Hetzner Cloud Python

Release 2.20.0

Hetzner Cloud GmbH

May 19, 2026

CONTENTS

CHAPTER

ONE

STABLE RELEASE

To install Hetzner Cloud Python, run this command in your terminal:

```
$ pip install hcloud
```

This is the preferred method to install Hetzner Cloud Python, as it will always install the most recent stable release.

If you don't have pip installed, this Python installation guide can guide you through the process.

CHAPTER

TWO

VIA CONDA (THIRD-PARTY)

Hetzner Cloud Python is also available as a `conda`-package via *conda-forge*. This package is not maintained by Hetzner Cloud and might be outdated._:

```
$ conda install -c conda-forge hcloud
```

CHAPTER

THREE

FROM SOURCES

The sources for Hetzner Cloud Python can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/hetznercloud/hcloud-python
```

Or download the tarball:

```
$ curl  -OL https://github.com/hetznercloud/hcloud-python/tarball/main
```

Once you have a copy of the source, you can install it with:

```
$ pip install .
```

CHAPTER

FOUR

API REFERENCES

4.1 Main Interface

class Client(*token: str*, *api_endpoint: str = 'https://api.hetzner.cloud/v1'*, *application_name: str | None = None*, *application_version: str | None = None*, *poll_interval: int | float | BackoffFunction = 1.0*, *poll_max_retries: int = 120*, *timeout: float | tuple[float, float] | None = None*, ** (Keyword-only parameters separator (PEP 3102))*, *api_endpoint_hetzner: str = 'https://api.hetzner.com/v1'*)

Client for the Hetzner Cloud API.

The Hetzner Cloud API reference is available at https://docs.hetzner.cloud.

Make sure to follow our API changelog available at https://docs.hetzner.cloud/changelog (or the RRS feed available at https://docs.hetzner.cloud/changelog/feed.rss) to be notified about additions, deprecations and removals.

Retry mechanism

The *Client.request* method will retry failed requests that match certain criteria. The default retry interval is defined by an exponential backoff algorithm truncated to 60s with jitter. The default maximal number of retries is 5.

The following rules define when a request can be retried:

- When the client returned a network timeout error.
- When the API returned an HTTP error, with the status code:
  - `502` Bad Gateway
  - `504` Gateway Timeout
- When the API returned an application error, with the code:
  - `conflict`
  - `rate_limit_exceeded`
  - `timeout`

Changes to the retry policy might occur between releases, and will not be considered breaking changes.

actions

ActionsClient Instance

Type
ActionsClient

certificates

CertificatesClient Instance

Type
CertificatesClient

datacenters

DatacentersClient Instance

Type
DatacentersClient

firewalls

FirewallsClient Instance

Type
FirewallsClient

floating_ips

FloatingIPsClient Instance

Type
FloatingIPsClient

images

ImagesClient Instance

Type
ImagesClient

isos

ImagesClient Instance

Type
IsosClient

load_balancer_types

LoadBalancerTypesClient Instance

Type
LoadBalancerTypesClient

load_balancers

LoadBalancersClient Instance

Type
LoadBalancersClient

locations

LocationsClient Instance

Type
LocationsClient

networks

NetworksClient Instance

Type
NetworksClient

placement_groups

PlacementGroupsClient Instance

Type
PlacementGroupsClient

primary_ips

PrimaryIPsClient Instance

Type
PrimaryIPsClient

request(*method: str*, *url: str*, ***kwargs*) → dict[str, Any]

Perform a request to the Hetzner Cloud API.

Parameters

- **method** – Method to perform the request.
- **url** – URL to perform the request.
- **timeout** – Requests timeout in seconds.

server_types

ServerTypesClient Instance

Type
ServerTypesClient

servers

ServersClient Instance

Type
ServersClient

ssh_keys

SSHKeysClient Instance

Type
SSHKeysClient

storage_box_types

StorageBoxTypesClient Instance

Type
StorageBoxTypesClient

storage_boxes

StorageBoxesClient Instance

Type
StorageBoxesClient

volumes

VolumesClient Instance

Type
VolumesClient

zones

ZonesClient Instance

Type
ZonesClient

4.2 API Clients

4.2.1 ActionsClient

class ResourceActionsClient(*client: ResourceClientBase | Client*, *resource: str | None*)

get_all(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_by_id(*id: int*) → *BoundAction*

Returns a specific Action by its ID.

Parameters
id – ID of the Action.

get_list(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

class ActionsClient(*client: Client*)

get_all(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Deprecated since version 1.28: Use `client.<resource>.actions.get_all()` instead, e.g. using *hcloud.certificates.client.CertificatesClient.actions*.

Starting 1 October 2023, it will no longer be available.

get_by_id(*id: int*) → *BoundAction*

Returns a specific Action by its ID.

Parameters
id – ID of the Action.

get_list(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Deprecated since version 1.28: Use `client.<resource>.actions.get_list()` instead, e.g. using *hcloud.certificates.client.CertificatesClient.actions*.

Starting 1 October 2023, it will no longer be available.

class BoundAction(*client: ResourceClientBase*, *data: dict[str, Any]*, *complete: bool = True*)

model

alias of *Action*

wait_until_finished(*max_retries: int | None = None*) → None

Wait until the specific action has status=finished.

Parameters

max_retries – int Specify how many retries will be performed before an ActionTimeoutException will be raised.

Raises

ActionFailedException when action is finished with status==error

Raises

ActionTimeoutException when Action is still in status==running after max_retries is reached.

class Action(*id: int*, *command: str | None = None*, *status: Literal['running', 'success', 'error'] | None = None*, *progress: int | None = None*, *started: str | None = None*, *finished: str | None = None*, *resources: list[ActionResource] | None = None*, *error: ActionError | None = None*)

Action Domain

Parameters

- **id** – int ID of an action
