Python Ironic Inspector Client Documentation

Release 4.4.1.dev2

OpenStack Foundation

Mar 18, 2024

CONTENTS

CHAPTER

ONE

IRONIC INSPECTOR CLIENT

This is a client library and tool for Ironic Inspector.

- Free software: Apache license
- Source: https://opendev.org/openstack/python-ironic-inspector-client/
- Documentation: https://docs.openstack.org/python-ironic-inspector-client/latest/
- Bugs: https://storyboard.openstack.org/#!/project/958
- Downloads: https://pypi.org/project/python-ironic-inspector-client
- Release Notes: https://docs.openstack.org/releasenotes/python-ironic-inspector-client/

Please follow usual OpenStack Gerrit Workflow to submit a patch, see Inspector contributing guide for more detail.

Refer to the HTTP API reference for information on the **Ironic Inspector** HTTP API.

CHAPTER

TWO

CONTENTS

2.1 Command Line Reference

Integration for two command lines tools are provided: ironicclient and OpenStackClient.

2.1.1 Integration with `baremetal` CLI

The `baremetal` command is provided by ironicclient, so it has to be installed, e.g.:

```
pip install 'python-ironicclient>=4.1.0'
```

All commands are prefixed with `baremetal introspection`. See standalone ironic CLI documentation for details on how to use it.

2.1.2 Integration with `openstack baremetal` CLI

The `openstack` command is provided by ironicclient, so it has to be installed, e.g.:

```
pip install python-openstackclient.
```

All commands are prefixed with `baremetal introspection`.

Common arguments

All commands accept the following arguments:

- `--inspector-url` the **Ironic Inspector** API endpoint. If missing, the endpoint will be fetched from the service catalog.
- `--inspector-api-version` requested API version, see *API Versioning* for details.

2.1.3 Start introspection on a node

```
$ openstack baremetal introspection start [--wait] [--check-errors] NODE_
↪ID [NODE_ID ...]
```

- NODE_ID - Ironic node UUID or name;

Note that the CLI call accepts several UUIDs and will stop on the first error.

Note: This CLI call doesnt rely on Ironic, and the introspected node will be left in MANAGEABLE state. This means that the Ironic node is not protected from other operations being performed by Ironic, which could cause inconsistency in the nodes state, and lead to operational errors.

With --wait flag it waits until introspection ends for all given nodes, then displays the results as a table.

The --check-errors flag verifies if any error occurred during the introspection of the selected nodes while waiting for the results. If any error has occurred in the introspection result of selected nodes no output is displayed, otherwise it shows the result as a table.

Note: --check-errors can only be used with --wait

2.1.4 Query introspection status

```
$ openstack baremetal introspection status NODE_ID
```

- NODE_ID - Ironic node UUID or name.

Returns following information about a node introspection status:

- error: an error string or None
- finished: True/False
- finished_at: an ISO8601 timestamp or None if not finished
- started_at: an ISO8601 timestamp
- uuid: node UUID

2.1.5 List introspection statuses

This command supports pagination.

```
$ openstack baremetal introspection list [--marker] [--limit]
```

- --marker the last item on the previous page, a UUID
- --limit the amount of items to list, an integer, 50 by default

Shows a table with the following columns:

- Error: an error string or None

- `Finished at`: an ISO8601 timestamp or `None` if not finished
- `Started at`: and ISO8601 timestamp
- `UUID`: node UUID

Note: The server orders the introspection status items according to the `Started at` column, newer items first.

2.1.6 Retrieving introspection data

```
$ openstack baremetal introspection data save [--file file_name] [--
↪unprocessed] NODE_ID
```

- `NODE_ID` - Ironic node UUID or name;
- `file_name` - file name to save data to. If file name is not provided, the data is dumped to stdout.
- `--unprocessed` - if set, retrieves the unprocessed data received from the ramdisk.

Note: This feature requires Swift support to be enabled in **Ironic Inspector** by setting `[processing]store_data` configuration option to `swift`.

2.1.7 Aborting introspection

```
$ openstack baremetal introspection abort NODE_ID
```

- `NODE_ID` - Ironic node UUID or name.

2.1.8 Reprocess stored introspection data

```
$ openstack baremetal introspection reprocess NODE_ID
```

- `NODE_ID` - Ironic node UUID or name.

