:

```
void set_name_description (session ref session_id, PVS_site ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_site ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_name_label

Overview:

Set the name/label field of the given PVS_site.

Signature:

```
void set_name_label (session ref session_id, PVS_site ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_site ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_PVS_uuid

Overview:

Update the PVS UUID of the PVS site

Signature:

```
void set_PVS_uuid (session ref session_id, PVS_site ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PVS_site ref`	self	this PVS site
`string`	value	PVS UUID to be used

Minimum Role: pool-operator

Return Type: `void`

Class: role

A set of permissions associated with a subject

Fields for class: role

Field	Type	Qualifier	Description
name_description	`string`	*RO/constructor*	what this role is for
name_label	`string`	*RO/constructor*	a short user-friendly name for the role
subroles	`role ref set`	*RO/constructor*	a list of pointers to other roles or permissions
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: role

RPC name: get_all

Overview:

Return a list of all the roles known to the system.

Signature:

```
role ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `role ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of role references to role records for all roles known to the system.

Signature:

```
(role ref -> role record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(role ref -> role record) map`

records of all objects

RPC name: get_by_name_label

Overview:

Get all the role instances with the given label.

Signature:

```
role ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `role ref set`

references to objects with matching names

RPC name: get_by_permission

Overview:

This call returns a list of roles given a permission

Signature:

```
role ref set get_by_permission (session ref session_id, role ref permission)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`role ref`	permission	a reference to a permission

Minimum Role: read-only

Return Type: `role ref set`

a list of references to roles

RPC name: get_by_permission_name_label

Overview:

This call returns a list of roles given a permission name

Signature:

```
role ref set get_by_permission_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	The short friendly name of the role

Minimum Role: read-only

Return Type: `role ref set`

a list of references to roles

RPC name: get_by_uuid

Overview:

Get a reference to the role instance with the specified UUID.

Signature:

```
role ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `role ref`

reference to the object

RPC name: get_name_description

Overview:

Get the name/description field of the given role.

Signature:

```
string get_name_description (session ref session_id, role ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`role ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

Overview:

Get the name/label field of the given role.

Signature:

```
string get_name_label (session ref session_id, role ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`role ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_permissions

Overview:

This call returns a list of permissions given a role

Signature:

```
role ref set get_permissions (session ref session_id, role ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`role ref`	self	a reference to a role

Minimum Role: read-only

Return Type: `role ref set`

a list of permissions

RPC name: get_permissions_name_label

Overview:

This call returns a list of permission names given a role

Signature:

```
string set get_permissions_name_label (session ref session_id, role ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`role ref`	self	a reference to a role

Minimum Role: read-only

Return Type: `string set`

a list of permission names

RPC name: get_record

Overview:

Get a record containing the current state of the given role.

Signature:

```
role record get_record (session ref session_id, role ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`role ref`	self	reference to the object

Minimum Role: read-only

Return Type: `role record`

all fields from the object

RPC name: get_subroles

Overview:

Get the subroles field of the given role.

Signature:

```
role ref set get_subroles (session ref session_id, role ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`role ref`	self	reference to the object

Minimum Role: read-only

Return Type: `role ref set`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given role.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`role ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

Class: SDN_controller

Describes the SDN controller that is to connect with the pool

Fields for class: SDN_controller

Field	Type	Qualifier	Description
address	`string`	*RO/constructor*	IP address of the controller
port	`int`	*RO/constructor*	TCP port of the controller
protocol	`sdn_controller_protocol`	*RO/constructor*	Protocol to connect with SDN controller
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: SDN_controller

RPC name: forget

Overview:

Remove the OVS manager of the pool and destroy the db record.

Signature:

```
void forget (session ref session_id, SDN_controller ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SDN_controller ref`	self	this SDN controller

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_address

Overview:

Get the address field of the given SDN_controller.

Signature:

```
string get_address (session ref session_id, SDN_controller ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SDN_controller ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_all

Overview:

Return a list of all the SDN_controllers known to the system.

Signature:

```
SDN_controller ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `SDN_controller ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of SDN_controller references to SDN_controller records for all SDN_controllers known to the system.

Signature:

```
(SDN_controller ref -> SDN_controller record) map get_all_records (session ref
session_id)
```

Minimum Role: read-only

Return Type: `(SDN_controller ref -> SDN_controller record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the SDN_controller instance with the specified UUID.

Signature:

```
SDN_controller ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `SDN_controller ref`

reference to the object

RPC name: get_port

Overview:

Get the port field of the given SDN_controller.

Signature:

```
int get_port (session ref session_id, SDN_controller ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SDN_controller ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_protocol

Overview:

Get the protocol field of the given SDN_controller.

Signature:

```
sdn_controller_protocol get_protocol (session ref session_id, SDN_controller ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SDN_controller ref`	self	reference to the object

Minimum Role: read-only

Return Type: `sdn_controller_protocol`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given SDN_controller.

Signature:

```
SDN_controller record get_record (session ref session_id, SDN_controller ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SDN_controller ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SDN_controller record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given SDN_controller.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SDN_controller ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: introduce

Overview:

Introduce an SDN controller to the pool.

Signature:

```
SDN_controller ref introduce (session ref session_id, sdn_controller_protocol
protocol, string address, int port)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`sdn_controller_protocol`	protocol	Protocol to connect with the controller.
`string`	address	IP address of the controller.
`int`	port	TCP port of the controller.

Minimum Role: pool-operator

Return Type: `SDN_controller ref`

the introduced SDN controller

Class: secret

A secret

Fields for class: secret

Field	Type	Qualifier	Description
other_config	`(string -> string) map`	*RW*	other_config
uuid	`string`	*RO/runtime*	Unique identifier/object reference

Field	Type	Qualifier	Description
value	`string`	*RW*	the secret

RPCs associated with class: secret

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given secret.

Signature:

```
void add_to_other_config (session ref session_id, secret ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`secret ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: create

Overview:

Create a new secret instance, and return its handle.

Signature:

```
secret ref create (session ref session_id, secret record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`secret record`	args	All constructor arguments

Minimum Role: pool-operator

Return Type: `secret ref`

reference to the newly created object

RPC name: destroy

Overview:

Destroy the specified secret instance.

Signature:

```
void destroy (session ref session_id, secret ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`secret ref`	self	reference to the object

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the secrets known to the system.

Signature:

```
secret ref set get_all (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `secret ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of secret references to secret records for all secrets known to the system.

Signature:

```
(secret ref -> secret record) map get_all_records (session ref session_id)
```

Minimum Role: pool-operator

Return Type: `(secret ref -> secret record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the secret instance with the specified UUID.

Signature:

```
secret ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: pool-operator

Return Type: `secret ref`

reference to the object

RPC name: get_other_config

Overview:

Get the other_config field of the given secret.

Signature:

```
(string -> string) map get_other_config (session ref session_id, secret ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`secret ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given secret.

Signature:

```
secret record get_record (session ref session_id, secret ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`secret ref`	self	reference to the object

Minimum Role: pool-operator

Return Type: `secret record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given secret.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`secret ref`	self	reference to the object

Minimum Role: pool-operator

Return Type: `string`

value of the field

RPC name: get_value

Overview:

Get the value field of the given secret.

Signature:

```
string get_value (session ref session_id, secret ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`secret ref`	self	reference to the object

Minimum Role: pool-operator

Return Type: `string`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given secret. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, secret ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`secret ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given secret.

Signature:

```
void set_other_config (session ref session_id, secret ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`secret ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_value

Overview:

Set the value field of the given secret.

Signature:

```
void set_value (session ref session_id, secret ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`secret ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: session

A session

Fields for class: session

Field	Type	Qualifier	Description

Field	Type	Qualifier	Description
auth_user_name	`string`	*RO/runtime*	the subject name of the user that was externally authenticated. If a session instance has is_local_superuser set, then the value of this field is undefined.
auth_user_sid	`string`	*RO/runtime*	the subject identifier of the user that was externally authenticated. If a session instance has is_local_superuser set, then the value of this field is undefined.
is_local_superuser	`bool`	*RO/runtime*	true iff this session was created using local superuser credentials
last_active	`datetime`	*RO/runtime*	Timestamp for last time session was active
originator	`string`	*RO/runtime*	a key string provided by a API user to distinguish itself from other users sharing the same login name
other_config	`(string -> string) map`	*RW*	additional configuration
parent	`session ref`	*RO/constructor*	references the parent session that created this session
pool	`bool`	*RO/runtime*	True if this session relates to a intra-pool login, false otherwise
rbac_permissions	`string set`	*RO/constructor*	list with all RBAC permissions for this session
subject	`subject ref`	*RO/runtime*	references the subject instance that created the session. If a session instance has is_local_superuser set, then the value of this field is undefined.
tasks	`task ref set`	*RO/runtime*	list of tasks created using the current session
this_host	`host ref`	*RO/runtime*	Currently connected host
this_user	`user ref`	*RO/runtime*	Currently connected user
uuid	`string`	*RO/runtime*	Unique identifier/object reference
validation_time	`datetime`	*RO/runtime*	time when session was last validated

RPCs associated with class: session

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given session.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-admin

Return Type: `void`

RPC name: change_password

Overview:

Change the account password; if your session is authenticated with root priviledges then the old_pwd is validated and the new_pwd is set regardless

Signature:

```
void change_password (session ref session_id, string old_pwd, string new_pwd)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	old_pwd	Old password for account
`string`	new_pwd	New password for account

Return Type: `void`

RPC name: create_from_db_file

Overview:

Signature:

```
session ref create_from_db_file (session ref session_id, string filename)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	filename	Database dump filename.

Return Type: `session ref`

ID of newly created session

RPC name: get_all_subject_identifiers

Overview:

Return a list of all the user subject-identifiers of all existing sessions

Signature:

```
string set get_all_subject_identifiers (session ref session_id)
```

Minimum Role: read-only

Return Type: `string set`

The list of user subject-identifiers of all existing sessions

RPC name: get_auth_user_name

Overview:

Get the auth_user_name field of the given session.

Signature:

```
string get_auth_user_name (session ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_auth_user_sid

Overview:

Get the auth_user_sid field of the given session.

Signature:

```
string get_auth_user_sid (session ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

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 ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `session ref`

reference to the object

RPC name: get_is_local_superuser

Overview:

Get the is_local_superuser field of the given session.

Signature:

```
bool get_is_local_superuser (session ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_last_active

Overview:

Get the last_active field of the given session.

Signature:

```
datetime get_last_active (session ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_originator

Overview:

Get the originator field of the given session.

Signature:

```
string get_originator (session ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given session.

Signature:

```
(string -> string) map get_other_config (session ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_parent

Overview:

Get the parent field of the given session.

Signature:

```
session ref get_parent (session ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `session ref`

value of the field

RPC name: get_pool

Overview:

Get the pool field of the given session.

Signature:

```
bool get_pool (session ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_rbac_permissions

Overview:

Get the rbac_permissions field of the given session.

Signature:

```
string set get_rbac_permissions (session ref session_id, session ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given session.

Signature:

```
session record get_record (session ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `session record`

all fields from the object

RPC name: get_subject

Overview:

Get the subject field of the given session.

Signature:

```
subject ref get_subject (session ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `subject ref`

value of the field

RPC name: get_tasks

Overview:

Get the tasks field of the given session.

Signature:

```
task ref set get_tasks (session ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `task ref set`

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 ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `user ref`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given session.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_validation_time

Overview:

Get the validation_time field of the given session.

Signature:

```
datetime get_validation_time (session ref session_id, session ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: local_logout

Overview:

Log out of local session.

Signature:

```
void local_logout (session ref session_id)
```

Minimum Role: pool-admin

Return Type: `void`

RPC name: login_with_password

Overview:

Attempt to authenticate the user, returning a session reference if successful

Signature:

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

Arguments:

type	name	description
`string`	uname	Username for login.
`string`	pwd	Password for login.
`string`	version	Client API version.
`string`	originator	Key string for distinguishing different API users sharing the same login name.

Minimum Role: read-only

Return Type: `session ref`

reference of newly created session

Possible Error Codes: `SESSION_AUTHENTICATION_FAILED`, `HOST_IS_SLAVE`

RPC name: logout

Overview:

Log out of a session

Signature:

```
void logout (session ref session_id)
```

Minimum Role: read-only

Return Type: `void`

RPC name: logout_subject_identifier

Overview:

Log out all sessions associated to a user subject-identifier, except the session associated with the context calling this function

Signature:

```
void logout_subject_identifier (session ref session_id, string subject_identifier)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	subject_identifier	User subject-identifier of the sessions to be destroyed

Minimum Role: pool-operator

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 session. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, session ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-admin

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given session.

Signature:

```
void set_other_config (session ref session_id, session ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`session ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-admin

Return Type: `void`

RPC name: slave_local_login_with_password

Overview:

Authenticate locally against a slave in emergency mode. Note the resulting sessions are only good for use on this host.

Signature:

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

Arguments:

type	name	description
`string`	uname	Username for login.
`string`	pwd	Password for login.

Minimum Role: pool-admin

Return Type: `session ref`

ID of newly created session

Class: SM

A storage manager plugin

Fields for class: SM

Field	Type	Qualifier	Description
capabilities	`string set`	*RO/runtime*	**Deprecated**. capabilities of the SM plugin
configuration	`(string -> string) map`	*RO/runtime*	names and descriptions of device config keys
copyright	`string`	*RO/runtime*	Entity which owns the copyright of this plugin
driver_filename	`string`	*RO/runtime*	filename of the storage driver
features	`(string -> int) map`	*RO/runtime*	capabilities of the SM plugin, with capability version numbers
name_description	`string`	*RO/runtime*	a notes field containing human-readable description
name_label	`string`	*RO/runtime*	a human-readable name
other_config	`(string -> string) map`	*RW*	additional configuration

Field	Type	Qualifier	Description
required_api_version	`string`	*RO/runtime*	Minimum SM API version required on the server
required_cluster_stack	`string set`	*RO/runtime*	The storage plugin requires that one of these cluster stacks is configured and running.
type	`string`	*RO/runtime*	SR.type
uuid	`string`	*RO/runtime*	Unique identifier/object reference
vendor	`string`	*RO/runtime*	Vendor who created this plugin
version	`string`	*RO/runtime*	Version of the plugin

RPCs associated with class: SM

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given SM.

Signature:

```
void add_to_other_config (session ref session_id, SM ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the SMs known to the system.

Signature:

```
SM ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `SM ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of SM references to SM records for all SMs known to the system.

Signature:

```
(SM ref -> SM record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(SM ref -> SM record) map`

records of all objects

RPC name: get_by_name_label

Overview:

Get all the SM instances with the given label.

Signature:

```
SM ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `SM ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the SM instance with the specified UUID.

Signature:

```
SM ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `SM ref`

reference to the object

RPC name: get_capabilities

This message is deprecated.

Overview:

Get the capabilities field of the given SM.

Signature:

```
string set get_capabilities (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_configuration

Overview:

Get the configuration field of the given SM.

Signature:

```
(string -> string) map get_configuration (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_copyright

Overview:

Get the copyright field of the given SM.

Signature:

```
string get_copyright (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_driver_filename

Overview:

Get the driver_filename field of the given SM.

Signature:

```
string get_driver_filename (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_features

Overview:

Get the features field of the given SM.

Signature:

```
(string -> int) map get_features (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> int) map`

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given SM.

Signature:

```
string get_name_description (session ref session_id, SM ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

Overview:

Get the name/label field of the given SM.

Signature:

```
string get_name_label (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given SM.

Signature:

```
(string -> string) map get_other_config (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given SM.

Signature:

```
SM record get_record (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SM record`

all fields from the object

RPC name: get_required_api_version

Overview:

Get the required_api_version field of the given SM.

Signature:

```
string get_required_api_version (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_required_cluster_stack

Overview:

Get the required_cluster_stack field of the given SM.

Signature:

```
string set get_required_cluster_stack (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_type

Overview:

Get the type field of the given SM.

Signature:

```
string get_type (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given SM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_vendor

Overview:

Get the vendor field of the given SM.

Signature:

```
string get_vendor (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_version

Overview:

Get the version field of the given SM.

Signature:

```
string get_version (session ref session_id, SM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given SM. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, SM ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given SM.

Signature:

```
void set_other_config (session ref session_id, SM ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SM ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: SR

A storage repository

Fields for class: SR

Field	Type	Qualifier	Description
allowed_operations	`storage_operations set`	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
blobs	`(string -> blob ref) map`	*RO/runtime*	Binary blobs associated with this SR
clustered	`bool`	*RO/runtime*	True if the SR is using aggregated local storage
content_type	`string`	*RO/constructor*	the type of the SR's content, if required (e.g. ISOs)
current_operations	`(string -> storage_operations) map`	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.
introduced_by	`DR_task ref`	*RO/runtime*	The disaster recovery task which introduced this SR
is_tools_sr	`bool`	*RO/runtime*	True if this is the SR that contains the Tools ISO VDIs
local_cache_enabled	`bool`	*RO/runtime*	True if this SR is assigned to be the local cache for its host

Field	Type	Qualifier	Description
name_description	`string`	*RO/constructor*	a notes field containing human-readable description
name_label	`string`	*RO/constructor*	a human-readable name
other_config	`(string -> string) map`	*RW*	additional configuration
PBDs	`PBD ref set`	*RO/runtime*	describes how particular hosts can see this storage repository
physical_size	`int`	*RO/constructor*	total physical size of the repository (in bytes)
physical_utilisation	`int`	*RO/runtime*	physical space currently utilised on this storage repository (in bytes). Note that for sparse disk formats, physical_utilisation may be less than virtual_allocation
shared	`bool`	*RO/runtime*	true if this SR is (capable of being) shared between multiple hosts
sm_config	`(string -> string) map`	*RW*	SM dependent data
tags	`string set`	*RW*	user-specified tags for categorization purposes
type	`string`	*RO/constructor*	type of the storage repository
uuid	`string`	*RO/runtime*	Unique identifier/object reference
VDIs	`VDI ref set`	*RO/runtime*	all virtual disks known to this storage repository
virtual_allocation	`int`	*RO/runtime*	sum of virtual_sizes of all VDIs in this storage repository (in bytes)

RPCs associated with class: SR

RPC name: add_tags

Overview:

Add the given value to the tags field of the given SR. If the value is already in that Set, then do nothing.

Signature:

```
void add_tags (session ref session_id, SR ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object
`string`	value	New value to add

Minimum Role: vm-operator

Return Type: `void`

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given SR.

Signature:

```
void add_to_other_config (session ref session_id, SR ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: add_to_sm_config

Overview:

Add the given key-value pair to the sm_config field of the given SR.

Signature:

```
void add_to_sm_config (session ref session_id, SR ref self, string key, string value)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
SR ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: assert_can_host_ha_statefile

Overview:

Returns successfully if the given SR can host an HA statefile. Otherwise returns an error to explain why not

Signature:

```
void assert_can_host_ha_statefile (session ref session_id, SR ref sr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
SR ref	sr	The SR to query

Minimum Role: pool-operator

Return Type: `void`

RPC name: assert_supports_database_replication

Overview:

Returns successfully if the given SR supports database replication. Otherwise returns an error to explain why not.

Signature:

```
void assert_supports_database_replication (session ref session_id, SR ref sr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
SR ref	sr	The SR to query

Minimum Role: pool-operator

Return Type: void

RPC name: create

Overview:

Create a new Storage Repository and introduce it into the managed system, creating both SR record and PBD record to attach it to current host (with specified device_config parameters)

Signature:

```
SR ref create (session ref session_id, host ref host, (string -> string) map
device_config, int physical_size, string name_label, string name_description, string
type, string content_type, bool shared, (string -> string) map sm_config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
host ref	host	The host to create/make the SR on
(string -> string) map	device_config	The device config string that will be passed to backend SR driver
int	physical_size	The physical size of the new storage repository
string	name_label	The name of the new storage repository
string	name_description	The description of the new storage repository
string	type	The type of the SR; used to specify the SR backend driver to use
string	content_type	The type of the new SRs content, if required (e.g. ISOs)
bool	shared	True if the SR (is capable of) being shared by multiple hosts
(string -> string) map	sm_config	Storage backend specific configuration options

Minimum Role: pool-operator

Return Type: SR ref

The reference of the newly created Storage Repository.

Possible Error Codes: SR_UNKNOWN_DRIVER

RPC name: create_new_blob

Overview:

Create a placeholder for a named binary blob of data that is associated with this SR

Signature:

```
blob ref create_new_blob (session ref session_id, SR ref sr, string name, string
mime_type, bool public)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	sr	The SR
`string`	name	The name associated with the blob
`string`	mime_type	The mime type for the data. Empty string translates to application/octet-stream
`bool`	public	True if the blob should be publicly available

Minimum Role: pool-operator

Return Type: `blob ref`

The reference of the blob, needed for populating its data

RPC name: destroy

Overview:

Destroy specified SR, removing SR-record from database and remove SR from disk. (In order to affect this operation the appropriate device_config is read from the specified SR's PBD on current host)

Signature:

```
void destroy (session ref session_id, SR ref sr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	sr	The SR to destroy

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `SR_HAS_PBD`

RPC name: disable_database_replication

Overview:

Signature:

```
void disable_database_replication (session ref session_id, SR ref sr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	sr	The SR to which metadata should be no longer replicated

Minimum Role: pool-operator

Return Type: `void`

RPC name: enable_database_replication

Overview:

Signature:

```
void enable_database_replication (session ref session_id, SR ref sr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	sr	The SR to which metadata should be replicated

Minimum Role: pool-operator

Return Type: `void`

RPC name: forget

Overview:

Removing specified SR-record from database, without attempting to remove SR from disk

Signature:

```
void forget (session ref session_id, SR ref sr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	sr	The SR to destroy

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `SR_HAS_PBD`

RPC name: forget_data_source_archives

Overview:

Forget the recorded statistics related to the specified data source

Signature:

```
void forget_data_source_archives (session ref session_id, SR ref sr, string
data_source)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	sr	The SR
`string`	data_source	The data source whose archives are to be forgotten

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the SRs known to the system.

Signature:

```
SR ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `SR ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of SR references to SR records for all SRs known to the system.

Signature:

```
(SR ref -> SR record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(SR ref -> SR record) map`

records of all objects

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given SR.

Signature:

```
storage_operations set get_allowed_operations (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `storage_operations set`

value of the field

RPC name: get_blobs

Overview:

Get the blobs field of the given SR.

Signature:

```
(string -> blob ref) map get_blobs (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> blob ref) map`

value of the field

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 ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `SR ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the SR instance with the specified UUID.

Signature:

```
SR ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `SR ref`

reference to the object

RPC name: get_clustered

Overview:

Get the clustered field of the given SR.

Signature:

```
bool get_clustered (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

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 ref session_id, SR ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_current_operations

Overview:

Get the current_operations field of the given SR.

Signature:

```
(string -> storage_operations) map get_current_operations (session ref session_id, SR
ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> storage_operations) map`

value of the field

RPC name: get_data_sources

Overview:

Signature:

```
data_source record set get_data_sources (session ref session_id, SR ref sr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	sr	The SR to interrogate

Minimum Role: read-only

Return Type: `data_source record set`

A set of data sources

RPC name: get_introduced_by

Overview:

Get the introduced_by field of the given SR.

Signature:

```
DR_task ref get_introduced_by (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `DR_task ref`

value of the field

RPC name: get_is_tools_sr

Overview:

Get the is_tools_sr field of the given SR.

Signature:

```
bool get_is_tools_sr (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_local_cache_enabled

Overview:

Get the local_cache_enabled field of the given SR.

Signature:

```
bool get_local_cache_enabled (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given SR.

Signature:

```
string get_name_description (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given SR.

Signature:

```
(string -> string) map get_other_config (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_PBDs

Overview:

Get the PBDs field of the given SR.

Signature:

```
PBD ref set get_PBDs (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PBD ref set`

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 ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given SR.

Signature:

```
SR record get_record (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SR record`

all fields from the object

RPC name: get_shared

Overview:

Get the shared field of the given SR.

Signature:

```
bool get_shared (session ref session_id, SR ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_sm_config

Overview:

Get the sm_config field of the given SR.

Signature:

```
(string -> string) map get_sm_config (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

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 ref session_id)
```

Minimum Role: read-only

Return Type: `string set`

the supported SR types

RPC name: get_tags

Overview:

Get the tags field of the given SR.

Signature:

```
string set get_tags (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_type

Overview:

Get the type field of the given SR.

Signature:

```
string get_type (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given SR.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VDIs

Overview:

Get the VDIs field of the given SR.

Signature:

```
VDI ref set get_VDIs (session ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VDI 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 ref session_id, SR ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: introduce

Overview:

Introduce a new Storage Repository into the managed system

Signature:

```
SR ref introduce (session ref session_id, string uuid, string name_label, string
name_description, string type, string content_type, bool shared, (string -> string)
map sm_config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	The uuid assigned to the introduced SR
`string`	name_label	The name of the new storage repository
`string`	name_description	The description of the new storage repository
`string`	type	The type of the SR; used to specify the SR backend driver to use
`string`	content_type	The type of the new SRs content, if required (e.g. ISOs)
`bool`	shared	True if the SR (is capable of) being shared by multiple hosts
`(string -> string) map`	sm_config	Storage backend specific configuration options

Minimum Role: pool-operator

Return Type: `SR ref`

The reference of the newly introduced Storage Repository.

RPC name: make

This message is deprecated.

Overview:

Create a new Storage Repository on disk. This call is deprecated: use SR.create instead.

Signature:

```
string make (session ref session_id, host ref host, (string -> string) map
device_config, int physical_size, string name_label, string name_description, string
type, string content_type, (string -> string) map sm_config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host to create/make the SR on
`(string -> string) map`	device_config	The device config string that will be passed to backend SR driver
`int`	physical_size	The physical size of the new storage repository
`string`	name_label	The name of the new storage repository
`string`	name_description	The description of the new storage repository
`string`	type	The type of the SR; used to specify the SR backend driver to use
`string`	content_type	The type of the new SRs content, if required (e.g. ISOs)
`(string -> string) map`	sm_config	Storage backend specific configuration options

Minimum Role: pool-operator

Return Type: `string`

The uuid of the newly created Storage Repository.

RPC name: probe

Overview:

Perform a backend-specific scan, using the given device_config. If the device_config is complete, then this will return a list of the SRs present of this type on the device, if any. If the device_config is partial, then a backend-specific scan will be performed, returning results that will guide the user in improving the device_config.

Signature:

```
string probe (session ref session_id, host ref host, (string -> string) map
device_config, string type, (string -> string) map sm_config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host to create/make the SR on
`(string -> string) map`	device_config	The device config string that will be passed to backend SR driver
`string`	type	The type of the SR; used to specify the SR backend driver to use
`(string -> string) map`	sm_config	Storage backend specific configuration options

Minimum Role: pool-operator

Return Type: `string`

An XML fragment containing the scan results. These are specific to the scan being performed, and the backend.

RPC name: probe_ext

Overview:

Perform a backend-specific scan, using the given device_config. If the device_config is complete, then this will return a list of the SRs present of this type on the device, if any. If the device_config is partial, then a backend-specific scan will be performed, returning results that will guide the user in improving the device_config.

Signature:

```
probe_result record set probe_ext (session ref session_id, host ref host, (string ->
string) map device_config, string type, (string -> string) map sm_config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`host ref`	host	The host to create/make the SR on
`(string -> string) map`	device_config	The device config string that will be passed to backend SR driver

type	name	description
`string`	type	The type of the SR; used to specify the SR backend driver to use
`(string -> string) map`	sm_config	Storage backend specific configuration options

Minimum Role: pool-operator

Return Type: `probe_result record set`

A set of records containing the scan results.

RPC name: query_data_source

Overview:

Query the latest value of the specified data source

Signature:

```
float query_data_source (session ref session_id, SR ref sr, string data_source)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	sr	The SR
`string`	data_source	The data source to query

Minimum Role: read-only

Return Type: `float`

The latest value, averaged over the last 5 seconds

RPC name: record_data_source

Overview:

Start recording the specified data source

Signature:

```
void record_data_source (session ref session_id, SR ref sr, string data_source)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	sr	The SR
`string`	data_source	The data source to record

Minimum Role: pool-operator

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 SR. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, SR ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: remove_from_sm_config

Overview:

Remove the given key and its corresponding value from the sm_config field of the given SR. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_sm_config (session ref session_id, SR ref self, string key)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: remove_tags

Overview:

Remove the given value from the tags field of the given SR. If the value is not in that Set, then do nothing.

Signature:

```
void remove_tags (session ref session_id, SR ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object
`string`	value	Value to remove

Minimum Role: vm-operator

Return Type: `void`

RPC name: scan

Overview:

Refreshes the list of VDIs associated with an SR

Signature:

```
void scan (session ref session_id, SR ref sr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
SR ref	sr	The SR to scan

Minimum Role: vm-power-admin

Return Type: void

RPC name: set_name_description

Overview:

Set the name description of the SR

Signature:

```
void set_name_description (session ref session_id, SR ref sr, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
SR ref	sr	The SR
string	value	The name description for the SR

Minimum Role: pool-operator

Return Type: void

RPC name: set_name_label

Overview:

Set the name label of the SR

Signature:

```
void set_name_label (session ref session_id, SR ref sr, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
SR ref	sr	The SR
string	value	The name label for the SR

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given SR.

Signature:

```
void set_other_config (session ref session_id, SR ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_physical_size

Overview:

Sets the SR's physical_size field

Signature:

```
void set_physical_size (session ref session_id, SR ref self, int value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	The SR to modify
`int`	value	The new value of the SR's physical_size

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_shared

Overview:

Sets the shared flag on the SR

Signature:

```
void set_shared (session ref session_id, SR ref sr, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	sr	The SR
`bool`	value	True if the SR is shared

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_sm_config

Overview:

Set the sm_config field of the given SR.

Signature:

```
void set_sm_config (session ref session_id, SR ref self, (string -> string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_tags

Overview:

Set the tags field of the given SR.

Signature:

```
void set_tags (session ref session_id, SR ref self, string set value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	self	reference to the object
`string set`	value	New value to set

Minimum Role: vm-operator

Return Type: `void`

RPC name: update

Overview:

Refresh the fields on the SR object

Signature:

```
void update (session ref session_id, SR ref sr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`SR ref`	sr	The SR whose fields should be refreshed

Minimum Role: pool-operator

Return Type: `void`

Class: sr_stat

A set of high-level properties associated with an SR.

Fields for class: sr_stat

Field	Type	Qualifier	Description
clustered	`bool`	*RO/runtime*	Indicates whether the SR uses clustered local storage.

Field	Type	Qualifier	Description
free_space	`int`	*RO/runtime*	Number of bytes free on the backing storage (in bytes)
health	`sr_health`	*RO/runtime*	The health status of the SR.
name_description	`string`	*RO/runtime*	Longer, human-readable description of the SR. Descriptions are generally only displayed by clients when the user is examining SRs in detail.
name_label	`string`	*RO/runtime*	Short, human-readable label for the SR.
total_space	`int`	*RO/runtime*	Total physical size of the backing storage (in bytes)
uuid	`string option`	*RO/runtime*	Uuid that uniquely identifies this SR, if one is available.

RPCs associated with class: sr_stat

Class sr_stat has no additional RPCs associated with it.

Class: subject

A user or group that can log in xapi

Fields for class: subject

Field	Type	Qualifier	Description
other_config	`(string -> string) map`	*RO/constructor*	additional configuration
roles	`role ref set`	*RO/runtime*	the roles associated with this subject
subject_identifier	`string`	*RO/constructor*	the subject identifier, unique in the external directory service
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: subject

RPC name: add_to_roles

Overview:

This call adds a new role to a subject

Signature:

```
void add_to_roles (session ref session_id, subject ref self, role ref role)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`subject ref`	self	The subject who we want to add the role to
`role ref`	role	The unique role reference

Minimum Role: pool-admin

Return Type: `void`

RPC name: create

Overview:

Create a new subject instance, and return its handle.

Signature:

```
subject ref create (session ref session_id, subject record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`subject record`	args	All constructor arguments

Minimum Role: pool-admin

Return Type: `subject ref`

reference to the newly created object

RPC name: destroy

Overview:

Destroy the specified subject instance.

Signature:

```
void destroy (session ref session_id, subject ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`subject ref`	self	reference to the object

Minimum Role: pool-admin

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the subjects known to the system.

Signature:

```
subject ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `subject ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of subject references to subject records for all subjects known to the system.

Signature:

```
(subject ref -> subject record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(subject ref -> subject record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the subject instance with the specified UUID.

Signature:

```
subject ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `subject ref`

reference to the object

RPC name: get_other_config

Overview:

Get the other_config field of the given subject.

Signature:

```
(string -> string) map get_other_config (session ref session_id, subject ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`subject ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_permissions_name_label

Overview:

This call returns a list of permission names given a subject

Signature:

```
string set get_permissions_name_label (session ref session_id, subject ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`subject ref`	self	The subject whose permissions will be retrieved

Minimum Role: read-only

Return Type: `string set`

a list of permission names

RPC name: get_record

Overview:

Get a record containing the current state of the given subject.

Signature:

```
subject record get_record (session ref session_id, subject ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`subject ref`	self	reference to the object

Minimum Role: read-only

Return Type: `subject record`

all fields from the object

RPC name: get_roles

Overview:

Get the roles field of the given subject.

Signature:

```
role ref set get_roles (session ref session_id, subject ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`subject ref`	self	reference to the object

Minimum Role: read-only

Return Type: `role ref set`

value of the field

RPC name: get_subject_identifier

Overview:

Get the subject_identifier field of the given subject.

Signature:

```
string get_subject_identifier (session ref session_id, subject ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`subject ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given subject.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`subject ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_roles

Overview:

This call removes a role from a subject

Signature:

```
void remove_from_roles (session ref session_id, subject ref self, role ref role)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`subject ref`	self	The subject from whom we want to remove the role
`role ref`	role	The unique role reference in the subject's roles field

Minimum Role: pool-admin

Return Type: `void`

Class: task

A long-running asynchronous task

Fields for class: task

Field	Type	Qualifier	Description
allowed_operations	`task_allowed_operations set`	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
backtrace	`string`	*RO/runtime*	Function call trace for debugging.
created	`datetime`	*RO/runtime*	Time task was created
current_operations	`(string -> task_allowed_operations) map`	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.

Field	Type	Qualifier	Description
error_info	`string set`	*RO/runtime*	if the task has failed, this field contains the set of associated error strings. Undefined otherwise.
finished	`datetime`	*RO/runtime*	Time task finished (i.e. succeeded or failed). If task-status is pending, then the value of this field has no meaning
name_description	`string`	*RO/runtime*	a notes field containing human-readable description
name_label	`string`	*RO/runtime*	a human-readable name
other_config	`(string -> string) map`	*RW*	additional configuration
progress	`float`	*RO/runtime*	This field contains the estimated fraction of the task which is complete. This field should not be used to determine whether the task is complete - for this the status field of the task should be used.
resident_on	`host ref`	*RO/runtime*	the host on which the task is running
result	`string`	*RO/runtime*	if the task has completed successfully, this field contains the result value (either Void or an object reference). Undefined otherwise.
status	`task_status_type`	*RO/runtime*	current status of the task
subtask_of	`task ref`	*RO/runtime*	Ref pointing to the task this is a substask of.
subtasks	`task ref set`	*RO/runtime*	List pointing to all the substasks.
type	`string`	*RO/runtime*	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.
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: task

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given task.

Signature:

```
void add_to_other_config (session ref session_id, task ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: cancel

Overview:

Request that a task be cancelled. Note that a task may fail to be cancelled and may complete or fail normally and note that, even when a task does cancel, it might take an arbitrary amount of time.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	task	The task

Minimum Role: read-only

Return Type: `void`

Possible Error Codes: `OPERATION_NOT_ALLOWED`

RPC name: create

Overview:

Create a new task object which must be manually destroyed.

Signature:

```
```

```
task ref create (session ref session_id, string label, string description)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	short label for the new task
`string`	description	longer description for the new task

Minimum Role: read-only

Return Type: `task ref`

The reference of the created task object

RPC name: destroy

Overview:

Destroy the task object

Signature:

```
void destroy (session ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	Reference to the task object

Minimum Role: read-only

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the tasks known to the system.

Signature:

```
task ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `task ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of task references to task records for all tasks known to the system.

Signature:

```
(task ref -> task record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(task ref -> task record) map`

records of all objects

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given task.

Signature:

```
task_allowed_operations set get_allowed_operations (session ref session_id, task ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `task_allowed_operations set`

value of the field

RPC name: get_backtrace

Overview:

Get the backtrace field of the given task.

Signature:

```
string get_backtrace (session ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

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 ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `task ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the task instance with the specified UUID.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `task ref`

reference to the object

RPC name: get_created

Overview:

Get the created field of the given task.

Signature:

```
datetime get_created (session ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_current_operations

Overview:

Get the current_operations field of the given task.

Signature:

```
(string -> task_allowed_operations) map get_current_operations (session ref
session_id, task ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> task_allowed_operations) map`

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 ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_finished

Overview:

Get the finished field of the given task.