- **command** – Command executed in the action
- **status** – Status of the action
- **progress** – Progress of action in percent
- **started** – Point in time when the action was started
- **finished** (*datetime,None*) – Point in time when the action was finished. Only set if the action is finished otherwise None
- **resources** – Resources the action relates to
- **error** – Error message for the action if error occurred, otherwise None.

STATUS_ERROR = 'error'

Action Status error

STATUS_RUNNING = 'running'

Action Status running

STATUS_SUCCESS = 'success'

Action Status success

4.2.2 CertificateClient

class CertificatesClient(*client:* Client)

actions: *ResourceActionsClient*

Certificates scoped actions client

Type
ResourceActionsClient

create(*name: str*, *certificate: str*, *private_key: str*, *labels: dict[str, str] | None = None*) → *BoundCertificate*

Creates a new Certificate with the given name, certificate and private_key. This methods allows only creating
custom uploaded certificates. If you want to create a managed certificate use *create_managed()*

Parameters

- **name** – str
- **certificate** – str Certificate and chain in PEM format, in order so that each record directly certifies the one preceding
- **private_key** – str Certificate key in PEM format
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundCertificate

create_managed(*name: str*, *domain_names: list[str]*, *labels: dict[str, str] | None = None*) → CreateManagedCertificateResponse

Creates a new managed Certificate with the given name and domain names. This methods allows only creating
managed certificates for domains that are using the Hetzner DNS service. If you want to create a custom uploaded certificate use *create()*

Parameters

- **name** – str
- **domain_names** – List[str] Domains and subdomains that should be contained in the Certificate
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundCertificate

delete(*certificate:* Certificate | BoundCertificate) → bool

Deletes a certificate.

Parameters
certificate – *BoundCertificate* or *Certificate*

Returns
True

get_actions(*certificate:* Certificate | BoundCertificate, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Certificate.

Parameters

- **certificate** – Certificate to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*certificate: Certificate | BoundCertificate*, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Certificate.

Parameters

- **certificate** – Certificate to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_all(*name: str | None = None*, *label_selector: str | None = None*) → list[*BoundCertificate*]

Get all certificates

Parameters

- **name** – str (optional) Can be used to filter certificates by their name.
- **label_selector** – str (optional) Can be used to filter certificates by labels. The response will only contain certificates matching the label selector.

Returns

List[*BoundCertificate*]

get_by_id(*id: int*) → *BoundCertificate*

Get a specific certificate by its ID.

Parameters

id – int

Returns

BoundCertificate

get_by_name(*name: str*) → *BoundCertificate* | None

Get certificate by name

Parameters

name – str Used to get certificate by name.

Returns

BoundCertificate

get_list(*name: str | None = None*, *label_selector: str | None = None*, *page: int | None = None*, *per_page: int | None = None*) → CertificatesPageResult

Get a list of certificates

Parameters

- **name** – str (optional) Can be used to filter certificates by their name.

- **label_selector** – str (optional) Can be used to filter certificates by labels. The response will only contain certificates matching the label selector.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page

Returns
(List[*BoundCertificate*], Meta)

retry_issuance(*certificate:* Certificate | BoundCertificate) → *BoundAction*

Returns all action objects for a Certificate.

Parameters
certificate – *BoundCertificate* or *Certificate*

Returns
BoundAction

update(*certificate:* Certificate | BoundCertificate, *name: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundCertificate*

Updates a Certificate. You can update a certificate name and labels.

Parameters

- **certificate** – *BoundCertificate* or *Certificate*
- **name** – str (optional) New name to set
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundCertificate

class BoundCertificate(*client:* CertificatesClient, *data: dict[str, Any]*, *complete: bool = True*)

delete() → bool

Deletes a certificate. :return: boolean

get_actions(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Certificate.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Certificate.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

model

alias of *Certificate*

retry_issuance() → *BoundAction*

Retry a failed Certificate issuance or renewal. :return: BoundAction

update(*name: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundCertificate*

Updates an certificate. You can update an certificate name and the certificate labels.

Parameters

- **name** – str (optional) New name to set
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns

BoundCertificate

class Certificate(*id: int | None = None*, *name: str | None = None*, *certificate: str | None = None*, *not_valid_before: str | None = None*, *not_valid_after: str | None = None*, *domain_names: list[str] | None = None*, *fingerprint: str | None = None*, *created: str | None = None*, *labels: dict[str, str] | None = None*, *type: str | None = None*, *status: ManagedCertificateStatus | None = None*)

Certificate Domain

Parameters

- **id** – int ID of Certificate
- **name** – str Name of Certificate
- **certificate** – str Certificate and chain in PEM format, in order so that each record directly certifies the one preceding
- **not_valid_before** – datetime Point in time when the Certificate becomes valid
- **not_valid_after** – datetime Point in time when the Certificate becomes invalid
