pypuppetdb Documentation

Release 0.1.0

Daniele Sluijters

January 14, 2014

Contents

Note: This is a very new project and still changing at a rapid pace. As such the only documentation currently available is the API documentation and a brief Getting Started guide. Once this settles down tutorials and other documentation will be added over time.

Getting started

The quickstart should get you up and running with pypuppetdb and familiarise you with how this library works.

1.1 Quickstart

Once you have pypuppetdb installed you can configure it to connect to PuppetDB and take it from there.

1.1.1 Connecting

The first thing you need to do is to connect with PuppetDB:

```
>>> from pypuppetdb import connect
>>> db = connect()
```

1.1.2 Nodes

The following will return a generator object yielding Node objects for every returned node from PuppetDB.

```
>>> nodes = db.nodes()
>>> for node in nodes:
>>>   print(node)
host1
host2
...
```

To query a single node the singular *node()* can be used:

```
>>> node = db.node('hostname')
>>> print(node)
hostname
```

Node scope

The Node objects are a bit more special in that they can query for facts and resources themselves. Using those methods from a node object will automatically add a query to the request scoping the request to the node.

```
>>> node = db.node('hostname')
>>> print(node.fact('osfamily'))
osfamily/hostname
```

1.1.3 Facts

```
>>> facts = db.facts('osfamily')
>>> for fact in facts:
>>>   print(fact)
osfamily/host1
osfamily/host2
```

That queries PuppetDB for the 'osfamily' fact and will yield Fact objects, one per node this fact is known for.

1.1.4 Resources

```
>>> resources = db.resources('file')
```

Will return a generator object containing all file resources you're managing across your infrastructure. This is probably a bad idea if you have a big number of nodes as the response will be huge.

1.1.5 SSL

If PuppetDB and the tool that's using pypuppetdb aren't located on the same machine you will have to connect securely to PuppetDB using client certificates according to PuppetDB's default configuration.

You can also tell PuppetDB to accept plain connections from anywhere instead of just the local machine but **don't do that**.

Pypuppetdb can handle this easily for you. It requires two things:

- Generate with your Puppet CA a key pair that you want to use
- Tell pypuppetdb to use this keypair.

Generate keypair

On your Puppet Master or dedicated Puppet CA server:

```
$ puppet cert generate <service_name>
```

Once that's done you'll need to get the public and private keyfile and copy them over. You can find those in Puppet's `$ssldir`, usually `/var/lib/puppet/ssl`:

- private key: `$ssldir/private_keys/<service_name>.pem`
- public key: `$ssldir/ca/signed/<service_name>.pem`

Configure pypuppetdb for SSL

Once you have those you can pass them to pypuppetdb's `connect()`:

```
>>> db = connect(ssl_key='/path/to/private.pem', ssl_cert='/path/to/public.pem')
```

If both `ssl_key` and `ssl_cert` are provided pypuppetdb will automatically switch over to using HTTPS instead.

By default pypuppetdb will also verify the certificate PuppetDB is serving. This means that the authority that signed PuppetDB's server certificate, most likely your Puppet Master, must be part of the trusted set of certificates for your OS or must be added to that set. Those certificates are usually found in `/etc/ssl/certs` on Linux-y machines.

For Debian, install your Puppet Master's certificate in `/usr/local/share/ca-certifiactes` with a `.crt` extension and then run `dpkg-reconfigure ca-certificates` as per `/usr/share/doc/ca-certificates/README.Debian`. This of course requires the `ca-certificates` package to be installed.

If you do not wish to do so or for whatever reason want to disable the verification of PuppetDB's certificate you can pass in `ssl_verify=False`.

CHAPTER 2

API Documentation

This part of the documentation focusses on the classes, methods and functions that make up this library.

2.1 Developer Interface

This part of the documentation covers all the interfaces of PyPuppetDB. It will cover how the API is set up and how to configure which version of the API to use.

2.1.1 Lazy objects

Note: Reading in the response from PuppetDB is currently greedy, it will read in the complete response no matter the size. This will change once streaming and pagination support are added to PuppetDB's endpoints.

In order for pypuppetdb to be able to deal with big datasets those functions that are expected to return more than a single item are implemented as generators.

This is usually the case for functions with a plural name like `nodes()` or `facts()`.