Signature:

```
datetime get_finished (session ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

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 ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given task.

Signature:

```
(string -> string) map get_other_config (session ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_progress

Overview:

Get the progress field of the given task.

Signature:

```
float get_progress (session ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `float`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given task.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `task record`

all fields from the object

RPC name: get_resident_on

Overview:

Get the resident_on field of the given task.

Signature:

```
host ref get_resident_on (session ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_result

Overview:

Get the result field of the given task.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `task_status_type`

value of the field

RPC name: get_subtask_of

Overview:

Get the subtask_of field of the given task.

Signature:

```
task ref get_subtask_of (session ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `task ref`

value of the field

RPC name: get_subtasks

Overview:

Get the subtasks field of the given task.

Signature:

```
task ref set get_subtasks (session ref session_id, task ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `task ref set`

value of the field

RPC name: get_type

Overview:

Get the type field of the given task.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given task.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given task. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, task ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given task.

Signature:

```
void set_other_config (session ref session_id, task ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_progress

Overview:

Set the task progress

Signature:

```
void set_progress (session ref session_id, task ref self, float value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`task ref`	self	Reference to the task object
`float`	value	Task progress value to be set

Minimum Role: read-only

Return Type: `void`

RPC name: set_status

Overview:

Set the task status

Signature:

```
void set_status (session ref session_id, task ref self, task_status_type value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`task ref`	self	Reference to the task object
`task_status_type`	value	task status value to be set

Minimum Role: read-only

Return Type: `void`

Class: tunnel

A tunnel for network traffic

Fields for class: tunnel

Field	Type	Qualifier	Description
access_PIF	`PIF ref`	*RO/constructor*	The interface through which the tunnel is accessed
other_config	`(string -> string) map`	*RW*	Additional configuration
status	`(string -> string) map`	*RW*	Status information about the tunnel
transport_PIF	`PIF ref`	*RO/constructor*	The interface used by the tunnel
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: tunnel

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given tunnel.

Signature:

```
void add_to_other_config (session ref session_id, tunnel ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: add_to_status

Overview:

Add the given key-value pair to the status field of the given tunnel.

Signature:

```
void add_to_status (session ref session_id, tunnel ref self, string key, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: create

Overview:

Create a tunnel

Signature:

```
tunnel ref create (session ref session_id, PIF ref transport_PIF, network ref network)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`PIF ref`	transport_PIF	PIF which receives the tagged traffic
`network ref`	network	Network to receive the tunnelled traffic

Minimum Role: pool-operator

Return Type: `tunnel ref`

The reference of the created tunnel object

Possible Error Codes: `OPENVSWITCH_NOT_ACTIVE`, `TRANSPORT_PIF_NOT_CONFIGURED`, `IS_TUNNEL_ACCESS_PIF`

RPC name: destroy

Overview:

Destroy a tunnel

Signature:

```
void destroy (session ref session_id, tunnel ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	tunnel to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_access_PIF

Overview:

Get the access_PIF field of the given tunnel.

Signature:

```
PIF ref get_access_PIF (session ref session_id, tunnel ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF ref`

value of the field

RPC name: get_all

Overview:

Return a list of all the tunnels known to the system.

Signature:

```
tunnel ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `tunnel ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of tunnel references to tunnel records for all tunnels known to the system.

Signature:

```
(tunnel ref -> tunnel record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(tunnel ref -> tunnel record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the tunnel instance with the specified UUID.

Signature:

```
tunnel ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `tunnel ref`

reference to the object

RPC name: get_other_config

Overview:

Get the other_config field of the given tunnel.

Signature:

```
(string -> string) map get_other_config (session ref session_id, tunnel ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given tunnel.

Signature:

```
tunnel record get_record (session ref session_id, tunnel ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	reference to the object

Minimum Role: read-only

Return Type: `tunnel record`

all fields from the object

RPC name: get_status

Overview:

Get the status field of the given tunnel.

Signature:

```
(string -> string) map get_status (session ref session_id, tunnel ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_transport_PIF

Overview:

Get the transport_PIF field of the given tunnel.

Signature:

```
PIF ref get_transport_PIF (session ref session_id, tunnel ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF ref`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given tunnel.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given tunnel. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, tunnel ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: remove_from_status

Overview:

Remove the given key and its corresponding value from the status field of the given tunnel. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_status (session ref session_id, tunnel ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given tunnel.

Signature:

```

```

```
void set_other_config (session ref session_id, tunnel ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_status

Overview:

Set the status field of the given tunnel.

Signature:

```
void set_status (session ref session_id, tunnel ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`tunnel ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: USB_group

A group of compatible USBs across the resource pool

Fields for class: USB_group

Field	Type	Qualifier	Description
name_description	`string`	*RW*	a notes field containing human-readable description

Field	Type	Qualifier	Description
name_label	`string`	*RW*	a human-readable name
other_config	`(string -> string) map`	*RW*	Additional configuration
PUSBs	`PUSB ref set`	*RO/runtime*	List of PUSBs in the group
uuid	`string`	*RO/runtime*	Unique identifier/object reference
VUSBs	`VUSB ref set`	*RO/runtime*	List of VUSBs using the group

RPCs associated with class: USB_group

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given USB_group.

Signature:

```
void add_to_other_config (session ref session_id, USB_group ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`USB_group ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-admin

Return Type: `void`

RPC name: create

Overview:

Signature:

```
USB_group ref create (session ref session_id, string name_label, string
name_description, (string -> string) map other_config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	name_label	
`string`	name_description	
`(string -> string) map`	other_config	

Minimum Role: pool-admin

Return Type: `USB_group ref`

RPC name: destroy

Overview:

Signature:

```
void destroy (session ref session_id, USB_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`USB_group ref`	self	The USB group to destroy

Minimum Role: pool-admin

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the USB_groups known to the system.

Signature:

```
USB_group ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `USB_group ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of USB_group references to USB_group records for all USB_groups known to the system.

Signature:

```
(USB_group ref -> USB_group record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(USB_group ref -> USB_group record) map`

records of all objects

RPC name: get_by_name_label

Overview:

Get all the USB_group instances with the given label.

Signature:

```
USB_group ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `USB_group ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the USB_group instance with the specified UUID.

Signature:

```
USB_group ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `USB_group ref`

reference to the object

RPC name: get_name_description

Overview:

Get the name/description field of the given USB_group.

Signature:

```
string get_name_description (session ref session_id, USB_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`USB_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

Overview:

Get the name/label field of the given USB_group.

Signature:

```
string get_name_label (session ref session_id, USB_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`USB_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given USB_group.

Signature:

```
(string -> string) map get_other_config (session ref session_id, USB_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`USB_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_PUSBs

Overview:

Get the PUSBs field of the given USB_group.

Signature:

```
PUSB ref set get_PUSBs (session ref session_id, USB_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`USB_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PUSB ref set`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given USB_group.

Signature:

```
USB_group record get_record (session ref session_id, USB_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`USB_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `USB_group record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given USB_group.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`USB_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VUSBs

Overview:

Get the VUSBs field of the given USB_group.

Signature:

```
VUSB ref set get_VUSBs (session ref session_id, USB_group ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`USB_group ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VUSB ref set`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given USB_group. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, USB_group ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`USB_group ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-admin

Return Type: `void`

RPC name: set_name_description

Overview:

Set the name/description field of the given USB_group.

Signature:

```
void set_name_description (session ref session_id, USB_group ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`USB_group ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-admin

Return Type: `void`

RPC name: set_name_label

Overview:

Set the name/label field of the given USB_group.

Signature:

```
void set_name_label (session ref session_id, USB_group ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`USB_group ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-admin

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given USB_group.

Signature:

```
void set_other_config (session ref session_id, USB_group ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`USB_group ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-admin

Return Type: `void`

Class: user

This class is deprecated.

A user of the system

Fields for class: user

Field	Type	Qualifier	Description
fullname	`string`	*RW*	**Deprecated**. full name
other_config	`(string -> string) map`	*RW*	**Deprecated**. additional configuration
short_name	`string`	*RO/constructor*	**Deprecated**. short name (e.g. userid)
uuid	`string`	*RO/runtime*	**Deprecated**. Unique identifier/object reference

RPCs associated with class: user

RPC name: add_to_other_config

This message is deprecated.

Overview:

Add the given key-value pair to the other_config field of the given user.

Signature:

```
void add_to_other_config (session ref session_id, user ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`user ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: pool-admin

Return Type: `void`

RPC name: create

This message is deprecated.

Overview:

Create a new user instance, and return its handle.

Signature:

```
user ref create (session ref session_id, user record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`user record`	args	All constructor arguments

Minimum Role: pool-admin

Return Type: `user ref`

reference to the newly created object

RPC name: destroy

This message is deprecated.

Overview:

Destroy the specified user instance.

Signature:

```
void destroy (session ref session_id, user ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`user ref`	self	reference to the object

Minimum Role: pool-admin

Return Type: `void`

RPC name: get_by_uuid

This message is deprecated.

Overview:

Get a reference to the user instance with the specified UUID.

Signature:

```
user ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `user ref`

reference to the object

RPC name: get_fullname

This message is deprecated.

Overview:

Get the fullname field of the given user.

Signature:

```
string get_fullname (session ref session_id, user ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`user ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_other_config

This message is deprecated.

Overview:

Get the other_config field of the given user.

Signature:

```
(string -> string) map get_other_config (session ref session_id, user ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`user ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

This message is deprecated.

Overview:

Get a record containing the current state of the given user.

Signature:

```
user record get_record (session ref session_id, user ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`user ref`	self	reference to the object

Minimum Role: read-only

Return Type: `user record`

all fields from the object

RPC name: get_short_name

This message is deprecated.

Overview:

Get the short_name field of the given user.

Signature:

```
string get_short_name (session ref session_id, user ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`user ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_uuid

This message is deprecated.

Overview:

Get the uuid field of the given user.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`user ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

This message is deprecated.

Overview:

Remove the given key and its corresponding value from the other_config field of the given user. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, user ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`user ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-admin

Return Type: `void`

RPC name: set_fullname

This message is deprecated.

Overview:

Set the fullname field of the given user.

Signature:

```
void set_fullname (session ref session_id, user ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`user ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-admin

Return Type: `void`

RPC name: set_other_config

This message is deprecated.

Overview:

Set the other_config field of the given user.

Signature:

```
void set_other_config (session ref session_id, user ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`user ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-admin

Return Type: `void`

Class: VBD

A virtual block device

Fields for class: VBD

Field	Type	Qualifier	Description
allowed_operations	`vbd_operations set`	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
bootable	`bool`	*RW*	true if this VBD is bootable

Field	Type	Qualifier	Description
current_operations	(string -> vbd_operations) map	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.
currently_attached	bool	*RO/runtime*	is the device currently attached (erased on reboot)
device	string	*RO/runtime*	device seen by the guest e.g. hda1
empty	bool	*RO/constructor*	if true this represents an empty drive
metrics	VBD_metrics ref	*RO/runtime*	**Removed**. metrics associated with this VBD
mode	vbd_mode	*RO/constructor*	the mode the VBD should be mounted with
other_config	(string -> string) map	*RW*	additional configuration
qos_algorithm_params	(string -> string) map	*RW*	parameters for chosen QoS algorithm
qos_algorithm_type	string	*RW*	QoS algorithm to use
qos_supported_algorithms	string set	*RO/runtime*	supported QoS algorithms for this VBD
runtime_properties	(string -> string) map	*RO/runtime*	Device runtime properties
status_code	int	*RO/runtime*	error/success code associated with last attach-operation (erased on reboot)
status_detail	string	*RO/runtime*	error/success information associated with last attach-operation status (erased on reboot)
storage_lock	bool	*RO/runtime*	true if a storage level lock was acquired
type	vbd_type	*RW*	how the VBD will appear to the guest (e.g. disk or CD)
unpluggable	bool	*RW*	true if this VBD will support hot-unplug
userdevice	string	*RW*	user-friendly device name e.g. 0,1,2,etc.
uuid	string	*RO/runtime*	Unique identifier/object reference
VDI	VDI ref	*RO/constructor*	the virtual disk
VM	VM ref	*RO/constructor*	the virtual machine

RPCs associated with class: VBD

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given VBD.

Signature:

```
void add_to_other_config (session ref session_id, VBD ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: vm-admin

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 ref session_id, VBD ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: vm-admin

Return Type: `void`

RPC name: assert_attachable

Overview:

Throws an error if this VBD could not be attached to this VM if the VM were running. Intended for debugging.

Signature:

```
void assert_attachable (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	The VBD to query

Minimum Role: vm-admin

Return Type: `void`

RPC name: create

Overview:

Create a new VBD instance, and return its handle.

Signature:

```
VBD ref create (session ref session_id, VBD record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD record`	args	All constructor arguments

Minimum Role: vm-admin

Return Type: `VBD ref`

reference to the newly created object

RPC name: destroy

Overview:

Destroy the specified VBD instance.

Signature:

```
void destroy (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	reference to the object

Minimum Role: vm-admin

Return Type: `void`

RPC name: eject

Overview:

Remove the media from the device and leave it empty

Signature:

```
void eject (session ref session_id, VBD ref vbd)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	vbd	The vbd representing the CDROM-like device

Minimum Role: vm-operator

Return Type: `void`

Possible Error Codes: `VBD_NOT_REMOVABLE_MEDIA`, `VBD_IS_EMPTY`

RPC name: get_all

Overview:

Return a list of all the VBDs known to the system.

Signature:

```
VBD ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `VBD ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of VBD references to VBD records for all VBDs known to the system.

Signature:

```
(VBD ref -> VBD record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VBD ref -> VBD record) map`

records of all objects

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given VBD.

Signature:

```
vbd_operations set get_allowed_operations (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vbd_operations set`

value of the field

RPC name: get_bootable

Overview:

Get the bootable field of the given VBD.

Signature:

```
bool get_bootable (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_by_uuid

Overview:

Get a reference to the VBD instance with the specified UUID.

Signature:

```
VBD ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VBD ref`

reference to the object

RPC name: get_current_operations

Overview:

Get the current_operations field of the given VBD.

Signature:

```

```

```
(string -> vbd_operations) map get_current_operations (session ref session_id, VBD ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> vbd_operations) map`

value of the field

RPC name: get_currently_attached

Overview:

Get the currently_attached field of the given VBD.

Signature:

```
bool get_currently_attached (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_device

Overview:

Get the device field of the given VBD.

Signature:

```
string get_device (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_empty

Overview:

Get the empty field of the given VBD.

Signature:

```
bool get_empty (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_metrics

This message is removed.

Overview:

Get the metrics field of the given VBD.

Signature:

```
VBD_metrics ref get_metrics (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	reference to the object

Minimum Role: read-only

Return Type: VBD_metrics ref

value of the field

RPC name: get_mode

Overview:

Get the mode field of the given VBD.

Signature:

```
vbd_mode get_mode (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	reference to the object

Minimum Role: read-only

Return Type: vbd_mode

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given VBD.

Signature:

```
(string -> string) map get_other_config (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VBD ref	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

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 ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	reference to the object

Minimum Role: read-only

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 ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

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 ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given VBD.

Signature:

```
VBD record get_record (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VBD record`

all fields from the object

RPC name: get_runtime_properties

Overview:

Get the runtime_properties field of the given VBD.

Signature:

```
(string -> string) map get_runtime_properties (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

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 ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_storage_lock

Overview:

Get the storage_lock field of the given VBD.

Signature:

```
bool get_storage_lock (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_type

Overview:

Get the type field of the given VBD.

Signature:

```
vbd_type get_type (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	reference to the object

Minimum Role: read-only

Return Type: vbd_type

value of the field

RPC name: get_unpluggable

Overview:

Get the unpluggable field of the given VBD.

Signature:

```
bool get_unpluggable (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	reference to the object

Minimum Role: read-only

Return Type: bool

value of the field

RPC name: get_userdevice

Overview:

Get the userdevice field of the given VBD.

Signature:

```
string get_userdevice (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given VBD.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VDI

Overview:

Get the VDI field of the given VBD.

Signature:

```
VDI ref get_VDI (session ref session_id, VBD ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	reference to the object

Minimum Role: read-only

Return Type: VDI ref

value of the field

RPC name: get_VM

Overview:

Get the VM field of the given VBD.

Signature:

```
VM ref get_VM (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	reference to the object

Minimum Role: read-only

Return Type: VM ref

value of the field

RPC name: insert

Overview:

Insert new media into the device

Signature:

```
void insert (session ref session_id, VBD ref vbd, VDI ref vdi)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VBD ref	vbd	The vbd representing the CDROM-like device
VDI ref	vdi	The new VDI to 'insert'

Minimum Role: vm-operator

Return Type: void

Possible Error Codes: VBD_NOT_REMOVABLE_MEDIA, VBD_NOT_EMPTY

RPC name: plug

Overview:

Hotplug the specified VBD, dynamically attaching it to the running VM

Signature:

```
void plug (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	The VBD to hotplug

Minimum Role: vm-admin

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 VBD. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, VBD ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	reference to the object

type	name	description
`string`	key	Key to remove

Minimum Role: vm-admin

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 ref session_id, VBD ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_bootable

Overview:

Set the bootable field of the given VBD.

Signature:

```
void set_bootable (session ref session_id, VBD ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object

type	name	description
bool	value	New value to set

Minimum Role: vm-admin

Return Type: void

RPC name: set_mode

Overview:

Sets the mode of the VBD. The power_state of the VM must be halted.

Signature:

```
void set_mode (session ref session_id, VBD ref self, vbd_mode value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	Reference to the object
vbd_mode	value	New value to set

Minimum Role: vm-admin

Return Type: void

RPC name: set_other_config

Overview:

Set the other_config field of the given VBD.

Signature:

```
void set_other_config (session ref session_id, VBD ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	reference to the object
(string -> string) map	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_qos_algorithm_params

Overview:

Set the qos/algorithm_params field of the given VBD.

Signature:

```
void set_qos_algorithm_params (session ref session_id, VBD ref self, (string ->
string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_qos_algorithm_type

Overview:

Set the qos/algorithm_type field of the given VBD.

Signature:

```
void set_qos_algorithm_type (session ref session_id, VBD ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_type

Overview:

Set the type field of the given VBD.

Signature:

```
void set_type (session ref session_id, VBD ref self, vbd_type value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object
`vbd_type`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_unpluggable

Overview:

Set the unpluggable field of the given VBD.

Signature:

```
void set_unpluggable (session ref session_id, VBD ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object
`bool`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_userdevice

Overview:

Set the userdevice field of the given VBD.

Signature:

```
void set_userdevice (session ref session_id, VBD ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: unplug

Overview:

Hot-unplug the specified VBD, dynamically unattaching it from the running VM

Signature:

```
void unplug (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD ref`	self	The VBD to hot-unplug

Minimum Role: vm-admin

Return Type: `void`

Possible Error Codes: `DEVICE_DETACH_REJECTED`, `DEVICE_ALREADY_DETACHED`

RPC name: unplug_force

Overview:

Forcibly unplug the specified VBD

Signature:

```
```

```
void unplug_force (session ref session_id, VBD ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD ref	self	The VBD to forcibly unplug

Minimum Role: vm-admin

Return Type: void

Class: VBD_metrics

This class is removed.

The metrics associated with a virtual block device

Fields for class: VBD_metrics

Field	Type	Qualifier	Description
io_read_kbs	float	*RO/runtime*	**Removed**. Read bandwidth (KiB/s)
io_write_kbs	float	*RO/runtime*	**Removed**. Write bandwidth (KiB/s)
last_updated	datetime	*RO/runtime*	**Removed**. Time at which this information was last updated
other_config	(string -> string) map	*RW*	**Removed**. additional configuration
uuid	string	*RO/runtime*	**Removed**. Unique identifier/object reference

RPCs associated with class: VBD_metrics

RPC name: add_to_other_config

This message is removed.

Overview:

Add the given key-value pair to the other_config field of the given VBD_metrics.

Signature:

```
void add_to_other_config (session ref session_id, VBD_metrics ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD_metrics ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: vm-admin

Return Type: `void`

RPC name: get_all

This message is removed.

Overview:

Return a list of all the VBD_metrics instances known to the system.

Signature:

```
VBD_metrics ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `VBD_metrics ref set`

references to all objects

RPC name: get_all_records

This message is removed.

Overview:

Return a map of VBD_metrics references to VBD_metrics records for all VBD_metrics instances known to the system.

Signature:

```
(VBD_metrics ref -> VBD_metrics record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VBD_metrics ref -> VBD_metrics record) map`

records of all objects

RPC name: get_by_uuid

This message is removed.

Overview:

Get a reference to the VBD_metrics instance with the specified UUID.

Signature:

```
VBD_metrics ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VBD_metrics ref`

reference to the object

RPC name: get_io_read_kbs

This message is removed.

Overview:

Get the io/read_kbs field of the given VBD_metrics.

Signature:

```
float get_io_read_kbs (session ref session_id, VBD_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `float`

value of the field

RPC name: get_io_write_kbs

This message is removed.

Overview:

Get the io/write_kbs field of the given VBD_metrics.

Signature:

```
float get_io_write_kbs (session ref session_id, VBD_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `float`

value of the field

RPC name: get_last_updated

This message is removed.

Overview:

Get the last_updated field of the given VBD_metrics.

Signature:

```
datetime get_last_updated (session ref session_id, VBD_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_other_config

This message is removed.

Overview:

Get the other_config field of the given VBD_metrics.

Signature:

```
(string -> string) map get_other_config (session ref session_id, VBD_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

This message is removed.

Overview:

Get a record containing the current state of the given VBD_metrics.

Signature:

```
VBD_metrics record get_record (session ref session_id, VBD_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VBD_metrics record`

all fields from the object

RPC name: get_uuid

This message is removed.

Overview:

Get the uuid field of the given VBD_metrics.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD_metrics ref	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

This message is removed.

Overview:

Remove the given key and its corresponding value from the other_config field of the given VBD_metrics. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, VBD_metrics ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VBD_metrics ref	self	reference to the object
string	key	Key to remove

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_other_config

This message is removed.

Overview:

Set the other_config field of the given VBD_metrics.

Signature:

```
void set_other_config (session ref session_id, VBD_metrics ref self, (string ->
string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VBD_metrics ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

Class: VDI

A virtual disk image

Fields for class: VDI

Field	Type	Qualifier	Description
allow_caching	`bool`	*RO/runtime*	true if this VDI is to be cached in the local cache SR
allowed_operations	`vdi_operations set`	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
cbt_enabled	`bool`	*RO/runtime*	True if changed blocks are tracked for this VDI
crash_dumps	`crashdump ref set`	*RO/runtime*	list of crash dumps that refer to this disk
current_operations	`(string -> vdi_operations) map`	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.
is_a_snapshot	`bool`	*RO/runtime*	true if this is a snapshot.
is_tools_iso	`bool`	*RO/runtime*	Whether this VDI is a Tools ISO
location	`string`	*RO/runtime*	location information

Field	Type	Qualifier	Description
managed	bool	*RO/runtime*	
metadata_latest	bool	*RO/runtime*	Whether this VDI contains the latest known accessible metadata for the pool
metadata_of_pool	pool ref	*RO/runtime*	The pool whose metadata is contained in this VDI
missing	bool	*RO/runtime*	true if SR scan operation reported this VDI as not present on disk
name_description	string	*RO/constructor*	a notes field containing human-readable description
name_label	string	*RO/constructor*	a human-readable name
on_boot	on_boot	*RO/runtime*	The behaviour of this VDI on a VM boot
other_config	(string -> string) map	*RW*	additional configuration
parent	VDI ref	*RO/runtime*	**Deprecated**. This field is always null. Deprecated
physical_utilisation	int	*RO/runtime*	amount of physical space that the disk image is currently taking up on the storage repository (in bytes)
read_only	bool	*RO/constructor*	true if this disk may ONLY be mounted read-only
sharable	bool	*RO/constructor*	true if this disk may be shared
sm_config	(string -> string) map	*RW*	SM dependent data
snapshot_of	VDI ref	*RO/runtime*	Ref pointing to the VDI this snapshot is of.
snapshot_time	datetime	*RO/runtime*	Date/time when this snapshot was created.
snapshots	VDI ref set	*RO/runtime*	List pointing to all the VDIs snapshots.
SR	SR ref	*RO/constructor*	storage repository in which the VDI resides
storage_lock	bool	*RO/runtime*	true if this disk is locked at the storage level
tags	string set	*RW*	user-specified tags for categorization purposes
type	vdi_type	*RO/constructor*	type of the VDI
uuid	string	*RO/runtime*	Unique identifier/object reference
VBDs	VBD ref set	*RO/runtime*	list of vbds that refer to this disk
virtual_size	int	*RO/constructor*	size of disk as presented to the guest (in bytes). Note that, depending on storage backend type, requested size may not be respected exactly

Field	Type	Qualifier	Description
xenstore_data	`(string -> string) map`	*RW*	data to be inserted into the xenstore tree (/local/domain/0/backend/vbd/<domid>/<device-id>/sm-data) after the VDI is attached. This is generally set by the SM backends on vdi_attach.

RPCs associated with class: VDI

RPC name: add_tags

Overview:

Add the given value to the tags field of the given VDI. If the value is already in that Set, then do nothing.

Signature:

```
void add_tags (session ref session_id, VDI ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object
`string`	value	New value to add

Minimum Role: vm-operator

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 ref session_id, VDI ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VDI ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: vm-admin

Return Type: void

RPC name: add_to_sm_config

Overview:

Add the given key-value pair to the sm_config field of the given VDI.

Signature:

```
void add_to_sm_config (session ref session_id, VDI ref self, string key, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VDI ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: vm-admin

Return Type: void

RPC name: add_to_xenstore_data

Overview:

Add the given key-value pair to the xenstore_data field of the given VDI.

Signature:

```
void add_to_xenstore_data (session ref session_id, VDI ref self, string key, string
value)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VDI ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: vm-admin

Return Type: void

RPC name: clone

Overview:

Take an exact copy of the VDI and return a reference to the new disk. If any driver_params are specified then these are passed through to the storage-specific substrate driver that implements the clone operation. NB the clone lives in the same Storage Repository as its parent.

Signature:

```
VDI ref clone (session ref session_id, VDI ref vdi, (string -> string) map
driver_params)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VDI ref	vdi	The VDI to clone
(string -> string) map	driver_params	Optional parameters that are passed through to the backend driver in order to specify storage-type-specific clone options

Minimum Role: vm-admin

Return Type: VDI ref

The ID of the newly created VDI.

RPC name: copy

Overview:

Copy either a full VDI or the block differences between two VDIs into either a fresh VDI or an existing VDI.

Signature:

```

```

```
VDI ref copy (session ref session_id, VDI ref vdi, SR ref sr, VDI ref base_vdi, VDI
ref into_vdi)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VDI ref	vdi	The VDI to copy
SR ref	sr	The destination SR (only required if the destination VDI is not specified
VDI ref	base_vdi	The base VDI (only required if copying only changed blocks, by default all blocks will be copied)
VDI ref	into_vdi	The destination VDI to copy blocks into (if omitted then a destination SR must be provided and a fresh VDI will be created)

Minimum Role: vm-admin

Return Type: VDI ref

The reference of the VDI where the blocks were written.

Possible Error Codes: VDI_READONLY, VDI_TOO_SMALL, VDI_NOT_SPARSE

RPC name: create

Overview:

Create a new VDI instance, and return its handle.

Signature:

```
VDI ref create (session ref session_id, VDI record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VDI record	args	All constructor arguments

Minimum Role: vm-admin

Return Type: VDI ref

reference to the newly created object

RPC name: data_destroy

Overview:

Delete the data of the snapshot VDI, but keep its changed block tracking metadata. When successful, this call changes the type of the VDI to cbt_metadata. This operation is idempotent: calling it on a VDI of type cbt_metadata results in a no-op, and no error will be thrown.

Signature:

```
void data_destroy (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	The VDI whose data should be deleted.

Minimum Role: vm-admin

Return Type: `void`

Possible Error Codes: `SR_OPERATION_NOT_SUPPORTED`, `VDI_MISSING`, `SR_NOT_ATTACHED`, `SR_HAS_NO_PBDS`, `OPERATION_NOT_ALLOWED`, `VDI_INCOMPATIBLE_TYPE`, `VDI_NO_CBT_METADATA`, `VDI_IN_USE`, `VDI_IS_A_PHYSICAL_DEVICE`

RPC name: destroy

Overview:

Destroy the specified VDI instance.

Signature:

```
void destroy (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: vm-admin

Return Type: `void`

RPC name: disable_cbt

Overview:

Disable changed block tracking for the VDI. This call is only allowed on VDIs that support enabling CBT. It is an idempotent operation - disabling CBT for a VDI for which CBT is not enabled results in a no-op, and no error will be thrown.

Signature:

```
void disable_cbt (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	The VDI for which CBT should be disabled

Minimum Role: vm-admin

Return Type: `void`

Possible Error Codes: `SR_OPERATION_NOT_SUPPORTED`, `VDI_MISSING`, `SR_NOT_ATTACHED`, `SR_HAS_NO_PBDS`, `OPERATION_NOT_ALLOWED`, `VDI_INCOMPATIBLE_TYPE`, `VDI_ON_BOOT_MODE_INCOMPATIBLE_WITH_OPERATION`

RPC name: enable_cbt

Overview:

Enable changed block tracking for the VDI. This call is idempotent - enabling CBT for a VDI for which CBT is already enabled results in a no-op, and no error will be thrown.

Signature:

```
void enable_cbt (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	The VDI for which CBT should be enabled

Minimum Role: vm-admin

Return Type: `void`

Possible Error Codes: `SR_OPERATION_NOT_SUPPORTED`, `VDI_MISSING`, `SR_NOT_ATTACHED`, `SR_HAS_NO_PBDS`, `OPERATION_NOT_ALLOWED`, `VDI_INCOMPATIBLE_TYPE`, `VDI_ON_BOOT_MODE_INCOMPATIBLE_WITH_OPERATION`

RPC name: forget

Overview:

Removes a VDI record from the database

Signature:

```
void forget (session ref session_id, VDI ref vdi)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	vdi	The VDI to forget about

Minimum Role: vm-admin

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the VDIs known to the system.

Signature:

```
VDI ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `VDI ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of VDI references to VDI records for all VDIs known to the system.

Signature:

```
(VDI ref -> VDI record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VDI ref -> VDI record) map`

records of all objects

RPC name: get_allow_caching

Overview:

Get the allow_caching field of the given VDI.

Signature:

```
bool get_allow_caching (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given VDI.

Signature:

```
vdi_operations set get_allowed_operations (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vdi_operations set`

value of the field

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 ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `VDI ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the VDI instance with the specified UUID.

Signature:

```
VDI ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VDI ref`

reference to the object

RPC name: get_cbt_enabled

Overview:

Get the cbt_enabled field of the given VDI.

Signature:

```
bool get_cbt_enabled (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

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 ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `crashdump ref set`

value of the field

RPC name: get_current_operations

Overview:

Get the current_operations field of the given VDI.

Signature:

```

```

```
(string -> vdi_operations) map get_current_operations (session ref session_id, VDI ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> vdi_operations) map`

value of the field

RPC name: get_is_a_snapshot

Overview:

Get the is_a_snapshot field of the given VDI.

Signature:

```
bool get_is_a_snapshot (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_is_tools_iso

Overview:

Get the is_tools_iso field of the given VDI.

Signature:

```
bool get_is_tools_iso (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_location

Overview:

Get the location field of the given VDI.

Signature:

```
string get_location (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_managed

Overview:

Get the managed field of the given VDI.

Signature:

```
bool get_managed (session ref session_id, VDI ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_metadata_latest

Overview:

Get the metadata_latest field of the given VDI.

Signature:

```
bool get_metadata_latest (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_metadata_of_pool

Overview:

Get the metadata_of_pool field of the given VDI.

Signature:

```
pool ref get_metadata_of_pool (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VDI ref	self	reference to the object

Minimum Role: read-only

Return Type: pool ref

value of the field

RPC name: get_missing

Overview:

Get the missing field of the given VDI.

Signature:

```
bool get_missing (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VDI ref	self	reference to the object

Minimum Role: read-only

Return Type: bool

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given VDI.

Signature:

```
string get_name_description (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VDI ref	self	reference to the object

Minimum Role: read-only

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 ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_nbd_info

Overview:

Get details specifying how to access this VDI via a Network Block Device server. For each of a set of NBD server addresses on which the VDI is available, the return value set contains a vdi_nbd_server_info object that contains an exportname to request once the NBD connection is established, and connection details for the address. An empty list is returned if there is no network that has a PIF on a host with access to the relevant SR, or if no such network has been assigned an NBD-related purpose in its purpose field. To access the given VDI, any of the vdi_nbd_server_info objects can be used to make a connection to a server, and then the VDI will be available by requesting the exportname.

Signature:

```
vdi_nbd_server_info record set get_nbd_info (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VDI ref	self	The VDI to access via Network Block Device protocol

Minimum Role: vm-admin

Return Type: vdi_nbd_server_info record set

The details necessary for connecting to the VDI over NBD. This includes an authentication token, so must be treated as sensitive material and must not be sent over insecure networks.

Possible Error Codes: VDI_INCOMPATIBLE_TYPE

RPC name: get_on_boot

Overview:

Get the on_boot field of the given VDI.

Signature:

```
on_boot get_on_boot (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VDI ref	self	reference to the object

Minimum Role: read-only

Return Type: on_boot

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given VDI.

Signature:

```
(string -> string) map get_other_config (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VDI ref	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_parent

This message is deprecated.

Overview:

Get the parent field of the given VDI.

Signature:

```
VDI ref get_parent (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VDI ref	self	reference to the object

Minimum Role: read-only

Return Type: `VDI ref`

value of the field

RPC name: get_physical_utilisation

Overview:

Get the physical_utilisation field of the given VDI.

Signature:

```
int get_physical_utilisation (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_read_only

Overview:

Get the read_only field of the given VDI.

Signature:

```
bool get_read_only (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given VDI.

Signature:

```
VDI record get_record (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VDI record`

all fields from the object

RPC name: get_sharable

Overview:

Get the sharable field of the given VDI.

Signature:

```
bool get_sharable (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_sm_config

Overview:

Get the sm_config field of the given VDI.

Signature:

```
(string -> string) map get_sm_config (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_snapshot_of

Overview:

Get the snapshot_of field of the given VDI.

Signature:

```
VDI ref get_snapshot_of (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VDI ref`

value of the field

RPC name: get_snapshot_time

Overview:

Get the snapshot_time field of the given VDI.

Signature:

```
datetime get_snapshot_time (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_snapshots

Overview:

Get the snapshots field of the given VDI.

Signature:

```
VDI ref set get_snapshots (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VDI ref set`

value of the field

RPC name: get_SR

Overview:

Get the SR field of the given VDI.

Signature:

```
SR ref get_SR (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SR ref`

value of the field

RPC name: get_storage_lock

Overview:

Get the storage_lock field of the given VDI.

Signature:

```
bool get_storage_lock (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_tags

Overview:

Get the tags field of the given VDI.

Signature:

```
string set get_tags (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_type

Overview:

Get the type field of the given VDI.

Signature:

```
vdi_type get_type (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vdi_type`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given VDI.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VBDs

Overview:

Get the VBDs field of the given VDI.

Signature:

```
VBD ref set get_VBDs (session ref session_id, VDI ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VBD 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 ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_xenstore_data

Overview:

Get the xenstore_data field of the given VDI.

Signature:

```
(string -> string) map get_xenstore_data (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VDI ref	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: introduce

Overview:

Create a new VDI record in the database only

Signature:

```
VDI ref introduce (session ref session_id, string uuid, string name_label, string
name_description, SR ref SR, vdi_type type, bool sharable, bool read_only, (string ->
string) map other_config, string location, (string -> string) map xenstore_data,
(string -> string) map sm_config, bool managed, int virtual_size, int
physical_utilisation, pool ref metadata_of_pool, bool is_a_snapshot, datetime
snapshot_time, VDI ref snapshot_of)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
string	uuid	The uuid of the disk to introduce
string	name_label	The name of the disk record
string	name_description	The description of the disk record
SR ref	SR	The SR that the VDI is in
vdi_type	type	The type of the VDI
bool	sharable	true if this disk may be shared
bool	read_only	true if this disk may ONLY be mounted read-only
(string -> string) map	other_config	additional configuration
string	location	location information
(string -> string) map	xenstore_data	Data to insert into xenstore
(string -> string) map	sm_config	Storage-specific config
bool	managed	Storage-specific config
int	virtual_size	Storage-specific config

type	name	description
int	physical_utilisation	Storage-specific config
pool ref	metadata_of_pool	Storage-specific config
bool	is_a_snapshot	Storage-specific config
datetime	snapshot_time	Storage-specific config
VDI ref	snapshot_of	Storage-specific config

Minimum Role: vm-admin

Return Type: VDI ref

The ref of the newly created VDI record.

Possible Error Codes: SR_OPERATION_NOT_SUPPORTED

RPC name: list_changed_blocks

Overview:

Compare two VDIs in 64k block increments and report which blocks differ. This operation is not allowed when vdi_to is attached to a VM.

Signature:

```
string list_changed_blocks (session ref session_id, VDI ref vdi_from, VDI ref vdi_to)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VDI ref	vdi_from	The first VDI.
VDI ref	vdi_to	The second VDI.