Note: This feature requires Swift store to be enabled for **Ironic Inspector** by setting `[processing]store_data` configuration option to `swift`.

2.1.9 Introspection Rules API

Creating a rule

```
$ openstack baremetal introspection rule import <JSON FILE>
```

- `rule_json` dictionary with a rule representation, see `ironic_inspector_client.RulesAPI.from_json()` for details.

Listing all rules

```
$ openstack baremetal introspection rule list
```

Returns list of short rule representations, containing only description, UUID and links.

Deleting all rules

```
$ openstack baremetal introspection rule purge
```

Deleting a rule

```
$ openstack baremetal introspection rule delete <UUID>
```

- `UUID` rule UUID.

2.1.10 Using names instead of UUID

Starting with baremetal introspection API 1.5 (provided by **Ironic Inspector** 3.3.0) its possible to use node names instead of UUIDs in all Python and CLI calls.

2.1.11 List interface data

```
$ openstack baremetal introspection interface list NODE_IDENT
[--fields=<field>] [--vlan=<vlan>]
```

- `NODE_IDENT` - Ironic node UUID or name
- `fields` - name of one or more interface columns to display.
- `vlan` - list only interfaces configured for this vlan id

Returns a list of interface data, including attached switch information, for each interface on the node.

2.1.12 Show interface data

```
$ openstack baremetal introspection interface show NODE_IDENT INTERFACE
[--fields=<field>]
```

- `NODE_IDENT` - Ironic node UUID or name
- `INTERFACE` - interface name on this node
- `fields` - name of one or more interface rows to display.

Show interface data, including attached switch information, for a particular node and interface.

2.2 Library User Reference

To use Python API first create a `ClientV1` object:

```
import ironic_inspector_client
client = ironic_inspector_client.ClientV1(session=keystone_session)
```

This code creates a client with API version *1.0* and a given Keystone session. The service URL is fetched from the service catalog in this case. See *ironic_inspector_client.ClientV1* documentation for details.

2.2.1 API Versioning

Starting with version 2.1.0 **Ironic Inspector** supports optional API versioning. Version is a tuple (X, Y), where X is always 1 for now.

The server has maximum and minimum supported versions. If no version is requested, the server assumes the maximum its supported.

Two constants are exposed for convenience:

- `ironic_inspector_client.DEFAULT_API_VERSION`
- `ironic_inspector_client.MAX_API_VERSION`

2.2.2 API Reference

ironic_inspector_client package

Subpackages

ironic_inspector_client.common package

Submodules

ironic_inspector_client.common.http module

Generic code for inspector client.

class ironic_inspector_client.common.http.**BaseClient**(*api_version*, *inspector_url=None*, *session=None*, *service_type='baremetal-introspection'*, *interface=None*, *region_name=None*)

Bases: object

Base class for clients, provides common HTTP code.

request(*method*, *url*, ***kwargs*)

Make an HTTP request.

Parameters

- **method** HTTP method
- **endpoint** relative endpoint
- **kwargs** arguments to pass to requests library

server_api_versions()

Get minimum and maximum supported API versions from a server.

Returns tuple (minimum version, maximum version) each version is returned as a tuple (X, Y)

Raises *requests* library exception on connection problems.

Raises ValueError if returned version cannot be parsed

exception ironic_inspector_client.common.http.**ClientError**(*response*)

Bases: requests.exceptions.HTTPError

Error returned from a server.

classmethod raise_if_needed(*response*)

Raise exception if response contains error.

exception ironic_inspector_client.common.http.**EndpointNotFound**(*service_type*)

Bases: Exception

Denotes that endpoint for the introspection service was not found.

Variables **service_type** requested service type

exception ironic_inspector_client.common.http.**VersionNotSupported**(*expected*, *supported*)

Bases: Exception

Denotes that requested API versions is not supported by the server.

Variables

- **expected** requested version.
- **supported** sequence with two items: minimum and maximum actually supported versions.

Module contents

Submodules

ironic_inspector_client.resource module

class ironic_inspector_client.resource.**InterfaceResource**(*field_ids=None*, *detailed=False*)

Bases: object

InterfaceResource class

This class is used to manage the fields including Link Layer Discovery Protocols (LLDP) fields, that an interface contains. An individual field consists of a field_id (key) and a label (value).