- **domain_names** – List[str] List of domains and subdomains covered by this certificate
- **fingerprint** – str Fingerprint of the Certificate
- **labels** – dict User-defined labels (key-value pairs)
- **created** – datetime Point in time when the certificate was created
- **type** – str Type of Certificate
- **status** – ManagedCertificateStatus Current status of a type managed Certificate, always none for type uploaded Certificates

4.2.3 DatacentersClient

class DatacentersClient(*client:* Client)

get_all(*name: str | None = None*) → list[*BoundDatacenter*]

Get all datacenters

Parameters

name – str (optional) Can be used to filter datacenters by their name.

Returns
List[*BoundDatacenter*]

get_by_id(*id: int*) → *BoundDatacenter*

Get a specific datacenter by its ID.

Parameters
id – int

Returns
BoundDatacenter

get_by_name(*name: str*) → *BoundDatacenter* | None

Get datacenter by name

Parameters
name – str Used to get datacenter by name.

Returns
BoundDatacenter

get_list(*name: str | None = None*, *page: int | None = None*, *per_page: int | None = None*) → DatacentersPageResult

Get a list of datacenters

Parameters

- **name** – str (optional) Can be used to filter datacenters by their name.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page

Returns
(List[*BoundDatacenter*], Meta)

class BoundDatacenter(*client: DatacentersClient*, *data: dict[str, Any]*)

model

alias of *Datacenter*

class Datacenter(*id: int | None = None*, *name: str | None = None*, *description: str | None = None*, *location: Location | None = None*, *server_types: DatacenterServerTypes | None = None*)

Datacenter Domain

Parameters

- **id** – int ID of Datacenter
- **name** – str Name of Datacenter
- **description** – str Description of Datacenter
- **location** – *BoundLocation*
- **server_types** – *DatacenterServerTypes*

property server_types: *DatacenterServerTypes* | None

Deprecated since version 2.18.0: The 'server_types' property is deprecated and will not be supported after 2026-10-01. Please use 'server_type.locations[]' instead.

See https://docs.hetzner.cloud/changelog#2026-04-01-datacenter-deprecations.

class DatacenterServerTypes(*available: list[BoundServerType]*, *supported: list[BoundServerType]*, *available_for_migration: list[BoundServerType]*)

DatacenterServerTypes Domain

Parameters

- **available** – List[BoundServerTypes] All available server types for this datacenter
- **supported** – List[BoundServerTypes] All supported server types for this datacenter
- **available_for_migration** – List[BoundServerTypes] All available for migration (change type) server types for this datacenter

4.2.4 FirewallsClient

class FirewallsClient(*client: Client*)

actions: *ResourceActionsClient*

Firewalls scoped actions client

Type
ResourceActionsClient

apply_to_resources(*firewall: Firewall | BoundFirewall*, *resources: list[FirewallResource]*) → list[*BoundAction*]

Applies one Firewall to multiple resources.

Parameters

- **firewall** – *BoundFirewall* or *Firewall*
- **resources** – List[*FirewallResource*]

Returns
List[*BoundAction*]

create(*name: str*, *rules: list[FirewallRule] | None = None*, *labels: str | None = None*, *resources: list[FirewallResource] | None = None*) → *CreateFirewallResponse*

Creates a new Firewall.

Parameters

- **name** – str Firewall Name
- **rules** – List[*FirewallRule*] (optional)
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)
- **resources** – List[*FirewallResource*] (optional)

Returns
CreateFirewallResponse

delete(*firewall: Firewall | BoundFirewall*) → bool

Deletes a Firewall.

Parameters
firewall – *BoundFirewall* or *Firewall*

Returns
boolean

get_actions(*firewall: Firewall | BoundFirewall*, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Firewall.

Parameters

- **firewall** – Firewall to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*firewall: Firewall | BoundFirewall*, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Firewall.

Parameters

- **firewall** – Firewall to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_all(*label_selector: str | None = None*, *name: str | None = None*, *sort: list[str] | None = None*) → list[*BoundFirewall*]

Get all floating ips from this account

Parameters

- **label_selector** – str (optional) Can be used to filter Firewalls by labels. The response will only contain Firewalls matching the label selector values.
- **name** – str (optional) Can be used to filter networks by their name.
- **sort** – List[str] (optional) Choices: id name created (You can add one of ":asc", ":desc" to modify sort order. (":asc" is default))

Returns

List[*BoundFirewall*]

get_by_id(*id: int*) → *BoundFirewall*

Returns a specific Firewall object.

Parameters

id – int

Returns

BoundFirewall

get_by_name(*name: str*) → *BoundFirewall* | None

Get Firewall by name

Parameters

name – str Used to get Firewall by name.

Returns
BoundFirewall

get_list(*label_selector: str | None = None*, *page: int | None = None*, *per_page: int | None = None*, *name: str | None = None*, *sort: list[str] | None = None*) → FirewallsPageResult

Get a list of floating ips from this account

Parameters

- **label_selector** – str (optional) Can be used to filter Firewalls by labels. The response will only contain Firewalls matching the label selector values.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page
- **name** – str (optional) Can be used to filter networks by their name.
- **sort** – List[str] (optional) Choices: id name created (You can add one of ":asc", ":desc" to modify sort order. (":asc" is default))

Returns
(List[*BoundFirewall*], Meta)

remove_from_resources(*firewall: Firewall | BoundFirewall*, *resources: list[FirewallResource]*) → list[*BoundAction*]

Removes one Firewall from multiple resources.

Parameters

- **firewall** – *BoundFirewall* or *Firewall*
- **resources** – List[*FirewallResource*]

Returns
List[*BoundAction*]

set_rules(*firewall: Firewall | BoundFirewall*, *rules: list[FirewallRule]*) → list[*BoundAction*]