Minimum Role: vm-operator

Return Type: string

A base64 string-encoding of the bitmap showing which blocks differ in the two VDIs.

Possible Error Codes: SR_OPERATION_NOT_SUPPORTED, VDI_MISSING, SR_NOT_ATTACHED, SR_HAS_NO_PBDS, VDI_IN_USE

RPC name: open_database

Overview:

Load the metadata found on the supplied VDI and return a session reference which can be used in API calls to query its contents.

Signature:

```
session ref open_database (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	The VDI which contains the database to open

Minimum Role: pool-operator

Return Type: `session ref`

A session which can be used to query the database

RPC name: pool_migrate

Overview:

Migrate a VDI, which may be attached to a running guest, to a different SR. The destination SR must be visible to the guest.

Signature:

```
VDI ref pool_migrate (session ref session_id, VDI ref vdi, SR ref sr, (string ->
string) map options)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	vdi	The VDI to migrate
`SR ref`	sr	The destination SR
`(string -> string) map`	options	Other parameters

Minimum Role: vm-power-admin

Return Type: `VDI ref`

The new reference of the migrated VDI.

RPC name: read_database_pool_uuid

Overview:

Check the VDI cache for the pool UUID of the database on this VDI.

Signature:

```
string read_database_pool_uuid (session ref session_id, VDI ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	The metadata VDI to look up in the cache.

Minimum Role: read-only

Return Type: `string`

The cached pool UUID of the database on the VDI.

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 ref session_id, VDI ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: vm-admin

Return Type: `void`

RPC name: remove_from_sm_config

Overview:

Remove the given key and its corresponding value from the sm_config field of the given VDI. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_sm_config (session ref session_id, VDI ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: vm-admin

Return Type: `void`

RPC name: remove_from_xenstore_data

Overview:

Remove the given key and its corresponding value from the xenstore_data field of the given VDI. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_xenstore_data (session ref session_id, VDI ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: vm-admin

Return Type: `void`

RPC name: remove_tags

Overview:

Remove the given value from the tags field of the given VDI. If the value is not in that Set, then do nothing.

Signature:

```
void remove_tags (session ref session_id, VDI ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object
`string`	value	Value to remove

Minimum Role: vm-operator

Return Type: `void`

RPC name: resize

Overview:

Resize the VDI.

Signature:

```
void resize (session ref session_id, VDI ref vdi, int size)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	vdi	The VDI to resize
`int`	size	The new size of the VDI

Minimum Role: vm-admin

Return Type: `void`

RPC name: resize_online

This message is removed.

Overview:

Resize the VDI which may or may not be attached to running guests.

Signature:

```
```

```
void resize_online (session ref session_id, VDI ref vdi, int size)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	vdi	The VDI to resize
`int`	size	The new size of the VDI

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_allow_caching

Overview:

Set the value of the allow_caching parameter. This value can only be changed when the VDI is not attached to a running VM. The caching behaviour is only affected by this flag for VHD-based VDIs that have one parent and no child VHDs. Moreover, caching only takes place when the host running the VM containing this VDI has a nominated SR for local caching.

Signature:

```
void set_allow_caching (session ref session_id, VDI ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	The VDI to modify
`bool`	value	The value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_name_description

Overview:

Set the name description of the VDI. This can only happen when its SR is currently attached.

Signature:

```

```

```
void set_name_description (session ref session_id, VDI ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	The VDI to modify
`string`	value	The name description for the VDI

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_name_label

Overview:

Set the name label of the VDI. This can only happen when then its SR is currently attached.

Signature:

```
void set_name_label (session ref session_id, VDI ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	The VDI to modify
`string`	value	The name lable for the VDI

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_on_boot

Overview:

Set the value of the on_boot parameter. This value can only be changed when the VDI is not attached to a running VM.

Signature:

```
void set_on_boot (session ref session_id, VDI ref self, on_boot value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	The VDI to modify
`on_boot`	value	The value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given VDI.

Signature:

```
void set_other_config (session ref session_id, VDI ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_read_only

Overview:

Sets the VDI's read_only field

Signature:

```
void set_read_only (session ref session_id, VDI ref self, bool value)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VDI ref	self	The VDI to modify
bool	value	The new value of the VDI's read_only field

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_sharable

Overview:

Sets the VDI's sharable field

Signature:

```
void set_sharable (session ref session_id, VDI ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VDI ref	self	The VDI to modify
bool	value	The new value of the VDI's sharable field

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_sm_config

Overview:

Set the sm_config field of the given VDI.

Signature:

```
void set_sm_config (session ref session_id, VDI ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VDI ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_tags

Overview:

Set the tags field of the given VDI.

Signature:

```
void set_tags (session ref session_id, VDI ref self, string set value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object
`string set`	value	New value to set

Minimum Role: vm-operator

Return Type: `void`

RPC name: set_xenstore_data

Overview:

Set the xenstore_data field of the given VDI.

Signature:

```
void set_xenstore_data (session ref session_id, VDI ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	self	reference to the object

type	name	description
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: snapshot

Overview:

Take a read-only snapshot of the VDI, returning a reference to the snapshot. If any driver_params are specified then these are passed through to the storage-specific substrate driver that takes the snapshot. NB the snapshot lives in the same Storage Repository as its parent.

Signature:

```
VDI ref snapshot (session ref session_id, VDI ref vdi, (string -> string) map
driver_params)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	vdi	The VDI to snapshot
`(string -> string) map`	driver_params	Optional parameters that can be passed through to backend driver in order to specify storage-type-specific snapshot options

Minimum Role: vm-admin

Return Type: `VDI ref`

The ID of the newly created VDI.

RPC name: update

Overview:

Ask the storage backend to refresh the fields in the VDI object

Signature:

```
void update (session ref session_id, VDI ref vdi)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VDI ref`	vdi	The VDI whose stats (eg size) should be updated

Minimum Role: vm-admin

Return Type: `void`

Possible Error Codes: `SR_OPERATION_NOT_SUPPORTED`

Class: vdi_nbd_server_info

Details for connecting to a VDI using the Network Block Device protocol

Fields for class: vdi_nbd_server_info

Field	Type	Qualifier	Description
address	`string`	*RO/runtime*	An address on which the server can be reached; this can be IPv4, IPv6, or a DNS name.
cert	`string`	*RO/runtime*	The TLS certificate of the server
exportname	`string`	*RO/runtime*	The exportname to request over NBD. This holds details including an authentication token, so it must be protected appropriately. Clients should regard the exportname as an opaque string or token.
port	`int`	*RO/runtime*	The TCP port
subject	`string`	*RO/runtime*	For convenience, this redundant field holds a DNS (hostname) subject of the certificate. This can be a wildcard, but only for a certificate that has a wildcard subject and no concrete hostname subjects.

RPCs associated with class: vdi_nbd_server_info

Class vdi_nbd_server_info has no additional RPCs associated with it.

Class: VGPU

A virtual GPU (vGPU)

Fields for class: VGPU

Field	Type	Qualifier	Description
compatibility_metadata	`(string -> string) map`	*RO/runtime*	VGPU metadata to determine whether a VGPU can migrate between two PGPUs
currently_attached	`bool`	*RO/runtime*	Reflects whether the virtual device is currently connected to a physical device

Field	Type	Qualifier	Description
device	string	*RO/runtime*	Order in which the devices are plugged into the VM
extra_args	string	*RW*	Extra arguments for vGPU and passed to demu
GPU_group	GPU_group ref	*RO/runtime*	GPU group used by the vGPU
other_config	(string -> string) map	*RW*	Additional configuration
PCI	PCI ref	*RO/runtime*	Device passed trough to VM, either as full device or SR-IOV virtual function
resident_on	PGPU ref	*RO/runtime*	The PGPU on which this VGPU is running
scheduled_to_be_resident_on	PGPU ref	*RO/runtime*	The PGPU on which this VGPU is scheduled to run
type	VGPU_type ref	*RO/runtime*	Preset type for this VGPU
uuid	string	*RO/runtime*	Unique identifier/object reference
VM	VM ref	*RO/runtime*	VM that owns the vGPU

RPCs associated with class: VGPU

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given VGPU.

Signature:

```
void add_to_other_config (session ref session_id, VGPU ref self, string key, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VGPU ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: create

Overview:

Signature:

```
VGPU ref create (session ref session_id, VM ref VM, GPU_group ref GPU_group, string
device, (string -> string) map other_config, VGPU_type ref type)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	VM	
`GPU_group ref`	GPU_group	
`string`	device	
`(string -> string) map`	other_config	
`VGPU_type ref`	type	

Minimum Role: pool-operator

Return Type: `VGPU ref`

reference to the newly created object

RPC name: destroy

Overview:

Signature:

```
void destroy (session ref session_id, VGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	The vGPU to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the VGPUs known to the system.

Signature:

```
VGPU ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `VGPU ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of VGPU references to VGPU records for all VGPUs known to the system.

Signature:

```
(VGPU ref -> VGPU record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VGPU ref -> VGPU record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the VGPU instance with the specified UUID.

Signature:

```
VGPU ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VGPU ref`

reference to the object

RPC name: get_compatibility_metadata

Overview:

Get the compatibility_metadata field of the given VGPU.

Signature:

```
(string -> string) map get_compatibility_metadata (session ref session_id, VGPU ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_currently_attached

Overview:

Get the currently_attached field of the given VGPU.

Signature:

```
bool get_currently_attached (session ref session_id, VGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_device

Overview:

Get the device field of the given VGPU.

Signature:

```
string get_device (session ref session_id, VGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_extra_args

Overview:

Get the extra_args field of the given VGPU.

Signature:

```
string get_extra_args (session ref session_id, VGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_GPU_group

Overview:

Get the GPU_group field of the given VGPU.

Signature:

```
GPU_group ref get_GPU_group (session ref session_id, VGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `GPU_group ref`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given VGPU.

Signature:

```
(string -> string) map get_other_config (session ref session_id, VGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_PCI

Overview:

Get the PCI field of the given VGPU.

Signature:

```
PCI ref get_PCI (session ref session_id, VGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PCI ref`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given VGPU.

Signature:

```
VGPU record get_record (session ref session_id, VGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VGPU record`

all fields from the object

RPC name: get_resident_on

Overview:

Get the resident_on field of the given VGPU.

Signature:

```
PGPU ref get_resident_on (session ref session_id, VGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PGPU ref`

value of the field

RPC name: get_scheduled_to_be_resident_on

Overview:

Get the scheduled_to_be_resident_on field of the given VGPU.

Signature:

```
PGPU ref get_scheduled_to_be_resident_on (session ref session_id, VGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PGPU ref`

value of the field

RPC name: get_type

Overview:

Get the type field of the given VGPU.

Signature:

```
VGPU_type ref get_type (session ref session_id, VGPU ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VGPU ref	self	reference to the object

Minimum Role: read-only

Return Type: VGPU_type ref

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given VGPU.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VGPU ref	self	reference to the object

Minimum Role: read-only

Return Type: string

value of the field

RPC name: get_VM

Overview:

Get the VM field of the given VGPU.

Signature:

```
VM ref get_VM (session ref session_id, VGPU ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VGPU ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given VGPU. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, VGPU ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_extra_args

Overview:

Set the extra_args field of the given VGPU.

Signature:

```
void set_extra_args (session ref session_id, VGPU ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object

type	name	description
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given VGPU.

Signature:

```
void set_other_config (session ref session_id, VGPU ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: VGPU_type

A type of virtual GPU

Fields for class: VGPU_type

Field	Type	Qualifier	Description
compatible_types_in_vm	`VGPU_type ref set`	*RO/runtime*	List of VGPU types which are compatible in one VM
enabled_on_GPU_groups	`GPU_group ref set`	*RO/runtime*	List of GPU groups in which at least one have this VGPU type enabled
enabled_on_PGPUs	`PGPU ref set`	*RO/runtime*	List of PGPUs that have this VGPU type enabled

Field	Type	Qualifier	Description
experimental	bool	*RO/constructor*	Indicates whether VGPUs of this type should be considered experimental
framebuffer_size	int	*RO/constructor*	Framebuffer size of the VGPU type, in bytes
identifier	string	*RO/constructor*	Key used to identify VGPU types and avoid creating duplicates - this field is used internally and not intended for interpretation by API clients
implementation	vgpu_type_implementation	*RO/constructor*	The internal implementation of this VGPU type
max_heads	int	*RO/constructor*	Maximum number of displays supported by the VGPU type
max_resolution_x	int	*RO/constructor*	Maximum resolution (width) supported by the VGPU type
max_resolution_y	int	*RO/constructor*	Maximum resolution (height) supported by the VGPU type
model_name	string	*RO/constructor*	Model name associated with the VGPU type
supported_on_GPU_groups	GPU_group ref set	*RO/runtime*	List of GPU groups in which at least one PGPU supports this VGPU type
supported_on_PGPUs	PGPU ref set	*RO/runtime*	List of PGPUs that support this VGPU type
uuid	string	*RO/runtime*	Unique identifier/object reference
vendor_name	string	*RO/constructor*	Name of VGPU vendor
VGPUs	VGPU ref set	*RO/runtime*	List of VGPUs of this type

RPCs associated with class: VGPU_type

RPC name: get_all

Overview:

Return a list of all the VGPU_types known to the system.

Signature:

```
VGPU_type ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `VGPU_type ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of VGPU_type references to VGPU_type records for all VGPU_types known to the system.

Signature:

```
(VGPU_type ref -> VGPU_type record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VGPU_type ref -> VGPU_type record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the VGPU_type instance with the specified UUID.

Signature:

```
VGPU_type ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VGPU_type ref`

reference to the object

RPC name: get_compatible_types_in_vm

Overview:

Get the compatible_types_in_vm field of the given VGPU_type.

Signature:

```
VGPU_type ref set get_compatible_types_in_vm (session ref session_id, VGPU_type ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VGPU_type ref set`

value of the field

RPC name: get_enabled_on_GPU_groups

Overview:

Get the enabled_on_GPU_groups field of the given VGPU_type.

Signature:

```
GPU_group ref set get_enabled_on_GPU_groups (session ref session_id, VGPU_type ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `GPU_group ref set`

value of the field

RPC name: get_enabled_on_PGPUs

Overview:

Get the enabled_on_PGPUs field of the given VGPU_type.

Signature:

```
PGPU ref set get_enabled_on_PGPUs (session ref session_id, VGPU_type ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PGPU ref set`

value of the field

RPC name: get_experimental

Overview:

Get the experimental field of the given VGPU_type.

Signature:

```
bool get_experimental (session ref session_id, VGPU_type ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_framebuffer_size

Overview:

Get the framebuffer_size field of the given VGPU_type.

Signature:

```
int get_framebuffer_size (session ref session_id, VGPU_type ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_identifier

Overview:

Get the identifier field of the given VGPU_type.

Signature:

```
string get_identifier (session ref session_id, VGPU_type ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_implementation

Overview:

Get the implementation field of the given VGPU_type.

Signature:

```
vgpu_type_implementation get_implementation (session ref session_id, VGPU_type ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vgpu_type_implementation`

value of the field

RPC name: get_max_heads

Overview:

Get the max_heads field of the given VGPU_type.

Signature:

```
int get_max_heads (session ref session_id, VGPU_type ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_max_resolution_x

Overview:

Get the max_resolution_x field of the given VGPU_type.

Signature:

```
int get_max_resolution_x (session ref session_id, VGPU_type ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_max_resolution_y

Overview:

Get the max_resolution_y field of the given VGPU_type.

Signature:

```
int get_max_resolution_y (session ref session_id, VGPU_type ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_model_name

Overview:

Get the model_name field of the given VGPU_type.

Signature:

```
string get_model_name (session ref session_id, VGPU_type ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given VGPU_type.

Signature:

```
VGPU_type record get_record (session ref session_id, VGPU_type ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VGPU_type record`

all fields from the object

RPC name: get_supported_on_GPU_groups

Overview:

Get the supported_on_GPU_groups field of the given VGPU_type.

Signature:

```
GPU_group ref set get_supported_on_GPU_groups (session ref session_id, VGPU_type ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `GPU_group ref set`

value of the field

RPC name: get_supported_on_PGPUs

Overview:

Get the supported_on_PGPUs field of the given VGPU_type.

Signature:

```
PGPU ref set get_supported_on_PGPUs (session ref session_id, VGPU_type ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PGPU ref set`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given VGPU_type.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_vendor_name

Overview:

Get the vendor_name field of the given VGPU_type.

Signature:

```
string get_vendor_name (session ref session_id, VGPU_type ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VGPUs

Overview:

Get the VGPUs field of the given VGPU_type.

Signature:

```
VGPU ref set get_VGPUs (session ref session_id, VGPU_type ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VGPU_type ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VGPU ref set`

value of the field

Class: VIF

A virtual network interface

Fields for class: VIF

Field	Type	Qualifier	Description
allowed_operations	`vif_operations set`	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
current_operations	`(string -> vif_operations) map`	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.
currently_attached	`bool`	*RO/runtime*	is the device currently attached (erased on reboot)
device	`string`	*RO/constructor*	order in which VIF backends are created by xapi
ipv4_addresses	`string set`	*RO/runtime*	IPv4 addresses in CIDR format
ipv4_allowed	`string set`	*RO/constructor*	A list of IPv4 addresses which can be used to filter traffic passing through this VIF
ipv4_configuration_mode	`vif_ipv4_configuration_mode`	*RO/runtime*	Determines whether IPv4 addresses are configured on the VIF
ipv4_gateway	`string`	*RO/runtime*	IPv4 gateway (the empty string means that no gateway is set)
ipv6_addresses	`string set`	*RO/runtime*	IPv6 addresses in CIDR format

Field	Type	Qualifier	Description
ipv6_allowed	`string set`	*RO/constructor*	A list of IPv6 addresses which can be used to filter traffic passing through this VIF
ipv6_configuration_mode	`vif_ipv6_configuration_mode`	*RO/runtime*	Determines whether IPv6 addresses are configured on the VIF
ipv6_gateway	`string`	*RO/runtime*	IPv6 gateway (the empty string means that no gateway is set)
locking_mode	`vif_locking_mode`	*RO/constructor*	current locking mode of the VIF
MAC	`string`	*RO/constructor*	ethernet MAC address of virtual interface, as exposed to guest
MAC_autogenerated	`bool`	*RO/runtime*	true if the MAC was autogenerated; false indicates it was set manually
metrics	`VIF_metrics ref`	*RO/runtime*	**Removed**. metrics associated with this VIF
MTU	`int`	*RO/constructor*	MTU in octets
network	`network ref`	*RO/constructor*	virtual network to which this vif is connected
other_config	`(string -> string) map`	*RW*	additional configuration
qos_algorithm_params	`(string -> string) map`	*RW*	parameters for chosen QoS algorithm
qos_algorithm_type	`string`	*RW*	QoS algorithm to use
qos_supported_algorithms	`string set`	*RO/runtime*	supported QoS algorithms for this VIF
runtime_properties	`(string -> string) map`	*RO/runtime*	Device runtime properties
status_code	`int`	*RO/runtime*	error/success code associated with last attach-operation (erased on reboot)

Field	Type	Qualifier	Description
status_detail	`string`	*RO/runtime*	error/success information associated with last attach-operation status (erased on reboot)
uuid	`string`	*RO/runtime*	Unique identifier/object reference
VM	`VM ref`	*RO/constructor*	virtual machine to which this vif is connected

RPCs associated with class: VIF

RPC name: add_ipv4_allowed

Overview:

Associates an IPv4 address with this VIF

Signature:

```
void add_ipv4_allowed (session ref session_id, VIF ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	The VIF which the IP address will be associated with
`string`	value	The IP address which will be associated with the VIF

Minimum Role: pool-operator

Return Type: `void`

RPC name: add_ipv6_allowed

Overview:

Associates an IPv6 address with this VIF

Signature:

```
void add_ipv6_allowed (session ref session_id, VIF ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	The VIF which the IP address will be associated with
`string`	value	The IP address which will be associated with the VIF

Minimum Role: pool-operator

Return Type: `void`

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given VIF.

Signature:

```
void add_to_other_config (session ref session_id, VIF ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: vm-admin

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 ref session_id, VIF ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VIF ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: vm-admin

Return Type: void

RPC name: configure_ipv4

Overview:

Configure IPv4 settings for this virtual interface

Signature:

```
void configure_ipv4 (session ref session_id, VIF ref self, vif_ipv4_configuration_mode
mode, string address, string gateway)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VIF ref	self	The VIF to configure
vif_ipv4_configuration_mode	mode	Whether to use static or no IPv4 assignment
string	address	The IPv4 address in <addr>/<prefix length> format (for static mode only)
string	gateway	The IPv4 gateway (for static mode only; leave empty to not set a gateway)

Minimum Role: vm-operator

Return Type: void

RPC name: configure_ipv6

Overview:

Configure IPv6 settings for this virtual interface

Signature:

```

```

```
void configure_ipv6 (session ref session_id, VIF ref self, vif_ipv6_configuration_mode
mode, string address, string gateway)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	The VIF to configure
`vif_ipv6_configuration_mode`	mode	Whether to use static or no IPv6 assignment
`string`	address	The IPv6 address in <addr>/<prefix length> format (for static mode only)
`string`	gateway	The IPv6 gateway (for static mode only; leave empty to not set a gateway)

Minimum Role: vm-operator

Return Type: `void`

RPC name: create

Overview:

Create a new VIF instance, and return its handle.

Signature:

```
VIF ref create (session ref session_id, VIF record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF record`	args	All constructor arguments

Minimum Role: vm-admin

Return Type: `VIF ref`

reference to the newly created object

RPC name: destroy

Overview:

Destroy the specified VIF instance.

Signature:

```
void destroy (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: vm-admin

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 ref session_id)
```

Minimum Role: read-only

Return Type: `VIF ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of VIF references to VIF records for all VIFs known to the system.

Signature:

```
(VIF ref -> VIF record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VIF ref -> VIF record) map`

records of all objects

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given VIF.

Signature:

```
vif_operations set get_allowed_operations (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vif_operations set`

value of the field

RPC name: get_by_uuid

Overview:

Get a reference to the VIF instance with the specified UUID.

Signature:

```
VIF ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VIF ref`

reference to the object

RPC name: get_current_operations

Overview:

Get the current_operations field of the given VIF.

Signature:

```
(string -> vif_operations) map get_current_operations (session ref session_id, VIF ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> vif_operations) map`

value of the field

RPC name: get_currently_attached

Overview:

Get the currently_attached field of the given VIF.

Signature:

```
bool get_currently_attached (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_device

Overview:

Get the device field of the given VIF.

Signature:

```

```

```
string get_device (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_ipv4_addresses

Overview:

Get the ipv4_addresses field of the given VIF.

Signature:

```
string set get_ipv4_addresses (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_ipv4_allowed

Overview:

Get the ipv4_allowed field of the given VIF.

Signature:

```
string set get_ipv4_allowed (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_ipv4_configuration_mode

Overview:

Get the ipv4_configuration_mode field of the given VIF.

Signature:

```
vif_ipv4_configuration_mode get_ipv4_configuration_mode (session ref session_id, VIF
ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vif_ipv4_configuration_mode`

value of the field

RPC name: get_ipv4_gateway

Overview:

Get the ipv4_gateway field of the given VIF.

Signature:

```
string get_ipv4_gateway (session ref session_id, VIF ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_ipv6_addresses

Overview:

Get the ipv6_addresses field of the given VIF.

Signature:

```
string set get_ipv6_addresses (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_ipv6_allowed

Overview:

Get the ipv6_allowed field of the given VIF.

Signature:

```
string set get_ipv6_allowed (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_ipv6_configuration_mode

Overview:

Get the ipv6_configuration_mode field of the given VIF.

Signature:

```
vif_ipv6_configuration_mode get_ipv6_configuration_mode (session ref session_id, VIF
ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vif_ipv6_configuration_mode`

value of the field

RPC name: get_ipv6_gateway

Overview:

Get the ipv6_gateway field of the given VIF.

Signature:

```
string get_ipv6_gateway (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_locking_mode

Overview:

Get the locking_mode field of the given VIF.

Signature:

```
vif_locking_mode get_locking_mode (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vif_locking_mode`

value of the field

RPC name: get_MAC

Overview:

Get the MAC field of the given VIF.

Signature:

```
string get_MAC (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_MAC_autogenerated

Overview:

Get the MAC_autogenerated field of the given VIF.

Signature:

```
bool get_MAC_autogenerated (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_metrics

This message is removed.

Overview:

Get the metrics field of the given VIF.

Signature:

```
VIF_metrics ref get_metrics (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VIF_metrics ref`

value of the field

RPC name: get_MTU

Overview:

Get the MTU field of the given VIF.

Signature:

```
int get_MTU (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_network

Overview:

Get the network field of the given VIF.

Signature:

```
network ref get_network (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `network ref`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given VIF.

Signature:

```
(string -> string) map get_other_config (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

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 ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

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 ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given VIF.

Signature:

```
VIF record get_record (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VIF record`

all fields from the object

RPC name: get_runtime_properties

Overview:

Get the runtime_properties field of the given VIF.

Signature:

```
(string -> string) map get_runtime_properties (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

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 ref session_id, VIF ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given VIF.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VM

Overview:

Get the VM field of the given VIF.

Signature:

```
VM ref get_VM (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref`

value of the field

RPC name: move

Overview:

Move the specified VIF to the specified network, even while the VM is running

Signature:

```
void move (session ref session_id, VIF ref self, network ref network)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	The VIF to move

type	name	description
network ref	network	The network to move it to

Minimum Role: vm-admin

Return Type: void

RPC name: plug

Overview:

Hotplug the specified VIF, dynamically attaching it to the running VM

Signature:

```
void plug (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VIF ref	self	The VIF to hotplug

Minimum Role: vm-admin

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 VIF. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, VIF ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VIF ref	self	reference to the object
string	key	Key to remove

Minimum Role: vm-admin

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 ref session_id, VIF ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: vm-admin

Return Type: `void`

RPC name: remove_ipv4_allowed

Overview:

Removes an IPv4 address from this VIF

Signature:

```
void remove_ipv4_allowed (session ref session_id, VIF ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	The VIF from which the IP address will be removed
`string`	value	The IP address which will be removed from the VIF

Minimum Role: pool-operator

Return Type: `void`

RPC name: remove_ipv6_allowed

Overview:

Removes an IPv6 address from this VIF

Signature:

```
void remove_ipv6_allowed (session ref session_id, VIF ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	The VIF from which the IP address will be removed
`string`	value	The IP address which will be removed from the VIF

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_ipv4_allowed

Overview:

Set the IPv4 addresses to which traffic on this VIF can be restricted

Signature:

```
void set_ipv4_allowed (session ref session_id, VIF ref self, string set value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	The VIF which the IP addresses will be associated with
`string set`	value	The IP addresses which will be associated with the VIF

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_ipv6_allowed

Overview:

Set the IPv6 addresses to which traffic on this VIF can be restricted

Signature:

```
void set_ipv6_allowed (session ref session_id, VIF ref self, string set value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	The VIF which the IP addresses will be associated with
`string set`	value	The IP addresses which will be associated with the VIF

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_locking_mode

Overview:

Set the locking mode for this VIF

Signature:

```
void set_locking_mode (session ref session_id, VIF ref self, vif_locking_mode value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	The VIF whose locking mode will be set
`vif_locking_mode`	value	The new locking mode for the VIF

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given VIF.

Signature:

```

```

```
void set_other_config (session ref session_id, VIF ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_qos_algorithm_params

Overview:

Set the qos/algorithm_params field of the given VIF.

Signature:

```
void set_qos_algorithm_params (session ref session_id, VIF ref self, (string ->
string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_qos_algorithm_type

Overview:

Set the qos/algorithm_type field of the given VIF.

Signature:

```
void set_qos_algorithm_type (session ref session_id, VIF ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: unplug

Overview:

Hot-unplug the specified VIF, dynamically unattaching it from the running VM

Signature:

```
void unplug (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF ref`	self	The VIF to hot-unplug

Minimum Role: vm-admin

Return Type: `void`

RPC name: unplug_force

Overview:

Forcibly unplug the specified VIF

Signature:

```
void unplug_force (session ref session_id, VIF ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VIF ref	self	The VIF to forcibly unplug

Minimum Role: vm-admin

Return Type: void

Class: VIF_metrics

This class is removed.

The metrics associated with a virtual network device

Fields for class: VIF_metrics

Field	Type	Qualifier	Description
io_read_kbs	float	*RO/runtime*	**Removed**. Read bandwidth (KiB/s)
io_write_kbs	float	*RO/runtime*	**Removed**. Write bandwidth (KiB/s)
last_updated	datetime	*RO/runtime*	**Removed**. Time at which this information was last updated
other_config	(string -> string) map	*RW*	**Removed**. additional configuration
uuid	string	*RO/runtime*	**Removed**. Unique identifier/object reference

RPCs associated with class: VIF_metrics

RPC name: add_to_other_config

This message is removed.

Overview:

Add the given key-value pair to the other_config field of the given VIF_metrics.

Signature:

```
void add_to_other_config (session ref session_id, VIF_metrics ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VIF_metrics ref	self	reference to the object
string	key	Key to add

type	name	description
`string`	value	Value to add

Minimum Role: vm-admin

Return Type: `void`

RPC name: get_all

This message is removed.

Overview:

Return a list of all the VIF_metrics instances known to the system.

Signature:

```
VIF_metrics ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `VIF_metrics ref set`

references to all objects

RPC name: get_all_records

This message is removed.

Overview:

Return a map of VIF_metrics references to VIF_metrics records for all VIF_metrics instances known to the system.

Signature:

```
(VIF_metrics ref -> VIF_metrics record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VIF_metrics ref -> VIF_metrics record) map`

records of all objects

RPC name: get_by_uuid

This message is removed.

Overview:

Get a reference to the VIF_metrics instance with the specified UUID.

Signature:

```
VIF_metrics ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VIF_metrics ref`

reference to the object

RPC name: get_io_read_kbs

This message is removed.

Overview:

Get the io/read_kbs field of the given VIF_metrics.

Signature:

```
float get_io_read_kbs (session ref session_id, VIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `float`

value of the field

RPC name: get_io_write_kbs

This message is removed.

Overview:

Get the io/write_kbs field of the given VIF_metrics.

Signature:

```
float get_io_write_kbs (session ref session_id, VIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `float`

value of the field

RPC name: get_last_updated

This message is removed.

Overview:

Get the last_updated field of the given VIF_metrics.

Signature:

```
datetime get_last_updated (session ref session_id, VIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_other_config

This message is removed.

Overview:

Get the other_config field of the given VIF_metrics.

Signature:

```
(string -> string) map get_other_config (session ref session_id, VIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

This message is removed.

Overview:

Get a record containing the current state of the given VIF_metrics.

Signature:

```
VIF_metrics record get_record (session ref session_id, VIF_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VIF_metrics record`

all fields from the object

RPC name: get_uuid

This message is removed.

Overview:

Get the uuid field of the given VIF_metrics.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

This message is removed.

Overview:

Remove the given key and its corresponding value from the other_config field of the given VIF_metrics. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, VIF_metrics ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF_metrics ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_other_config

This message is removed.

Overview:

Set the other_config field of the given VIF_metrics.

Signature:

```
void set_other_config (session ref session_id, VIF_metrics ref self, (string ->
string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VIF_metrics ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

Class: VLAN

A VLAN mux/demux

Fields for class: VLAN

Field	Type	Qualifier	Description
other_config	`(string -> string) map`	*RW*	additional configuration
tag	`int`	*RO/constructor*	VLAN tag in use
tagged_PIF	`PIF ref`	*RO/constructor*	interface on which traffic is tagged
untagged_PIF	`PIF ref`	*RO/runtime*	interface on which traffic is untagged
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: VLAN

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given VLAN.

Signature:

```
void add_to_other_config (session ref session_id, VLAN ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VLAN ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: create

Overview:

Create a VLAN mux/demuxer

Signature:

```
VLAN ref create (session ref session_id, PIF ref tagged_PIF, int tag, network ref
network)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
PIF ref	tagged_PIF	PIF which receives the tagged traffic
int	tag	VLAN tag to use
network ref	network	Network to receive the untagged traffic

Minimum Role: pool-operator

Return Type: `VLAN ref`

The reference of the created VLAN object

RPC name: destroy

Overview:

Destroy a VLAN mux/demuxer

Signature:

```
void destroy (session ref session_id, VLAN ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VLAN ref`	self	VLAN mux/demuxer to destroy

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the VLANs known to the system.

Signature:

```
VLAN ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `VLAN ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of VLAN references to VLAN records for all VLANs known to the system.

Signature:

```
(VLAN ref -> VLAN record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VLAN ref -> VLAN record) map`

records of all objects

RPC name: get_by_uuid

Overview:

Get a reference to the VLAN instance with the specified UUID.

Signature:

```
VLAN ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VLAN ref`

reference to the object

RPC name: get_other_config

Overview:

Get the other_config field of the given VLAN.

Signature:

```
(string -> string) map get_other_config (session ref session_id, VLAN ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VLAN ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given VLAN.

Signature:

```
VLAN record get_record (session ref session_id, VLAN ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VLAN ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VLAN record`

all fields from the object

RPC name: get_tag

Overview:

Get the tag field of the given VLAN.

Signature:

```
int get_tag (session ref session_id, VLAN ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VLAN ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_tagged_PIF

Overview:

Get the tagged_PIF field of the given VLAN.

Signature:

```
PIF ref get_tagged_PIF (session ref session_id, VLAN ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VLAN ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF ref`

value of the field

RPC name: get_untagged_PIF

Overview:

Get the untagged_PIF field of the given VLAN.

Signature:

```
PIF ref get_untagged_PIF (session ref session_id, VLAN ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VLAN ref`	self	reference to the object

Minimum Role: read-only

Return Type: `PIF ref`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given VLAN.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VLAN ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given VLAN. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, VLAN ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VLAN ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given VLAN.