DEFAULT_FIELD_IDS = ['interface', 'mac', 'switch_port_vlan_ids', 'switch_cha
Interface fields displayed by default.

FIELDS = {'interface': 'Interface', 'mac': 'MAC Address', 'node_ident': '
A mapping of all known interface fields to their descriptions.

property fields
List of fields displayed for this resource.

property labels
List of labels for fields displayed for this resource.

ironic_inspector_client.v1 module

Client for V1 API.

class ironic_inspector_client.v1.**ClientV1**(***kwargs*)

Bases: *ironic_inspector_client.common.http.BaseClient*

Client for API v1.

Create this object to use Python API, for example:

```
import ironic_inspector_client
client = ironic_inspector_client.ClientV1(session=keystone_session)
```

This code creates a client with API version *1.0* and a given Keystone session. The service URL is fetched from the service catalog in this case. Optional arguments `service_type`, `interface` and `region_name` can be provided to modify how the URL is looked up.

If the catalog lookup fails, the local host with port 5050 is tried. However, this behaviour is deprecated and should not be relied on. Also an explicit `inspector_url` can be passed to bypass service catalog.

Optional `api_version` argument is a minimum API version that a server must support. It can be a tuple (MAJ, MIN), string MAJ.MIN or integer (only major, minimum supported minor version is assumed).

Variables rules Reference to the introspection rules API. Instance of *ironic_inspector_client.v1.RulesAPI*.

abort (*node_id=None*, *uuid=None*)
Abort running introspection for a node.

Parameters

- **uuid** node UUID or name, deprecated
- **node_id** node node_id or name

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

Raises TypeError if uuid is not a string.

get_all_interface_data (*node_ident*, *field_sel*, *vlan=None*)
Get interface data for all of the interfaces on this node

Parameters

- **node_ident** node UUID or name
- **field_sel** list of all fields for which to get data
- **vlan** list of vlans used to filter the lists returned

Returns list of interface data, each interface in a list

get_data (*node_id=None*, *raw=False*, *uuid=None*, *processed=True*)
Get introspection data from the last introspection of a node.

If swift support is disabled, introspection data wont be stored, this request will return error response with 404 code.

Parameters

- **uuid** node UUID or name, deprecated
- **node_id** node node_id or name
- **raw** whether to return raw binary data or parsed JSON data
- **processed** whether to return the final processed data or the raw unprocessed data received from the ramdisk.

Returns bytes or a dict depending on the raw argument

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

Raises TypeError if uuid is not a string

get_interface_data (*node_ident*, *interface*, *field_sel*)
Get interface data for the input node and interface

To get LLDP data, collection must be enabled by the kernel parameter `ipa-collect-lldp=1`, and the inspector plugin `basic_lldp` must be enabled.

Parameters

- **node_ident** node UUID or name
- **interface** interface name
- **field_sel** list of all fields for which to get data

Returns interface data in OrderedDict

Raises ValueError if interface is not found.

get_status(*node_id=None*, *uuid=None*)
Get introspection status for a node.

Parameters

- **uuid** node UUID or name, deprecated
- **node_id** node node_id or name

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

Returns

dictionary with the keys:

- *error* an error string or None,
- *finished* whether introspection was finished,
- *finished_at* an ISO8601 timestamp or None,
- *links* with a self-link URL,
- *started_at* an ISO8601 timestamp,
- *uuid* the node UUID

introspect(*node_id=None*, *manage_boot=None*, *uuid=None*)
Start introspection for a node.

Parameters

- **uuid** node UUID or name, deprecated
- **node_id** node node_id or name
- **manage_boot** whether to manage boot during introspection of this node. If it is None (the default), then this argument is not passed to API and the server default is used instead.

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

list_statuses(*marker=None*, *limit=None*)

List introspection statuses.

Supports pagination via the marker and limit params. The items are sorted by the server according to the *started_at* attribute, newer items first.

Parameters

- **marker** pagination maker, UUID or None
- **limit** pagination limit, int or None

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

Returns

a list of status dictionaries with the keys:

- *error* an error string or None,
- *finished* whether introspection was finished,
- *finished_at* an ISO8601 timestamp or None,
- *links* with a self-link URL,
- *started_at* an ISO8601 timestamp,
- *uuid* the node UUID

reprocess(*node_id=None*, *uuid=None*)

Reprocess stored introspection data.