Sets the rules of a Firewall. All existing rules will be overwritten. Pass an empty rules array to remove all rules.

Parameters

- **firewall** – *BoundFirewall* or *Firewall*
- **rules** – List[*FirewallRule*]

Returns
List[*BoundAction*]

update(*firewall: Firewall | BoundFirewall*, *labels: dict[str, str] | None = None*, *name: str | None = None*) → *BoundFirewall*

Updates the description or labels of a Firewall.

Parameters

- **firewall** – *BoundFirewall* or *Firewall*
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)
- **name** – str (optional) New name to set

Returns
BoundFirewall

class **BoundFirewall**(*client: FirewallsClient*, *data: dict[str, Any]*, *complete: bool = True*)

apply_to_resources(*resources: list[FirewallResource]*) → list[*BoundAction*]

Applies one Firewall to multiple resources. :param resources: List[*FirewallResource*] :return: List[*BoundAction*]

delete() → bool

Deletes a Firewall.

Returns
boolean

get_actions(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Firewall.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Firewall.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

model

alias of *Firewall*

remove_from_resources(*resources: list[FirewallResource]*) → list[*BoundAction*]

Removes one Firewall from multiple resources. :param resources: List[*FirewallResource*] :return: List[*BoundAction*]

set_rules(*rules: list[FirewallRule]*) → list[*BoundAction*]

Sets the rules of a Firewall. All existing rules will be overwritten. Pass an empty rules array to remove all rules. :param rules: List[*FirewallRule*] :return: List[*BoundAction*]

update(*name: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundFirewall*

Updates the name or labels of a Firewall.

Parameters

- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)
- **name** – str (optional) New Name to set

Returns
BoundFirewall

class Firewall(*id: int | None = None*, *name: str | None = None*, *labels: dict[str, str] | None = None*, *rules: list[FirewallRule] | None = None*, *applied_to: list[FirewallResource] | None = None*, *created: str | None = None*)

Firewall Domain

Parameters

- **id** – int ID of the Firewall
- **name** – str Name of the Firewall
- **labels** – dict User-defined labels (key-value pairs)
- **rules** – List[*FirewallRule*] Rules of the Firewall
- **applied_to** – List[*FirewallResource*] Resources currently using the Firewall
- **created** – datetime Point in time when the image was created

class FirewallRule(*direction: str*, *protocol: str*, *source_ips: list[str] | None = None*, *port: str | None = None*, *destination_ips: list[str] | None = None*, *description: str | None = None*)

Firewall Rule Domain

Parameters

- **direction** – str The Firewall which was created
- **port** – str Port to which traffic will be allowed, only applicable for protocols TCP and UDP, specify port ranges by using - as a indicator, Sample: 80-85 means all ports between 80 & 85 (80, 82, 83, 84, 85)
- **protocol** – str Select traffic direction on which rule should be applied. Use source_ips for direction in and destination_ips for direction out.
- **source_ips** – List[str] List of permitted IPv4/IPv6 addresses in CIDR notation. Use 0.0.0.0/0 to allow all IPv4 addresses and ::/0 to allow all IPv6 addresses. You can specify 100 CIDRs at most.
- **destination_ips** – List[str] List of permitted IPv4/IPv6 addresses in CIDR notation. Use 0.0.0.0/0 to allow all IPv4 addresses and ::/0 to allow all IPv6 addresses. You can specify 100 CIDRs at most.
- **description** – str Short description of the firewall rule

DIRECTION_IN = 'in'

Firewall Rule Direction In

DIRECTION_OUT = 'out'

Firewall Rule Direction Out

PROTOCOL_ESP = 'esp'

Firewall Rule Protocol ESP

PROTOCOL_GRE = 'gre'

Firewall Rule Protocol GRE

PROTOCOL_ICMP = 'icmp'

Firewall Rule Protocol ICMP

PROTOCOL_TCP = 'tcp'

Firewall Rule Protocol TCP

PROTOCOL_UDP = 'udp'

Firewall Rule Protocol UDP

to_payload() → dict[str, Any]

Generates the request payload from this domain object.

class FirewallResource(*type: str*, *server:* Server | BoundServer | *None = None*, *label_selector:* *FirewallResourceLabelSelector | None = None*, *applied_to_resources:* *list[FirewallResourceAppliedToResources] | None = None*)

Firewall Used By Domain

Parameters

- **type** – str Type of resource referenced
- **server** – Optional[Server] Server the Firewall is applied to
- **label_selector** – Optional[FirewallResourceLabelSelector] Label Selector for Servers the Firewall should be applied to
- **applied_to_resources** – (read-only) List of effective resources the firewall is applied to.

TYPE_LABEL_SELECTOR = 'label_selector'

Firewall Used By Type label_selector

TYPE_SERVER = 'server'

Firewall Used By Type Server

to_payload() → dict[str, Any]

Generates the request payload from this domain object.

class CreateFirewallResponse(*firewall:* BoundFirewall, *actions: list[*BoundAction*] | None*)

Create Firewall Response Domain

Parameters

- **firewall** – *BoundFirewall* The Firewall which was created
- **actions** – List[*BoundAction*] The Action which shows the progress of the Firewall Creation

4.2.5 Floating IPsClient

class FloatingIPsClient(*client:* Client)

actions: *ResourceActionsClient*

Floating IPs scoped actions client

Type
ResourceActionsClient

assign(*floating_ip:* FloatingIP | BoundFloatingIP, *server:* Server | BoundServer) → *BoundAction*

Assigns a Floating IP to a server.

Parameters

- **floating_ip** – *BoundFloatingIP* or *FloatingIP*
- **server** – *BoundServer* or *Server* Server the Floating IP shall be assigned to

Returns
BoundAction

change_dns_ptr(*floating_ip:* FloatingIP | BoundFloatingIP, *ip: str*, *dns_ptr: str*) → *BoundAction*

Changes the hostname that will appear when getting the hostname belonging to this Floating IP.

Parameters

- **floating_ip** – *BoundFloatingIP* or *FloatingIP*