Signature:

```
void set_other_config (session ref session_id, VLAN ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VLAN ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: VM

A virtual machine (or 'guest').

Fields for class: VM

Field	Type	Qualifier	Description
actions_after_crash	`on_crash_behaviour`	*RO/constructor*	action to take if the guest crashes
actions_after_reboot	`on_normal_exit`	*RW*	action to take after the guest has rebooted itself
actions_after_shutdown	`on_normal_exit`	*RW*	action to take after the guest has shutdown itself
affinity	`host ref`	*RW*	A host which the VM has some affinity for (or NULL). This is used as a hint to the start call when it decides where to run the VM. Resource constraints may cause the VM to be started elsewhere.
allowed_operations	`vm_operations set`	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
appliance	`VM_appliance ref`	*RO/constructor*	the appliance to which this VM belongs
attached_PCIs	`PCI ref set`	*RO/runtime*	Currently passed-through PCI devices
bios_strings	`(string -> string) map`	*RO/runtime*	BIOS strings
blobs	`(string -> blob ref) map`	*RO/runtime*	Binary blobs associated with this VM

Field	Type	Qualifier	Description
blocked_operations	`(vm_operations -> string) map`	*RW*	List of operations which have been explicitly blocked and an error code
children	`VM ref set`	*RO/runtime*	List pointing to all the children of this VM
consoles	`console ref set`	*RO/runtime*	virtual console devices
crash_dumps	`crashdump ref set`	*RO/runtime*	crash dumps associated with this VM
current_operations	`(string -> vm_operations) map`	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.
domain_type	`domain_type`	*RO/constructor*	The type of domain that will be created when the VM is started
domarch	`string`	*RO/runtime*	Domain architecture (if available, null string otherwise)
domid	`int`	*RO/runtime*	domain ID (if available, -1 otherwise)
generation_id	`string`	*RO/constructor*	Generation ID of the VM
guest_metrics	`VM_guest_metrics ref`	*RO/runtime*	metrics associated with the running guest
ha_always_run	`bool`	*RO/constructor*	**Deprecated**. if true then the system will attempt to keep the VM running as much as possible.
ha_restart_priority	`string`	*RO/constructor*	has possible values: "best-effort" meaning "try to restart this VM if possible but don't consider the Pool to be overcommitted if this is not possible"; "restart" meaning "this VM should be restarted"; "" meaning "do not try to restart this VM"
hardware_platform_version	`int`	*RW*	The host virtual hardware platform version the VM can run on

Field	Type	Qualifier	Description
has_vendor_device	`bool`	*RO/constructor*	When an HVM guest starts, this controls the presence of the emulated C000 PCI device which triggers Windows Update to fetch or update PV drivers.
HVM_boot_params	`(string -> string) map`	*RW*	HVM boot params
HVM_boot_policy	`string`	*RO/constructor*	**Deprecated**. HVM boot policy
HVM_shadow_multiplier	`float`	*RO/constructor*	multiplier applied to the amount of shadow that will be made available to the guest
is_a_snapshot	`bool`	*RO/runtime*	true if this is a snapshot. Snapshotted VMs can never be started, they are used only for cloning other VMs
is_a_template	`bool`	*RW*	true if this is a template. Template VMs can never be started, they are used only for cloning other VMs
is_control_domain	`bool`	*RO/runtime*	true if this is a control domain (domain 0 or a driver domain)
is_default_template	`bool`	*RO/runtime*	true if this is a default template. Default template VMs can never be started or migrated, they are used only for cloning other VMs
is_snapshot_from_vmpp	`bool`	*RO/constructor*	**Deprecated**. true if this snapshot was created by the protection policy
is_vmss_snapshot	`bool`	*RO/constructor*	true if this snapshot was created by the snapshot schedule
last_boot_CPU_flags	`(string -> string) map`	*RO/runtime*	describes the CPU flags on which the VM was last booted
last_booted_record	`string`	*RO/runtime*	marshalled value containing VM record at time of last boot, updated dynamically to reflect the runtime state of the domain
memory_dynamic_max	`int`	*RO/constructor*	Dynamic maximum (bytes)
memory_dynamic_min	`int`	*RO/constructor*	Dynamic minimum (bytes)

Field	Type	Qualifier	Description
memory_overhead	int	*RO/runtime*	Virtualization memory overhead (bytes).
memory_static_max	int	*RO/constructor*	Statically-set (i.e. absolute) maximum (bytes). The value of this field at VM start time acts as a hard limit of the amount of memory a guest can use. New values only take effect on reboot.
memory_static_min	int	*RO/constructor*	Statically-set (i.e. absolute) mininum (bytes). The value of this field indicates the least amount of memory this VM can boot with without crashing.
memory_target	int	*RO/constructor*	**Deprecated**. Dynamically-set memory target (bytes). The value of this field indicates the current target for memory available to this VM.
metrics	VM_metrics ref	*RO/runtime*	metrics associated with this VM
name_description	string	*RW*	a notes field containing human-readable description
name_label	string	*RW*	a human-readable name
NVRAM	(string -> string) map	*RO/constructor*	initial value for guest NVRAM (containing UEFI variables, etc). Cannot be changed while the VM is running
order	int	*RO/constructor*	The point in the startup or shutdown sequence at which this VM will be started
other_config	(string -> string) map	*RW*	additional configuration
parent	VM ref	*RO/runtime*	Ref pointing to the parent of this VM
PCI_bus	string	*RW*	**Deprecated**. PCI bus path for pass-through devices
platform	(string -> string) map	*RW*	platform-specific configuration
power_state	vm_power_state	*RO/runtime*	Current power state of the machine

Field	Type	Qualifier	Description
protection_policy	VMPP ref	*RO/constructor*	**Deprecated**. Ref pointing to a protection policy for this VM
PV_args	string	*RW*	kernel command-line arguments
PV_bootloader	string	*RW*	name of or path to bootloader
PV_bootloader_args	string	*RW*	miscellaneous arguments for the bootloader
PV_kernel	string	*RW*	path to the kernel
PV_legacy_args	string	*RW*	to make Zurich guests boot
PV_ramdisk	string	*RW*	path to the initrd
recommendations	string	*RW*	An XML specification of recommended values and ranges for properties of this VM
reference_label	string	*RO/constructor*	Textual reference to the template used to create a VM. This can be used by clients in need of an immutable reference to the template since the latter's uuid and name_label may change, for example, after a package installation or upgrade.
requires_reboot	bool	*RO/runtime*	Indicates whether a VM requires a reboot in order to update its configuration, e.g. its memory allocation.
resident_on	host ref	*RO/runtime*	the host the VM is currently resident on
scheduled_to_be_resident_on	host ref	*RO/runtime*	the host on which the VM is due to be started/resumed/migrated. This acts as a memory reservation indicator
shutdown_delay	int	*RO/constructor*	The delay to wait before proceeding to the next order in the shutdown sequence (seconds)
snapshot_info	(string -> string) map	*RO/runtime*	Human-readable information concerning this snapshot
snapshot_metadata	string	*RO/runtime*	Encoded information about the VM's metadata this is a snapshot of

Field	Type	Qualifier	Description
snapshot_of	VM ref	*RO/runtime*	Ref pointing to the VM this snapshot is of.
snapshot_schedule	VMSS ref	*RO/constructor*	Ref pointing to a snapshot schedule for this VM
snapshot_time	datetime	*RO/runtime*	Date/time when this snapshot was created.
snapshots	VM ref set	*RO/runtime*	List pointing to all the VM snapshots.
start_delay	int	*RO/constructor*	The delay to wait before proceeding to the next order in the startup sequence (seconds)
suspend_SR	SR ref	*RW*	The SR on which a suspend image is stored
suspend_VDI	VDI ref	*RO/runtime*	The VDI that a suspend image is stored on. (Only has meaning if VM is currently suspended)
tags	string set	*RW*	user-specified tags for categorization purposes
transportable_snapshot_id	string	*RO/runtime*	Transportable ID of the snapshot VM
user_version	int	*RW*	Creators of VMs and templates may store version information here.
uuid	string	*RO/runtime*	Unique identifier/object reference
VBDs	VBD ref set	*RO/runtime*	virtual block devices
VCPUs_at_startup	int	*RO/constructor*	Boot number of VCPUs
VCPUs_max	int	*RO/constructor*	Max number of VCPUs
VCPUs_params	(string -> string) map	*RW*	configuration parameters for the selected VCPU policy
version	int	*RO/constructor*	The number of times this VM has been recovered
VGPUs	VGPU ref set	*RO/runtime*	Virtual GPUs
VIFs	VIF ref set	*RO/runtime*	virtual network interfaces
VTPMs	VTPM ref set	*RO/runtime*	virtual TPMs
VUSBs	VUSB ref set	*RO/runtime*	vitual usb devices

Field	Type	Qualifier	Description
xenstore_data	`(string -> string) map`	*RW*	data to be inserted into the xenstore tree (/local/domain/<domid>/vm-data) after the VM is created.

RPCs associated with class: VM

RPC name: add_tags

Overview:

Add the given value to the tags field of the given VM. If the value is already in that Set, then do nothing.

Signature:

```
void add_tags (session ref session_id, VM ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	value	New value to add

Minimum Role: vm-operator

Return Type: `void`

RPC name: add_to_blocked_operations

Overview:

Add the given key-value pair to the blocked_operations field of the given VM.

Signature:

```
void add_to_blocked_operations (session ref session_id, VM ref self, vm_operations
key, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

type	name	description
vm_operations	key	Key to add
string	value	Value to add

Minimum Role: vm-admin

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 ref session_id, VM ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: vm-admin

Return Type: void

RPC name: add_to_NVRAM

Overview:

Signature:

```
void add_to_NVRAM (session ref session_id, VM ref self, string key, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM

type	name	description
`string`	key	The key
`string`	value	The value

Minimum Role: vm-admin

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 ref session_id, VM ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: vm-admin

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 ref session_id, VM ref self, string key, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

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

Minimum Role: vm-admin

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 ref session_id, VM ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: vm-admin

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 ref session_id, VM ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
string	key	The key
string	value	The value

Minimum Role: vm-admin

Return Type: void

RPC name: add_to_xenstore_data

Overview:

Add the given key-value pair to the xenstore_data field of the given VM.

Signature:

```
void add_to_xenstore_data (session ref session_id, VM ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: vm-admin

Return Type: void

RPC name: assert_agile

Overview:

Returns an error if the VM is not considered agile e.g. because it is tied to a resource local to a host

Signature:

```
void assert_agile (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM

Minimum Role: read-only

Return Type: void

RPC name: assert_can_be_recovered

Overview:

Assert whether all SRs required to recover this VM are available.

Signature:

```
void assert_can_be_recovered (session ref session_id, VM ref self, session ref
session_to)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM to recover
session ref	session_to	The session to which the VM is to be recovered.

Minimum Role: read-only

Return Type: void

Possible Error Codes: VM_IS_PART_OF_AN_APPLIANCE, VM_REQUIRES_SR

RPC name: assert_can_boot_here

Overview:

Returns an error if the VM could not boot on this host for some reason

Signature:

```
void assert_can_boot_here (session ref session_id, VM ref self, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VM ref	self	The VM
host ref	host	The host

Minimum Role: read-only

Return Type: void

Possible Error Codes: HOST_NOT_ENOUGH_FREE_MEMORY, HOST_NOT_ENOUGH_PCPUS, NETWORK_SRIOV_INSUFFICIENT_CAPACITY, HOST_NOT_LIVE, HOST_DISABLED, HOST_CANNOT_ATTACH_NETWORK, VM_HVM_REQUIRED, VM_REQUIRES_GPU, VM_REQUIRES_IOMMU, VM_REQUIRES_NETWORK, VM_REQUIRES_SR, VM_REQUIRES_VGPU, VM_HOST_INCOMPATIBLE_VERSION, VM_HOST_INCOMPATIBLE_VIRTUAL_HARDWARE_PLATFORM_VERSION, INVALID_VALUE, MEMORY_CONSTRAINT_VIOLATION, OPERATION_NOT_ALLOWED, VALUE_NOT_SUPPORTED, VM_INCOMPATIBLE_WITH_THIS_HOST

RPC name: assert_can_migrate

Overview:

Assert whether a VM can be migrated to the specified destination.

Signature:

```
void assert_can_migrate (session ref session_id, VM ref vm, (string -> string) map
dest, bool live, (VDI ref -> SR ref) map vdi_map, (VIF ref -> network ref) map
vif_map, (string -> string) map options, (VGPU ref -> GPU_group ref) map vgpu_map)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM
(string -> string) map	dest	The result of a VM.migrate_receive call.
bool	live	Live migration
(VDI ref -> SR ref) map	vdi_map	Map of source VDI to destination SR
(VIF ref -> network ref) map	vif_map	Map of source VIF to destination network
(string -> string) map	options	Other parameters
(VGPU ref -> GPU_group ref) map	vgpu_map	Map of source vGPU to destination GPU group

Minimum Role: vm-power-admin

Return Type: void

Possible Error Codes: LICENCE_RESTRICTION

RPC name: assert_operation_valid

Overview:

Check to see whether this operation is acceptable in the current state of the system, raising an error if the operation is invalid for some reason

Signature:

```
void assert_operation_valid (session ref session_id, VM ref self, vm_operations op)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`vm_operations`	op	proposed operation

Minimum Role: read-only

Return Type: `void`

RPC name: call_plugin

Overview:

Call an API plugin on this vm

Signature:

```
string call_plugin (session ref session_id, VM ref vm, string plugin, string fn,
(string -> string) map args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	vm	The vm
`string`	plugin	The name of the plugin
`string`	fn	The name of the function within the plugin
`(string -> string) map`	args	Arguments for the function

Minimum Role: vm-operator

Return Type: `string`

Result from the plugin

RPC name: checkpoint

Overview:

Checkpoints the specified VM, making a new VM. Checkpoint automatically exploits the capabilities of the underlying storage repository in which the VM's disk images are stored (e.g. Copy on Write) and saves the memory image as well.

Signature:

```
VM ref checkpoint (session ref session_id, VM ref vm, string new_name)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	vm	The VM to be checkpointed
`string`	new_name	The name of the checkpointed VM

Minimum Role: vm-power-admin

Return Type: `VM ref`

The reference of the newly created VM.

Possible Error Codes: `VM_BAD_POWER_STATE`, `SR_FULL`, `OPERATION_NOT_ALLOWED`, `VM_CHECKPOINT_SUSPEND_FAILED`, `VM_CHECKPOINT_RESUME_FAILED`

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

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	vm	The VM to shutdown

Minimum Role: vm-operator

Return Type: `void`

Possible Error Codes: `VM_BAD_POWER_STATE`, `OTHER_OPERATION_IN_PROGRESS`, `OPERATION_NOT_ALLOWED`, `VM_IS_TEMPLATE`

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

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	vm	The VM to shutdown

Minimum Role: vm-operator

Return Type: `void`

Possible Error Codes: `VM_BAD_POWER_STATE`, `OTHER_OPERATION_IN_PROGRESS`, `OPERATION_NOT_ALLOWED`, `VM_IS_TEMPLATE`

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 ref session_id, VM ref vm, string new_name)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

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

Minimum Role: vm-admin

Return Type: VM ref

The reference of the newly created VM.

Possible Error Codes: VM_BAD_POWER_STATE, SR_FULL, OPERATION_NOT_ALLOWED, LICENCE_RESTRICTION

RPC name: compute_memory_overhead

Overview:

Computes the virtualization memory overhead of a VM.

Signature:

```
int compute_memory_overhead (session ref session_id, VM ref vm)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM for which to compute the memory overhead

Minimum Role: read-only

Return Type: int

the virtualization memory overhead of the VM.

RPC name: copy

Overview:

Copied the specified VM, making a new VM. Unlike clone, copy does not exploits the capabilities of the underlying storage repository in which the VM's disk images are stored. Instead, copy guarantees that the disk images of the newly created VM will be 'full disks' - i.e. not part of a CoW chain. This function can only be called when the VM is in the Halted State.

Signature:

```
VM ref copy (session ref session_id, VM ref vm, string new_name, SR ref sr)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM to be copied
string	new_name	The name of the copied VM
SR ref	sr	An SR to copy all the VM's disks into (if an invalid reference then it uses the existing SRs)

Minimum Role: vm-admin

Return Type: VM ref

The reference of the newly created VM.

Possible Error Codes: VM_BAD_POWER_STATE, SR_FULL, OPERATION_NOT_ALLOWED, LICENCE_RESTRICTION

RPC name: copy_bios_strings

Overview:

Copy the BIOS strings from the given host to this VM

Signature:

```
void copy_bios_strings (session ref session_id, VM ref vm, host ref host)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM to modify
host ref	host	The host to copy the BIOS strings from

Minimum Role: vm-admin

Return Type: void

RPC name: create

Overview:

NOT RECOMMENDED! VM.clone or VM.copy (or VM.import) is a better choice in almost all situations. The standard way to obtain a new VM is to call VM.clone on a template VM, then call VM.provision on the new clone. Caution: if VM.create is used and then the new VM is attached to a virtual disc that has an operating system already installed, then there is no guarantee that the operating system will boot and run. Any software that calls VM.create on a future version of this API may fail or give unexpected results. For example this could happen if an

additional parameter were added to VM.create. VM.create is intended only for use in the automatic creation of the system VM templates. It creates a new VM instance, and returns its handle.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM record`	args	All constructor arguments

Minimum Role: vm-admin

Return Type: `VM ref`

reference to the newly created object

RPC name: create_new_blob

Overview:

Create a placeholder for a named binary blob of data that is associated with this VM

Signature:

```
blob ref create_new_blob (session ref session_id, VM ref vm, string name, string
mime_type, bool public)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	vm	The VM
`string`	name	The name associated with the blob
`string`	mime_type	The mime type for the data. Empty string translates to application/octet-stream
`bool`	public	True if the blob should be publicly available

Minimum Role: vm-power-admin

Return Type: `blob ref`

The reference of the blob, needed for populating its data

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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: vm-admin

Return Type: `void`

RPC name: forget_data_source_archives

Overview:

Forget the recorded statistics related to the specified data source

Signature:

```
void forget_data_source_archives (session ref session_id, VM ref self, string
data_source)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM
`string`	data_source	The data source whose archives are to be forgotten

Minimum Role: vm-admin

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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: on_crash_behaviour

value of the field

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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: on_normal_exit

value of the field

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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `on_normal_exit`

value of the field

RPC name: get_affinity

Overview:

Get the affinity field of the given VM.

Signature:

```
host ref get_affinity (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_all

Overview:

Return a list of all the VMs known to the system.

Signature:

```
VM ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `VM ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of VM references to VM records for all VMs known to the system.

Signature:

```
(VM ref -> VM record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VM ref -> VM record) map`

records of all objects

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given VM.

Signature:

```
vm_operations set get_allowed_operations (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vm_operations set`

value of the field

RPC name: get_allowed_VBD_devices

Overview:

Returns a list of the allowed values that a VBD device field can take

Signature:

```
string set get_allowed_VBD_devices (session ref session_id, VM ref vm)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	vm	The VM to query

Minimum Role: read-only

Return Type: `string set`

The allowed values

RPC name: get_allowed_VIF_devices

Overview:

Returns a list of the allowed values that a VIF device field can take

Signature:

```
string set get_allowed_VIF_devices (session ref session_id, VM ref vm)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	vm	The VM to query

Minimum Role: read-only

Return Type: `string set`

The allowed values

RPC name: get_appliance

Overview:

Get the appliance field of the given VM.

Signature:

```
VM_appliance ref get_appliance (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: VM_appliance ref

value of the field

RPC name: get_attached_PCIs

Overview:

Get the attached_PCIs field of the given VM.

Signature:

```
PCI ref set get_attached_PCIs (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: PCI ref set

value of the field

RPC name: get_bios_strings

Overview:

Get the bios_strings field of the given VM.

Signature:

```
(string -> string) map get_bios_strings (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_blobs

Overview:

Get the blobs field of the given VM.

Signature:

```
(string -> blob ref) map get_blobs (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> blob ref) map`

value of the field

RPC name: get_blocked_operations

Overview:

Get the blocked_operations field of the given VM.

Signature:

```
(vm_operations -> string) map get_blocked_operations (session ref session_id, VM ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `(vm_operations -> string) map`

value of the field

RPC name: get_boot_record

This message is deprecated.

Overview:

Returns a record describing the VM's dynamic state, initialised when the VM boots and updated to reflect runtime configuration changes e.g. CPU hotplug

Signature:

```
VM record get_boot_record (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM whose boot-time state to return

Minimum Role: read-only

Return Type: `VM record`

A record describing the VM

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 ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `VM ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the VM instance with the specified UUID.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VM ref`

reference to the object

RPC name: get_children

Overview:

Get the children field of the given VM.

Signature:

```
VM ref set get_children (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref set`

value of the field

RPC name: get_consoles

Overview:

Get the consoles field of the given VM.

Signature:

```
console ref set get_consoles (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `console ref set`

value of the field

RPC name: get_cooperative

This message is deprecated.

Overview:

Return true if the VM is currently 'co-operative' i.e. is expected to reach a balloon target and actually has done

Signature:

```
bool get_cooperative (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM

Minimum Role: read-only

Return Type: `bool`

true if the VM is currently 'co-operative'; false otherwise

RPC name: get_crash_dumps

Overview:

Get the crash_dumps field of the given VM.

Signature:

```
crashdump ref set get_crash_dumps (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `crashdump ref set`

value of the field

RPC name: get_current_operations

Overview:

Get the current_operations field of the given VM.

Signature:

```
(string -> vm_operations) map get_current_operations (session ref session_id, VM ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> vm_operations) map`

value of the field

RPC name: get_data_sources

Overview:

Signature:

```
data_source record set get_data_sources (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM to interrogate

Minimum Role: read-only

Return Type: `data_source record set`

A set of data sources

RPC name: get_domain_type

Overview:

Get the domain_type field of the given VM.

Signature:

```
domain_type get_domain_type (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `domain_type`

value of the field

RPC name: get_domarch

Overview:

Get the domarch field of the given VM.

Signature:

```
string get_domarch (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_domid

Overview:

Get the domid field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_generation_id

Overview:

Get the generation_id field of the given VM.

Signature:

```
string get_generation_id (session ref session_id, VM ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `string`

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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `VM_guest_metrics ref`

value of the field

RPC name: get_ha_always_run

This message is deprecated.

Overview:

Get the ha_always_run field of the given VM.

Signature:

```
bool get_ha_always_run (session ref session_id, VM ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: bool

value of the field

RPC name: get_ha_restart_priority

Overview:

Get the ha_restart_priority field of the given VM.

Signature:

```
string get_ha_restart_priority (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: string

value of the field

RPC name: get_hardware_platform_version

Overview:

Get the hardware_platform_version field of the given VM.

Signature:

```
int get_hardware_platform_version (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: int

value of the field

RPC name: get_has_vendor_device

Overview:

Get the has_vendor_device field of the given VM.

Signature:

```
bool get_has_vendor_device (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: bool

value of the field

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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_HVM_boot_policy

This message is deprecated.

Overview:

Get the HVM/boot_policy field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_HVM_shadow_multiplier

Overview:

Get the HVM/shadow_multiplier field of the given VM.

Signature:

```
float get_HVM_shadow_multiplier (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `float`

value of the field

RPC name: get_is_a_snapshot

Overview:

Get the is_a_snapshot field of the given VM.

Signature:

```
bool get_is_a_snapshot (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_is_a_template

Overview:

Get the is_a_template field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: bool

value of the field

RPC name: get_is_default_template

Overview:

Get the is_default_template field of the given VM.

Signature:

```
bool get_is_default_template (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: bool

value of the field

RPC name: get_is_snapshot_from_vmpp

This message is deprecated.

Overview:

Get the is_snapshot_from_vmpp field of the given VM.

Signature:

```
bool get_is_snapshot_from_vmpp (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_is_vmss_snapshot

Overview:

Get the is_vmss_snapshot field of the given VM.

Signature:

```
bool get_is_vmss_snapshot (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_last_boot_CPU_flags

Overview:

Get the last_boot_CPU_flags field of the given VM.

Signature:

```
(string -> string) map get_last_boot_CPU_flags (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_last_booted_record

Overview:

Get the last_booted_record field of the given VM.

Signature:

```
string get_last_booted_record (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_memory_dynamic_max

Overview:

Get the memory/dynamic_max field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_memory_dynamic_min

Overview:

Get the memory/dynamic_min field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_memory_overhead

Overview:

Get the memory/overhead field of the given VM.

Signature:

```
int get_memory_overhead (session ref session_id, VM ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_memory_static_min

Overview:

Get the memory/static_min field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: int

value of the field

RPC name: get_memory_target

This message is deprecated.

Overview:

Get the memory/target field of the given VM.

Signature:

```
int get_memory_target (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: int

value of the field

RPC name: get_metrics

Overview:

Get the metrics field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: VM_metrics ref

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: string

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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_NVRAM

Overview:

Get the NVRAM field of the given VM.

Signature:

```
(string -> string) map get_NVRAM (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_order

Overview:

Get the order field of the given VM.

Signature:

```
int get_order (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given VM.

Signature:

```
(string -> string) map get_other_config (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_parent

Overview:

Get the parent field of the given VM.

Signature:

```
VM ref get_parent (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref`

value of the field

RPC name: get_PCI_bus

This message is deprecated.

Overview:

Get the PCI_bus field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_platform

Overview:

Get the platform field of the given VM.

Signature:

```
(string -> string) map get_platform (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_possible_hosts

Overview:

Return the list of hosts on which this VM may run.

Signature:

```
host ref set get_possible_hosts (session ref session_id, VM ref vm)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM

Minimum Role: read-only

Return Type: `host ref set`

The possible hosts

RPC name: get_power_state

Overview:

Get the power_state field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `vm_power_state`

value of the field

RPC name: get_protection_policy

This message is deprecated.

Overview:

Get the protection_policy field of the given VM.

Signature:

```
VMPP ref get_protection_policy (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: VMPP ref

value of the field

RPC name: get_PV_args

Overview:

Get the PV/args field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: string

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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_PV_bootloader_args

Overview:

Get the PV/bootloader_args field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_PV_kernel

Overview:

Get the PV/kernel field of the given VM.

Signature:

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

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_PV_legacy_args

Overview:

Get the PV/legacy_args field of the given VM.

Signature:

```
string get_PV_legacy_args (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_PV_ramdisk

Overview:

Get the PV/ramdisk field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_recommendations

Overview:

Get the recommendations field of the given VM.

Signature:

```
string get_recommendations (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given VM.

Signature:

```
VM record get_record (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `VM record`

all fields from the object

RPC name: get_reference_label

Overview:

Get the reference_label field of the given VM.

Signature:

```
string get_reference_label (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_requires_reboot

Overview:

Get the requires_reboot field of the given VM.

Signature:

```
bool get_requires_reboot (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_scheduled_to_be_resident_on

Overview:

Get the scheduled_to_be_resident_on field of the given VM.

Signature:

```
host ref get_scheduled_to_be_resident_on (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `host ref`

value of the field

RPC name: get_shutdown_delay

Overview:

Get the shutdown_delay field of the given VM.

Signature:

```
int get_shutdown_delay (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_snapshot_info

Overview:

Get the snapshot_info field of the given VM.

Signature:

```
(string -> string) map get_snapshot_info (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_snapshot_metadata

Overview:

Get the snapshot_metadata field of the given VM.

Signature:

```
string get_snapshot_metadata (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_snapshot_of

Overview:

Get the snapshot_of field of the given VM.

Signature:

```
VM ref get_snapshot_of (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref`

value of the field

RPC name: get_snapshot_schedule

Overview:

Get the snapshot_schedule field of the given VM.

Signature:

```
VMSS ref get_snapshot_schedule (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VMSS ref`

value of the field

RPC name: get_snapshot_time

Overview:

Get the snapshot_time field of the given VM.

Signature:

```
datetime get_snapshot_time (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_snapshots

Overview:

Get the snapshots field of the given VM.

Signature:

```
VM ref set get_snapshots (session ref session_id, VM ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref set`

value of the field

RPC name: get_SRs_required_for_recovery

Overview:

List all the SR's that are required for the VM to be recovered

Signature:

```
SR ref set get_SRs_required_for_recovery (session ref session_id, VM ref self, session
ref session_to)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM for which the SRs have to be recovered
`session ref`	session_to	The session to which the SRs of the VM have to be recovered.

Minimum Role: read-only

Return Type: `SR ref set`

refs for SRs required to recover the VM

RPC name: get_start_delay

Overview:

Get the start_delay field of the given VM.

Signature:

```
int get_start_delay (session ref session_id, VM ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_suspend_SR

Overview:

Get the suspend_SR field of the given VM.

Signature:

```
SR ref get_suspend_SR (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `SR ref`

value of the field

RPC name: get_suspend_VDI

Overview:

Get the suspend_VDI field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: VDI ref

value of the field

RPC name: get_tags

Overview:

Get the tags field of the given VM.

Signature:

```
string set get_tags (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: string set

value of the field

RPC name: get_transportable_snapshot_id

Overview:

Get the transportable_snapshot_id field of the given VM.

Signature:

```
string get_transportable_snapshot_id (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_user_version

Overview:

Get the user_version field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VBDs

Overview:

Get the VBDs field of the given VM.

Signature:

```
VBD ref set get_VBDs (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VBD ref set`

value of the field

RPC name: get_VCPUs_at_startup

Overview:

Get the VCPUs/at_startup field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_VCPUs_max

Overview:

Get the VCPUs/max field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_VCPUs_params

Overview:

Get the VCPUs/params field of the given VM.

Signature:

```
(string -> string) map get_VCPUs_params (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_version

Overview:

Get the version field of the given VM.

Signature:

```
int get_version (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_VGPUs

Overview:

Get the VGPUs field of the given VM.

Signature:

```
VGPU ref set get_VGPUs (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VGPU 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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VIF 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 ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VTPM ref set`

value of the field

RPC name: get_VUSBs

Overview:

Get the VUSBs field of the given VM.

Signature:

```
VUSB ref set get_VUSBs (session ref session_id, VM ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: VUSB ref set

value of the field

RPC name: get_xenstore_data

Overview:

Get the xenstore_data field of the given VM.

Signature:

```
(string -> string) map get_xenstore_data (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

Minimum Role: read-only

Return Type: (string -> string) map

value of the field

RPC name: hard_reboot

Overview:

Stop executing the specified VM without attempting a clean shutdown and immediately restart the VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VM ref	vm	The VM to reboot

Minimum Role: vm-operator

Return Type: `void`

Possible Error Codes: `VM_BAD_POWER_STATE`, `OTHER_OPERATION_IN_PROGRESS`, `OPERATION_NOT_ALLOWED`, `VM_IS_TEMPLATE`

RPC name: hard_shutdown

Overview:

Stop executing the specified VM without attempting a clean shutdown.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM to destroy

Minimum Role: vm-operator

Return Type: `void`

Possible Error Codes: `VM_BAD_POWER_STATE`, `OTHER_OPERATION_IN_PROGRESS`, `OPERATION_NOT_ALLOWED`, `VM_IS_TEMPLATE`

RPC name: import

Overview:

Import an XVA from a URI

Signature:

```
VM ref set import (session ref session_id, string url, SR ref sr, bool full_restore,
bool force)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`string`	url	The URL of the XVA file
`SR ref`	sr	The destination SR for the disks
`bool`	full_restore	Perform a full restore
`bool`	force	Force the import

Minimum Role: pool-operator

Return Type: `VM ref set`

Imported VM reference

RPC name: import_convert

Overview:

Import using a conversion service.

Signature:

```
void import_convert (session ref session_id, string type, string username, string
password, SR ref sr, (string -> string) map remote_config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	type	Type of the conversion
`string`	username	Admin username on the host
`string`	password	Password on the host
`SR ref`	sr	The destination SR
`(string -> string) map`	remote_config	Remote configuration options

Minimum Role: vm-admin

Return Type: `void`

RPC name: maximise_memory

Overview:

Returns the maximum amount of guest memory which will fit, together with overheads, in the supplied amount of physical memory. If 'exact' is true then an exact calculation is performed using the VM's current settings. If 'exact'

is false then a more conservative approximation is used

Signature:

```
int maximise_memory (session ref session_id, VM ref self, int total, bool approximate)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
int	total	Total amount of physical RAM to fit within
bool	approximate	If false the limit is calculated with the guest's current exact configuration. Otherwise a more approximate calculation is performed

Minimum Role: read-only

Return Type: int

The maximum possible static-max

RPC name: migrate_send

Overview:

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

Signature:

```
VM ref migrate_send (session ref session_id, VM ref vm, (string -> string) map dest,
bool live, (VDI ref -> SR ref) map vdi_map, (VIF ref -> network ref) map vif_map,
(string -> string) map options, (VGPU ref -> GPU_group ref) map vgpu_map)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM
(string -> string) map	dest	The result of a Host.migrate_receive call.
bool	live	Live migration
(VDI ref -> SR ref) map	vdi_map	Map of source VDI to destination SR
(VIF ref -> network ref) map	vif_map	Map of source VIF to destination network

type	name	description
`(string -> string) map`	options	Other parameters
`(VGPU ref -> GPU_group ref) map`	vgpu_map	Map of source vGPU to destination GPU group

Minimum Role: vm-power-admin

Return Type: `VM ref`

The reference of the newly created VM in the destination pool

Possible Error Codes: `VM_BAD_POWER_STATE`, `LICENCE_RESTRICTION`

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 ref session_id, VM ref vm)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	vm	The VM to pause

Minimum Role: vm-operator

Return Type: `void`

Possible Error Codes: `VM_BAD_POWER_STATE`, `OTHER_OPERATION_IN_PROGRESS`, `OPERATION_NOT_ALLOWED`, `VM_IS_TEMPLATE`

RPC name: pool_migrate

Overview:

Migrate a VM to another Host.

Signature:

```
void pool_migrate (session ref session_id, VM ref vm, host ref host, (string ->
string) map options)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM to migrate
host ref	host	The target host
(string -> string) map	options	Extra configuration operations

Minimum Role: vm-power-admin

Return Type: void

Possible Error Codes: VM_BAD_POWER_STATE, OTHER_OPERATION_IN_PROGRESS, VM_IS_TEMPLATE, OPERATION_NOT_ALLOWED, VM_MIGRATE_FAILED

RPC name: power_state_reset

Overview:

Reset the power-state of the VM to halted in the database only. (Used to recover from slave failures in pooling scenarios by resetting the power-states of VMs running on dead slaves to halted.) This is a potentially dangerous operation; use with care.

Signature:

```
void power_state_reset (session ref session_id, VM ref vm)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM to reset

Minimum Role: pool-operator

Return Type: void

RPC name: provision

Overview:

Inspects the disk configuration contained within the VM's other_config, creates VDIs and VBDs and then executes any applicable post-install script.

Signature:

```
void provision (session ref session_id, VM ref vm)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	vm	The VM to be provisioned

Minimum Role: vm-admin

Return Type: `void`

Possible Error Codes: `VM_BAD_POWER_STATE`, `SR_FULL`, `OPERATION_NOT_ALLOWED`, `LICENCE_RESTRICTION`

RPC name: query_data_source

Overview:

Query the latest value of the specified data source

Signature:

```
float query_data_source (session ref session_id, VM ref self, string data_source)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM
`string`	data_source	The data source to query

Minimum Role: read-only

Return Type: `float`

The latest value, averaged over the last 5 seconds

RPC name: query_services

Overview:

Query the system services advertised by this VM and register them. This can only be applied to a system domain.

Signature:

```
(string -> string) map query_services (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM

Minimum Role: pool-admin

Return Type: `(string -> string) map`

map of service type to name

RPC name: record_data_source

Overview:

Start recording the specified data source

Signature:

```
void record_data_source (session ref session_id, VM ref self, string data_source)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
string	data_source	The data source to record

Minimum Role: vm-admin

Return Type: `void`

RPC name: recover

Overview:

Recover the VM

Signature:

```
void recover (session ref session_id, VM ref self, session ref session_to, bool force)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VM ref	self	The VM to recover
session ref	session_to	The session to which the VM is to be recovered.
bool	force	Whether the VM should replace newer versions of itself.

Minimum Role: read-only

Return Type: void

RPC name: remove_from_blocked_operations

Overview:

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

Signature:

```
void remove_from_blocked_operations (session ref session_id, VM ref self,
vm_operations key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object
vm_operations	key	Key to remove

Minimum Role: vm-admin

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 ref session_id, VM ref self, string key)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object
string	key	Key to remove

Minimum Role: vm-admin

Return Type: void

RPC name: remove_from_NVRAM

Overview:

Signature:

```
void remove_from_NVRAM (session ref session_id, VM ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
string	key	The key

Minimum Role: vm-admin

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 ref session_id, VM ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object

type	name	description
`string`	key	Key to remove

Minimum Role: vm-admin

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 ref session_id, VM ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: vm-admin

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 ref session_id, VM ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object

type	name	description
`string`	key	Key to remove

Minimum Role: vm-admin

Return Type: `void`

RPC name: remove_from_xenstore_data

Overview:

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

Signature:

```
void remove_from_xenstore_data (session ref session_id, VM ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: vm-admin

Return Type: `void`

RPC name: remove_tags

Overview:

Remove the given value from the tags field of the given VM. If the value is not in that Set, then do nothing.

Signature:

```
void remove_tags (session ref session_id, VM ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	value	Value to remove

Minimum Role: vm-operator

Return Type: `void`

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 ref session_id, VM ref vm, bool start_paused, bool force)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	vm	The VM to resume
`bool`	start_paused	Resume VM in paused state if set to true.
`bool`	force	Attempt to force the VM to resume. If this flag is false then the VM may fail pre-resume safety checks (e.g. if the CPU the VM was running on looks substantially different to the current one)

Minimum Role: vm-operator

Return Type: `void`

Possible Error Codes: `VM_BAD_POWER_STATE`, `OPERATION_NOT_ALLOWED`, `VM_IS_TEMPLATE`

RPC name: resume_on

Overview:

Awaken the specified VM and resume it on a particular Host. This can only be called when the specified VM is in the Suspended state.

Signature:

```
void resume_on (session ref session_id, VM ref vm, host ref host, bool start_paused,
bool force)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM to resume
host ref	host	The Host on which to resume the VM
bool	start_paused	Resume VM in paused state if set to true.
bool	force	Attempt to force the VM to resume. If this flag is false then the VM may fail pre-resume safety checks (e.g. if the CPU the VM was running on looks substantially different to the current one)

Minimum Role: vm-power-admin

Return Type: `void`

Possible Error Codes: `VM_BAD_POWER_STATE`, `OPERATION_NOT_ALLOWED`, `VM_IS_TEMPLATE`

RPC name: retrieve_wlb_recommendations

Overview:

Returns mapping of hosts to ratings, indicating the suitability of starting the VM at that location according to wlb. Rating is replaced with an error if the VM cannot boot there.

Signature:

```
(host ref -> string set) map retrieve_wlb_recommendations (session ref session_id, VM ref vm)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM

Minimum Role: read-only

Return Type: `(host ref -> string set) map`

The potential hosts and their corresponding recommendations or errors

RPC name: revert

Overview:

Reverts the specified VM to a previous state.

Signature:

```
void revert (session ref session_id, VM ref snapshot)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	snapshot	The snapshotted state that we revert to

Minimum Role: vm-power-admin

Return Type: void

Possible Error Codes: VM_BAD_POWER_STATE, OPERATION_NOT_ALLOWED, SR_FULL, VM_REVERT_FAILED

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 ref session_id, VM ref vm, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM
string	key	The key to send

Minimum Role: pool-admin

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 ref session_id, VM ref vm, string trigger)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM
string	trigger	The trigger to send

Minimum Role: pool-admin

Return Type: void

Possible Error Codes: VM_BAD_POWER_STATE

RPC name: set_actions_after_crash

Overview:

Sets the actions_after_crash parameter

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM to set
on_crash_behaviour	value	The new value to set

Minimum Role: vm-admin

Return Type: void

RPC name: set_actions_after_reboot

Overview:

Set the actions/after_reboot field of the given VM.