If swift support is disabled, introspection data wont be stored, this request will return error response with 404 code.

Parameters

- **uuid** node UUID or name, deprecated
- **node_id** node node_id or name

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

Raises TypeError if uuid is not a string.

wait_for_finish(*node_ids=None*, *retry_interval=10*, *max_retries=3600*, *sleep_function=<built-in function sleep>*, *uuids=None*)

Wait for introspection finishing for given nodes.

Parameters

- **uuids** collection of node UUIDs or names, deprecated
- **node_ids** collection of node node_ids or names
- **retry_interval** sleep interval between retries.
- **max_retries** maximum number of retries.
- **sleep_function** function used for sleeping between retries.

Raises `ironic_inspector_client.WaitTimeoutError` on timeout

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

Returns dictionary UUID -> status (the same as in get_status).

`ironic_inspector_client.v1.`**DEFAULT_API_VERSION = (1, 0)**

Server API version used by default.

`ironic_inspector_client.v1.`**DEFAULT_MAX_RETRIES = 3600**

Default number of retries when waiting for introspection to finish.

`ironic_inspector_client.v1.`**DEFAULT_RETRY_INTERVAL = 10**

Default interval (in seconds) between retries when waiting for introspection to finish.

`ironic_inspector_client.v1.`**MAX_API_VERSION = (1, 13)**

Maximum API version this client was designed to work with.

This does not mean that other versions wont work at all - the server might still support them.

class `ironic_inspector_client.v1.`**RulesAPI**(*requester*)

Bases: `object`

Introspection rules API.

Do not create instances of this class directly, use `ironic_inspector_client.v1.ClientV1.rules` instead.

create(*conditions*, *actions*, *uuid=None*, *description=None*)

Create a new introspection rule.

Parameters

- **conditions** list of rule conditions
- **actions** list of rule actions
- **uuid** rule UUID, will be generated if not specified
- **description** optional rule description

Returns rule representation

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

delete(*uuid*)
Delete an introspection rule.

Parameters **uuid** rule UUID

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

delete_all()
Delete all introspection rules.

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

from_json(*json_rule*)
Import an introspection rule from JSON data.

Parameters **json_rule** rule information as a dict with keys matching arguments of *RulesAPI.create()*.

Returns rule representation

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

get(*uuid*)
Get detailed information about an introspection rule.

Parameters **uuid** rule UUID

Returns rule representation

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

get_all()
List all introspection rules.

Returns list of short rule representations (uuid, description and links)

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

exception ironic_inspector_client.v1.**WaitTimeoutError**
Bases: Exception

Timeout while waiting for nodes to finish introspection.

ironic_inspector_client.version module

ironic_inspector_client.version.**version_info = pbr.version.VersionInfo(python-ir**
Installed package version.

Module contents

exception ironic_inspector_client.**ClientError**(*response*)
Bases: requests.exceptions.HTTPError

Error returned from a server.

classmethod raise_if_needed(*response*)
Raise exception if response contains error.

class ironic_inspector_client.**ClientV1**(***kwargs*)
Bases: *ironic_inspector_client.common.http.BaseClient*

Client for API v1.

Create this object to use Python API, for example:

```
import ironic_inspector_client
client = ironic_inspector_client.ClientV1(session=keystone_session)
```

This code creates a client with API version *1.0* and a given Keystone session. The service URL is fetched from the service catalog in this case. Optional arguments `service_type`, `interface` and `region_name` can be provided to modify how the URL is looked up.

If the catalog lookup fails, the local host with port 5050 is tried. However, this behaviour is deprecated and should not be relied on. Also an explicit `inspector_url` can be passed to bypass service catalog.

Optional `api_version` argument is a minimum API version that a server must support. It can be a tuple (MAJ, MIN), string MAJ.MIN or integer (only major, minimum supported minor version is assumed).

Variables **rules** Reference to the introspection rules API. Instance of *ironic_inspector_client.v1.RulesAPI*.

abort(*node_id=None*, *uuid=None*)
Abort running introspection for a node.