Because of this we'll only query PuppetDB once you start iterating over the generator object. Until that time not a single request is fired at PuppetDB.

Most singular functions are implemented by calling their plural counterpart and then iterating over the generator, immediately exhausting the generator and returning a single/the first object.

2.1.2 Main Interface

What you'll usually need to do is use the `connect()` method to set up a connection with PuppetDB and indicate which version of the API you want to talk. .. autofunction:: connect

2.1.3 API objects

The PuppetDB API is versioned. We currently have a v1, v2 and v3.

In order to work with this structure PyPuppetDB consists of a `BaseAPI` class that factors out identical code between different versions.

Every version of the API has its own class which inherits from our `BaseAPI`.

pypuppetdb.**API_VERSIONS**

`dict` of `int:string` pairs representing the API version and it's URL prefix.

We currently only handle API version 2 though it should be fairly easy to support version 1 should we want to.

BaseAPI

class pypuppetdb.api.**BaseAPI**(*api_version*, *host=u'localhost'*, *port=8080*, *ssl_verify=True*, *ssl_key=None*, *ssl_cert=None*, *timeout=10*)

This is a Base or Abstract class and is not meant to be instantiated or used directly.

The BaseAPI object defines a set of methods that can be reused across different versions of the PuppetDB API. If querying for a certain resource is done in an identical fashion across different versions it will be implemented here and should be overridden in their respective versions if they deviate.

If `ssl` is set to *True* but either `ssl_key` or `ssl_cert` are *None* this will raise an error.

When at initialisation `api_version` isn't found in `API_VERSIONS` this will raise an error.

Parameters

- **api_version** (`int`) – Version of the API we're initialising.
- **host** (`string`) – (optional) Hostname or IP of PuppetDB.
- **port** (`int`) – (optional) Port on which to talk to PuppetDB.
- **ssl_verify** (`bool`) – (optional) Verify PuppetDB server certificate.
- **ssl_key** (`None` or `string` representing a filesystem path.) – (optional) Path to our client secret key.
- **ssl_cert** (`None` or `string` representing a filesystem path.) – (optional) Path to our client certificate.
- **timeout** (`int`) – (optional) Number of seconds to wait for a response.

Raises `ImproperlyConfiguredError`

Raises `UnsupportedVersionError`

_query(*endpoint*, *path=None*, *query=None*, *order_by=None*, *limit=None*, *offset=None*, *include_total=False*, *summarize_by=None*, *count_by=None*, *count_filter=None*)

This method actually querries PuppetDB. Provided an endpoint and an optional path and/or query it will fire a request at PuppetDB. If PuppetDB can be reached and answers within the timeout we'll decode the response and give it back or raise for the HTTP Status Code PuppetDB gave back.

Parameters

- **endpoint** (`string`) – The PuppetDB API endpoint we want to query.
- **path** (`string`) – An additional path if we don't wish to query the bare endpoint.
- **query** (`string`) – (optional) A query to further narrow down the resultset.
- **order_by** (`bool`) – (optional) Set the order parameters for the resultset.
- **limit** (`int`) – (optional) Tell PuppetDB to limit it's response to this number of objects.
- **offset** (`string`) – (optional) Tell PuppetDB to start it's response from the given offset. This is useful for implementing pagination but is not supported just yet.
- **include_total** – (optional) Include the total number of results
- **summarize_by** (`string`) – (optional) Specify what type of object you'd like to see counts at the event-counts and aggregate-event-counts endpoints

- **count_by** (`string`) – (optional) Specify what type of object is counted
- **count_filter** (`string`) – (optional) Specify a filter for the results

Raises `EmptyResponseError`

Returns The decoded response from PuppetDB

Return type `dict` or `list`

`_url`(*endpoint*, *path=None*)

The complete URL we will end up querying. Depending on the endpoint we pass in this will result in different URL's with different prefixes.

Parameters

- **endpoint** (`string`) – The PuppetDB API endpoint we want to query.
- **path** (`string`) – An additional path if we don't wish to query the bare endpoint.

Returns A URL constructed from `base_url()` with the apropraite API version/prefix and the rest of the path added to it.

Return type `string`

`base_url`

A base_url that will be used to construct the final URL we're going to query against.

Returns A URL of the form: `proto://host:port`.

Return type `string`

`metric`(*metric*)

Query for a specific metrc.