Signature:

```

```

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`on_normal_exit`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_actions_after_shutdown

Overview:

Set the actions/after_shutdown field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`on_normal_exit`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_affinity

Overview:

Set the affinity field of the given VM.

Signature:

```
void set_affinity (session ref session_id, VM ref self, host ref value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`host ref`	value	New value to set

Minimum Role: vm-power-admin

Return Type: `void`

RPC name: set_appliance

Overview:

Assign this VM to an appliance.

Signature:

```
void set_appliance (session ref session_id, VM ref self, VM_appliance ref value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM to assign to an appliance.
`VM_appliance ref`	value	The appliance to which this VM should be assigned.

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_bios_strings

Overview:

Set custom BIOS strings to this VM. VM will be given a default set of BIOS strings, only some of which can be overridden by the supplied values. Allowed keys are: 'bios-vendor', 'bios-version', 'system-manufacturer', 'system-product-name', 'system-version', 'system-serial-number', 'enclosure-asset-tag', 'baseboard-manufacturer', 'baseboard-product-name', 'baseboard-version', 'baseboard-serial-number', 'baseboard-asset-tag', 'baseboard-location-in-chassis', 'enclosure-asset-tag'

Signature:

```
void set_bios_strings (session ref session_id, VM ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM to modify
`(string -> string) map`	value	The custom BIOS strings as a list of key-value pairs

Minimum Role: vm-admin

Return Type: `void`

Possible Error Codes: `VM_BIOS_STRINGS_ALREADY_SET`, `INVALID_VALUE`

RPC name: set_blocked_operations

Overview:

Set the blocked_operations field of the given VM.

Signature:

```
void set_blocked_operations (session ref session_id, VM ref self, (vm_operations ->
string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`(vm_operations -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_domain_type

Overview:

Set the VM.domain_type field of the given VM, which will take effect when it is next started

Signature:

```
void set_domain_type (session ref session_id, VM ref self, domain_type value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
domain_type	value	The new domain type

Minimum Role: vm-admin

Return Type: void

RPC name: set_ha_always_run

This message is deprecated.

Overview:

Set the value of the ha_always_run

Signature:

```
void set_ha_always_run (session ref session_id, VM ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
bool	value	The value

Minimum Role: pool-operator

Return Type: void

RPC name: set_ha_restart_priority

Overview:

Set the value of the ha_restart_priority field

Signature:

```
void set_ha_restart_priority (session ref session_id, VM ref self, string value)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
string	value	The value

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_hardware_platform_version

Overview:

Set the hardware_platform_version field of the given VM.

Signature:

```
void set_hardware_platform_version (session ref session_id, VM ref self, int value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object
int	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_has_vendor_device

Overview:

Controls whether, when the VM starts in HVM mode, its virtual hardware will include the emulated PCI device for which drivers may be available through Windows Update. Usually this should never be changed on a VM on which Windows has been installed: changing it on such a VM is likely to lead to a crash on next start.

Signature:

```
void set_has_vendor_device (session ref session_id, VM ref self, bool value)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM on which to set this flag
bool	value	True to provide the vendor PCI device.

Minimum Role: vm-admin

Return Type: void

RPC name: set_HVM_boot_params

Overview:

Set the HVM/boot_params field of the given VM.

Signature:

```
void set_HVM_boot_params (session ref session_id, VM ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object
(string -> string) map	value	New value to set

Minimum Role: vm-admin

Return Type: void

RPC name: set_HVM_boot_policy

This message is deprecated.

Overview:

Set the VM.HVM_boot_policy field of the given VM, which will take effect when it is next started

Signature:

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

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
string	value	The new HVM boot policy

Minimum Role: vm-admin

Return Type: void

RPC name: set_HVM_shadow_multiplier

Overview:

Set the shadow memory multiplier on a halted VM

Signature:

```
void set_HVM_shadow_multiplier (session ref session_id, VM ref self, float value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
float	value	The new shadow memory multiplier to set

Minimum Role: vm-power-admin

Return Type: void

RPC name: set_is_a_template

Overview:

Set the is_a_template field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

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

Minimum Role: vm-admin

Return Type: void

RPC name: set_memory

Overview:

Set the memory allocation of this VM. Sets all of memory_static_max, memory_dynamic_min, and memory_dynamic_max to the given value, and leaves memory_static_min untouched.

Signature:

```
void set_memory (session ref session_id, VM ref self, int value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
int	value	The new memory allocation (bytes).

Minimum Role: vm-power-admin

Return Type: void

RPC name: set_memory_dynamic_max

Overview:

Set the value of the memory_dynamic_max field

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM to modify

type	name	description
int	value	The new value of memory_dynamic_max

Minimum Role: vm-power-admin

Return Type: void

RPC name: set_memory_dynamic_min

Overview:

Set the value of the memory_dynamic_min field

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM to modify
int	value	The new value of memory_dynamic_min

Minimum Role: vm-power-admin

Return Type: void

RPC name: set_memory_dynamic_range

Overview:

Set the minimum and maximum amounts of physical memory the VM is allowed to use.

Signature:

```
void set_memory_dynamic_range (session ref session_id, VM ref self, int min, int max)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
int	min	The new minimum value

type	name	description
int	max	The new maximum value

Minimum Role: vm-power-admin

Return Type: void

RPC name: set_memory_limits

Overview:

Set the memory limits of this VM.

Signature:

```
void set_memory_limits (session ref session_id, VM ref self, int static_min, int
static_max, int dynamic_min, int dynamic_max)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
int	static_min	The new value of memory_static_min.
int	static_max	The new value of memory_static_max.
int	dynamic_min	The new value of memory_dynamic_min.
int	dynamic_max	The new value of memory_dynamic_max.

Minimum Role: vm-power-admin

Return Type: void

RPC name: set_memory_static_max

Overview:

Set the value of the memory_static_max field

Signature:

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

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM to modify
int	value	The new value of memory_static_max

Minimum Role: vm-power-admin

Return Type: void

Possible Error Codes: HA_OPERATION_WOULD_BREAK_FAILOVER_PLAN

RPC name: set_memory_static_min

Overview:

Set the value of the memory_static_min field

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM to modify
int	value	The new value of memory_static_min

Minimum Role: vm-power-admin

Return Type: void

RPC name: set_memory_static_range

Overview:

Set the static (ie boot-time) range of virtual memory that the VM is allowed to use.

Signature:

```
void set_memory_static_range (session ref session_id, VM ref self, int min, int max)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
int	min	The new minimum value
int	max	The new maximum value

Minimum Role: vm-power-admin

Return Type: void

RPC name: set_memory_target_live

This message is deprecated.

Overview:

Set the memory target for a running VM

Signature:

```
void set_memory_target_live (session ref session_id, VM ref self, int target)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
int	target	The target in bytes

Minimum Role: vm-power-admin

Return Type: void

RPC name: set_name_description

Overview:

Set the name/description field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object
string	value	New value to set

Minimum Role: vm-admin

Return Type: void

RPC name: set_name_label

Overview:

Set the name/label field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object
string	value	New value to set

Minimum Role: vm-admin

Return Type: void

RPC name: set_NVRAM

Overview:

Signature:

```
void set_NVRAM (session ref session_id, VM ref self, (string -> string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
(string -> string) map	value	The value

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_order

Overview:

Set this VM's boot order

Signature:

```
void set_order (session ref session_id, VM ref self, int value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM
`int`	value	This VM's boot order

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given VM.

Signature:

```
void set_other_config (session ref session_id, VM ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_PCI_bus

This message is deprecated.

Overview:

Set the PCI_bus field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_platform

Overview:

Set the platform field of the given VM.

Signature:

```
void set_platform (session ref session_id, VM ref self, (string -> string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_protection_policy

Overview:

Set the value of the protection_policy field

Signature:

```
void set_protection_policy (session ref session_id, VM ref self, VMPP ref value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM
`VMPP ref`	value	The value

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_PV_args

Overview:

Set the PV/args field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_PV_bootloader

Overview:

Set the PV/bootloader field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_PV_bootloader_args

Overview:

Set the PV/bootloader_args field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_PV_kernel

Overview:

Set the PV/kernel field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_PV_legacy_args

Overview:

Set the PV/legacy_args field of the given VM.

Signature:

```
void set_PV_legacy_args (session ref session_id, VM ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_PV_ramdisk

Overview:

Set the PV/ramdisk field of the given VM.

Signature:

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

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_recommendations

Overview:

Set the recommendations field of the given VM.

Signature:

```
void set_recommendations (session ref session_id, VM ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_shadow_multiplier_live

Overview:

Set the shadow memory multiplier on a running VM

Signature:

```
void set_shadow_multiplier_live (session ref session_id, VM ref self, float
multiplier)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VM ref`	self	The VM
`float`	multiplier	The new shadow memory multiplier to set

Minimum Role: vm-power-admin

Return Type: `void`

RPC name: set_shutdown_delay

Overview:

Set this VM's shutdown delay in seconds

Signature:

```
void set_shutdown_delay (session ref session_id, VM ref self, int value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM
`int`	value	This VM's shutdown delay in seconds

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_snapshot_schedule

Overview:

Set the value of the snapshot schedule field

Signature:

```
void set_snapshot_schedule (session ref session_id, VM ref self, VMSS ref value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM

type	name	description
VMSS ref	value	The value

Minimum Role: pool-operator

Return Type: void

RPC name: set_start_delay

Overview:

Set this VM's start delay in seconds

Signature:

```
void set_start_delay (session ref session_id, VM ref self, int value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
int	value	This VM's start delay in seconds

Minimum Role: pool-operator

Return Type: void

RPC name: set_suspend_SR

Overview:

Set the suspend_SR field of the given VM.

Signature:

```
void set_suspend_SR (session ref session_id, VM ref self, SR ref value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object
SR ref	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_suspend_VDI

Overview:

Set this VM's suspend VDI, which must be indentical to its current one

Signature:

```
void set_suspend_VDI (session ref session_id, VM ref self, VDI ref value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM
`VDI ref`	value	The suspend VDI uuid

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_tags

Overview:

Set the tags field of the given VM.

Signature:

```
void set_tags (session ref session_id, VM ref self, string set value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`string set`	value	New value to set

Minimum Role: vm-operator

Return Type: `void`

RPC name: set_user_version

Overview:

Set the user_version field of the given VM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	reference to the object
int	value	New value to set

Minimum Role: vm-admin

Return Type: void

RPC name: set_VCPUs_at_startup

Overview:

Set the number of startup VCPUs for a halted VM

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM
int	value	The new maximum number of VCPUs

Minimum Role: vm-admin

Return Type: void

RPC name: set_VCPUs_max

Overview:

Set the maximum number of VCPUs for a halted VM

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM
`int`	value	The new maximum number of VCPUs

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_VCPUs_number_live

Overview:

Set the number of VCPUs for a running VM

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	The VM
`int`	nvcpu	The number of VCPUs

Minimum Role: vm-admin

Return Type: `void`

Possible Error Codes: `OPERATION_NOT_ALLOWED`, `LICENCE_RESTRICTION`

RPC name: set_VCPUs_params

Overview:

Set the VCPUs/params field of the given VM.

Signature:

```
void set_VCPUs_params (session ref session_id, VM ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_xenstore_data

Overview:

Set the xenstore_data field of the given VM.

Signature:

```
void set_xenstore_data (session ref session_id, VM ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

RPC name: shutdown

Overview:

Attempts to first clean shutdown a VM and if it should fail then perform a hard shutdown on it.

Signature:

```
void shutdown (session ref session_id, VM ref vm)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM to shutdown

Minimum Role: vm-operator

Return Type: void

Possible Error Codes: VM_BAD_POWER_STATE, OTHER_OPERATION_IN_PROGRESS, OPERATION_NOT_ALLOWED, VM_IS_TEMPLATE

RPC name: snapshot

Overview:

Snapshots the specified VM, making a new VM. Snapshot automatically exploits the capabilities of the underlying storage repository in which the VM's disk images are stored (e.g. Copy on Write).

Signature:

```
VM ref snapshot (session ref session_id, VM ref vm, string new_name)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM to be snapshotted
string	new_name	The name of the snapshotted VM

Minimum Role: vm-power-admin

Return Type: VM ref

The reference of the newly created VM.

Possible Error Codes: VM_BAD_POWER_STATE, SR_FULL, OPERATION_NOT_ALLOWED

RPC name: snapshot_with_quiesce

This message is removed.

Overview:

Snapshots the specified VM with quiesce, making a new VM. Snapshot automatically exploits the capabilities of the underlying storage repository in which the VM's disk images are stored (e.g. Copy on Write).

Signature:

```
VM ref snapshot_with_quiesce (session ref session_id, VM ref vm, string new_name)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM to be snapshotted
string	new_name	The name of the snapshotted VM

Minimum Role: vm-power-admin

Return Type: VM ref

The reference of the newly created VM.

Possible Error Codes: VM_BAD_POWER_STATE, SR_FULL, OPERATION_NOT_ALLOWED, VM_SNAPSHOT_WITH_QUIESCE_FAILED, VM_SNAPSHOT_WITH_QUIESCE_TIMEOUT, VM_SNAPSHOT_WITH_QUIESCE_PLUGIN_DEOS_NOT_RESPOND, VM_SNAPSHOT_WITH_QUIESCE_NOT_SUPPORTED

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 ref session_id, VM ref vm, bool start_paused, bool force)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM to start
bool	start_paused	Instantiate VM in paused state if set to true.
bool	force	Attempt to force the VM to start. If this flag is false then the VM may fail pre-boot safety checks (e.g. if the CPU the VM last booted on looks substantially different to the current one)

Minimum Role: vm-operator

Return Type: void

Possible Error Codes: `VM_BAD_POWER_STATE`, `VM_HVM_REQUIRED`, `VM_IS_TEMPLATE`, `OTHER_OPERATION_IN_PROGRESS`, `OPERATION_NOT_ALLOWED`, `BOOTLOADER_FAILED`, `UNKNOWN_BOOTLOADER`, `NO_HOSTS_AVAILABLE`, `LICENCE_RESTRICTION`

RPC name: start_on

Overview:

Start the specified VM on a particular host. This function can only be called with the VM is in the Halted State.

Signature:

```
void start_on (session ref session_id, VM ref vm, host ref host, bool start_paused,
bool force)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	vm	The VM to start
`host ref`	host	The Host on which to start the VM
`bool`	start_paused	Instantiate VM in paused state if set to true.
`bool`	force	Attempt to force the VM to start. If this flag is false then the VM may fail pre-boot safety checks (e.g. if the CPU the VM last booted on looks substantially different to the current one)

Minimum Role: vm-power-admin

Return Type: `void`

Possible Error Codes: `VM_BAD_POWER_STATE`, `VM_IS_TEMPLATE`, `OTHER_OPERATION_IN_PROGRESS`, `OPERATION_NOT_ALLOWED`, `BOOTLOADER_FAILED`, `UNKNOWN_BOOTLOADER`

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 ref session_id, VM ref vm)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM to suspend

Minimum Role: vm-operator

Return Type: `void`

Possible Error Codes: `VM_BAD_POWER_STATE`, `OTHER_OPERATION_IN_PROGRESS`, `OPERATION_NOT_ALLOWED`, `VM_IS_TEMPLATE`

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 ref session_id, VM ref vm)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	vm	The VM to unpause

Minimum Role: vm-operator

Return Type: `void`

Possible Error Codes: `VM_BAD_POWER_STATE`, `OPERATION_NOT_ALLOWED`, `VM_IS_TEMPLATE`

RPC name: update_allowed_operations

Overview:

Recomputes the list of acceptable operations

Signature:

```
void update_allowed_operations (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VM ref	self	reference to the object

Minimum Role: pool-admin

Return Type: void

RPC name: wait_memory_target_live

This message is deprecated.

Overview:

Wait for a running VM to reach its current memory target

Signature:

```
void wait_memory_target_live (session ref session_id, VM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	self	The VM

Minimum Role: read-only

Return Type: void

Class: VM_appliance

VM appliance

Fields for class: VM_appliance

Field	Type	Qualifier	Description
allowed_operations	vm_appliance_operation set	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
current_operations	(string -> vm_appliance_operation) map	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.
name_description	string	*RW*	a notes field containing human-readable description

Field	Type	Qualifier	Description
name_label	`string`	*RW*	a human-readable name
uuid	`string`	*RO/runtime*	Unique identifier/object reference
VMs	`VM ref set`	*RO/runtime*	all VMs in this appliance

RPCs associated with class: VM_appliance

RPC name: assert_can_be_recovered

Overview:

Assert whether all SRs required to recover this VM appliance are available.

Signature:

```
void assert_can_be_recovered (session ref session_id, VM_appliance ref self, session
ref session_to)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	The VM appliance to recover
`session ref`	session_to	The session to which the VM appliance is to be recovered.

Minimum Role: read-only

Return Type: `void`

Possible Error Codes: `VM_REQUIRES_SR`

RPC name: clean_shutdown

Overview:

Perform a clean shutdown of all the VMs in the appliance

Signature:

```
void clean_shutdown (session ref session_id, VM_appliance ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VM_appliance ref`	self	The VM appliance

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `OPERATION_PARTIALLY_FAILED`

RPC name: create

Overview:

Create a new VM_appliance instance, and return its handle.

Signature:

```
VM_appliance ref create (session ref session_id, VM_appliance record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance record`	args	All constructor arguments

Minimum Role: pool-operator

Return Type: `VM_appliance ref`

reference to the newly created object

RPC name: destroy

Overview:

Destroy the specified VM_appliance instance.

Signature:

```
void destroy (session ref session_id, VM_appliance ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	reference to the object

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the VM_appliances known to the system.

Signature:

```
VM_appliance ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `VM_appliance ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of VM_appliance references to VM_appliance records for all VM_appliances known to the system.

Signature:

```
(VM_appliance ref -> VM_appliance record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VM_appliance ref -> VM_appliance record) map`

records of all objects

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given VM_appliance.

Signature:

```
vm_appliance_operation set get_allowed_operations (session ref session_id,
VM_appliance ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vm_appliance_operation set`

value of the field

RPC name: get_by_name_label

Overview:

Get all the VM_appliance instances with the given label.

Signature:

```
VM_appliance ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `VM_appliance ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the VM_appliance instance with the specified UUID.

Signature:

```
VM_appliance ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VM_appliance ref`

reference to the object

RPC name: get_current_operations

Overview:

Get the current_operations field of the given VM_appliance.

Signature:

```
(string -> vm_appliance_operation) map get_current_operations (session ref session_id,
VM_appliance ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> vm_appliance_operation) map`

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given VM_appliance.

Signature:

```
string get_name_description (session ref session_id, VM_appliance ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

Overview:

Get the name/label field of the given VM_appliance.

Signature:

```
string get_name_label (session ref session_id, VM_appliance ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given VM_appliance.

Signature:

```
VM_appliance record get_record (session ref session_id, VM_appliance ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM_appliance record`

all fields from the object

RPC name: get_SRs_required_for_recovery

Overview:

Get the list of SRs required by the VM appliance to recover.

Signature:

```
SR ref set get_SRs_required_for_recovery (session ref session_id, VM_appliance ref
self, session ref session_to)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	The VM appliance for which the required list of SRs has to be recovered.
`session ref`	session_to	The session to which the list of SRs have to be recovered .

Minimum Role: read-only

Return Type: `SR ref set`

refs for SRs required to recover the VM

RPC name: get_uuid

Overview:

Get the uuid field of the given VM_appliance.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VMs

Overview:

Get the VMs field of the given VM_appliance.

Signature:

```
VM ref set get_VMs (session ref session_id, VM_appliance ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref set`

value of the field

RPC name: hard_shutdown

Overview:

Perform a hard shutdown of all the VMs in the appliance

Signature:

```
void hard_shutdown (session ref session_id, VM_appliance ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	The VM appliance

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `OPERATION_PARTIALLY_FAILED`

RPC name: recover

Overview:

Recover the VM appliance

Signature:

```
void recover (session ref session_id, VM_appliance ref self, session ref session_to,
bool force)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM_appliance ref	self	The VM appliance to recover
session ref	session_to	The session to which the VM appliance is to be recovered.
bool	force	Whether the VMs should replace newer versions of themselves.

Minimum Role: read-only

Return Type: void

Possible Error Codes: VM_REQUIRES_SR

RPC name: set_name_description

Overview:

Set the name/description field of the given VM_appliance.

Signature:

```
void set_name_description (session ref session_id, VM_appliance ref self, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM_appliance ref	self	reference to the object
string	value	New value to set

Minimum Role: pool-operator

Return Type: void

RPC name: set_name_label

Overview:

Set the name/label field of the given VM_appliance.

Signature:

```
void set_name_label (session ref session_id, VM_appliance ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: shutdown

Overview:

For each VM in the appliance, try to shut it down cleanly. If this fails, perform a hard shutdown of the VM.

Signature:

```
void shutdown (session ref session_id, VM_appliance ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_appliance ref`	self	The VM appliance

Minimum Role: pool-operator

Return Type: `void`

Possible Error Codes: `OPERATION_PARTIALLY_FAILED`

RPC name: start

Overview:

Start all VMs in the appliance

Signature:

```
```

```
void start (session ref session_id, VM_appliance ref self, bool paused)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM_appliance ref	self	The VM appliance
bool	paused	Instantiate all VMs belonging to this appliance in paused state if set to true.

Minimum Role: pool-operator

Return Type: void

Possible Error Codes: OPERATION_PARTIALLY_FAILED

Class: VM_guest_metrics

The metrics reported by the guest (as opposed to inferred from outside)

Fields for class: VM_guest_metrics

Field	Type	Qualifier	Description
can_use_hotplug_vbd	tristate_type	*RO/runtime*	The guest's statement of whether it supports VBD hotplug, i.e. whether it is capable of responding immediately to instantiation of a new VBD by bringing online a new PV block device. If the guest states that it is not capable, then the VBD plug and unplug operations will not be allowed while the guest is running.
can_use_hotplug_vif	tristate_type	*RO/runtime*	The guest's statement of whether it supports VIF hotplug, i.e. whether it is capable of responding immediately to instantiation of a new VIF by bringing online a new PV network device. If the guest states that it is not capable, then the VIF plug and unplug operations will not be allowed while the guest is running.
disks	(string -> string) map	*RO/runtime*	**Removed**. This field exists but has no data.
last_updated	datetime	*RO/runtime*	Time at which this information was last updated
live	bool	*RO/runtime*	True if the guest is sending heartbeat messages via the guest agent

Field	Type	Qualifier	Description
memory	`(string -> string) map`	*RO/runtime*	**Removed**. This field exists but has no data. Use the memory and memory_internal_free RRD data-sources instead.
networks	`(string -> string) map`	*RO/runtime*	network configuration
os_version	`(string -> string) map`	*RO/runtime*	version of the OS
other	`(string -> string) map`	*RO/runtime*	anything else
other_config	`(string -> string) map`	*RW*	additional configuration
PV_drivers_detected	`bool`	*RO/runtime*	At least one of the guest's devices has successfully connected to the backend.
PV_drivers_up_to_date	`bool`	*RO/runtime*	**Deprecated**. Logically equivalent to PV_drivers_detected
PV_drivers_version	`(string -> string) map`	*RO/runtime*	version of the PV drivers
uuid	`string`	*RO/runtime*	Unique identifier/object reference

RPCs associated with class: VM_guest_metrics

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given VM_guest_metrics.

Signature:

```
void add_to_other_config (session ref session_id, VM_guest_metrics ref self, string
key, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_guest_metrics ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: vm-admin

Return Type: `void`

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 ref session_id)
```

Minimum Role: read-only

Return Type: `VM_guest_metrics ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of VM_guest_metrics references to VM_guest_metrics records for all VM_guest_metrics instances known to the system.

Signature:

```
(VM_guest_metrics ref -> VM_guest_metrics record) map get_all_records (session ref
session_id)
```

Minimum Role: read-only

Return Type: `(VM_guest_metrics ref -> VM_guest_metrics record) map`

records of all objects

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 ref session_id, string uuid)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VM_guest_metrics ref`

reference to the object

RPC name: get_can_use_hotplug_vbd

Overview:

Get the can_use_hotplug_vbd field of the given VM_guest_metrics.

Signature:

```
tristate_type get_can_use_hotplug_vbd (session ref session_id, VM_guest_metrics ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_guest_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `tristate_type`

value of the field

RPC name: get_can_use_hotplug_vif

Overview:

Get the can_use_hotplug_vif field of the given VM_guest_metrics.

Signature:

```
tristate_type get_can_use_hotplug_vif (session ref session_id, VM_guest_metrics ref
self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM_guest_metrics ref	self	reference to the object

Minimum Role: read-only

Return Type: `tristate_type`

value of the field

RPC name: get_disks

This message is removed.

Overview:

Get the disks field of the given VM_guest_metrics.

Signature:

```
(string -> string) map get_disks (session ref session_id, VM_guest_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM_guest_metrics ref	self	reference to the object

Minimum Role: read-only

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 ref session_id, VM_guest_metrics ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM_guest_metrics ref	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_live

Overview:

Get the live field of the given VM_guest_metrics.

Signature:

```
bool get_live (session ref session_id, VM_guest_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM_guest_metrics ref	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_memory

This message is removed.

Overview:

Get the memory field of the given VM_guest_metrics.

Signature:

```
(string -> string) map get_memory (session ref session_id, VM_guest_metrics ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VM_guest_metrics ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, VM_guest_metrics ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_guest_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

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 ref session_id, VM_guest_metrics ref
self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VM_guest_metrics ref	self	reference to the object

Minimum Role: read-only

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 ref session_id, VM_guest_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM_guest_metrics ref	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given VM_guest_metrics.

Signature:

```
(string -> string) map get_other_config (session ref session_id, VM_guest_metrics ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VM_guest_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_PV_drivers_detected

Overview:

Get the PV_drivers_detected field of the given VM_guest_metrics.

Signature:

```
bool get_PV_drivers_detected (session ref session_id, VM_guest_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_guest_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_PV_drivers_up_to_date

This message is deprecated.

Overview:

Get the PV_drivers_up_to_date field of the given VM_guest_metrics.

Signature:

```
bool get_PV_drivers_up_to_date (session ref session_id, VM_guest_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VM_guest_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

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 ref session_id,
VM_guest_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_guest_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

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 ref session_id, VM_guest_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_guest_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM_guest_metrics record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given VM_guest_metrics.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_guest_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given VM_guest_metrics. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, VM_guest_metrics ref self,
string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_guest_metrics ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given VM_guest_metrics.

Signature:

```
void set_other_config (session ref session_id, VM_guest_metrics ref self, (string ->
string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_guest_metrics ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

Class: VM_metrics

The metrics associated with a VM

Fields for class: VM_metrics

Field	Type	Qualifier	Description
current_domain_type	`domain_type`	*RO/runtime*	The current domain type of the VM (for running,suspended, or paused VMs). The last-known domain type for halted VMs.
hvm	`bool`	*RO/runtime*	hardware virtual machine
install_time	`datetime`	*RO/runtime*	Time at which the VM was installed
last_updated	`datetime`	*RO/runtime*	Time at which this information was last updated
memory_actual	`int`	*RO/runtime*	Guest's actual memory (bytes)
nested_virt	`bool`	*RO/runtime*	VM supports nested virtualisation
nomigrate	`bool`	*RO/runtime*	VM is immobile and can't migrate between hosts
other_config	`(string -> string) map`	*RW*	additional configuration

Field	Type	Qualifier	Description
start_time	`datetime`	*RO/runtime*	Time at which this VM was last booted
state	`string set`	*RO/runtime*	The state of the guest, eg blocked, dying etc
uuid	`string`	*RO/runtime*	Unique identifier/object reference
VCPUs_CPU	`(int -> int) map`	*RO/runtime*	VCPU to PCPU map
VCPUs_flags	`(int -> string set) map`	*RO/runtime*	CPU flags (blocked,online,running)
VCPUs_number	`int`	*RO/runtime*	Current number of VCPUs
VCPUs_params	`(string -> string) map`	*RO/runtime*	The live equivalent to VM.VCPUs_params
VCPUs_utilisation	`(int -> float) map`	*RO/runtime*	**Removed**. Utilisation for all of guest's current VCPUs

RPCs associated with class: VM_metrics

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given VM_metrics.

Signature:

```
void add_to_other_config (session ref session_id, VM_metrics ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object
`string`	key	Key to add
`string`	value	Value to add

Minimum Role: vm-admin

Return Type: `void`

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 ref session_id)
```

Minimum Role: read-only

Return Type: `VM_metrics ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of VM_metrics references to VM_metrics records for all VM_metrics instances known to the system.

Signature:

```
(VM_metrics ref -> VM_metrics record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VM_metrics ref -> VM_metrics record) map`

records of all objects

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 ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VM_metrics ref`

reference to the object

RPC name: get_current_domain_type

Overview:

Get the current_domain_type field of the given VM_metrics.

Signature:

```
domain_type get_current_domain_type (session ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `domain_type`

value of the field

RPC name: get_hvm

Overview:

Get the hvm field of the given VM_metrics.

Signature:

```
bool get_hvm (session ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_install_time

Overview:

Get the install_time field of the given VM_metrics.

Signature:

```
datetime get_install_time (session ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

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 ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_nested_virt

Overview:

Get the nested_virt field of the given VM_metrics.

Signature:

```
bool get_nested_virt (session ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_nomigrate

Overview:

Get the nomigrate field of the given VM_metrics.

Signature:

```
bool get_nomigrate (session ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given VM_metrics.

Signature:

```
(string -> string) map get_other_config (session ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given VM_metrics.

Signature:

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

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM_metrics record`

all fields from the object

RPC name: get_start_time

Overview:

Get the start_time field of the given VM_metrics.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_state

Overview:

Get the state field of the given VM_metrics.

Signature:

```
string set get_state (session ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given VM_metrics.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

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 ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(int -> int) 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 ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(int -> string set) map`

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 ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

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 ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_VCPUs_utilisation

This message is removed.

Overview:

Get the VCPUs/utilisation field of the given VM_metrics.

Signature:

```
(int -> float) map get_VCPUs_utilisation (session ref session_id, VM_metrics ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(int -> float) map`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given VM_metrics. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, VM_metrics ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: vm-admin

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given VM_metrics.

Signature:

```
void set_other_config (session ref session_id, VM_metrics ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM_metrics ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: vm-admin

Return Type: `void`

Class: VMPP

This class is removed.

VM Protection Policy

Fields for class: VMPP

Field	Type	Qualifier	Description
alarm_config	`(string -> string) map`	*RO/constructor*	**Removed**. configuration for the alarm
archive_frequency	`vmpp_archive_frequency`	*RO/constructor*	**Removed**. frequency of the archive schedule
archive_last_run_time	`datetime`	*RO/runtime*	**Removed**. time of the last archive
archive_schedule	`(string -> string) map`	*RO/constructor*	**Removed**. schedule of the archive containing 'hour', 'min', 'days'. Date/time-related information is in Local Timezone
archive_target_config	`(string -> string) map`	*RO/constructor*	**Removed**. configuration for the archive, including its 'location', 'username', 'password'
archive_target_type	`vmpp_archive_target_type`	*RO/constructor*	**Removed**. type of the archive target config
backup_frequency	`vmpp_backup_frequency`	*RO/constructor*	**Removed**. frequency of the backup schedule
backup_last_run_time	`datetime`	*RO/runtime*	**Removed**. time of the last backup
backup_retention_value	`int`	*RO/constructor*	**Removed**. maximum number of backups that should be stored at any time
backup_schedule	`(string -> string) map`	*RO/constructor*	**Removed**. schedule of the backup containing 'hour', 'min', 'days'. Date/time-related information is in Local Timezone
backup_type	`vmpp_backup_type`	*RW*	**Removed**. type of the backup sub-policy
is_alarm_enabled	`bool`	*RO/constructor*	**Removed**. true if alarm is enabled for this policy
is_archive_running	`bool`	*RO/runtime*	**Removed**. true if this protection policy's archive is running

Field	Type	Qualifier	Description
is_backup_running	`bool`	*RO/runtime*	**Removed**. true if this protection policy's backup is running
is_policy_enabled	`bool`	*RW*	**Removed**. enable or disable this policy
name_description	`string`	*RW*	**Removed**. a notes field containing human-readable description
name_label	`string`	*RW*	**Removed**. a human-readable name
recent_alerts	`string set`	*RO/runtime*	**Removed**. recent alerts
uuid	`string`	*RO/runtime*	**Removed**. Unique identifier/object reference
VMs	`VM ref set`	*RO/runtime*	**Removed**. all VMs attached to this protection policy

RPCs associated with class: VMPP

RPC name: add_to_alarm_config

This message is removed.

Overview:

Signature:

```
void add_to_alarm_config (session ref session_id, VMPP ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	The protection policy
`string`	key	the key to add
`string`	value	the value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: add_to_archive_schedule

This message is removed.

Overview:

Signature:

```
void add_to_archive_schedule (session ref session_id, VMPP ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	The protection policy
`string`	key	the key to add
`string`	value	the value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: add_to_archive_target_config

This message is removed.

Overview:

Signature:

```
void add_to_archive_target_config (session ref session_id, VMPP ref self, string key,
string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	The protection policy
`string`	key	the key to add
`string`	value	the value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: add_to_backup_schedule

This message is removed.

Overview:

Signature:

```
void add_to_backup_schedule (session ref session_id, VMPP ref self, string key, string
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	The protection policy
`string`	key	the key to add
`string`	value	the value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: archive_now

This message is removed.

Overview:

This call archives the snapshot provided as a parameter

Signature:

```
string archive_now (session ref session_id, VM ref snapshot)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VM ref`	snapshot	The snapshot to archive

Minimum Role: vm-power-admin

Return Type: `string`

An XMLRPC result

RPC name: create

This message is removed.

Overview:

Create a new VMPP instance, and return its handle.

Signature:

```
VMPP ref create (session ref session_id, VMPP record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP record`	args	All constructor arguments

Minimum Role: pool-operator

Return Type: `VMPP ref`

reference to the newly created object

RPC name: destroy

This message is removed.

Overview:

Destroy the specified VMPP instance.

Signature:

```
void destroy (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_alarm_config

This message is removed.

Overview:

Get the alarm_config field of the given VMPP.

Signature:

```
(string -> string) map get_alarm_config (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_alerts

This message is removed.

Overview:

This call fetches a history of alerts for a given protection policy

Signature:

```
string set get_alerts (session ref session_id, VMPP ref vmpp, int hours_from_now)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	vmpp	The protection policy
`int`	hours_from_now	how many hours in the past the oldest record to fetch is

Minimum Role: pool-operator

Return Type: `string set`

A list of alerts encoded in xml

RPC name: get_all

This message is removed.

Overview:

Return a list of all the VMPPs known to the system.

Signature:

```
VMPP ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `VMPP ref set`

references to all objects

RPC name: get_all_records

This message is removed.

Overview:

Return a map of VMPP references to VMPP records for all VMPPs known to the system.

Signature:

```
(VMPP ref -> VMPP record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VMPP ref -> VMPP record) map`

records of all objects

RPC name: get_archive_frequency

This message is removed.

Overview:

Get the archive_frequency field of the given VMPP.

Signature:

```
vmpp_archive_frequency get_archive_frequency (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VMPP ref	self	reference to the object

Minimum Role: read-only

Return Type: vmpp_archive_frequency

value of the field

RPC name: get_archive_last_run_time

This message is removed.

Overview:

Get the archive_last_run_time field of the given VMPP.

Signature:

```
datetime get_archive_last_run_time (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VMPP ref	self	reference to the object

Minimum Role: read-only

Return Type: datetime

value of the field

RPC name: get_archive_schedule

This message is removed.

Overview:

Get the archive_schedule field of the given VMPP.

Signature:

```
(string -> string) map get_archive_schedule (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_archive_target_config

This message is removed.

Overview:

Get the archive_target_config field of the given VMPP.

Signature:

```
(string -> string) map get_archive_target_config (session ref session_id, VMPP ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_archive_target_type

This message is removed.

Overview:

Get the archive_target_type field of the given VMPP.

Signature:

```
vmpp_archive_target_type get_archive_target_type (session ref session_id, VMPP ref
self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vmpp_archive_target_type`

value of the field

RPC name: get_backup_frequency

This message is removed.

Overview:

Get the backup_frequency field of the given VMPP.

Signature:

```
vmpp_backup_frequency get_backup_frequency (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vmpp_backup_frequency`

value of the field

RPC name: get_backup_last_run_time

This message is removed.

Overview:

Get the backup_last_run_time field of the given VMPP.

Signature:

```
datetime get_backup_last_run_time (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_backup_retention_value

This message is removed.

Overview:

Get the backup_retention_value field of the given VMPP.

Signature:

```
int get_backup_retention_value (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_backup_schedule

This message is removed.

Overview:

Get the backup_schedule field of the given VMPP.

Signature:

```
(string -> string) map get_backup_schedule (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_backup_type

This message is removed.

Overview:

Get the backup_type field of the given VMPP.