- **ip** – str The IP address for which to set the reverse DNS entry
- **dns_ptr** – str Hostname to set as a reverse DNS PTR entry, will reset to original default value if *None*

Returns
BoundAction

change_protection(*floating_ip:* FloatingIP | BoundFloatingIP, *delete: bool | None = None*) → *BoundAction*

Changes the protection configuration of the Floating IP.

Parameters

- **floating_ip** – *BoundFloatingIP* or *FloatingIP*
- **delete** – boolean If true, prevents the Floating IP from being deleted

Returns
BoundAction

create(*type: str*, *description: str | None = None*, *labels: dict[str, str] | None = None*, *home_location:* Location | BoundLocation | *None = None*, *server:* Server | BoundServer | *None = None*, *name: str | None = None*) → *CreateFloatingIPResponse*

Creates a new Floating IP assigned to a server.

Parameters

- **type** – str Floating IP type Choices: ipv4, ipv6
- **description** – str (optional)
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)
- **home_location** – *BoundLocation* or *Location* (Home location (routing is optimized for that location). Only optional if server argument is passed.
- **server** – *BoundServer* or *Server* Server to assign the Floating IP to
- **name** – str (optional)

Returns
CreateFloatingIPResponse

delete(*floating_ip:* FloatingIP | BoundFloatingIP) → bool

Deletes a Floating IP. If it is currently assigned to a server it will automatically get unassigned.

Parameters
floating_ip – *BoundFloatingIP* or *FloatingIP*

Returns
boolean

get_actions(*floating_ip: FloatingIP | BoundFloatingIP*, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Floating IP.

Parameters

- **floating_ip** – Floating IP to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*floating_ip: FloatingIP | BoundFloatingIP*, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Floating IP.

Parameters

- **floating_ip** – Floating IP to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_all(*label_selector: str | None = None*, *name: str | None = None*) → list[*BoundFloatingIP*]

Get all floating ips from this account

Parameters

- **label_selector** – str (optional) Can be used to filter Floating IPs by labels. The response will only contain Floating IPs matching the label selector.able values.
- **name** – str (optional) Can be used to filter networks by their name.

Returns

List[*BoundFloatingIP*]

get_by_id(*id: int*) → *BoundFloatingIP*

Returns a specific Floating IP object.

Parameters

id – int

Returns

BoundFloatingIP

get_by_name(*name: str*) → *BoundFloatingIP* | None

Get Floating IP by name

Parameters

name – str Used to get Floating IP by name.

Returns

BoundFloatingIP

get_list(*label_selector: str | None = None*, *page: int | None = None*, *per_page: int | None = None*, *name: str | None = None*) → FloatingIPsPageResult

Get a list of floating ips from this account

Parameters

- **label_selector** – str (optional) Can be used to filter Floating IPs by labels. The response will only contain Floating IPs matching the label selector.able values.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page
- **name** – str (optional) Can be used to filter networks by their name.

Returns
(List[*BoundFloatingIP*], Meta)

unassign(*floating_ip: FloatingIP | BoundFloatingIP*) → *BoundAction*

Unassigns a Floating IP, resulting in it being unreachable. You may assign it to a server again at a later time.

Parameters
floating_ip – *BoundFloatingIP* or *FloatingIP*

Returns
BoundAction

update(*floating_ip: FloatingIP | BoundFloatingIP*, *description: str | None = None*, *labels: dict[str, str] | None = None*, *name: str | None = None*) → *BoundFloatingIP*

Updates the description or labels of a Floating IP.

Parameters

- **floating_ip** – *BoundFloatingIP* or *FloatingIP*
- **description** – str (optional) New Description to set
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)
- **name** – str (optional) New name to set

Returns
BoundFloatingIP

class BoundFloatingIP(*client: FloatingIPsClient*, *data: dict[str, Any]*, *complete: bool = True*)

assign(*server: Server | BoundServer*) → *BoundAction*

Assigns a Floating IP to a server.

Parameters
server – *BoundServer* or *Server* Server the Floating IP shall be assigned to

Returns
BoundAction

change_dns_ptr(*ip: str*, *dns_ptr: str*) → *BoundAction*

Changes the hostname that will appear when getting the hostname belonging to this Floating IP.

Parameters

- **ip** – str The IP address for which to set the reverse DNS entry

- **dns_ptr** – str Hostname to set as a reverse DNS PTR entry, will reset to original default value if *None*

Returns
BoundAction

change_protection(*delete: bool | None = None*) → *BoundAction*

Changes the protection configuration of the Floating IP.

Parameters
delete – boolean If true, prevents the Floating IP from being deleted

Returns
BoundAction

delete() → bool

Deletes a Floating IP. If it is currently assigned to a server it will automatically get unassigned.

Returns
boolean

get_actions(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Floating IP.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Floating IP.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

model

alias of *FloatingIP*

unassign() → *BoundAction*

Unassigns a Floating IP, resulting in it being unreachable. You may assign it to a server again at a later time.

Returns
BoundAction

update(*description: str | None = None*, *labels: dict[str, str] | None = None*, *name: str | None = None*) → *BoundFloatingIP*

Updates the description or labels of a Floating IP.

Parameters

- **description** – str (optional) New Description to set
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)
- **name** – str (optional) New Name to set

Returns

BoundFloatingIP