Parameters **metric** (`string`) – The name of the metric we want.

Returns The return of `_query()`.

`total`

The total-count of the last request to PuppetDB if enabled as parameter in _query method

:returns Number of total results :rtype `int`

`version`

The version of the API we're querying against.

Returns Current API version.

Return type `string`

v2.API

class `pypuppetdb.api.v2.`**`API`**(**args*, ***kwargs*)

Bases: `pypuppetdb.api.BaseAPI`

The API object for version 2 of the PuppetDB API. This object contains all v2 specific methods and ways of doing things.

Parameters ****kwargs** – Rest of the keyword arguments passed on to our parent `BaseAPI`.

`_query`(*endpoint*, *path=None*, *query=None*, *order_by=None*, *limit=None*, *offset=None*, *include_total=False*, *summarize_by=None*, *count_by=None*, *count_filter=None*)

This method actually querries PuppetDB. Provided an endpoint and an optional path and/or query it will fire a request at PuppetDB. If PuppetDB can be reached and answers within the timeout we'll decode the response and give it back or raise for the HTTP Status Code PuppetDB gave back.

Parameters

- **endpoint** (`string`) – The PuppetDB API endpoint we want to query.
- **path** (`string`) – An additional path if we don't wish to query the bare endpoint.
- **query** (`string`) – (optional) A query to further narrow down the resultset.
- **order_by** (`bool`) – (optional) Set the order parameters for the resultset.
- **limit** (`int`) – (optional) Tell PuppetDB to limit it's response to this number of objects.
- **offset** (`string`) – (optional) Tell PuppetDB to start it's response from the given offset. This is useful for implementing pagination but is not supported just yet.
- **include_total** – (optional) Include the total number of results
- **summarize_by** (`string`) – (optional) Specify what type of object you'd like to see counts at the event-counts and aggregate-event-counts endpoints
- **count_by** (`string`) – (optional) Specify what type of object is counted
- **count_filter** (`string`) – (optional) Specify a filter for the results

Raises `EmptyResponseError`

Returns The decoded response from PuppetDB

Return type `dict` or `list`

`_url`(*endpoint*, *path=None*)
The complete URL we will end up querying. Depending on the endpoint we pass in this will result in different URL's with different prefixes.

Parameters

- **endpoint** (`string`) – The PuppetDB API endpoint we want to query.
- **path** (`string`) – An additional path if we don't wish to query the bare endpoint.

Returns A URL constructed from `base_url()` with the apropraite API version/prefix and the rest of the path added to it.

Return type `string`

`base_url`
A base_url that will be used to construct the final URL we're going to query against.

Returns A URL of the form: `proto://host:port`.

Return type `string`

`fact_names`()
Get a list of all known facts.

`facts`(*name=None*, *value=None*, *query=None*)
Query for facts limited by either name, value and/or query. This will yield a single Fact object at a time.

`metric`(*metric*)
Query for a specific metrc.

Parameters **metric** (`string`) – The name of the metric we want.

Returns The return of `_query()`.

node(*name*)
Gets a single node from PuppetDB.

nodes(*name=None*, *query=None*)
Query for nodes by either name or query. If both aren't provided this will return a list of all nodes.

Parameters

- **name** (`None` or `string`) – (optional)
- **query** (`None` or `string`) – (optional)

Returns A generator yieling Nodes.

Return type `pypuppetdb.types.Node`

resources(*type_=None*, *title=None*, *query=None*)
Query for resources limited by either type and/or title or query. This will yield a Resources object for every returned resource.

total
The total-count of the last request to PuppetDB if enabled as parameter in _query method

:returns Number of total results :rtype `int`

version
The version of the API we're querying against.

Returns Current API version.

Return type `string`

v3.API

class `pypuppetdb.api.v3.`**API**(**args*, ***kwargs*)
Bases: `pypuppetdb.api.BaseAPI`

The API object for version 3 of the PuppetDB API. This object contains all v3 specific methods and ways of doing things.

Parameters ****kwargs** – Rest of the keyword arguments passed on to our parent `BaseAPI`.