Signature:

```
vmpp_backup_type get_backup_type (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vmpp_backup_type`

value of the field

RPC name: get_by_name_label

This message is removed.

Overview:

Get all the VMPP instances with the given label.

Signature:

```
VMPP ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `VMPP ref set`

references to objects with matching names

RPC name: get_by_uuid

This message is removed.

Overview:

Get a reference to the VMPP instance with the specified UUID.

Signature:

```
VMPP ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VMPP ref`

reference to the object

RPC name: get_is_alarm_enabled

This message is removed.

Overview:

Get the is_alarm_enabled field of the given VMPP.

Signature:

```
bool get_is_alarm_enabled (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_is_archive_running

This message is removed.

Overview:

Get the is_archive_running field of the given VMPP.

Signature:

```
bool get_is_archive_running (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_is_backup_running

This message is removed.

Overview:

Get the is_backup_running field of the given VMPP.

Signature:

```
bool get_is_backup_running (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_is_policy_enabled

This message is removed.

Overview:

Get the is_policy_enabled field of the given VMPP.

Signature:

```
bool get_is_policy_enabled (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_name_description

This message is removed.

Overview:

Get the name/description field of the given VMPP.

Signature:

```
string get_name_description (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

This message is removed.

Overview:

Get the name/label field of the given VMPP.

Signature:

```
string get_name_label (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_recent_alerts

This message is removed.

Overview:

Get the recent_alerts field of the given VMPP.

Signature:

```
string set get_recent_alerts (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string set`

value of the field

RPC name: get_record

This message is removed.

Overview:

Get a record containing the current state of the given VMPP.

Signature:

```
VMPP record get_record (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VMPP record`

all fields from the object

RPC name: get_uuid

This message is removed.

Overview:

Get the uuid field of the given VMPP.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VMs

This message is removed.

Overview:

Get the VMs field of the given VMPP.

Signature:

```
VM ref set get_VMs (session ref session_id, VMPP ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref set`

value of the field

RPC name: protect_now

This message is removed.

Overview:

This call executes the protection policy immediately

Signature:

```
string protect_now (session ref session_id, VMPP ref vmpp)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	vmpp	The protection policy to execute

Minimum Role: pool-operator

Return Type: `string`

An XMLRPC result

RPC name: remove_from_alarm_config

This message is removed.

Overview:

Signature:

```
void remove_from_alarm_config (session ref session_id, VMPP ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	The protection policy
`string`	key	the key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: remove_from_archive_schedule

This message is removed.

Overview:

Signature:

```
void remove_from_archive_schedule (session ref session_id, VMPP ref self, string key)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VMPP ref	self	The protection policy
string	key	the key to remove

Minimum Role: pool-operator

Return Type: void

RPC name: remove_from_archive_target_config

This message is removed.

Overview:

Signature:

```
void remove_from_archive_target_config (session ref session_id, VMPP ref self, string
key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VMPP ref	self	The protection policy
string	key	the key to remove

Minimum Role: pool-operator

Return Type: void

RPC name: remove_from_backup_schedule

This message is removed.

Overview:

Signature:

```
void remove_from_backup_schedule (session ref session_id, VMPP ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VMPP ref	self	The protection policy
string	key	the key to remove

Minimum Role: pool-operator

Return Type: void

RPC name: set_alarm_config

This message is removed.

Overview:

Signature:

```
void set_alarm_config (session ref session_id, VMPP ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VMPP ref	self	The protection policy
(string -> string) map	value	the value to set

Minimum Role: pool-operator

Return Type: void

RPC name: set_archive_frequency

This message is removed.

Overview:

Set the value of the archive_frequency field

Signature:

```
void set_archive_frequency (session ref session_id, VMPP ref self,
vmpp_archive_frequency value)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VMPP ref	self	The protection policy
vmpp_archive_frequency	value	the archive frequency

Minimum Role: pool-operator

Return Type: void

RPC name: set_archive_last_run_time

This message is removed.

Overview:

Signature:

```
void set_archive_last_run_time (session ref session_id, VMPP ref self, datetime value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VMPP ref	self	The protection policy
datetime	value	the value to set

Return Type: void

RPC name: set_archive_schedule

This message is removed.

Overview:

Signature:

```
void set_archive_schedule (session ref session_id, VMPP ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VMPP ref	self	The protection policy

type	name	description
`(string -> string) map`	value	the value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_archive_target_config

This message is removed.

Overview:

Signature:

```
void set_archive_target_config (session ref session_id, VMPP ref self, (string ->
string) map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	The protection policy
`(string -> string) map`	value	the value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_archive_target_type

This message is removed.

Overview:

Set the value of the archive_target_config_type field

Signature:

```
void set_archive_target_type (session ref session_id, VMPP ref self,
vmpp_archive_target_type value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VMPP ref	self	The protection policy
vmpp_archive_target_type	value	the archive target config type

Minimum Role: pool-operator

Return Type: void

RPC name: set_backup_frequency

This message is removed.

Overview:

Set the value of the backup_frequency field

Signature:

```
void set_backup_frequency (session ref session_id, VMPP ref self,
vmpp_backup_frequency value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VMPP ref	self	The protection policy
vmpp_backup_frequency	value	the backup frequency

Minimum Role: pool-operator

Return Type: void

RPC name: set_backup_last_run_time

This message is removed.

Overview:

Signature:

```
void set_backup_last_run_time (session ref session_id, VMPP ref self, datetime value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VMPP ref	self	The protection policy
datetime	value	the value to set

Return Type: void

RPC name: set_backup_retention_value

This message is removed.

Overview:

Signature:

```
void set_backup_retention_value (session ref session_id, VMPP ref self, int value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VMPP ref	self	The protection policy
int	value	the value to set

Minimum Role: pool-operator

Return Type: void

RPC name: set_backup_schedule

This message is removed.

Overview:

Signature:

```
void set_backup_schedule (session ref session_id, VMPP ref self, (string -> string)
map value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VMPP ref	self	The protection policy
(string -> string) map	value	the value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_backup_type

This message is removed.

Overview:

Set the backup_type field of the given VMPP.

Signature:

```
void set_backup_type (session ref session_id, VMPP ref self, vmpp_backup_type value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object
`vmpp_backup_type`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_is_alarm_enabled

This message is removed.

Overview:

Set the value of the is_alarm_enabled field

Signature:

```
void set_is_alarm_enabled (session ref session_id, VMPP ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	The protection policy
`bool`	value	true if alarm is enabled for this policy

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_is_policy_enabled

This message is removed.

Overview:

Set the is_policy_enabled field of the given VMPP.

Signature:

```
void set_is_policy_enabled (session ref session_id, VMPP ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object
`bool`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_name_description

This message is removed.

Overview:

Set the name/description field of the given VMPP.

Signature:

```
void set_name_description (session ref session_id, VMPP ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_name_label

This message is removed.

Overview:

Set the name/label field of the given VMPP.

Signature:

```
void set_name_label (session ref session_id, VMPP ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMPP ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

Class: VMSS

VM Snapshot Schedule

Fields for class: VMSS

Field	Type	Qualifier	Description
enabled	`bool`	*RW*	enable or disable this snapshot schedule
frequency	`vmss_frequency`	*RO/constructor*	frequency of taking snapshot from snapshot schedule
last_run_time	`datetime`	*RO/runtime*	time of the last snapshot
name_description	`string`	*RW*	a notes field containing human-readable description
name_label	`string`	*RW*	a human-readable name
retained_snapshots	`int`	*RO/constructor*	maximum number of snapshots that should be stored at any time
schedule	`(string -> string) map`	*RO/constructor*	schedule of the snapshot containing 'hour', 'min', 'days'. Date/time-related information is in Local Timezone
type	`vmss_type`	*RO/constructor*	type of the snapshot schedule

Field	Type	Qualifier	Description
uuid	`string`	*RO/runtime*	Unique identifier/object reference
VMs	`VM ref set`	*RO/runtime*	all VMs attached to this snapshot schedule

RPCs associated with class: VMSS

RPC name: add_to_schedule

Overview:

Signature:

```
void add_to_schedule (session ref session_id, VMSS ref self, string key, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	The snapshot schedule
`string`	key	the key to add
`string`	value	the value to add

Minimum Role: pool-operator

Return Type: `void`

RPC name: create

Overview:

Create a new VMSS instance, and return its handle.

Signature:

```
VMSS ref create (session ref session_id, VMSS record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS record`	args	All constructor arguments

Minimum Role: pool-operator

Return Type: `VMSS ref`

reference to the newly created object

RPC name: destroy

Overview:

Destroy the specified VMSS instance.

Signature:

```
void destroy (session ref session_id, VMSS ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	reference to the object

Minimum Role: pool-operator

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the VMSSs known to the system.

Signature:

```
VMSS ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `VMSS ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of VMSS references to VMSS records for all VMSSs known to the system.

Signature:

```
(VMSS ref -> VMSS record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VMSS ref -> VMSS record) map`

records of all objects

RPC name: get_by_name_label

Overview:

Get all the VMSS instances with the given label.

Signature:

```
VMSS ref set get_by_name_label (session ref session_id, string label)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	label	label of object to return

Minimum Role: read-only

Return Type: `VMSS ref set`

references to objects with matching names

RPC name: get_by_uuid

Overview:

Get a reference to the VMSS instance with the specified UUID.

Signature:

```
VMSS ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VMSS ref`

reference to the object

RPC name: get_enabled

Overview:

Get the enabled field of the given VMSS.

Signature:

```
bool get_enabled (session ref session_id, VMSS ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_frequency

Overview:

Get the frequency field of the given VMSS.

Signature:

```
vmss_frequency get_frequency (session ref session_id, VMSS ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vmss_frequency`

value of the field

RPC name: get_last_run_time

Overview:

Get the last_run_time field of the given VMSS.

Signature:

```
datetime get_last_run_time (session ref session_id, VMSS ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	reference to the object

Minimum Role: read-only

Return Type: `datetime`

value of the field

RPC name: get_name_description

Overview:

Get the name/description field of the given VMSS.

Signature:

```
string get_name_description (session ref session_id, VMSS ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_name_label

Overview:

Get the name/label field of the given VMSS.

Signature:

```
string get_name_label (session ref session_id, VMSS ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given VMSS.

Signature:

```
VMSS record get_record (session ref session_id, VMSS ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VMSS record`

all fields from the object

RPC name: get_retained_snapshots

Overview:

Get the retained_snapshots field of the given VMSS.

Signature:

```
int get_retained_snapshots (session ref session_id, VMSS ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	reference to the object

Minimum Role: read-only

Return Type: `int`

value of the field

RPC name: get_schedule

Overview:

Get the schedule field of the given VMSS.

Signature:

```
(string -> string) map get_schedule (session ref session_id, VMSS ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_type

Overview:

Get the type field of the given VMSS.

Signature:

```
vmss_type get_type (session ref session_id, VMSS ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VMSS ref	self	reference to the object

Minimum Role: read-only

Return Type: vmss_type

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given VMSS.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VMSS ref	self	reference to the object

Minimum Role: read-only

Return Type: string

value of the field

RPC name: get_VMs

Overview:

Get the VMs field of the given VMSS.

Signature:

```
VM ref set get_VMs (session ref session_id, VMSS ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VMSS ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref set`

value of the field

RPC name: remove_from_schedule

Overview:

Signature:

```
void remove_from_schedule (session ref session_id, VMSS ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	The snapshot schedule
`string`	key	the key to remove

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_enabled

Overview:

Set the enabled field of the given VMSS.

Signature:

```
void set_enabled (session ref session_id, VMSS ref self, bool value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	reference to the object
`bool`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_frequency

Overview:

Set the value of the frequency field

Signature:

```
void set_frequency (session ref session_id, VMSS ref self, vmss_frequency value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	The snapshot schedule
`vmss_frequency`	value	the snapshot schedule frequency

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_last_run_time

Overview:

Signature:

```
void set_last_run_time (session ref session_id, VMSS ref self, datetime value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	The snapshot schedule
`datetime`	value	the value to set

Return Type: `void`

RPC name: set_name_description

Overview:

Set the name/description field of the given VMSS.

Signature:

```
void set_name_description (session ref session_id, VMSS ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_name_label

Overview:

Set the name/label field of the given VMSS.

Signature:

```
void set_name_label (session ref session_id, VMSS ref self, string value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	reference to the object
`string`	value	New value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_retained_snapshots

Overview:

Signature:

```
void set_retained_snapshots (session ref session_id, VMSS ref self, int value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	The schedule snapshot
`int`	value	the value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_schedule

Overview:

Signature:

```
void set_schedule (session ref session_id, VMSS ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	self	The snapshot schedule
`(string -> string) map`	value	the value to set

Minimum Role: pool-operator

Return Type: `void`

RPC name: set_type

Overview:

Signature:

```
void set_type (session ref session_id, VMSS ref self, vmss_type value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VMSS ref`	self	The snapshot schedule
`vmss_type`	value	the snapshot schedule type

Minimum Role: pool-operator

Return Type: `void`

RPC name: snapshot_now

Overview:

This call executes the snapshot schedule immediately

Signature:

```
string snapshot_now (session ref session_id, VMSS ref vmss)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VMSS ref`	vmss	Snapshot Schedule to execute

Minimum Role: pool-operator

Return Type: `string`

An XMLRPC result

Class: VTPM

A virtual TPM device

Fields for class: VTPM

Field	Type	Qualifier	Description
backend	`VM ref`	*RO/constructor*	the domain where the backend is located
uuid	`string`	*RO/runtime*	Unique identifier/object reference
VM	`VM ref`	*RO/constructor*	the virtual machine

RPCs associated with class: VTPM

RPC name: create

Overview:

Create a new VTPM instance, and return its handle.

Signature:

```
VTPM ref create (session ref session_id, VTPM record args)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VTPM record`	args	All constructor arguments

Minimum Role: vm-admin

Return Type: `VTPM ref`

reference to the newly created object

RPC name: destroy

Overview:

Destroy the specified VTPM instance.

Signature:

```
void destroy (session ref session_id, VTPM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VTPM ref`	self	reference to the object

Minimum Role: vm-admin

Return Type: `void`

RPC name: get_backend

Overview:

Get the backend field of the given VTPM.

Signature:

```
VM ref get_backend (session ref session_id, VTPM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VTPM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref`

value of the field

RPC name: get_by_uuid

Overview:

Get a reference to the VTPM instance with the specified UUID.

Signature:

```
VTPM ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

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 ref session_id, VTPM ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
`VTPM ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VTPM record`

all fields from the object

RPC name: get_uuid

Overview:

Get the uuid field of the given VTPM.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VTPM ref`	self	reference to the object

Minimum Role: read-only

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 ref session_id, VTPM ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
VTPM ref	self	reference to the object

Minimum Role: read-only

Return Type: VM ref

value of the field

Class: VUSB

Describes the vusb device

Fields for class: VUSB

Field	Type	Qualifier	Description
allowed_operations	vusb_operations set	*RO/runtime*	list of the operations allowed in this state. This list is advisory only and the server state may have changed by the time this field is read by a client.
current_operations	(string -> vusb_operations) map	*RO/runtime*	links each of the running tasks using this object (by reference) to a current_operation enum which describes the nature of the task.
currently_attached	bool	*RO/runtime*	is the device currently attached
other_config	(string -> string) map	*RW*	Additional configuration
USB_group	USB_group ref	*RO/runtime*	USB group used by the VUSB
uuid	string	*RO/runtime*	Unique identifier/object reference
VM	VM ref	*RO/runtime*	VM that owns the VUSB

RPCs associated with class: VUSB

RPC name: add_to_other_config

Overview:

Add the given key-value pair to the other_config field of the given VUSB.

Signature:

```
void add_to_other_config (session ref session_id, VUSB ref self, string key, string
value)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VUSB ref	self	reference to the object
string	key	Key to add
string	value	Value to add

Minimum Role: pool-admin

Return Type: void

RPC name: create

Overview:

Create a new VUSB record in the database only

Signature:

```
VUSB ref create (session ref session_id, VM ref VM, USB_group ref USB_group, (string -
> string) map other_config)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VM ref	VM	The VM
USB_group ref	USB_group	
(string -> string) map	other_config	

Minimum Role: pool-admin

Return Type: VUSB ref

The ref of the newly created VUSB record.

RPC name: destroy

Overview:

Removes a VUSB record from the database

Signature:

```
void destroy (session ref session_id, VUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VUSB ref`	self	The VUSB to destroy about

Minimum Role: pool-admin

Return Type: `void`

RPC name: get_all

Overview:

Return a list of all the VUSBs known to the system.

Signature:

```
VUSB ref set get_all (session ref session_id)
```

Minimum Role: read-only

Return Type: `VUSB ref set`

references to all objects

RPC name: get_all_records

Overview:

Return a map of VUSB references to VUSB records for all VUSBs known to the system.

Signature:

```
(VUSB ref -> VUSB record) map get_all_records (session ref session_id)
```

Minimum Role: read-only

Return Type: `(VUSB ref -> VUSB record) map`

records of all objects

RPC name: get_allowed_operations

Overview:

Get the allowed_operations field of the given VUSB.

Signature:

```
vusb_operations set get_allowed_operations (session ref session_id, VUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `vusb_operations set`

value of the field

RPC name: get_by_uuid

Overview:

Get a reference to the VUSB instance with the specified UUID.

Signature:

```
VUSB ref get_by_uuid (session ref session_id, string uuid)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`string`	uuid	UUID of object to return

Minimum Role: read-only

Return Type: `VUSB ref`

reference to the object

RPC name: get_current_operations

Overview:

Get the current_operations field of the given VUSB.

Signature:

```
(string -> vusb_operations) map get_current_operations (session ref session_id, VUSB
ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> vusb_operations) map`

value of the field

RPC name: get_currently_attached

Overview:

Get the currently_attached field of the given VUSB.

Signature:

```
bool get_currently_attached (session ref session_id, VUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `bool`

value of the field

RPC name: get_other_config

Overview:

Get the other_config field of the given VUSB.

Signature:

```
(string -> string) map get_other_config (session ref session_id, VUSB ref self)
```

Arguments:

type	name	description

type	name	description
session ref	session_id	Reference to a valid session
VUSB ref	self	reference to the object

Minimum Role: read-only

Return Type: `(string -> string) map`

value of the field

RPC name: get_record

Overview:

Get a record containing the current state of the given VUSB.

Signature:

```
VUSB record get_record (session ref session_id, VUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
VUSB ref	self	reference to the object

Minimum Role: read-only

Return Type: `VUSB record`

all fields from the object

RPC name: get_USB_group

Overview:

Get the USB_group field of the given VUSB.

Signature:

```
USB_group ref get_USB_group (session ref session_id, VUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session

type	name	description
`VUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `USB_group ref`

value of the field

RPC name: get_uuid

Overview:

Get the uuid field of the given VUSB.

Signature:

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

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `string`

value of the field

RPC name: get_VM

Overview:

Get the VM field of the given VUSB.

Signature:

```
VM ref get_VM (session ref session_id, VUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VUSB ref`	self	reference to the object

Minimum Role: read-only

Return Type: `VM ref`

value of the field

RPC name: remove_from_other_config

Overview:

Remove the given key and its corresponding value from the other_config field of the given VUSB. If the key is not in that Map, then do nothing.

Signature:

```
void remove_from_other_config (session ref session_id, VUSB ref self, string key)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VUSB ref`	self	reference to the object
`string`	key	Key to remove

Minimum Role: pool-admin

Return Type: `void`

RPC name: set_other_config

Overview:

Set the other_config field of the given VUSB.

Signature:

```
void set_other_config (session ref session_id, VUSB ref self, (string -> string) map
value)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VUSB ref`	self	reference to the object
`(string -> string) map`	value	New value to set

Minimum Role: pool-admin

Return Type: `void`

RPC name: unplug

Overview:

Unplug the vusb device from the vm.

Signature:

```
void unplug (session ref session_id, VUSB ref self)
```

Arguments:

type	name	description
session ref	session_id	Reference to a valid session
`VUSB ref`	self	vusb deivce

Minimum Role: pool-admin

Return Type: `void`

API Reference - Error Handling

When a low-level transport error occurs, or a request is malformed at the HTTP or RPC level, the server may send an HTTP 500 error 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 when using the XML-RPC protocol:

```
$curl -D - -X POST https://server -H 'Content-Type: application/xml' \
> -d '<?xml version="1.0"?>
> <methodCall>
>   <methodName>session.logout</methodName>
> </methodCall>'
HTTP/1.1 500 Internal Error
content-length: 297
content-type:text/html
connection:close
cache-control:no-cache, no-store

<html><body><h1>HTTP 500 internal server error</h1>An unexpected error occurred;
 please wait a while and try again. If the problem persists, please contact your
 support representative.<h1> Additional information </h1>Xmlrpc.Parse_error(&quo
t;close_tag&quot;, &quot;open_tag&quot;, _)</body></html>
```

When using the JSON-RPC protocol:

```
$curl -D - -X POST https://server/jsonrpc -H 'Content-Type: application/json' \
> -d '{
>     "jsonrpc": "2.0",
>     "method": "session.login_with_password",
>     "id": 0
> }'
HTTP/1.1 500 Internal Error
content-length: 308
content-type:text/html
connection:close
cache-control:no-cache, no-store

<html><body><h1>HTTP 500 internal server error</h1>An unexpected error occurred;
 please wait a while and try again. If the problem persists, please contact your
 support representative.<h1> Additional information </h1>Jsonrpc.Malformed_metho
d_request(&quot;{jsonrpc=...,method=...,id=...}&quot;)</body></html>
```

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 when using the XML-RPC protocol:

```
    <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>eSpiel Inc.</value>
              <value>eSpiel 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 -> eSpiel Incorporated* to a Map, but it already contains the mapping *Customer -> eSpiel Inc.*, and so the request has failed.

When using the JSON-RPC protocol v2.0, the above error is transmitted as:

```
{
    "jsonrpc": "2.0",
    "error": {
        "code": 1,
        "message": "MAP_DUPLICATE_KEY",
        "data": [
            "Customer","eSpiel Inc.","eSpiel Incorporated"
        ]
    },
    "id": 3
  }
```

Finally, when using the JSON-RPC protocol v1.0:

```
{
  "result": null,
  "error": [
      "MAP_DUPLICATE_KEY","Customer","eSpiel Inc.","eSpiel Incorporated"
  ],
  "id": "xyz"
}
```

Each possible error code is documented in the following section.

Error Codes

ACTIVATION_WHILE_NOT_FREE

An activation key can only be applied when the edition is set to 'free'.

No parameters.

ADDRESS_VIOLATES_LOCKING_CONSTRAINT

The specified IP address violates the VIF locking configuration.

Signature:

```
ADDRESS_VIOLATES_LOCKING_CONSTRAINT(address)
```

AUTH_ALREADY_ENABLED

External authentication for this server is already enabled.

Signature:

```
AUTH_ALREADY_ENABLED(current auth_type, current service_name)
```

AUTH_DISABLE_FAILED

The host failed to disable external authentication.

Signature:

```
AUTH_DISABLE_FAILED(message)
```

AUTH_DISABLE_FAILED_PERMISSION_DENIED

The host failed to disable external authentication.

Signature:

```
AUTH_DISABLE_FAILED_PERMISSION_DENIED(message)
```

AUTH_DISABLE_FAILED_WRONG_CREDENTIALS

The host failed to disable external authentication.

Signature:

```
AUTH_DISABLE_FAILED_WRONG_CREDENTIALS(message)
```

AUTH_ENABLE_FAILED

The host failed to enable external authentication.

Signature:

```
AUTH_ENABLE_FAILED(message)
```

AUTH_ENABLE_FAILED_DOMAIN_LOOKUP_FAILED

The host failed to enable external authentication.

Signature:

```
AUTH_ENABLE_FAILED_DOMAIN_LOOKUP_FAILED(message)
```

AUTH_ENABLE_FAILED_INVALID_ACCOUNT

The host failed to enable external authentication.

Signature:

```
AUTH_ENABLE_FAILED_INVALID_ACCOUNT(message)
```

AUTH_ENABLE_FAILED_INVALID_OU

The host failed to enable external authentication.

Signature:

```
AUTH_ENABLE_FAILED_INVALID_OU(message)
```

AUTH_ENABLE_FAILED_PERMISSION_DENIED

The host failed to enable external authentication.

Signature:

```
AUTH_ENABLE_FAILED_PERMISSION_DENIED(message)
```

AUTH_ENABLE_FAILED_UNAVAILABLE

The host failed to enable external authentication.

Signature:

```
AUTH_ENABLE_FAILED_UNAVAILABLE(message)
```

AUTH_ENABLE_FAILED_WRONG_CREDENTIALS

The host failed to enable external authentication.

Signature:

```
AUTH_ENABLE_FAILED_WRONG_CREDENTIALS(message)
```

AUTH_IS_DISABLED

External authentication is disabled, unable to resolve subject name.

No parameters.

AUTH_SERVICE_ERROR

Error querying the external directory service.

Signature:

```
AUTH_SERVICE_ERROR(message)
```

AUTH_UNKNOWN_TYPE

Unknown type of external authentication.

Signature:

```
AUTH_UNKNOWN_TYPE(type)
```

BACKUP_SCRIPT_FAILED

The backup could not be performed because the backup script failed.

Signature:

```
BACKUP_SCRIPT_FAILED(log)
```

BALLOONING_TIMEOUT_BEFORE_MIGRATION

Timeout trying to balloon down memory before VM migration. If the error occurs repeatedly, consider increasing the memory-dynamic-min value.

Signature:

```
BALLOONING_TIMEOUT_BEFORE_MIGRATION(vm)
```

BOOTLOADER_FAILED

The bootloader returned an error

Signature:

```
BOOTLOADER_FAILED(vm, msg)
```

BRIDGE_NAME_EXISTS

The specified bridge already exists.

Signature:

```
BRIDGE_NAME_EXISTS(bridge)
```

BRIDGE_NOT_AVAILABLE

Could not find bridge required by VM.

Signature:

```
BRIDGE_NOT_AVAILABLE(bridge)
```

CANNOT_ADD_TUNNEL_TO_BOND_SLAVE

This PIF is a bond member and cannot have a tunnel on it.

Signature:

```
CANNOT_ADD_TUNNEL_TO_BOND_SLAVE(PIF)
```

CANNOT_ADD_TUNNEL_TO_SRIOV_LOGICAL

This is a network SR-IOV logical PIF and cannot have a tunnel on it.

Signature:

```
CANNOT_ADD_TUNNEL_TO_SRIOV_LOGICAL(PIF)
```

CANNOT_ADD_TUNNEL_TO_VLAN_ON_SRIOV_LOGICAL

This is a vlan PIF on network SR-IOV and cannot have a tunnel on it.

Signature:

```
CANNOT_ADD_TUNNEL_TO_VLAN_ON_SRIOV_LOGICAL(PIF)
```

CANNOT_ADD_VLAN_TO_BOND_SLAVE

This PIF is a bond member and cannot have a VLAN on it.

Signature:

```
CANNOT_ADD_VLAN_TO_BOND_SLAVE(PIF)
```

CANNOT_CHANGE_PIF_PROPERTIES

This properties of this PIF cannot be changed. Only the properties of non-bonded physical PIFs, or bond masters can be changed.

Signature:

```
CANNOT_CHANGE_PIF_PROPERTIES(PIF)
```

CANNOT_CONTACT_HOST

Cannot forward messages because the server cannot be contacted. The server may be switched off or there may be network connectivity problems.

Signature:

```
CANNOT_CONTACT_HOST(host)
```

CANNOT_CREATE_STATE_FILE

An HA statefile could not be created, perhaps because no SR with the appropriate capability was found.

No parameters.

CANNOT_DESTROY_DISASTER_RECOVERY_TASK

The disaster recovery task could not be cleanly destroyed.

Signature:

```
CANNOT_DESTROY_DISASTER_RECOVERY_TASK(reason)
```

CANNOT_DESTROY_SYSTEM_NETWORK

You tried to destroy a system network: these cannot be destroyed.

Signature:

```
CANNOT_DESTROY_SYSTEM_NETWORK(network)
```

CANNOT_ENABLE_REDO_LOG

Could not enable redo log.

Signature:

```
CANNOT_ENABLE_REDO_LOG(reason)
```

CANNOT_EVACUATE_HOST

This server cannot be evacuated.

Signature:

```
CANNOT_EVACUATE_HOST(errors)
```

CANNOT_FETCH_PATCH

The requested update could not be obtained from the master.

Signature:

```
CANNOT_FETCH_PATCH(uuid)
```

CANNOT_FIND_OEM_BACKUP_PARTITION

The backup partition to stream the update to cannot be found.

No parameters.

CANNOT_FIND_PATCH

The requested update could not be found. This can occur when you designate a new master or xe patch-clean. Please upload the update again.

No parameters.

CANNOT_FIND_STATE_PARTITION

This operation could not be performed because the state partition could not be found

No parameters.

CANNOT_FIND_UPDATE

The requested update could not be found. Please upload the update again. This can occur when you run xe update-pool-clean before xe update-apply.

No parameters.

CANNOT_FORGET_SRIOV_LOGICAL

This is a network SR-IOV logical PIF and cannot do forget on it

Signature:

```
CANNOT_FORGET_SRIOV_LOGICAL(PIF)
```

CANNOT_PLUG_BOND_SLAVE

This PIF is a bond member and cannot be plugged.

Signature:

```
CANNOT_PLUG_BOND_SLAVE(PIF)
```

CANNOT_PLUG_VIF

Cannot plug VIF

Signature:

```
CANNOT_PLUG_VIF(VIF)
```

CANNOT_RESET_CONTROL_DOMAIN

The power-state of a control domain cannot be reset.

Signature:

```

```



```
CANNOT_RESET_CONTROL_DOMAIN(vm)
```

CERTIFICATE_ALREADY_EXISTS

A certificate already exists with the specified name.

Signature:

```
CERTIFICATE_ALREADY_EXISTS(name)
```

CERTIFICATE_CORRUPT

The specified certificate is corrupt or unreadable.

Signature:

```
CERTIFICATE_CORRUPT(name)
```

CERTIFICATE_DOES_NOT_EXIST

The specified certificate does not exist.

Signature:

```
CERTIFICATE_DOES_NOT_EXIST(name)
```

CERTIFICATE_LIBRARY_CORRUPT

The certificate library is corrupt or unreadable.

No parameters.

CERTIFICATE_NAME_INVALID

The specified certificate name is invalid.

Signature:

```
CERTIFICATE_NAME_INVALID(name)
```

CHANGE_PASSWORD_REJECTED

The system rejected the password change request; perhaps the new password was too short?

Signature:

```
CHANGE_PASSWORD_REJECTED(msg)
```

CLUSTERED_SR_DEGRADED

An SR is using clustered local storage. It is not safe to reboot a host at the moment.

Signature:

```
CLUSTERED_SR_DEGRADED(sr)
```

CLUSTERING_DISABLED

An operation was attempted while clustering was disabled on the cluster_host.

Signature:

```
CLUSTERING_DISABLED(cluster_host)
```

CLUSTERING_ENABLED

An operation was attempted while clustering was enabled on the cluster_host.

Signature:

```
CLUSTERING_ENABLED(cluster_host)
```

CLUSTER_ALREADY_EXISTS

A cluster already exists in the pool.

No parameters.

CLUSTER_CREATE_IN_PROGRESS

The operation could not be performed because cluster creation is in progress.

No parameters.

CLUSTER_DOES_NOT_HAVE_ONE_NODE

An operation failed as it expected the cluster to have only one node but found multiple cluster_hosts.

Signature:

```
CLUSTER_DOES_NOT_HAVE_ONE_NODE(number_of_nodes)
```

CLUSTER_FORCE_DESTROY_FAILED

Force destroy failed on a Cluster_host while force destroying the cluster.

Signature:

```
CLUSTER_FORCE_DESTROY_FAILED(cluster)
```

CLUSTER_HOST_IS_LAST

The last cluster host cannot be destroyed. Destroy the cluster instead

Signature:

```
CLUSTER_HOST_IS_LAST(cluster_host)
```

CLUSTER_HOST_NOT_JOINED

Cluster_host operation failed as the cluster_host has not joined the cluster.

Signature:

```
CLUSTER_HOST_NOT_JOINED(cluster_host)
```

CLUSTER_STACK_IN_USE

The cluster stack is still in use by at least one plugged PBD.

Signature:

```
CLUSTER_STACK_IN_USE(cluster_stack)
```

COULD_NOT_FIND_NETWORK_INTERFACE_WITH_SPECIFIED_DEVICE_NAME_AND_MAC_ADDRESS

Could not find a network interface with the specified device name and MAC address.

Signature:

```
COULD_NOT_FIND_NETWORK_INTERFACE_WITH_SPECIFIED_DEVICE_NAME_AND_MAC_ADDRESS(device,
mac)
```

COULD_NOT_IMPORT_DATABASE

An error occurred while attempting to import a database from a metadata VDI

Signature:

```
COULD_NOT_IMPORT_DATABASE(reason)
```

COULD_NOT_UPDATE_IGMP_SNOOPING_EVERYWHERE

The IGMP Snooping setting cannot be applied for some of the host, network(s).

No parameters.

CPU_FEATURE_MASKING_NOT_SUPPORTED

The CPU does not support masking of features.

Signature:

```
CPU_FEATURE_MASKING_NOT_SUPPORTED(details)
```

CRL_ALREADY_EXISTS

A CRL already exists with the specified name.

Signature:

```
CRL_ALREADY_EXISTS(name)
```

CRL_CORRUPT

The specified CRL is corrupt or unreadable.

Signature:

```
CRL_CORRUPT(name)
```

CRL_DOES_NOT_EXIST

The specified CRL does not exist.

Signature:

```
CRL_DOES_NOT_EXIST(name)
```

CRL_NAME_INVALID

The specified CRL name is invalid.

Signature:

```
CRL_NAME_INVALID(name)
```

DB_UNIQUENESS_CONSTRAINT_VIOLATION

You attempted an operation which would have resulted in duplicate keys in the database.

Signature:

```
DB_UNIQUENESS_CONSTRAINT_VIOLATION(table, field, value)
```

DEFAULT_SR_NOT_FOUND

The default SR reference does not point to a valid SR

Signature:

```
DEFAULT_SR_NOT_FOUND(sr)
```

DEVICE_ALREADY_ATTACHED

The device is already attached to a VM

Signature:

```
DEVICE_ALREADY_ATTACHED(device)
```

DEVICE_ALREADY_DETACHED

The device is not currently attached

Signature:

```
DEVICE_ALREADY_DETACHED(device)
```

DEVICE_ALREADY_EXISTS

A device with the name given already exists on the selected VM

Signature:

```
DEVICE_ALREADY_EXISTS(device)
```

DEVICE_ATTACH_TIMEOUT

A timeout happened while attempting to attach a device to a VM.

Signature:

```
DEVICE_ATTACH_TIMEOUT(type, ref)
```

DEVICE_DETACH_REJECTED

The VM rejected the attempt to detach the device.

Signature:

```
DEVICE_DETACH_REJECTED(type, ref, msg)
```

DEVICE_DETACH_TIMEOUT

A timeout happened while attempting to detach a device from a VM.

Signature:

```
DEVICE_DETACH_TIMEOUT(type, ref)
```

DEVICE_NOT_ATTACHED

The operation could not be performed because the VBD was not connected to the VM.

Signature:

```
DEVICE_NOT_ATTACHED(VBD)
```

DISK_VBD_MUST_BE_READWRITE_FOR_HVM

All VBDs of type 'disk' must be read/write for HVM guests

Signature:

```
DISK_VBD_MUST_BE_READWRITE_FOR_HVM(vbd)
```

DOMAIN_BUILDER_ERROR

An internal error generated by the domain builder.

Signature:

```
DOMAIN_BUILDER_ERROR(function, code, message)
```

DOMAIN_EXISTS

The operation could not be performed because a domain still exists for the specified VM.

Signature:

```
DOMAIN_EXISTS(vm, domid)
```

DUPLICATE_MAC_SEED

This MAC seed is already in use by a VM in the pool

Signature:

```
DUPLICATE_MAC_SEED(seed)
```

DUPLICATE_PIF_DEVICE_NAME

A PIF with this specified device name already exists.

Signature:

```
DUPLICATE_PIF_DEVICE_NAME(device)
```

DUPLICATE_VM

Cannot restore this VM because it would create a duplicate

Signature:

```
DUPLICATE_VM(vm)
```

EVENTS_LOST

Some events have been lost from the queue and cannot be retrieved.

No parameters.

EVENT_FROM_TOKEN_PARSE_FAILURE

The event.from token could not be parsed. Valid values include: '', and a value returned from a previous event.from call.

Signature:

```
EVENT_FROM_TOKEN_PARSE_FAILURE(token)
```

EVENT_SUBSCRIPTION_PARSE_FAILURE

The server failed to parse your event subscription. Valid values include: *, class-name, class-name/object-reference.

Signature:

```
EVENT_SUBSCRIPTION_PARSE_FAILURE(subscription)
```

FAILED_TO_START_EMULATOR

An emulator required to run this VM failed to start

Signature:

```
FAILED_TO_START_EMULATOR(vm, name, msg)
```

FEATURE_REQUIRES_HVM

The VM is set up to use a feature that requires it to boot as HVM.