class FloatingIP(*id: int | None = None*, *type: str | None = None*, *description: str | None = None*, *ip: str | None = None*, *server: BoundServer | None = None*, *dns_ptr: list[DNSPtr] | None = None*, *home_location: BoundLocation | None = None*, *blocked: bool | None = None*, *protection: FloatingIPProtection | None = None*, *labels: dict[str, str] | None = None*, *created: str | None = None*, *name: str | None = None*)

Floating IP Domain

Parameters

- **id** – int ID of the Floating IP
- **description** – str, None Description of the Floating IP
- **ip** – str IP address of the Floating IP
- **type** – str Type of Floating IP. Choices: *ipv4*, *ipv6*
- **server** – *BoundServer*, None Server the Floating IP is assigned to, None if it is not assigned at all
- **dns_ptr** – List[Dict] Array of reverse DNS entries
- **home_location** – *BoundLocation* Location the Floating IP was created in. Routing is optimized for this location.
- **blocked** – boolean Whether the IP is blocked
- **protection** – dict Protection configuration for the Floating IP
- **labels** – dict User-defined labels (key-value pairs)
- **created** – datetime Point in time when the Floating IP was created
- **name** – str Name of the Floating IP

class CreateFloatingIPResponse(*floating_ip: BoundFloatingIP*, *action: BoundAction | None*)

Create Floating IP Response Domain

Parameters

- **floating_ip** – *BoundFloatingIP* The Floating IP which was created
- **action** – *BoundAction* The Action which shows the progress of the Floating IP Creation

4.2.6 ImagesClient

class ImagesClient(*client: Client*)

actions: *ResourceActionsClient*

Images scoped actions client

Type
ResourceActionsClient

change_protection(*image:* Image | BoundImage, *delete: bool | None = None*) → *BoundAction*

Changes the protection configuration of the image. Can only be used on snapshots.

Parameters

- **image** – *BoundImage* or *Image*
- **delete** – bool If true, prevents the snapshot from being deleted

Returns
BoundAction

delete(*image:* Image | BoundImage) → bool

Deletes an Image. Only images of type snapshot and backup can be deleted.

:param *BoundImage* or *Image* :return: bool

get_actions(*image:* Image | BoundImage, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Image.

Parameters

- **image** – Image to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*image:* Image | BoundImage, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Image.

Parameters

- **image** – Image to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_all(*name: str | None = None*, *label_selector: str | None = None*, *bound_to: list[str] | None = None*, *type: list[str] | None = None*, *architecture: list[str] | None = None*, *sort: list[str] | None = None*, *status: list[str] | None = None*, *include_deprecated: bool | None = None*) → list[*BoundImage*]

Get all images

Parameters

- **name** – str (optional) Can be used to filter images by their name.

- **label_selector** – str (optional) Can be used to filter servers by labels. The response will only contain servers matching the label selector.
- **bound_to** – List[str] (optional) Server Id linked to the image. Only available for images of type backup
- **type** – List[str] (optional) Choices: system snapshot backup
- **architecture** – List[str] (optional) Choices: x86 arm
- **status** – List[str] (optional) Can be used to filter images by their status. The response will only contain images matching the status.
- **sort** – List[str] (optional) Choices: id name created (You can add one of ":asc", ":desc" to modify sort order. (":asc" is default))
- **include_deprecated** – bool (optional) Include deprecated images in the response. Default: False

Returns
List[*BoundImage*]

get_by_id(*id: int*) → *BoundImage*

Get a specific Image

Parameters
id – int

Returns
BoundImage <hcloud.images.client.BoundImage

get_by_name(*name: str*) → *BoundImage* | None

Get image by name

Parameters
name – str Used to get image by name.

Returns
BoundImage

Deprecated since version 1.19: Use *hcloud.images.client.ImagesClient.get_by_name_and_architecture()* instead.

get_by_name_and_architecture(*name: str*, *architecture: str*, ***, *include_deprecated: bool | None = None*) → *BoundImage* | None

Get image by name

Parameters

- **name** – str Used to identify the image.
- **architecture** – str Used to identify the image.
- **include_deprecated** – bool (optional) Include deprecated images. Default: False

Returns
BoundImage

get_list(*name: str | None = None*, *label_selector: str | None = None*, *bound_to: list[str] | None = None*, *type: list[str] | None = None*, *architecture: list[str] | None = None*, *sort: list[str] | None = None*, *page: int | None = None*, *per_page: int | None = None*, *status: list[str] | None = None*, *include_deprecated: bool | None = None*) → ImagesPageResult

Get all images

Parameters

- **name** – str (optional) Can be used to filter images by their name.
- **label_selector** – str (optional) Can be used to filter servers by labels. The response will only contain servers matching the label selector.
- **bound_to** – List[str] (optional) Server Id linked to the image. Only available for images of type backup
- **type** – List[str] (optional) Choices: system snapshot backup
- **architecture** – List[str] (optional) Choices: x86 arm
- **status** – List[str] (optional) Can be used to filter images by their status. The response will only contain images matching the status.
- **sort** – List[str] (optional) Choices: id id:asc id:desc name name:asc name:desc created created:asc created:desc
- **include_deprecated** – bool (optional) Include deprecated images in the response. Default: False
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page

Returns
(List[*BoundImage*], Meta)

update(*image:* Image | BoundImage, *description: str | None = None*, *type: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundImage*

Updates the Image. You may change the description, convert a Backup image to a Snapshot Image or change the image labels.

Parameters

- **image** – *BoundImage* or *Image*
- **description** – str (optional) New description of Image