_query(*endpoint*, *path=None*, *query=None*, *order_by=None*, *limit=None*, *offset=None*, *include_total=False*, *summarize_by=None*, *count_by=None*, *count_filter=None*)
This method actually querries PuppetDB. Provided an endpoint and an optional path and/or query it will fire a request at PuppetDB. If PuppetDB can be reached and answers within the timeout we'll decode the response and give it back or raise for the HTTP Status Code PuppetDB gave back.

Parameters

- **endpoint** (`string`) – The PuppetDB API endpoint we want to query.
- **path** (`string`) – An additional path if we don't wish to query the bare endpoint.
- **query** (`string`) – (optional) A query to further narrow down the resultset.
- **order_by** (`bool`) – (optional) Set the order parameters for the resultset.
- **limit** (`int`) – (optional) Tell PuppetDB to limit it's response to this number of objects.
- **offset** (`string`) – (optional) Tell PuppetDB to start it's response from the given offset. This is useful for implementing pagination but is not supported just yet.
- **include_total** – (optional) Include the total number of results

- **summarize_by** (`string`) – (optional) Specify what type of object you'd like to see counts at the event-counts and aggregate-event-counts endpoints
- **count_by** (`string`) – (optional) Specify what type of object is counted
- **count_filter** (`string`) – (optional) Specify a filter for the results

Raises `EmptyResponseError`

Returns The decoded response from PuppetDB

Return type `dict` or `list`

`_url`(*endpoint*, *path=None*)
The complete URL we will end up querying. Depending on the endpoint we pass in this will result in different URL's with different prefixes.

Parameters

- **endpoint** (`string`) – The PuppetDB API endpoint we want to query.
- **path** (`string`) – An additional path if we don't wish to query the bare endpoint.

Returns A URL constructed from `base_url()` with the apropraite API version/prefix and the rest of the path added to it.

Return type `string`

`aggregate_event_counts`(*query*, *summarize_by*, *count_by=None*, *count_filter=None*)
Get event counts from puppetdb

`base_url`
A base_url that will be used to construct the final URL we're going to query against.

Returns A URL of the form: `proto://host:port`.

Return type `string`

`catalog`(*node*)
Get the most recent catalog for a given node

`current_version`()
Get version information about the running PuppetDB server

`event_counts`(*query*, *summarize_by*, *count_by=None*, *count_filter=None*)
Get event counts from puppetdb

`events`(*query*)
A report is made up of events. This allows to query for events based on the reprt hash. This yields an Event object for every returned event.

`fact_names`()
Get a list of all known facts.

`facts`(*name=None*, *value=None*, *query=None*)
Query for facts limited by either name, value and/or query. This will yield a single Fact object at a time.

`metric`(*metric*)
Query for a specific metrc.

Parameters **metric** (`string`) – The name of the metric we want.

Returns The return of `_query()`.

`node`(*name*)
Gets a single node from PuppetDB.

nodes(*name=None*, *query=None*, *unreported=2*, *with_status=False*)
Query for nodes by either name or query. If both aren't provided this will return a list of all nodes. This method also fetches the nodes status and event counts of the latest report from puppetdb.

Parameters

- **name** (`None` or `string`) – (optional)
- **query** (`None` or `string`) – (optional)
- **with_status** – (optional) include the node status in the returned nodes
- **unreported** (`None` or integer) – (optional) amount of hours when a node gets marked as unreported

Returns A generator yieling Nodes.

Return type `pypuppetdb.types.Node`

reports(*query*)
Get reports for our infrastructure. Currently reports can only be filtered through a query which requests a specific certname. If not it will return all reports.

This yields a Report object for every returned report.

resources(*type_=None*, *title=None*, *query=None*)
Query for resources limited by either type and/or title or query. This will yield a Resources object for every returned resource.

server_time()
Get the current time of the clock on the PuppetDB server

total
The total-count of the last request to PuppetDB if enabled as parameter in _query method

:returns Number of total results :rtype `int`

version
The version of the API we're querying against.

Returns Current API version.

Return type `string`

2.1.4 Types

In order to facilitate working with the API most methods like `nodes()` don't return the decoded JSON response but return an object representation of the querried endpoints data.

class `pypuppetdb.types.`**Node**(*api*, *name*, *deactivated=None*, *report_timestamp=None*, *catalog_timestamp=None*, *facts_timestamp=None*, *status=None*, *events=None*, *unreported_time=None*)
This object represents a node. It additionally has some helper methods so that you can query for resources or facts directly from the node scope.