Signature:

```
FEATURE_REQUIRES_HVM(details)
```

FEATURE_RESTRICTED

The use of this feature is restricted.

No parameters.

FIELD_TYPE_ERROR

The value specified is of the wrong type

Signature:

```
FIELD_TYPE_ERROR(field)
```

GPU_GROUP_CONTAINS_NO_PGPUS

The GPU group does not contain any PGPUs.

Signature:

```
GPU_GROUP_CONTAINS_NO_PGPUS(gpu_group)
```

GPU_GROUP_CONTAINS_PGPU

The GPU group contains active PGPUs and cannot be deleted.

Signature:

```
GPU_GROUP_CONTAINS_PGPU(pgpus)
```

GPU_GROUP_CONTAINS_VGPU

The GPU group contains active VGPUs and cannot be deleted.

Signature:

```
GPU_GROUP_CONTAINS_VGPU(vgpus)
```

HANDLE_INVALID

You gave an invalid object reference. 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)
```

HA_ABORT_NEW_MASTER

This server cannot accept the proposed new master setting at this time.

Signature:

```
HA_ABORT_NEW_MASTER(reason)
```

HA_CANNOT_CHANGE_BOND_STATUS_OF_MGMT_IFACE

This operation cannot be performed because creating or deleting a bond involving the management interface is not allowed while HA is on. In order to do that, disable HA, create or delete the bond then re-enable HA.

No parameters.

HA_CONSTRAINT_VIOLATION_NETWORK_NOT_SHARED

This operation cannot be performed because the referenced network is not properly shared. The network must either be entirely virtual or must be physically present via a currently_attached PIF on every host.

Signature:

```
HA_CONSTRAINT_VIOLATION_NETWORK_NOT_SHARED(network)
```

HA_CONSTRAINT_VIOLATION_SR_NOT_SHARED

This operation cannot be performed because the referenced SR is not properly shared. The SR must both be marked as shared and a currently_attached PBD must exist for each host.

Signature:

```
HA_CONSTRAINT_VIOLATION_SR_NOT_SHARED(SR)
```

HA_DISABLE_IN_PROGRESS

The operation could not be performed because HA disable is in progress

No parameters.

HA_ENABLE_IN_PROGRESS

The operation could not be performed because HA enable is in progress

No parameters.

HA_FAILED_TO_FORM_LIVESET

HA could not be enabled on the Pool because a liveset could not be formed: check storage and network heartbeat paths.

No parameters.

HA_HEARTBEAT_DAEMON_STARTUP_FAILED

The server could not join the liveset because the HA daemon failed to start.

No parameters.

HA_HOST_CANNOT_ACCESS_STATEFILE

The server could not join the liveset because the HA daemon could not access the heartbeat disk.

No parameters.

HA_HOST_CANNOT_SEE_PEERS

The operation failed because the HA software on the specified server could not see a subset of other servers. Check your network connectivity.

Signature:

```
HA_HOST_CANNOT_SEE_PEERS(host, all, subset)
```

HA_HOST_IS_ARMED

The operation could not be performed while the server is still armed; it must be disarmed first.

Signature:

```
HA_HOST_IS_ARMED(host)
```

HA_IS_ENABLED

The operation could not be performed because HA is enabled on the Pool

No parameters.

HA_LOST_STATEFILE

This server lost access to the HA statefile.

No parameters.

HA_NOT_ENABLED

The operation could not be performed because HA is not enabled on the Pool

No parameters.

HA_NOT_INSTALLED

The operation could not be performed because the HA software is not installed on this server.

Signature:

```
HA_NOT_INSTALLED(host)
```

HA_NO_PLAN

Cannot find a plan for placement of VMs as there are no other servers available.

No parameters.

HA_OPERATION_WOULD_BREAK_FAILOVER_PLAN

This operation cannot be performed because it would invalidate VM failover planning such that the system would be unable to guarantee to restart protected VMs after a Host failure.

No parameters.

HA_POOL_IS_ENABLED_BUT_HOST_IS_DISABLED

This server cannot join the pool because the pool has HA enabled but this server has HA disabled.

No parameters.

HA_SHOULD_BE_FENCED

Server cannot rejoin pool because it should have fenced (it is not in the master's partition).

Signature:

```
HA_SHOULD_BE_FENCED(host)
```

HA_TOO_FEW_HOSTS

HA can only be enabled for 2 servers or more. Note that 2 servers requires a pre-configured quorum tiebreak script.

No parameters.

HOSTS_NOT_COMPATIBLE

The hosts in this pool are not compatible.

No parameters.

HOSTS_NOT_HOMOGENEOUS

The hosts in this pool are not homogeneous.

Signature:

```
HOSTS_NOT_HOMOGENEOUS(reason)
```

HOST_BROKEN

This server failed in the middle of an automatic failover operation and needs to retry the failover action.

No parameters.

HOST_CANNOT_ATTACH_NETWORK

Server cannot attach network (in the case of NIC bonding, this may be because attaching the network on this server would require other networks - that are currently active - to be taken down).

Signature:

```
HOST_CANNOT_ATTACH_NETWORK(host, network)
```

HOST_CANNOT_DESTROY_SELF

The pool master host cannot be removed.

Signature:

```
HOST_CANNOT_DESTROY_SELF(host)
```

HOST_CANNOT_READ_METRICS

The metrics of this server could not be read.

No parameters.

HOST_CD_DRIVE_EMPTY

The host CDROM drive does not contain a valid CD

No parameters.

HOST_DISABLED

The specified server is disabled.

Signature:

```
HOST_DISABLED(host)
```

HOST_DISABLED_UNTIL_REBOOT

The specified server is disabled and cannot be re-enabled until after it has rebooted.

Signature:

```
HOST_DISABLED_UNTIL_REBOOT(host)
```

HOST_EVACUATE_IN_PROGRESS

This host is being evacuated.

Signature:

```

```

```
HOST_EVACUATE_IN_PROGRESS(host)
```

HOST_HAS_NO_MANAGEMENT_IP

The server failed to acquire an IP address on its management interface and therefore cannot contact the master.

No parameters.

HOST_HAS_RESIDENT_VMS

This server cannot be forgotten because there are user VMs still running.

Signature:

```
HOST_HAS_RESIDENT_VMS(host)
```

HOST_IN_EMERGENCY_MODE

Cannot perform operation as the host is running in emergency mode.

No parameters.

HOST_IN_USE

This operation cannot be completed as the host is in use by (at least) the object of type and ref echoed below.

Signature:

```
HOST_IN_USE(host, type, ref)
```

HOST_IS_LIVE

This operation cannot be completed because the server is still live.

Signature:

```
HOST_IS_LIVE(host)
```

HOST_IS_SLAVE

You cannot make regular API calls directly on a pool member. Please pass API calls via the master host.

Signature:

```
HOST_IS_SLAVE(Master IP address)
```

HOST_ITS_OWN_SLAVE

The host is its own pool member. Please use pool-emergency-transition-to-master or pool-emergency-reset-master.

No parameters.

HOST_MASTER_CANNOT_TALK_BACK

The master reports that it cannot talk back to the pool member on the supplied management IP address.

Signature:

```
HOST_MASTER_CANNOT_TALK_BACK(ip)
```

HOST_NAME_INVALID

The server name is invalid.

Signature:

```
HOST_NAME_INVALID(reason)
```

HOST_NOT_DISABLED

This operation cannot be performed because the host is not disabled. Please disable the host and then try again.

No parameters.

HOST_NOT_ENOUGH_FREE_MEMORY

Not enough server memory is available to perform this operation.

Signature:

```
HOST_NOT_ENOUGH_FREE_MEMORY(needed, available)
```

HOST_NOT_ENOUGH_PCPUS

The host does not have enough pCPUs to run the VM. It needs at least as many as the VM has vCPUs.

Signature:

```
HOST_NOT_ENOUGH_PCPUS(vcpus, pcpus)
```

HOST_NOT_LIVE

This operation cannot be completed as the server is not live.

No parameters.

HOST_OFFLINE

You attempted an operation which involves a host which could not be contacted.

Signature:

```
HOST_OFFLINE(host)
```

HOST_POWER_ON_MODE_DISABLED

This operation cannot be completed because the server power on mode is disabled.

No parameters.

HOST_STILL_BOOTING

The host toolstack is still initialising. Please wait.

No parameters.

HOST_UNKNOWN_TO_MASTER

The master says the server is not known to it. Is the server in the master's database and pointing to the correct master? Are all servers using the same pool secret?

Signature:

```
HOST_UNKNOWN_TO_MASTER(host)
```

ILLEGAL_VBD_DEVICE

The specified VBD device is not recognized: please use a non-negative integer

Signature:

```
ILLEGAL_VBD_DEVICE(vbd, device)
```

IMPORT_ERROR

The VM could not be imported.

Signature:

```
IMPORT_ERROR(msg)
```

IMPORT_ERROR_ATTACHED_DISKS_NOT_FOUND

The VM could not be imported because attached disks could not be found.

No parameters.

IMPORT_ERROR_CANNOT_HANDLE_CHUNKED

Cannot import VM using chunked encoding.

No parameters.

IMPORT_ERROR_FAILED_TO_FIND_OBJECT

The VM could not be imported because a required object could not be found.

Signature:

```
IMPORT_ERROR_FAILED_TO_FIND_OBJECT(id)
```

IMPORT_ERROR_PREMATURE_EOF

The VM could not be imported; the end of the file was reached prematurely.

No parameters.

IMPORT_ERROR_SOME_CHECKSUMS_FAILED

Some data checksums were incorrect; the VM may be corrupt.

No parameters.

IMPORT_ERROR_UNEXPECTED_FILE

The VM could not be imported because the XVA file is invalid: an unexpected file was encountered.

Signature:

```
IMPORT_ERROR_UNEXPECTED_FILE(filename_expected, filename_found)
```

IMPORT_INCOMPATIBLE_VERSION

The import failed because this export has been created by a different (incompatible) product version

No parameters.

INCOMPATIBLE_CLUSTER_STACK_ACTIVE

This operation cannot be performed, because it is incompatible with the currently active HA cluster stack.

Signature:

```
INCOMPATIBLE_CLUSTER_STACK_ACTIVE(cluster_stack)
```

INCOMPATIBLE_PIF_PROPERTIES

These PIFs cannot be bonded, because their properties are different.

No parameters.

INCOMPATIBLE_STATEFILE_SR

The specified SR is incompatible with the selected HA cluster stack.

Signature:

```
INCOMPATIBLE_STATEFILE_SR(SR type)
```

INTERFACE_HAS_NO_IP

The specified interface cannot be used because it has no IP address

Signature:

```
INTERFACE_HAS_NO_IP(interface)
```

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)
```

INVALID_CIDR_ADDRESS_SPECIFIED

A required parameter contained an invalid CIDR address (<addr>/<prefix length>)

Signature:

```
INVALID_CIDR_ADDRESS_SPECIFIED(parameter)
```

INVALID_CLUSTER_STACK

The cluster stack provided is not supported.

Signature:

```
INVALID_CLUSTER_STACK(cluster_stack)
```

INVALID_DEVICE

The device name is invalid

Signature:

```
INVALID_DEVICE(device)
```

INVALID_EDITION

The edition you supplied is invalid.

Signature:

```
INVALID_EDITION(edition)
```

INVALID_FEATURE_STRING

The given feature string is not valid.

Signature:

```
INVALID_FEATURE_STRING(details)
```

INVALID_IP_ADDRESS_SPECIFIED

A required parameter contained an invalid IP address

Signature:

```
INVALID_IP_ADDRESS_SPECIFIED(parameter)
```

INVALID_PATCH

The uploaded patch file is invalid

No parameters.

INVALID_PATCH_WITH_LOG

The uploaded patch file is invalid. See attached log for more details.

Signature:

```
INVALID_PATCH_WITH_LOG(log)
```

INVALID_UPDATE

The uploaded update package is invalid.

Signature:

```
INVALID_UPDATE(info)
```

INVALID_VALUE

The value given is invalid

Signature:

```
INVALID_VALUE(field, value)
```

IS_TUNNEL_ACCESS_PIF

Cannot create a VLAN or tunnel on top of a tunnel access PIF - use the underlying transport PIF instead.

Signature:

```
IS_TUNNEL_ACCESS_PIF(PIF)
```

JOINING_HOST_CANNOT_BE_MASTER_OF_OTHER_HOSTS

The server joining the pool cannot already be a master of another pool.

No parameters.

JOINING_HOST_CANNOT_CONTAIN_SHARED_SRS

The server joining the pool cannot contain any shared storage.

No parameters.

JOINING_HOST_CANNOT_HAVE_RUNNING_OR_SUSPENDED_VMS

The server joining the pool cannot have any running or suspended VMs.

No parameters.

JOINING_HOST_CANNOT_HAVE_RUNNING_VMS

The server joining the pool cannot have any running VMs.

No parameters.

JOINING_HOST_CANNOT_HAVE_VMS_WITH_CURRENT_OPERATIONS

The host joining the pool cannot have any VMs with active tasks.

No parameters.

JOINING_HOST_CONNECTION_FAILED

There was an error connecting to the host while joining it in the pool.

No parameters.

JOINING_HOST_SERVICE_FAILED

There was an error connecting to the server. The service contacted didn't reply properly.

No parameters.

LICENCE_RESTRICTION

This operation is not allowed because your license lacks a needed feature. Please contact your support representative.

Signature:

```
LICENCE_RESTRICTION(feature)
```

LICENSE_CANNOT_DOWNGRADE_WHILE_IN_POOL

Cannot downgrade license while in pool. Please disband the pool first, then downgrade licenses on hosts separately.

No parameters.

LICENSE_CHECKOUT_ERROR

The license for the edition you requested is not available.

Signature:

```
LICENSE_CHECKOUT_ERROR(reason)
```

LICENSE_DOES_NOT_SUPPORT_POOLING

This server cannot join a pool because its license does not support pooling.

No parameters.

LICENSE_DOES_NOT_SUPPORT_XHA

HA cannot be enabled because this server's license does not allow it.

No parameters.

LICENSE_EXPIRED

Your license has expired. Please contact your support representative.

No parameters.

LICENSE_FILE_DEPRECATED

This type of license file is for previous versions of the server. Please upgrade to the new licensing system.

No parameters.

LICENSE_HOST_POOL_MISMATCH

Host and pool have incompatible licenses (editions).

No parameters.

LICENSE_PROCESSING_ERROR

There was an error processing your license. Please contact your support representative.

No parameters.

LOCATION_NOT_UNIQUE

A VDI with the specified location already exists within the SR

Signature:

```
LOCATION_NOT_UNIQUE(SR, location)
```

MAC_DOES_NOT_EXIST

The MAC address specified does not exist on this server.

Signature:

```
MAC_DOES_NOT_EXIST(MAC)
```

MAC_INVALID

The MAC address specified is not valid.

Signature:

```
MAC_INVALID(MAC)
```

MAC_STILL_EXISTS

The MAC address specified still exists on this server.

Signature:

```
MAC_STILL_EXISTS(MAC)
```

MAP_DUPLICATE_KEY

You tried to add a key-value pair to a map, but that key is already there.

Signature:

```
MAP_DUPLICATE_KEY(type, param_name, uuid, key)
```

MEMORY_CONSTRAINT_VIOLATION

The dynamic memory range does not satisfy the following constraint.

Signature:

```
MEMORY_CONSTRAINT_VIOLATION(constraint)
```

MEMORY_CONSTRAINT_VIOLATION_MAXPIN

The dynamic memory range violates constraint static_min = dynamic_min = dynamic_max = static_max.

Signature:

```
MEMORY_CONSTRAINT_VIOLATION_MAXPIN(reason)
```

MEMORY_CONSTRAINT_VIOLATION_ORDER

The dynamic memory range violates constraint static_min <= dynamic_min <= dynamic_max <= static_max.

No parameters.

MESSAGE_DEPRECATED

This message has been deprecated.

No parameters.

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)
```

MESSAGE_REMOVED

This function is no longer available.

No parameters.

MIRROR_FAILED

The VDI mirroring cannot be performed

Signature:

```
MIRROR_FAILED(vdi)
```

MISSING_CONNECTION_DETAILS

The license-server connection details (address or port) were missing or incomplete.

No parameters.

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)
```

NETWORK_CONTAINS_PIF

The network contains active PIFs and cannot be deleted.

Signature:

```
NETWORK_CONTAINS_PIF(pifs)
```

NETWORK_CONTAINS_VIF

The network contains active VIFs and cannot be deleted.

Signature:

```
NETWORK_CONTAINS_VIF(vifs)
```

NETWORK_HAS_INCOMPATIBLE_SRIOV_PIFS

The PIF is not compatible with the selected SR-IOV network

Signature:

```
NETWORK_HAS_INCOMPATIBLE_SRIOV_PIFS(PIF, network)
```

NETWORK_HAS_INCOMPATIBLE_VLAN_ON_SRIOV_PIFS

VLAN on the PIF is not compatible with the selected SR-IOV VLAN network

Signature:

```
NETWORK_HAS_INCOMPATIBLE_VLAN_ON_SRIOV_PIFS(PIF, network)
```

NETWORK_INCOMPATIBLE_PURPOSES

You tried to add a purpose to a network but the new purpose is not compatible with an existing purpose of the network or other networks.

Signature:

```
NETWORK_INCOMPATIBLE_PURPOSES(new_purpose, conflicting_purpose)
```

NETWORK_INCOMPATIBLE_WITH_BOND

The network is incompatible with bond

Signature:

```
NETWORK_INCOMPATIBLE_WITH_BOND(network)
```

NETWORK_INCOMPATIBLE_WITH_SRIOV

The network is incompatible with sriov

Signature:

```
NETWORK_INCOMPATIBLE_WITH_SRIOV(network)
```

NETWORK_INCOMPATIBLE_WITH_TUNNEL

The network is incompatible with tunnel

Signature:

```
NETWORK_INCOMPATIBLE_WITH_TUNNEL(network)
```

NETWORK_INCOMPATIBLE_WITH_VLAN_ON_BRIDGE

The network is incompatible with vlan on bridge

Signature:

```
NETWORK_INCOMPATIBLE_WITH_VLAN_ON_BRIDGE(network)
```

NETWORK_INCOMPATIBLE_WITH_VLAN_ON_SRIOV

The network is incompatible with vlan on sriov

Signature:

```
NETWORK_INCOMPATIBLE_WITH_VLAN_ON_SRIOV(network)
```

NETWORK_SRIOV_ALREADY_ENABLED

The PIF selected for the SR-IOV network is already enabled

Signature:

```
NETWORK_SRIOV_ALREADY_ENABLED(PIF)
```

NETWORK_SRIOV_DISABLE_FAILED

Failed to disable SR-IOV on PIF

Signature:

```
NETWORK_SRIOV_DISABLE_FAILED(PIF, msg)
```

NETWORK_SRIOV_ENABLE_FAILED

Failed to enable SR-IOV on PIF

Signature:

```
NETWORK_SRIOV_ENABLE_FAILED(PIF, msg)
```

NETWORK_SRIOV_INSUFFICIENT_CAPACITY

There is insufficient capacity for VF reservation

Signature:

```
NETWORK_SRIOV_INSUFFICIENT_CAPACITY(network)
```

NETWORK_UNMANAGED

The network is not managed by xapi.

Signature:

```
NETWORK_UNMANAGED(network)
```

NOT_ALLOWED_ON_OEM_EDITION

This command is not allowed on the OEM edition.

Signature:

```
NOT_ALLOWED_ON_OEM_EDITION(command)
```

NOT_IMPLEMENTED

The function is not implemented

Signature:

```

```

```
NOT_IMPLEMENTED(function)
```

NOT_IN_EMERGENCY_MODE

This pool is not in emergency mode.

No parameters.

NOT_SUPPORTED_DURING_UPGRADE

This operation is not supported during an upgrade.

No parameters.

NOT_SYSTEM_DOMAIN

The given VM is not registered as a system domain. This operation can only be performed on a registered system domain.

Signature:

```
NOT_SYSTEM_DOMAIN(vm)
```

NO_CLUSTER_HOSTS_REACHABLE

No other cluster host was reachable when joining

Signature:

```
NO_CLUSTER_HOSTS_REACHABLE(cluster)
```

NO_COMPATIBLE_CLUSTER_HOST

Clustering is not enabled on this host or pool.

Signature:

```
NO_COMPATIBLE_CLUSTER_HOST(host)
```

NO_HOSTS_AVAILABLE

There were no servers available to complete the specified operation.

No parameters.

NO_MORE_REDO_LOGS_ALLOWED

The upper limit of active redo log instances was reached.

No parameters.

NVIDIA_SRIOV_MISCONFIGURED

The NVidia GPU is not configured for SR-IOV as expected

Signature:

```
NVIDIA_SRIOV_MISCONFIGURED(host, device_name)
```

NVIDIA_TOOLS_ERROR

Nvidia tools error. Please ensure that the latest Nvidia tools are installed

Signature:

```
NVIDIA_TOOLS_ERROR(host)
```

OBJECT_NOLONGER_EXISTS

The specified object no longer exists.

No parameters.

ONLY_ALLOWED_ON_OEM_EDITION

This command is only allowed on the OEM edition.

Signature:

```
ONLY_ALLOWED_ON_OEM_EDITION(command)
```

OPENVSWITCH_NOT_ACTIVE

This operation needs the OpenVSwitch networking backend to be enabled on all hosts in the pool.

No parameters.

OPERATION_BLOCKED

You attempted an operation that was explicitly blocked (see the blocked_operations field of the given object).

Signature:

```
OPERATION_BLOCKED(ref, code)
```

OPERATION_NOT_ALLOWED

You attempted an operation that was not allowed.

Signature:

```
OPERATION_NOT_ALLOWED(reason)
```

OPERATION_PARTIALLY_FAILED

Some VMs belonging to the appliance threw an exception while carrying out the specified operation

Signature:

```
OPERATION_PARTIALLY_FAILED(operation)
```

OTHER_OPERATION_IN_PROGRESS

Another operation involving the object is currently in progress

Signature:

```
OTHER_OPERATION_IN_PROGRESS(class, object)
```

OUT_OF_SPACE

There is not enough space to upload the update

Signature:

```
OUT_OF_SPACE(location)
```

PATCH_ALREADY_APPLIED

This patch has already been applied

Signature:

```
PATCH_ALREADY_APPLIED(patch)
```

PATCH_ALREADY_EXISTS

The uploaded patch file already exists

Signature:

```
PATCH_ALREADY_EXISTS(uuid)
```

PATCH_APPLY_FAILED

The patch apply failed. Please see attached output.

Signature:

```
PATCH_APPLY_FAILED(output)
```

PATCH_APPLY_FAILED_BACKUP_FILES_EXIST

The patch apply failed: there are backup files created while applying patch. Please remove these backup files before applying patch again.

Signature:

```
PATCH_APPLY_FAILED_BACKUP_FILES_EXIST(output)
```

PATCH_IS_APPLIED

The specified patch is applied and cannot be destroyed.

No parameters.

PATCH_PRECHECK_FAILED_ISO_MOUNTED

Tools ISO must be ejected from all running VMs.

Signature:

```
PATCH_PRECHECK_FAILED_ISO_MOUNTED(patch)
```

PATCH_PRECHECK_FAILED_OUT_OF_SPACE

The patch pre-check stage failed: the server does not have enough space.

Signature:

```
PATCH_PRECHECK_FAILED_OUT_OF_SPACE(patch, found_space, required_required)
```

PATCH_PRECHECK_FAILED_PREREQUISITE_MISSING

The patch pre-check stage failed: prerequisite patches are missing.

Signature:

```
PATCH_PRECHECK_FAILED_PREREQUISITE_MISSING(patch, prerequisite_patch_uuid_list)
```

PATCH_PRECHECK_FAILED_UNKNOWN_ERROR

The patch pre-check stage failed with an unknown error. See attached info for more details.

Signature:

```
PATCH_PRECHECK_FAILED_UNKNOWN_ERROR(patch, info)
```

PATCH_PRECHECK_FAILED_VM_RUNNING

The patch pre-check stage failed: there are one or more VMs still running on the server. All VMs must be suspended before the patch can be applied.

Signature:

```
PATCH_PRECHECK_FAILED_VM_RUNNING(patch)
```

PATCH_PRECHECK_FAILED_WRONG_SERVER_BUILD

The patch pre-check stage failed: the server is of an incorrect build.

Signature:

```
PATCH_PRECHECK_FAILED_WRONG_SERVER_BUILD(patch, found_build, required_build)
```

PATCH_PRECHECK_FAILED_WRONG_SERVER_VERSION

The patch pre-check stage failed: the server is of an incorrect version.

Signature:

```
PATCH_PRECHECK_FAILED_WRONG_SERVER_VERSION(patch, found_version, required_version)
```

PBD_EXISTS

A PBD already exists connecting the SR to the server.

Signature:

```
PBD_EXISTS(sr, host, pbd)
```

PERMISSION_DENIED

Caller not allowed to perform this operation.

Signature:

```
PERMISSION_DENIED(message)
```

PGPU_INSUFFICIENT_CAPACITY_FOR_VGPU

There is insufficient capacity on this PGPU to run the VGPU.

Signature:

```
PGPU_INSUFFICIENT_CAPACITY_FOR_VGPU(pgpu, vgpu_type)
```

PGPU_IN_USE_BY_VM

This PGPU is currently in use by running VMs.

Signature:

```
PGPU_IN_USE_BY_VM(VMs)
```

PGPU_NOT_COMPATIBLE_WITH_GPU_GROUP

PGPU type not compatible with destination group.

Signature:

```
PGPU_NOT_COMPATIBLE_WITH_GPU_GROUP(type, group_types)
```

PIF_ALLOWS_UNPLUG

The operation you requested cannot be performed because the specified PIF allows unplug.

Signature:

```
PIF_ALLOWS_UNPLUG(PIF)
```

PIF_ALREADY_BONDED

This operation cannot be performed because the pif is bonded.

Signature:

```
PIF_ALREADY_BONDED(PIF)
```

PIF_BOND_MORE_THAN_ONE_IP

Only one PIF on a bond is allowed to have an IP configuration.

No parameters.

PIF_BOND_NEEDS_MORE_MEMBERS

A bond must consist of at least two member interfaces

No parameters.

PIF_CANNOT_BOND_CROSS_HOST

You cannot bond interfaces across different servers.

No parameters.

PIF_CONFIGURATION_ERROR

An unknown error occurred while attempting to configure an interface.

Signature:

```
PIF_CONFIGURATION_ERROR(PIF, msg)
```

PIF_DEVICE_NOT_FOUND

The specified device was not found.

No parameters.

PIF_DOES_NOT_ALLOW_UNPLUG

The operation you requested cannot be performed because the specified PIF does not allow unplug.

Signature:

```
PIF_DOES_NOT_ALLOW_UNPLUG(PIF)
```

PIF_HAS_FCOE_SR_IN_USE

The operation you requested cannot be performed because the specified PIF has FCoE SR in use.

Signature:

```
PIF_HAS_FCOE_SR_IN_USE(PIF, SR)
```

PIF_HAS_NO_NETWORK_CONFIGURATION

PIF has no IP configuration (mode currently set to 'none')

Signature:

```
PIF_HAS_NO_NETWORK_CONFIGURATION(PIF)
```

PIF_HAS_NO_V6_NETWORK_CONFIGURATION

PIF has no IPv6 configuration (mode currently set to 'none')

Signature:

```
PIF_HAS_NO_V6_NETWORK_CONFIGURATION(PIF)
```

PIF_INCOMPATIBLE_PRIMARY_ADDRESS_TYPE

The primary address types are not compatible

Signature:

```
PIF_INCOMPATIBLE_PRIMARY_ADDRESS_TYPE(PIF)
```

PIF_IS_MANAGEMENT_INTERFACE

The operation you requested cannot be performed because the specified PIF is the management interface.

Signature:

```
PIF_IS_MANAGEMENT_INTERFACE(PIF)
```

PIF_IS_NOT_PHYSICAL

You tried to perform an operation which is only available on physical PIF

Signature:

```
PIF_IS_NOT_PHYSICAL(PIF)
```

PIF_IS_NOT_SRIOV_CAPABLE

The selected PIF is not capable of network SR-IOV

Signature:

```
PIF_IS_NOT_SRIOV_CAPABLE(PIF)
```

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)
```

PIF_IS_SRIOV_LOGICAL

You tried to create a bond on top of a network SR-IOV logical PIF - use the underlying physical PIF instead

Signature:

```
PIF_IS_SRIOV_LOGICAL(PIF)
```

PIF_IS_VLAN

You tried to create a VLAN on top of another VLAN - use the underlying physical PIF/bond instead

Signature:

```
PIF_IS_VLAN(PIF)
```

PIF_NOT_ATTACHED_TO_HOST

Cluster_host creation failed as the PIF provided is not attached to the host.

Signature:

```
PIF_NOT_ATTACHED_TO_HOST(pif, host)
```

PIF_NOT_PRESENT

This host has no PIF on the given network.

Signature:

```
PIF_NOT_PRESENT(host, network)
```

PIF_SRIOV_STILL_EXISTS

The PIF is still related with a network SR-IOV

Signature:

```
PIF_SRIOV_STILL_EXISTS(PIF)
```

PIF_TUNNEL_STILL_EXISTS

Operation cannot proceed while a tunnel exists on this interface.

Signature:

```
PIF_TUNNEL_STILL_EXISTS(PIF)
```

PIF_UNMANAGED

The operation you requested cannot be performed because the specified PIF is not managed by xapi.

Signature:

```
PIF_UNMANAGED(PIF)
```

PIF_VLAN_EXISTS

You tried to create a PIF, but it already exists.

Signature:

```
PIF_VLAN_EXISTS(PIF)
```

PIF_VLAN_STILL_EXISTS

Operation cannot proceed while a VLAN exists on this interface.

Signature:

```
PIF_VLAN_STILL_EXISTS(PIF)
```

POOL_AUTH_ALREADY_ENABLED

External authentication is already enabled for at least one server in this pool.

Signature:

```
POOL_AUTH_ALREADY_ENABLED(host)
```

POOL_AUTH_DISABLE_FAILED

The pool failed to disable the external authentication of at least one host.

Signature:

```
POOL_AUTH_DISABLE_FAILED(host, message)
```

POOL_AUTH_DISABLE_FAILED_INVALID_ACCOUNT

External authentication has been disabled with errors: Some AD machine accounts were not disabled on the AD server due to invalid account.

Signature:

```
POOL_AUTH_DISABLE_FAILED_INVALID_ACCOUNT(host, message)
```

POOL_AUTH_DISABLE_FAILED_PERMISSION_DENIED

External authentication has been disabled with errors: Your AD machine account was not disabled on the AD server as permission was denied.

Signature:

```
POOL_AUTH_DISABLE_FAILED_PERMISSION_DENIED(host, message)
```

POOL_AUTH_DISABLE_FAILED_WRONG_CREDENTIALS

External authentication has been disabled with errors: Some AD machine accounts were not disabled on the AD server due to invalid credentials.

Signature:

```
POOL_AUTH_DISABLE_FAILED_WRONG_CREDENTIALS(host, message)
```

POOL_AUTH_ENABLE_FAILED

The pool failed to enable external authentication.

Signature:

```
POOL_AUTH_ENABLE_FAILED(host, message)
```

POOL_AUTH_ENABLE_FAILED_DOMAIN_LOOKUP_FAILED

The pool failed to enable external authentication.

Signature:

```
POOL_AUTH_ENABLE_FAILED_DOMAIN_LOOKUP_FAILED(host, message)
```

POOL_AUTH_ENABLE_FAILED_DUPLICATE_HOSTNAME

The pool failed to enable external authentication.

Signature:

```
POOL_AUTH_ENABLE_FAILED_DUPLICATE_HOSTNAME(host, message)
```

POOL_AUTH_ENABLE_FAILED_INVALID_ACCOUNT

The pool failed to enable external authentication.

Signature:

```
POOL_AUTH_ENABLE_FAILED_INVALID_ACCOUNT(host, message)
```

POOL_AUTH_ENABLE_FAILED_INVALID_OU

The pool failed to enable external authentication.

Signature:

```
POOL_AUTH_ENABLE_FAILED_INVALID_OU(host, message)
```

POOL_AUTH_ENABLE_FAILED_PERMISSION_DENIED

The pool failed to enable external authentication.

Signature:

```
POOL_AUTH_ENABLE_FAILED_PERMISSION_DENIED(host, message)
```

POOL_AUTH_ENABLE_FAILED_UNAVAILABLE

The pool failed to enable external authentication.

Signature:

```
POOL_AUTH_ENABLE_FAILED_UNAVAILABLE(host, message)
```

POOL_AUTH_ENABLE_FAILED_WRONG_CREDENTIALS

The pool failed to enable external authentication.

Signature:

```
POOL_AUTH_ENABLE_FAILED_WRONG_CREDENTIALS(host, message)
```

POOL_JOINING_EXTERNAL_AUTH_MISMATCH

Cannot join pool whose external authentication configuration is different.

No parameters.

POOL_JOINING_HOST_HAS_BONDS

The host joining the pool must not have any bonds.

No parameters.

POOL_JOINING_HOST_HAS_NETWORK_SRIOVS

The host joining the pool must not have any network SR-IOVs.

No parameters.

POOL_JOINING_HOST_HAS_NON_MANAGEMENT_VLANS

The host joining the pool must not have any non-management vlans.

No parameters.

POOL_JOINING_HOST_HAS_TUNNELS

The host joining the pool must not have any tunnels.

No parameters.

POOL_JOINING_HOST_MANAGEMENT_VLAN_DOES_NOT_MATCH

The host joining the pool must have the same management vlan.

Signature:

```
POOL_JOINING_HOST_MANAGEMENT_VLAN_DOES_NOT_MATCH(local, remote)
```

POOL_JOINING_HOST_MUST_HAVE_PHYSICAL_MANAGEMENT_NIC

The server joining the pool must have a physical management NIC (i.e. the management NIC must not be on a VLAN or bonded PIF).

No parameters.

POOL_JOINING_HOST_MUST_HAVE_SAME_API_VERSION

The host joining the pool must have the same API version as the pool master.

Signature:

```
POOL_JOINING_HOST_MUST_HAVE_SAME_API_VERSION(host_api_version, master_api_version)
```

POOL_JOINING_HOST_MUST_HAVE_SAME_DB_SCHEMA

The host joining the pool must have the same database schema as the pool master.

Signature:

```
POOL_JOINING_HOST_MUST_HAVE_SAME_DB_SCHEMA(host_db_schema, master_db_schema)
```

POOL_JOINING_HOST_MUST_HAVE_SAME_PRODUCT_VERSION

The server joining the pool must have the same product version as the pool master.

No parameters.

POOL_JOINING_HOST_MUST_ONLY_HAVE_PHYSICAL_PIFS

The host joining the pool must not have any bonds, VLANs or tunnels.

No parameters.

PROVISION_FAILED_OUT_OF_SPACE

The provision call failed because it ran out of space.

No parameters.

PROVISION_ONLY_ALLOWED_ON_TEMPLATE

The provision call can only be invoked on templates, not regular VMs.

No parameters.

PUSB_VDI_CONFLICT

The VDI corresponding to this PUSB has existing VBDs.

Signature:

```
PUSB_VDI_CONFLICT(PUSB, VDI)
```

PVS_CACHE_STORAGE_ALREADY_PRESENT

The PVS site already has cache storage configured for the host.

Signature:

```
PVS_CACHE_STORAGE_ALREADY_PRESENT(site, host)
```

PVS_CACHE_STORAGE_IS_IN_USE

The PVS cache storage is in use by the site and cannot be removed.

Signature:

```
PVS_CACHE_STORAGE_IS_IN_USE(PVS_cache_storage)
```

PVS_PROXY_ALREADY_PRESENT

The VIF is already associated with a PVS proxy

Signature:

```
PVS_PROXY_ALREADY_PRESENT(proxies)
```

PVS_SERVER_ADDRESS_IN_USE

The address specified is already in use by an existing PVS_server object

Signature:

```
PVS_SERVER_ADDRESS_IN_USE(address)
```

PVS_SITE_CONTAINS_RUNNING_PROXIES

The PVS site contains running proxies.

Signature:

```

```

```
PVS_SITE_CONTAINS_RUNNING_PROXIES(proxies)
```

PVS_SITE_CONTAINS_SERVERS

The PVS site contains servers and cannot be forgotten.

Signature:

```
PVS_SITE_CONTAINS_SERVERS(servers)
```

RBAC_PERMISSION_DENIED

RBAC permission denied.

Signature:

```
RBAC_PERMISSION_DENIED(permission, message)
```

REDO_LOG_IS_ENABLED

The operation could not be performed because a redo log is enabled on the Pool.

No parameters.

REQUIRED_PIF_IS_UNPLUGGED

The operation you requested cannot be performed because the specified PIF is currently unplugged.

Signature:

```
REQUIRED_PIF_IS_UNPLUGGED(PIF)
```

RESTORE_INCOMPATIBLE_VERSION

The restore could not be performed because this backup has been created by a different (incompatible) product version

No parameters.

RESTORE_SCRIPT_FAILED

The restore could not be performed because the restore script failed. Is the file corrupt?

Signature:

```
RESTORE_SCRIPT_FAILED(log)
```

RESTORE_TARGET_MGMT_IF_NOT_IN_BACKUP

The restore could not be performed because the server's current management interface is not in the backup. The interfaces mentioned in the backup are:

No parameters.

RESTORE_TARGET_MISSING_DEVICE

The restore could not be performed because a network interface is missing

Signature:

```
RESTORE_TARGET_MISSING_DEVICE(device)
```

ROLE_ALREADY_EXISTS

Role already exists.

No parameters.

ROLE_NOT_FOUND

Role cannot be found.

No parameters.

SERVER_CERTIFICATE_CHAIN_INVALID

The provided intermediate certificates are not in a pem-encoded X509.

No parameters.

SERVER_CERTIFICATE_EXPIRED

The provided certificate has expired.

Signature:

```
SERVER_CERTIFICATE_EXPIRED(now, not_after)
```

SERVER_CERTIFICATE_INVALID

The provided certificate is not in a pem-encoded X509.

No parameters.

SERVER_CERTIFICATE_KEY_ALGORITHM_NOT_SUPPORTED

The provided key uses an unsupported algorithm.

Signature:

```
SERVER_CERTIFICATE_KEY_ALGORITHM_NOT_SUPPORTED(algorithm_oid)
```

SERVER_CERTIFICATE_KEY_INVALID

The provided key is not in a pem-encoded PKCS#8 format.

No parameters.

SERVER_CERTIFICATE_KEY_MISMATCH

The provided key does not match the provided certificate's public key.

No parameters.

SERVER_CERTIFICATE_KEY_RSA_LENGTH_NOT_SUPPORTED

The provided RSA key does not have a length between 2048 and 4096.

Signature:

```
SERVER_CERTIFICATE_KEY_RSA_LENGTH_NOT_SUPPORTED(length)
```

SERVER_CERTIFICATE_KEY_RSA_MULTI_NOT_SUPPORTED

The provided RSA key is using more than 2 primes, expecting only 2.

No parameters.

SERVER_CERTIFICATE_NOT_VALID_YET

The provided certificate is not valid yet.

Signature:

```
SERVER_CERTIFICATE_NOT_VALID_YET(now, not_before)
```

SERVER_CERTIFICATE_SIGNATURE_NOT_SUPPORTED

The provided certificate is not using the SHA256 (SHA2) signature algorithm.

No parameters.

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 reference. 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)
```

SLAVE_REQUIRES_MANAGEMENT_INTERFACE

The management interface on a pool member cannot be disabled because the pool member would enter emergency mode.