Parameters

- **uuid** node UUID or name, deprecated
- **node_id** node node_id or name

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

Raises TypeError if uuid is not a string.

get_all_interface_data(*node_ident*, *field_sel*, *vlan=None*)
Get interface data for all of the interfaces on this node

Parameters

- **node_ident** node UUID or name
- **field_sel** list of all fields for which to get data
- **vlan** list of vlans used to filter the lists returned

Returns list of interface data, each interface in a list

get_data(*node_id=None*, *raw=False*, *uuid=None*, *processed=True*)
Get introspection data from the last introspection of a node.

If swift support is disabled, introspection data wont be stored, this request will return error response with 404 code.

Parameters

- **uuid** node UUID or name, deprecated
- **node_id** node node_id or name
- **raw** whether to return raw binary data or parsed JSON data
- **processed** whether to return the final processed data or the raw unprocessed data received from the ramdisk.

Returns bytes or a dict depending on the raw argument

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

Raises TypeError if uuid is not a string

get_interface_data(*node_ident*, *interface*, *field_sel*)
Get interface data for the input node and interface

To get LLDP data, collection must be enabled by the kernel parameter `ipa-collect-lldp=1`, and the inspector plugin `basic_lldp` must be enabled.

Parameters

- **node_ident** node UUID or name
- **interface** interface name
- **field_sel** list of all fields for which to get data

Returns interface data in OrderedDict

Raises ValueError if interface is not found.

get_status(*node_id=None*, *uuid=None*)
Get introspection status for a node.

Parameters

- **uuid** node UUID or name, deprecated
- **node_id** node node_id or name

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

Returns

dictionary with the keys:

- *error* an error string or None,
- *finished* whether introspection was finished,
- *finished_at* an ISO8601 timestamp or None,
- *links* with a self-link URL,
- *started_at* an ISO8601 timestamp,
- *uuid* the node UUID

introspect(*node_id=None*, *manage_boot=None*, *uuid=None*)
Start introspection for a node.

Parameters

- **uuid** node UUID or name, deprecated
- **node_id** node node_id or name
- **manage_boot** whether to manage boot during introspection of this node. If it is None (the default), then this argument is not passed to API and the server default is used instead.

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

list_statuses(*marker=None*, *limit=None*)
List introspection statuses.

Supports pagination via the marker and limit params. The items are sorted by the server according to the *started_at* attribute, newer items first.

Parameters

- **marker** pagination maker, UUID or None
- **limit** pagination limit, int or None

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

Returns

a list of status dictionaries with the keys:

- *error* an error string or None,
- *finished* whether introspection was finished,
- *finished_at* an ISO8601 timestamp or None,
- *links* with a self-link URL,
- *started_at* an ISO8601 timestamp,
- *uuid* the node UUID

reprocess(*node_id=None*, *uuid=None*)

Reprocess stored introspection data.

If swift support is disabled, introspection data wont be stored, this request will return error response with 404 code.

Parameters

- **uuid** node UUID or name, deprecated
- **node_id** node node_id or name

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

Raises TypeError if uuid is not a string.

wait_for_finish(*node_ids=None*, *retry_interval=10*, *max_retries=3600*, *sleep_function=<built-in function sleep>*, *uuids=None*)

Wait for introspection finishing for given nodes.

Parameters

- **uuids** collection of node UUIDs or names, deprecated
- **node_ids** collection of node node_ids or names
- **retry_interval** sleep interval between retries.
- **max_retries** maximum number of retries.
- **sleep_function** function used for sleeping between retries.

Raises ironic_inspector_client.WaitTimeoutError on timeout

Raises *ironic_inspector_client.ClientError* on error reported from a server

Raises *ironic_inspector_client.VersionNotSupported* if requested api_version is not supported

Raises *requests* library exception on connection problems.

Returns dictionary UUID -> status (the same as in get_status).

exception ironic_inspector_client.**EndpointNotFound**(*service_type*)

Bases: Exception

Denotes that endpoint for the introspection service was not found.

Variables **service_type** requested service type

exception ironic_inspector_client.**VersionNotSupported**(*expected*, *supported*)

Bases: Exception

Denotes that requested API versions is not supported by the server.

Variables

- **expected** requested version.
- **supported** sequence with two items: minimum and maximum actually supported versions.

ironic_inspector_client

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX

L

M

R

S

V

W