Parameters

- **api** – API object.
- **name** – Hostname of this node.
- **deactivated** (`string` formatted as `%Y-%m-%dT%H:%M:%S.%fZ`) – (default *None*) Time this node was deactivated at.

- **report_timestamp** (`string` formatted as `%Y-%m-%dT%H:%M:%S.%fZ`) – (default *None*) Time of the last report.
- **catalog_timestamp** (`string` formatted as `%Y-%m-%dT%H:%M:%S.%fZ`) – (default *None*) Time the last time a catalog was compiled.
- **facts_timestamp** (`string` formatted as `%Y-%m-%dT%H:%M:%S.%fZ`) – (default *None*) Time the last time facts were collected.
- **status** (`string`) – (default *None*) Status of the node changed | unchanged | unreported | failed
- **events** (`dict`) – (default *None*) Counted events from latest Report
- **unreported_time** (`string`) – (default *None*) Time since last report

Variables

- **name** – Hostname of this node.
- **deactivated** – `datetime.datetime` when this host was deactivated or *False*.
- **report_timestamp** – `datetime.datetime` when the last run occured or *None*.
- **catalog_timestamp** – `datetime.datetime` last time a catalog was compiled or *None*.
- **facts_timestamp** – `datetime.datetime` last time when facts were collected or *None*.

fact(*name*)
Get a single fact from this node.

facts()
Get all facts of this node.

reports()
Get all reports for this node.

resource(*type_*, *title*)
Get a resource matching the supplied type and title.

resources(*type_=None*, *title=None*)
Get all resources of this node or all resources of the specified type.

class `pypuppetdb.types.`**Fact**(*node*, *name*, *value*)
his object represents a fact.

Parameters

- **node** – The hostname this fact was collected from.
- **name** – The fact's name, such as 'osfamily'
- **value** – The fact's value, such as 'Debian'

Variables

- **node** – `string` holding the hostname.
- **name** – `string` holding the fact's name.
- **value** – `string` holding the fact's value.

class `pypuppetdb.types.`**Resource**(*node*, *name*, *type_*, *tags*, *exported*, *sourcefile*, *sourceline*, *parameters={}*)
This object represents a resource.

Parameters

- **node** – The hostname this resource is located on.
- **name** – The name of the resource in the Puppet manifest.
- **type_** – Type of the Puppet resource.
- **tags** (`list`) – Tags associated with this resource.
- **exported** (`bool`) – If it's an exported resource.
- **sourcefile** – The Puppet manifest this resource is declared in.
- **sourceline** – The line this resource is declared at.
- **parameters** (`dict`) – The parameters this resource has been declared with.

Variables

- **node** – The hostname this resources is located on.
- **name** – The name of the resource in the Puppet manifest.
- **type_** – The type of Puppet resource.
- **exported** – `bool` if the resource is exported.
- **sourcefile** – The Puppet manifest this resource is declared in.
- **sourceline** – The line this resource is declared at.
- **parameters** – `dict` with key:value pairs of parameters.

class `pypuppetdb.types.`**`Event`** (*node*, *status*, *timestamp*, *hash_*, *title*, *property_*, *message*, *new_value*, *old_value*, *type_*)

This object represents an event.

Parameters

- **node** – The hostname of the node this event fired on.
- **status** – The status for the event.
- **timestamp** – A timestamp of when this event occured.
- **hash_** – The hash of this event.
- **title** – The resource title this event was fired for.
- **property_** – The property of the resource this event was fired for.
- **message** – A message associated with this event.
- **new_value** – The new value/state of the resource.
- **old_value** – The old value/state of the resource.
- **type_** – The type of the resource this event fired for.

Variables

- **status** – A `string` of this event's status.
- **failed** – The `bool` equivalent of *status*.
- **timestamp** – A `datetime.datetime` of when this event happend.
- **node** – The hostname of the machine this event occured on.
- **hash_** – The hash of this event.
- **item** – `dict` with information about the item/resource this event was triggered for.

class `pypuppetdb.types.`**`Report`**(*node*, *hash_*, *start*, *end*, *received*, *version*, *format_*, *agent_version*, *transaction*)

This object represents a report.