No parameters.

SM_PLUGIN_COMMUNICATION_FAILURE

The SM plug-in did not respond to a query.

Signature:

```
SM_PLUGIN_COMMUNICATION_FAILURE(sm)
```

SR_ATTACH_FAILED

Attaching this SR failed.

Signature:

```
SR_ATTACH_FAILED(sr)
```

SR_BACKEND_FAILURE

There was an SR backend failure.

Signature:

```
SR_BACKEND_FAILURE(status, stdout, stderr)
```

SR_DEVICE_IN_USE

The SR operation cannot be performed because a device underlying the SR is in use by the server.

No parameters.

SR_DOES_NOT_SUPPORT_MIGRATION

Cannot migrate a VDI to or from an SR that doesn't support migration.

Signature:

```
SR_DOES_NOT_SUPPORT_MIGRATION(sr)
```

SR_FULL

The SR is full. Requested new size exceeds the maximum size

Signature:

```
SR_FULL(requested, maximum)
```

SR_HAS_MULTIPLE_PBDS

The SR.shared flag cannot be set to false while the SR remains connected to multiple servers.

Signature:

```
SR_HAS_MULTIPLE_PBDS(PBD)
```

SR_HAS_NO_PBDS

The SR has no attached PBDs

Signature:

```
SR_HAS_NO_PBDS(sr)
```

SR_HAS_PBD

The SR is still connected to a host via a PBD. It cannot be destroyed or forgotten.

Signature:

```
SR_HAS_PBD(sr)
```

SR_INDESTRUCTIBLE

The SR could not be destroyed because the 'indestructible' flag was set on it.

Signature:

```
SR_INDESTRUCTIBLE(sr)
```

SR_IS_CACHE_SR

The SR is currently being used as a local cache SR.

Signature:

```
SR_IS_CACHE_SR(host)
```

SR_NOT_ATTACHED

The SR is not attached.

Signature:

```
SR_NOT_ATTACHED(sr)
```

SR_NOT_EMPTY

The SR operation cannot be performed because the SR is not empty.

No parameters.

SR_NOT_SHARABLE

The PBD could not be plugged because the SR is in use by another host and is not marked as sharable.

Signature:

```
SR_NOT_SHARABLE(sr, host)
```

SR_OPERATION_NOT_SUPPORTED

The SR backend does not support the operation (check the SR's allowed operations)

Signature:

```
SR_OPERATION_NOT_SUPPORTED(sr)
```

SR_REQUIRES_UPGRADE

The operation cannot be performed until the SR has been upgraded

Signature:

```
SR_REQUIRES_UPGRADE(SR)
```

SR_SOURCE_SPACE_INSUFFICIENT

The source SR does not have sufficient temporary space available to proceed the operation.

Signature:

```
SR_SOURCE_SPACE_INSUFFICIENT(sr)
```

SR_UNKNOWN_DRIVER

The SR could not be connected because the driver was not recognised.

Signature:

```
SR_UNKNOWN_DRIVER(driver)
```

SR_UUID_EXISTS

An SR with that uuid already exists.

Signature:

```
SR_UUID_EXISTS(uuid)
```

SR_VDI_LOCKING_FAILED

The operation could not proceed because necessary VDIs were already locked at the storage level.

No parameters.

SSL_VERIFY_ERROR

The remote system's SSL certificate failed to verify against our certificate library.

Signature:

```
SSL_VERIFY_ERROR(reason)
```

SUBJECT_ALREADY_EXISTS

Subject already exists.

No parameters.

SUBJECT_CANNOT_BE_RESOLVED

Subject cannot be resolved by the external directory service.

No parameters.

SUSPEND_IMAGE_NOT_ACCESSIBLE

The suspend image of a checkpoint is not accessible from the host on which the VM is running

Signature:

```
SUSPEND_IMAGE_NOT_ACCESSIBLE(vdi)
```

SYSTEM_STATUS_MUST_USE_TAR_ON_OEM

You must use tar output to retrieve system status from an OEM server.

No parameters.

SYSTEM_STATUS_RETRIEVAL_FAILED

Retrieving system status from the host failed. A diagnostic reason suitable for support organisations is also returned.

Signature:

```
SYSTEM_STATUS_RETRIEVAL_FAILED(reason)
```

TASK_CANCELLED

The request was asynchronously canceled.

Signature:

```
TASK_CANCELLED(task)
```

TLS_CONNECTION_FAILED

Cannot contact the other host using TLS on the specified address and port

Signature:

```
TLS_CONNECTION_FAILED(address, port)
```

TOO_BUSY

The request was rejected because the server is too busy.

No parameters.

TOO_MANY_PENDING_TASKS

The request was rejected because there are too many pending tasks on the server.

No parameters.

TOO_MANY_STORAGE_MIGRATES

You reached the maximal number of concurrently migrating VMs.

Signature:

```
TOO_MANY_STORAGE_MIGRATES(number)
```

TOO_MANY_VUSBS

The VM has too many VUSBs.

Signature:

```
TOO_MANY_VUSBS(number)
```

TRANSPORT_PIF_NOT_CONFIGURED

The tunnel transport PIF has no IP configuration set.

Signature:

```
TRANSPORT_PIF_NOT_CONFIGURED(PIF)
```

UNIMPLEMENTED_IN_SM_BACKEND

You have attempted a function which is not implemented

Signature:

```
UNIMPLEMENTED_IN_SM_BACKEND(message)
```

UNKNOWN_BOOTLOADER

The requested bootloader is unknown

Signature:

```
UNKNOWN_BOOTLOADER(vm, bootloader)
```

UPDATE_ALREADY_APPLIED

This update has already been applied.

Signature:

```
UPDATE_ALREADY_APPLIED(update)
```

UPDATE_ALREADY_APPLIED_IN_POOL

This update has already been applied to all hosts in the pool.

Signature:

```
UPDATE_ALREADY_APPLIED_IN_POOL(update)
```

UPDATE_ALREADY_EXISTS

The uploaded update already exists

Signature:

```
UPDATE_ALREADY_EXISTS(uuid)
```

UPDATE_APPLY_FAILED

The update failed to apply. Please see attached output.

Signature:

```
UPDATE_APPLY_FAILED(output)
```

UPDATE_IS_APPLIED

The specified update has been applied and cannot be destroyed.

No parameters.

UPDATE_POOL_APPLY_FAILED

The update cannot be applied for the following host(s).

Signature:

```
UPDATE_POOL_APPLY_FAILED(hosts)
```

UPDATE_PRECHECK_FAILED_CONFLICT_PRESENT

The update pre-check stage failed: conflicting update(s) are present.

Signature:

```
UPDATE_PRECHECK_FAILED_CONFLICT_PRESENT(update, conflict_update)
```

UPDATE_PRECHECK_FAILED_GPGKEY_NOT_IMPORTED

The update pre-check stage failed: RPM package validation requires a GPG key that is not present on the host.

Signature:

```
UPDATE_PRECHECK_FAILED_GPGKEY_NOT_IMPORTED(update)
```

UPDATE_PRECHECK_FAILED_OUT_OF_SPACE

The update pre-check stage failed: the server does not have enough space.

Signature:

```
UPDATE_PRECHECK_FAILED_OUT_OF_SPACE(update, available_space, required_space )
```

UPDATE_PRECHECK_FAILED_PREREQUISITE_MISSING

The update pre-check stage failed: prerequisite update(s) are missing.

Signature:

```
UPDATE_PRECHECK_FAILED_PREREQUISITE_MISSING(update, prerequisite_update)
```

UPDATE_PRECHECK_FAILED_UNKNOWN_ERROR

The update pre-check stage failed with an unknown error.

Signature:

```
UPDATE_PRECHECK_FAILED_UNKNOWN_ERROR(update, info)
```

UPDATE_PRECHECK_FAILED_WRONG_SERVER_VERSION

The update pre-check stage failed: the server is of an incorrect version.

Signature:

```
UPDATE_PRECHECK_FAILED_WRONG_SERVER_VERSION(update, installed_version,
required_version )
```

USB_ALREADY_ATTACHED

The USB device is currently attached to a VM.

Signature:

```
USB_ALREADY_ATTACHED(PUSB, VM)
```

USB_GROUP_CONFLICT

USB_groups are currently restricted to contain no more than one VUSB.

Signature:

```
USB_GROUP_CONFLICT(USB_group)
```

USB_GROUP_CONTAINS_NO_PUSBS

The USB group does not contain any PUSBs.

Signature:

```
USB_GROUP_CONTAINS_NO_PUSBS(usb_group)
```

USB_GROUP_CONTAINS_PUSB

The USB group contains active PUSBs and cannot be deleted.

Signature:

```
USB_GROUP_CONTAINS_PUSB(pusbs)
```

USB_GROUP_CONTAINS_VUSB

The USB group contains active VUSBs and cannot be deleted.

Signature:

```
USB_GROUP_CONTAINS_VUSB(vusbs)
```

USER_IS_NOT_LOCAL_SUPERUSER

Only the local superuser can perform this operation.

Signature:

```
USER_IS_NOT_LOCAL_SUPERUSER(msg)
```

UUID_INVALID

The uuid you supplied was invalid.

Signature:

```
UUID_INVALID(type, uuid)
```

V6D_FAILURE

There was a problem with the license daemon (v6d).

No parameters.

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)
```

VBD_CDS_MUST_BE_READONLY

Read/write CDs are not supported

No parameters.

VBD_IS_EMPTY

Operation could not be performed because the drive is empty

Signature:

```
VBD_IS_EMPTY(vbd)
```

VBD_NOT_EMPTY

Operation could not be performed because the drive is not empty

Signature:

```
VBD_NOT_EMPTY(vbd)
```

VBD_NOT_REMOVABLE_MEDIA

Media could not be ejected because it is not removable

Signature:

```
VBD_NOT_REMOVABLE_MEDIA(vbd)
```

VBD_NOT_UNPLUGGABLE

Drive could not be hot-unplugged because it is not marked as unpluggable

Signature:

```
VBD_NOT_UNPLUGGABLE(vbd)
```

VBD_TRAY_LOCKED

This VM has locked the DVD drive tray, so the disk cannot be ejected

Signature:

```
VBD_TRAY_LOCKED(vbd)
```

VDI_CBT_ENABLED

The requested operation is not allowed for VDIs with CBT enabled or VMs having such VDIs, and CBT is enabled for the specified VDI.

Signature:

```
VDI_CBT_ENABLED(vdi)
```

VDI_CONTAINS_METADATA_OF_THIS_POOL

The VDI could not be opened for metadata recovery as it contains the current pool's metadata.

Signature:

```
VDI_CONTAINS_METADATA_OF_THIS_POOL(vdi, pool)
```

VDI_COPY_FAILED

The VDI copy action has failed

No parameters.

VDI_HAS_RRDS

The operation cannot be performed because this VDI has rrd stats

Signature:

```
VDI_HAS_RRDS(vdi)
```

VDI_INCOMPATIBLE_TYPE

This operation cannot be performed because the specified VDI is of an incompatible type (eg: an HA statefile cannot be attached to a guest)

Signature:

```
VDI_INCOMPATIBLE_TYPE(vdi, type)
```

VDI_IN_USE

This operation cannot be performed because this VDI is in use by some other operation

Signature:

```
VDI_IN_USE(vdi, operation)
```

VDI_IS_A_PHYSICAL_DEVICE

The operation cannot be performed on physical device

Signature:

```
VDI_IS_A_PHYSICAL_DEVICE(vdi)
```

VDI_IS_ENCRYPTED

The requested operation is not allowed because the specified VDI is encrypted.

Signature:

```
VDI_IS_ENCRYPTED(vdi)
```

VDI_IS_NOT_ISO

This operation can only be performed on CD VDIs (iso files or CDROM drives)

Signature:

```
VDI_IS_NOT_ISO(vdi, type)
```

VDI_LOCATION_MISSING

This operation cannot be performed because the specified VDI could not be found in the specified SR

Signature:

```
VDI_LOCATION_MISSING(sr, location)
```

VDI_MISSING

This operation cannot be performed because the specified VDI could not be found on the storage substrate

Signature:

```
VDI_MISSING(sr, vdi)
```

VDI_NEEDS_VM_FOR_MIGRATE

Cannot migrate a VDI which is not attached to a running VM.

Signature:

```
VDI_NEEDS_VM_FOR_MIGRATE(vdi)
```

VDI_NOT_AVAILABLE

This operation cannot be performed because this VDI could not be properly attached to the VM.

Signature:

```
VDI_NOT_AVAILABLE(vdi)
```

VDI_NOT_IN_MAP

This VDI was not mapped to a destination SR in VM.migrate_send operation

Signature:

```
VDI_NOT_IN_MAP(vdi)
```

VDI_NOT_MANAGED

This operation cannot be performed because the system does not manage this VDI

Signature:

```
VDI_NOT_MANAGED(vdi)
```

VDI_NOT_SPARSE

The VDI is not stored using a sparse format. It is not possible to query and manipulate only the changed blocks (or 'block differences' or 'disk deltas') between two VDIs. Please select a VDI which uses a sparse-aware technology such as VHD.

Signature:

```
VDI_NOT_SPARSE(vdi)
```

VDI_NO_CBT_METADATA

The requested operation is not allowed because the specified VDI does not have changed block tracking metadata.

Signature:

```
VDI_NO_CBT_METADATA(vdi)
```

VDI_ON_BOOT_MODE_INCOMPATIBLE_WITH_OPERATION

This operation is not permitted on VDIs in the 'on-boot=reset' mode, or on VMs having such VDIs.

No parameters.

VDI_READONLY

The operation required write access but this VDI is read-only

Signature:

```
VDI_READONLY(vdi)
```

VDI_TOO_LARGE

The VDI is too large.

Signature:

```
VDI_TOO_LARGE(vdi, maximum size)
```

VDI_TOO_SMALL

The VDI is too small. Please resize it to at least the minimum size.

Signature:

```
VDI_TOO_SMALL(vdi, minimum size)
```

VGPU_DESTINATION_INCOMPATIBLE

The VGPU is not compatible with any PGPU in the destination.

Signature:

```
VGPU_DESTINATION_INCOMPATIBLE(reason, vgpu, host)
```

VGPU_GUEST_DRIVER_LIMIT

The guest driver does not support VGPU migration.

Signature:

```
VGPU_GUEST_DRIVER_LIMIT(reason, vm, host)
```

VGPU_SUSPENSION_NOT_SUPPORTED

The VGPU configuration does not support suspension.

Signature:

```
VGPU_SUSPENSION_NOT_SUPPORTED(reason, vgpu, host)
```

VGPU_TYPE_NOT_COMPATIBLE

Cannot create a virtual GPU that is incompatible with the existing types on the VM.

Signature:

```
VGPU_TYPE_NOT_COMPATIBLE(type)
```

VGPU_TYPE_NOT_COMPATIBLE_WITH_RUNNING_TYPE

The VGPU type is incompatible with one or more of the VGPU types currently running on this PGPU

Signature:

```
VGPU_TYPE_NOT_COMPATIBLE_WITH_RUNNING_TYPE(pgpu, type, running_type)
```

VGPU_TYPE_NOT_ENABLED

VGPU type is not one of the PGPU's enabled types.

Signature:

```
VGPU_TYPE_NOT_ENABLED(type, enabled_types)
```

VGPU_TYPE_NOT_SUPPORTED

VGPU type is not one of the PGPU's supported types.

Signature:

```
VGPU_TYPE_NOT_SUPPORTED(type, supported_types)
```

VIF_IN_USE

Network has active VIFs

Signature:

```
VIF_IN_USE(network, VIF)
```

VIF_NOT_IN_MAP

This VIF was not mapped to a destination Network in VM.migrate_send operation

Signature:

```
VIF_NOT_IN_MAP(vif)
```

VLAN_IN_USE

Operation cannot be performed because this VLAN is already in use. Please check your network configuration.

Signature:

```
VLAN_IN_USE(device, vlan)
```

VLAN_TAG_INVALID

You tried to create a VLAN, but the tag you gave was invalid -- it must be between 0 and 4094. The parameter echoes the VLAN tag you gave.

Signature:

```
VLAN_TAG_INVALID(VLAN)
```

VMPP_ARCHIVE_MORE_FREQUENT_THAN_BACKUP

Archive more frequent than backup.

No parameters.

VMPP_HAS_VM

There is at least one VM assigned to this protection policy.

No parameters.

VMSS_HAS_VM

There is at least one VM assigned to snapshot schedule.

No parameters.

VMS_FAILED_TO_COOPERATE

The given VMs failed to release memory when instructed to do so

No parameters.

VM_ASSIGNED_TO_PROTECTION_POLICY

This VM is assigned to a protection policy.

Signature:

```
VM_ASSIGNED_TO_PROTECTION_POLICY(vm, vmpp)
```

VM_ASSIGNED_TO_SNAPSHOT_SCHEDULE

This VM is assigned to a snapshot schedule.

Signature:

```
VM_ASSIGNED_TO_SNAPSHOT_SCHEDULE(vm, vmss)
```

VM_ATTACHED_TO_MORE_THAN_ONE_VDI_WITH_TIMEOFFSET_MARKED_AS_RESET_ON_BOOT

You attempted to start a VM that's attached to more than one VDI with a timeoffset marked as reset-on-boot.

Signature:

```
VM_ATTACHED_TO_MORE_THAN_ONE_VDI_WITH_TIMEOFFSET_MARKED_AS_RESET_ON_BOOT(vm)
```

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_BIOS_STRINGS_ALREADY_SET

The BIOS strings for this VM have already been set and cannot be changed.

No parameters.

VM_CALL_PLUGIN_RATE_LIMIT

There is a minimal interval required between consecutive plug-in calls made on the same VM, please wait before retry.

Signature:

```
VM_CALL_PLUGIN_RATE_LIMIT(VM, interval, wait)
```

VM_CANNOT_DELETE_DEFAULT_TEMPLATE

You cannot delete the specified default template.

Signature:

```
VM_CANNOT_DELETE_DEFAULT_TEMPLATE(vm)
```

VM_CHECKPOINT_RESUME_FAILED

An error occured while restoring the memory image of the specified virtual machine

Signature:

```
VM_CHECKPOINT_RESUME_FAILED(vm)
```

VM_CHECKPOINT_SUSPEND_FAILED

An error occured while saving the memory image of the specified virtual machine

Signature:

```
VM_CHECKPOINT_SUSPEND_FAILED(vm)
```

VM_CRASHED

The VM crashed

Signature:

```
VM_CRASHED(vm)
```

VM_DUPLICATE_VBD_DEVICE

The specified VM has a duplicate VBD device and cannot be started.

Signature:

```
VM_DUPLICATE_VBD_DEVICE(vm, vbd, device)
```

VM_FAILED_SHUTDOWN_ACKNOWLEDGMENT

VM didn't acknowledge the need to shutdown.

Signature:

```
VM_FAILED_SHUTDOWN_ACKNOWLEDGMENT(vm)
```

VM_FAILED_SUSPEND_ACKNOWLEDGMENT

VM didn't acknowledge the need to suspend.

Signature:

```
VM_FAILED_SUSPEND_ACKNOWLEDGMENT(vm)
```

VM_HALTED

The VM unexpectedly halted

Signature:

```
VM_HALTED(vm)
```

VM_HAS_CHECKPOINT

Cannot migrate a VM which has a checkpoint.

Signature:

```
VM_HAS_CHECKPOINT(vm)
```

VM_HAS_NO_SUSPEND_VDI

VM cannot be resumed because it has no suspend VDI

Signature:

```
VM_HAS_NO_SUSPEND_VDI(vm)
```

VM_HAS_PCI_ATTACHED

This operation could not be performed, because the VM has one or more PCI devices passed through.

Signature:

```
VM_HAS_PCI_ATTACHED(vm)
```

VM_HAS_SRIOV_VIF

This operation could not be performed, because the VM has one or more SR-IOV VIFs.

Signature:

```
VM_HAS_SRIOV_VIF(vm)
```

VM_HAS_TOO_MANY_SNAPSHOTS

Cannot migrate a VM with more than one snapshot.

Signature:

```
VM_HAS_TOO_MANY_SNAPSHOTS(vm)
```

VM_HAS_VGPU

This operation could not be performed, because the VM has one or more virtual GPUs.

Signature:

```
VM_HAS_VGPU(vm)
```

VM_HAS_VUSBS

The operation is not allowed when the VM has VUSBs.

Signature:

```
VM_HAS_VUSBS(VM)
```

VM_HOST_INCOMPATIBLE_VERSION

This VM operation cannot be performed on an older-versioned host during an upgrade.

Signature:

```
VM_HOST_INCOMPATIBLE_VERSION(host, vm)
```

VM_HOST_INCOMPATIBLE_VERSION_MIGRATE

Cannot migrate a VM to a destination host which is older than the source host.

Signature:

```
VM_HOST_INCOMPATIBLE_VERSION_MIGRATE(host, vm)
```

VM_HOST_INCOMPATIBLE_VIRTUAL_HARDWARE_PLATFORM_VERSION

You attempted to run a VM on a host that cannot provide the VM's required Virtual Hardware Platform version.

Signature:

```
VM_HOST_INCOMPATIBLE_VIRTUAL_HARDWARE_PLATFORM_VERSION(host, host_versions, vm,
vm_version)
```

VM_HVM_REQUIRED

HVM is required for this operation

Signature:

```
VM_HVM_REQUIRED(vm)
```

VM_INCOMPATIBLE_WITH_THIS_HOST

The VM is incompatible with the CPU features of this host.

Signature:

```
VM_INCOMPATIBLE_WITH_THIS_HOST(vm, host, reason)
```

VM_IS_IMMOBILE

The VM is configured in a way that prevents it from being mobile.

Signature:

```
VM_IS_IMMOBILE(VM)
```

VM_IS_PART_OF_AN_APPLIANCE

This operation is not allowed as the VM is part of an appliance.

Signature:

```
VM_IS_PART_OF_AN_APPLIANCE(vm, appliance)
```

VM_IS_PROTECTED

This operation cannot be performed because the specified VM is protected by HA

Signature:

```
VM_IS_PROTECTED(vm)
```

VM_IS_TEMPLATE

The operation attempted is not valid for a template VM

Signature:

```
VM_IS_TEMPLATE(vm)
```

VM_IS_USING_NESTED_VIRT

This operation is illegal because the VM is using nested virtualization.

Signature:

```
VM_IS_USING_NESTED_VIRT(VM)
```

VM_LACKS_FEATURE

You attempted an operation on a VM which lacks the feature.

Signature:

```
VM_LACKS_FEATURE(vm)
```

VM_LACKS_FEATURE_SHUTDOWN

You attempted an operation which needs the cooperative shutdown feature on a VM which lacks it.

Signature:

```
VM_LACKS_FEATURE_SHUTDOWN(vm)
```

VM_LACKS_FEATURE_STATIC_IP_SETTING

You attempted an operation which needs the VM static-ip-setting feature on a VM which lacks it.

Signature:

```
VM_LACKS_FEATURE_STATIC_IP_SETTING(vm)
```

VM_LACKS_FEATURE_SUSPEND

You attempted an operation which needs the VM cooperative suspend feature on a VM which lacks it.

Signature:

```
VM_LACKS_FEATURE_SUSPEND(vm)
```

VM_LACKS_FEATURE_VCPU_HOTPLUG

You attempted an operation which needs the VM hotplug-vcpu feature on a VM which lacks it.

Signature:

```
VM_LACKS_FEATURE_VCPU_HOTPLUG(vm)
```

VM_MEMORY_SIZE_TOO_LOW

The specified VM has too little memory to be started.

Signature:

```
VM_MEMORY_SIZE_TOO_LOW(vm)
```

VM_MIGRATE_CONTACT_REMOTE_SERVICE_FAILED

Failed to contact service on the destination host.

No parameters.

VM_MIGRATE_FAILED

An error occurred during the migration process.

Signature:

```
VM_MIGRATE_FAILED(vm, source, destination, msg)
```

VM_MISSING_PV_DRIVERS

You attempted an operation on a VM which requires PV drivers to be installed but the drivers were not detected.

Signature:

```
VM_MISSING_PV_DRIVERS(vm)
```

VM_NOT_RESIDENT_HERE

The specified VM is not currently resident on the specified server.

Signature:

```
VM_NOT_RESIDENT_HERE(vm, host)
```

VM_NO_CRASHDUMP_SR

This VM does not have a crash dump SR specified.

Signature:

```
VM_NO_CRASHDUMP_SR(vm)
```

VM_NO_EMPTY_CD_VBD

The VM has no empty CD drive (VBD).

Signature:

```
VM_NO_EMPTY_CD_VBD(vm)
```

VM_NO_SUSPEND_SR

This VM does not have a suspend SR specified.

Signature:

```
VM_NO_SUSPEND_SR(vm)
```

VM_NO_VCPUS

You need at least 1 VCPU to start a VM

Signature:

```
VM_NO_VCPUS(vm)
```

VM_OLD_PV_DRIVERS

You attempted an operation on a VM which requires a more recent version of the PV drivers. Please upgrade your PV drivers.

Signature:

```
VM_OLD_PV_DRIVERS(vm, major, minor)
```

VM_PCI_BUS_FULL

The VM does not have any free PCI slots

Signature:

```
VM_PCI_BUS_FULL(VM)
```

VM_PV_DRIVERS_IN_USE

VM PV drivers still in use

Signature:

```
VM_PV_DRIVERS_IN_USE(vm)
```

VM_REBOOTED

The VM unexpectedly rebooted

Signature:

```
VM_REBOOTED(vm)
```

VM_REQUIRES_GPU

You attempted to run a VM on a host which doesn't have a pGPU available in the GPU group needed by the VM. The VM has a vGPU attached to this GPU group.

Signature:

```
VM_REQUIRES_GPU(vm, GPU_group)
```

VM_REQUIRES_IOMMU

You attempted to run a VM on a host which doesn't have I/O virtualization (IOMMU/VT-d) enabled, which is needed by the VM.

Signature:

```
VM_REQUIRES_IOMMU(host)
```

VM_REQUIRES_NETWORK

You attempted to run a VM on a host which doesn't have a PIF on a Network needed by the VM. The VM has at least one VIF attached to the Network.

Signature:

```
VM_REQUIRES_NETWORK(vm, network)
```

VM_REQUIRES_SR

You attempted to run a VM on a host which doesn't have access to an SR needed by the VM. The VM has at least one VBD attached to a VDI in the SR.

Signature:

```
VM_REQUIRES_SR(vm, sr)
```

VM_REQUIRES_VDI

VM cannot be started because it requires a VDI which cannot be attached

Signature:

```
VM_REQUIRES_VDI(vm, vdi)
```

VM_REQUIRES_VGPU

You attempted to run a VM on a host on which the vGPU required by the VM cannot be allocated on any pGPUs in the GPU_group needed by the VM.

Signature:

```
VM_REQUIRES_VGPU(vm, GPU_group, vGPU_type)
```

VM_REQUIRES_VUSB

You attempted to run a VM on a host on which the VUSB required by the VM cannot be allocated on any PUSBs in the USB_group needed by the VM.

Signature:

```
VM_REQUIRES_VUSB(vm, USB_group)
```

VM_REVERT_FAILED

An error occured while reverting the specified virtual machine to the specified snapshot

Signature:

```
VM_REVERT_FAILED(vm, snapshot)
```

VM_SHUTDOWN_TIMEOUT

VM failed to shutdown before the timeout expired

Signature:

```
VM_SHUTDOWN_TIMEOUT(vm, timeout)
```

VM_SNAPSHOT_WITH_QUIESCE_FAILED

The quiesced-snapshot operation failed for an unexpected reason

Signature:

```
VM_SNAPSHOT_WITH_QUIESCE_FAILED(vm)
```

VM_SNAPSHOT_WITH_QUIESCE_NOT_SUPPORTED

The VSS plug-in is not installed on this virtual machine

Signature:

```
VM_SNAPSHOT_WITH_QUIESCE_NOT_SUPPORTED(vm, error)
```

VM_SNAPSHOT_WITH_QUIESCE_PLUGIN_DEOS_NOT_RESPOND

The VSS plug-in cannot be contacted

Signature:

```
VM_SNAPSHOT_WITH_QUIESCE_PLUGIN_DEOS_NOT_RESPOND(vm)
```

VM_SNAPSHOT_WITH_QUIESCE_TIMEOUT

The VSS plug-in has timed out

Signature:

```
VM_SNAPSHOT_WITH_QUIESCE_TIMEOUT(vm)
```

VM_SUSPEND_TIMEOUT

VM failed to suspend before the timeout expired

Signature:

```
VM_SUSPEND_TIMEOUT(vm, timeout)
```

VM_TOO_MANY_VCPUS

Too many VCPUs to start this VM

Signature:

```
VM_TOO_MANY_VCPUS(vm)
```

VM_TO_IMPORT_IS_NOT_NEWER_VERSION

The VM cannot be imported unforced because it is either the same version or an older version of an existing VM.

Signature:

```
VM_TO_IMPORT_IS_NOT_NEWER_VERSION(vm, existing_version, version_to_import)
```

VM_UNSAFE_BOOT

You attempted an operation on a VM that was judged to be unsafe by the server. This can happen if the VM would run on a CPU that has a potentially incompatible set of feature flags to those the VM requires. If you want to override this warning then use the 'force' option.

Signature:

```
VM_UNSAFE_BOOT(vm)
```

WLB_AUTHENTICATION_FAILED

WLB rejected our configured authentication details.

No parameters.

WLB_CONNECTION_REFUSED

WLB refused a connection to the server.

No parameters.

WLB_CONNECTION_RESET

The connection to the WLB server was reset.

No parameters.

WLB_DISABLED

This pool has wlb-enabled set to false.

No parameters.

WLB_INTERNAL_ERROR

WLB reported an internal error.

No parameters.

WLB_MALFORMED_REQUEST

WLB rejected the server's request as malformed.

No parameters.

WLB_MALFORMED_RESPONSE

WLB said something that the server wasn't expecting or didn't understand. The method called on WLB, a diagnostic reason, and the response from WLB are returned.

Signature:

```
WLB_MALFORMED_RESPONSE(method, reason, response)
```

WLB_NOT_INITIALIZED

No WLB connection is configured.

No parameters.

WLB_TIMEOUT

The communication with the WLB server timed out.

Signature:

```
WLB_TIMEOUT(configured_timeout)
```

WLB_UNKNOWN_HOST

The configured WLB server name failed to resolve in DNS.

No parameters.

WLB_URL_INVALID

The WLB URL is invalid. Ensure it is in the format: <ipaddress>:<port>. The configured/given URL is returned.

Signature:

```
WLB_URL_INVALID(url)
```

WLB_XENSERVER_AUTHENTICATION_FAILED

WLB reported that the server rejected its configured authentication details.

No parameters.

WLB_XENSERVER_CONNECTION_REFUSED

WLB reported that the server refused to let it connect (even though we're connecting perfectly fine in the other direction).

No parameters.

WLB_XENSERVER_MALFORMED_RESPONSE

WLB reported that the server said something to it that WLB wasn't expecting or didn't understand.

No parameters.

WLB_XENSERVER_TIMEOUT

WLB reported that communication with the server timed out.

No parameters.

WLB_XENSERVER_UNKNOWN_HOST

WLB reported that its configured server name for this server instance failed to resolve in DNS.

No parameters.

XAPI_HOOK_FAILED

3rd party xapi hook failed

Signature:

```
XAPI_HOOK_FAILED(hook_name, reason, stdout, exit_code)
```

XENAPI_MISSING_PLUGIN

The requested plug-in could not be found.

Signature:

```
XENAPI_MISSING_PLUGIN(name)
```

XENAPI_PLUGIN_FAILURE

There was a failure communicating with the plug-in.

Signature:

```
XENAPI_PLUGIN_FAILURE(status, stdout, stderr)
```

XEN_INCOMPATIBLE

The current version of Xen or its control libraries is incompatible with the Toolstack.

No parameters.

XEN_VSS_REQ_ERROR_ADDING_VOLUME_TO_SNAPSET_FAILED

Some volumes to be snapshot could not be added to the VSS snapshot set

Signature:

```
XEN_VSS_REQ_ERROR_ADDING_VOLUME_TO_SNAPSET_FAILED(vm, error_code)
```

XEN_VSS_REQ_ERROR_CREATING_SNAPSHOT

An attempt to create the snapshots failed

Signature:

```

```

```
XEN_VSS_REQ_ERROR_CREATING_SNAPSHOT(vm, error_code)
```

XEN_VSS_REQ_ERROR_CREATING_SNAPSHOT_XML_STRING

Could not create the XML string generated by the transportable snapshot

Signature:

```
XEN_VSS_REQ_ERROR_CREATING_SNAPSHOT_XML_STRING(vm, error_code)
```

XEN_VSS_REQ_ERROR_INIT_FAILED

Initialization of the VSS requester failed

Signature:

```
XEN_VSS_REQ_ERROR_INIT_FAILED(vm, error_code)
```

XEN_VSS_REQ_ERROR_NO_VOLUMES_SUPPORTED

Could not find any volumes supported by the VSS Provider

Signature:

```
XEN_VSS_REQ_ERROR_NO_VOLUMES_SUPPORTED(vm, error_code)
```

XEN_VSS_REQ_ERROR_PREPARING_WRITERS

An attempt to prepare VSS writers for the snapshot failed

Signature:

```
XEN_VSS_REQ_ERROR_PREPARING_WRITERS(vm, error_code)
```

XEN_VSS_REQ_ERROR_PROV_NOT_LOADED

The VSS Provider is not loaded

Signature:

```
XEN_VSS_REQ_ERROR_PROV_NOT_LOADED(vm, error_code)
```

XEN_VSS_REQ_ERROR_START_SNAPSHOT_SET_FAILED

An attempt to start a new VSS snapshot failed

Signature:

```
XEN_VSS_REQ_ERROR_START_SNAPSHOT_SET_FAILED(vm, error_code)
```

XMLRPC_UNMARSHAL_FAILURE

The server failed to unmarshal the XMLRPC message; it was expecting one element and received something else.

Signature:

```
XMLRPC_UNMARSHAL_FAILURE(expected, received)
```

Locations

Corporate Headquarters | 851 Cypress Creek Road Fort Lauderdale, FL 33309, United States

Silicon Valley | 4988 Great America Parkway Santa Clara, CA 95054, United States

Citrix Product Documentation | http://docs.citrix.com

December 3, 2021