- **type** – str (optional) Destination image type to convert to Choices: snapshot
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundImage

class BoundImage(*client:* ImagesClient, *data: dict[str, Any]*)

change_protection(*delete: bool | None = None*) → *BoundAction*

Changes the protection configuration of the image. Can only be used on snapshots.

Parameters
delete – bool If true, prevents the snapshot from being deleted

Returns
BoundAction

delete() → bool

Deletes an Image. Only images of type snapshot and backup can be deleted.

Returns
bool

get_actions(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Image.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Image.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

model

alias of *Image*

update(*description: str | None = None*, *type: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundImage*

Updates the Image. You may change the description, convert a Backup image to a Snapshot Image or change the image labels.

Parameters

- **description** – str (optional) New description of Image
- **type** – str (optional) Destination image type to convert to Choices: snapshot
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns

BoundImage

class Image(*id: int | None = None*, *name: str | None = None*, *type: str | None = None*, *created: str | None = None*, *description: str | None = None*, *image_size: int | None = None*, *disk_size: int | None = None*, *deprecated: str | None = None*, *bound_to:* Server | BoundServer *| None = None*, *os_flavor: str | None = None*, *os_version: str | None = None*, *architecture: str | None = None*, *rapid_deploy: bool | None = None*, *created_from:* Server | BoundServer *| None = None*, *protection: ImageProtection | None = None*, *labels: dict[str, str] | None = None*, *status: str | None = None*)

Image Domain

Parameters

- **id** – int ID of the image
- **type** – str Type of the image Choices: *system*, *snapshot*, *backup*, *app*

- **status** – str Whether the image can be used or if it's still being created Choices: *available*, *creating*
- **name** – str, None Unique identifier of the image. This value is only set for system images.
- **description** – str Description of the image
- **image_size** – number, None Size of the image file in our storage in GB. For snapshot images this is the value relevant for calculating costs for the image.
- **disk_size** – number Size of the disk contained in the image in GB.
- **created** – datetime Point in time when the image was created
- **created_from** – *BoundServer*, None Information about the server the image was created from
- **bound_to** – *BoundServer*, None ID of server the image is bound to. Only set for images of type *backup*.
- **os_flavor** – str Flavor of operating system contained in the image Choices: *ubuntu*, *centos*, *debian*, *fedora*, *unknown*
- **os_version** – str, None Operating system version
- **architecture** – str CPU Architecture that the image is compatible with. Choices: *x86*, *arm*
- **rapid_deploy** – bool Indicates that rapid deploy of the image is available
- **protection** – dict Protection configuration for the image
- **deprecated** – datetime, None Point in time when the image is considered to be deprecated (in ISO-8601 format)
- **labels** – Dict User-defined labels (key-value pairs)

class CreateImageResponse(*action:* BoundAction, *image:* BoundImage)

Create Image Response Domain

Parameters

- **image** – *BoundImage* The Image which was created
- **action** – *BoundAction* The Action which shows the progress of the Floating IP Creation

4.2.7 ISOsClient

class IsosClient(*client:* Client)

get_all(*name: str | None = None*, *architecture: list[str] | None = None*, *include_architecture_wildcard: bool | None = None*) → list[*BoundIso*]

Get all ISOs

Parameters

- **name** – str (optional) Can be used to filter ISOs by their name.
- **architecture** – List[str] (optional) Can be used to filter ISOs by their architecture. Choices: x86 arm
- **include_architecture_wildcard** – bool (optional) Custom ISOs do not have an architecture set. You must also set this flag to True if you are filtering by architecture and also want custom ISOs.

Returns
List[*BoundIso*]

get_by_id(*id: int*) → *BoundIso*

Get a specific ISO by its id

Parameters
id – int

Returns
BoundIso

get_by_name(*name: str*) → *BoundIso* | None

Get iso by name

Parameters
name – str Used to get iso by name.

Returns
BoundIso

get_list(*name: str | None = None*, *architecture: list[str] | None = None*, *include_architecture_wildcard: bool | None = None*, *page: int | None = None*, *per_page: int | None = None*) → IsosPageResult

Get a list of ISOs

Parameters

- **name** – str (optional) Can be used to filter ISOs by their name.
- **architecture** – List[str] (optional) Can be used to filter ISOs by their architecture. Choices: x86 arm
- **include_architecture_wildcard** – bool (optional) Custom ISOs do not have an architecture set. You must also set this flag to True if you are filtering by architecture and also want custom ISOs.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page

Returns
(List[*BoundIso*], Meta)

class BoundIso(*client: ResourceClientBase*, *data: dict[str, Any]*, *complete: bool = True*)

model

alias of *Iso*

class Iso(*id: int | None = None*, *name: str | None = None*, *type: str | None = None*, *architecture: str | None = None*, *description: str | None = None*, *deprecated: str | None = None*, *deprecation: dict[str, Any] | None = None*)

Iso Domain

Parameters

- **id** – int ID of the ISO
- **name** – str, None Unique identifier of the ISO. Only set for public ISOs
- **description** – str Description of the ISO
- **type** – str Type of the ISO. Choices: *public*, *private*

- **architecture** – str, None CPU Architecture that the ISO is compatible with. None means that the compatibility is unknown. Choices: *x86*, *arm*