Parameters

- **node** – The hostname of the node this report originated on.
- **hash_** – A string uniquely identifying this report.
- **start** (`string` formatted as `%Y-%m-%dT%H:%M:%S.%fZ`) – The start time of the agent run.
- **end** (`string` formatted as `%Y-%m-%dT%H:%M:%S.%fZ`) – The time the agent finished its run.
- **received** (`string` formatted as `%Y-%m-%dT%H:%M:%S.%fZ`) – The time PuppetDB received the report.
- **version** (`string`) – The catalog / configuration version.
- **format_** (`int`) – The catalog format version.
- **agent_version** (`string`) – The Puppet agent version.
- **transaction** (`string`) – The UUID of this transaction.

Variables

- **node** – The hostname this report originated from.
- **hash_** – Unique identifier of this report.
- **start** – `datetime.datetime` when the Puppet agent run started.
- **end** – `datetime.datetime` when the Puppet agent run ended.
- **received** – `datetime.datetime` when the report finished uploading.
- **version** – `string` catalog configuration version.
- **format_** – `int` catalog format version.
- **agent_version** – `string` Puppet Agent version.
- **run_time** – `datetime.timedelta` of **end** - **start**.
- **transaction** – UUID identifying this transaction.

class `pypuppetdb.types.`**`Catalog`**(*node*, *edges*, *resources*, *version*, *transaction_uuid*)

This object represents a compiled catalog from puppet. It contains Resource and Edge object that represent the dependency graph.

Parameters

- **node** – Name of the host
- **edges** (`list` containing `dict` with Edge information) – Edges returned from Catalog data
- **resources** (`list` containing `dict` with Resources) – Resources returned from Catalog data
- **version** (`string`) – Catalog version from Puppet (unique for each node)
- **transaction_uuid** (`string`) – A string used to match the catalog with the corresponding report that was issued during the same puppet run

Variables

- **node** – `string` Name of the host

- **version** – `string` Catalog version from Puppet (unique for each node)
- **transaction_uuid** – `string` used to match the catalog with corresponding report
- **edges** – `list` of `Edge` The source Resource object of the relationship
- **resources** – `dict` of `Resource` The source Resource object of the relationship

class `pypuppetdb.types.`**`Edge`**(*source*, *target*, *relationship*)
This object represents the connection between two Resource objects

Parameters

- **source** (`Resource`) – The source Resource object of the relationship
- **target** (`Resource`) – The target Resource object of the relationship
- **relaptionship** – Name of the Puppet Ressource Relationship

Variables

- **source** – `Resource` The source Resource object
- **target** – `Resource` The target Resource object
- **relationship** – `string` Name of the Puppet Resource relationship

2.1.5 Errors

Unfortunately things can go haywire. PuppetDB might not be reachable or complain about our query, requests might have to wait too long to recieve a response or the body is just too big to handle.

In that case, we'll throw an exception at you.

exception `pypuppetdb.errors.`**`APIError`**
Our base exception the other errors inherit from.

exception `pypuppetdb.errors.`**`ImproperlyConfiguredError`**
Bases: `pypuppetdb.errors.APIError`

This exception is thrown when the API is initialised and it detects incompatbile configuration such as SSL turned on but no certificates provided.

exception `pypuppetdb.errors.`**`UnsupportedVersionError`**
Bases: `pypuppetdb.errors.APIError`

Triggers when using the `connect()` function and providing it with an unknown API version.

exception `pypuppetdb.errors.`**`DoesNotComputeError`**
Bases: `pypuppetdb.errors.APIError`

This error will be thrown when a function is called with an incompatible set of optional parameters. This is the 'you are being a naughty developer, go read the docs' error.

exception `pypuppetdb.errors.`**`EmptyResponseError`**
Bases: `pypuppetdb.errors.APIError`

Will be thrown when we did recieve a response but the response is empty.

2.1.6 Utilities

A few functions that are used across this library have been put into their own `utils` module.

class `pypuppetdb.utils.`**`UTC`**

UTC

`pypuppetdb.utils.`**`json_to_datetime`**`(`*date*`)`

Tranforms a JSON datetime string into a timezone aware datetime object with a UTC tzinfo object.

Parameters **date** (`string`) – The datetime representation.

Returns A timezone aware datetime object.

Return type `datetime.datetime`

CHAPTER 3

Indices and tables

- *genindex*
- *search*

Python Module Index