- **deprecated** – datetime, None ISO 8601 timestamp of deprecation, None if ISO is still available. After the deprecation time it will no longer be possible to attach the ISO to servers. This field is deprecated. Use *deprecation* instead.
- **deprecation** – *DeprecationInfo*, None Describes if, when & how the resources was deprecated. If this field is set to None the resource is not deprecated. If it has a value, it is considered deprecated.

property deprecated: datetime | None

ISO 8601 timestamp of deprecation, None if ISO is still available.

4.2.8 LoadBalancerTypesClient

class LoadBalancerTypesClient(*client:* Client)

get_all(*name: str | None = None*) → list[BoundLoadBalancerType]

Get all Load Balancer types

Parameters
name – str (optional) Can be used to filter Load Balancer type by their name.

Returns
List[BoundLoadBalancerType]

get_by_id(*id: int*) → BoundLoadBalancerType

Returns a specific Load Balancer Type.

Parameters
id – int

Returns
BoundLoadBalancerType

get_by_name(*name: str*) → BoundLoadBalancerType | None

Get Load Balancer type by name

Parameters
name – str Used to get Load Balancer type by name.

Returns
BoundLoadBalancerType

get_list(*name: str | None = None*, *page: int | None = None*, *per_page: int | None = None*) → LoadBalancerTypesPageResult

Get a list of Load Balancer types

Parameters

- **name** – str (optional) Can be used to filter Load Balancer type by their name.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page

Returns
(List[BoundLoadBalancerType], Meta)

class LoadBalancerType(*id: int | None = None*, *name: str | None = None*, *description: str | None = None*, *max_connections: int | None = None*, *max_services: int | None = None*, *max_targets: int | None = None*, *max_assigned_certificates: int | None = None*, *prices: list[dict[str, Any]] | None = None*)

LoadBalancerType Domain

Parameters

- **id** – int ID of the Load Balancer type
- **name** – str Name of the Load Balancer type
- **description** – str Description of the Load Balancer type
- **max_connections** – int Max amount of connections the Load Balancer can handle
- **max_services** – int Max amount of services the Load Balancer can handle
- **max_targets** – int Max amount of targets the Load Balancer can handle
- **max_assigned_certificates** – int Max amount of certificates the Load Balancer can serve
- **prices** – List of dict Prices in different locations

4.2.9 LoadBalancerClient

class LoadBalancersClient(*client: Client*)

actions: *ResourceActionsClient*

Load Balancers scoped actions client

Type
ResourceActionsClient

add_service(*load_balancer: LoadBalancer | BoundLoadBalancer*, *service: LoadBalancerService*) → *BoundAction*

Adds a service to a Load Balancer.

Parameters

- **load_balancer** – *BoundLoadBalancer* or *LoadBalancer*
- **service** – *LoadBalancerService* The LoadBalancerService you want to add to the Load Balancer

Returns
BoundAction

add_target(*load_balancer: LoadBalancer | BoundLoadBalancer*, *target: LoadBalancerTarget*) → *BoundAction*

Adds a target to a Load Balancer.

Parameters

- **load_balancer** – *BoundLoadBalancer* or *LoadBalancer*
- **target** – *LoadBalancerTarget* The LoadBalancerTarget you want to add to the Load Balancer

Returns
BoundAction

attach_to_network(*load_balancer:* LoadBalancer | BoundLoadBalancer, *network:* Network | BoundNetwork, *ip: str | None = None*, *ip_range: str | None = None*) → *BoundAction*

Attach a Load Balancer to a Network.

Parameters

- **load_balancer** – :class:` <hcloud.load_balancers.client.BoundLoadBalancer>` or *LoadBalancer*
- **network** – *BoundNetwork* or *Network*
- **ip** – str IP to request to be assigned to this Load Balancer
- **ip_range** – str IP range in CIDR block notation of the subnet to attach to.

Returns
BoundAction

change_algorithm(*load_balancer:* LoadBalancer | BoundLoadBalancer, *algorithm:* LoadBalancerAlgorithm) → *BoundAction*

Changes the algorithm used by the Load Balancer

Parameters

- **load_balancer** – :class:` <hcloud.load_balancers.client.BoundLoadBalancer>` or *LoadBalancer*
- **algorithm** – *LoadBalancerAlgorithm* The LoadBalancerSubnet you want to add to the Load Balancer

Returns
BoundAction

change_dns_ptr(*load_balancer:* LoadBalancer | BoundLoadBalancer, *ip: str*, *dns_ptr: str*) → *BoundAction*

Changes the hostname that will appear when getting the hostname belonging to the public IPs (IPv4 and IPv6) of this Load Balancer.

Parameters

- **ip** – str The IP address for which to set the reverse DNS entry
- **dns_ptr** – str Hostname to set as a reverse DNS PTR entry, will reset to original default value if *None*

Returns
BoundAction

change_protection(*load_balancer:* LoadBalancer | BoundLoadBalancer, *delete: bool | None = None*) → *BoundAction*

Changes the protection configuration of a Load Balancer.

Parameters

- **load_balancer** – :class:` <hcloud.load_balancers.client.BoundLoadBalancer>` or *LoadBalancer*
- **delete** – boolean If True, prevents the Load Balancer from being deleted

Returns
BoundAction

change_type(*load_balancer:* LoadBalancer | BoundLoadBalancer, *load_balancer_type:* LoadBalancerType | *BoundLoadBalancerType*) → *BoundAction*

Changes the type of a Load Balancer.

Parameters

- **load_balancer** – *BoundLoadBalancer* or *LoadBalancer*
- **load_balancer_type** – BoundLoadBalancerType or *LoadBalancerType* Load Balancer type the Load Balancer should migrate to

Returns
BoundAction

create(*name: str*, *load_balancer_type:* LoadBalancerType | *BoundLoadBalancerType*, *algorithm:* LoadBalancerAlgorithm | *None = None*, *services: list[*LoadBalancerService*]* | *None = None*, *targets: list[*LoadBalancerTarget*]* | *None = None*, *labels: dict[str, str]* | *None = None*, *location:* Location | BoundLocation | *None = None*, *network_zone: str* | *None = None*, *public_interface: bool* | *None = None*, *network:* Network | BoundNetwork | *None = None*) → CreateLoadBalancerResponse

Creates a Load Balancer .

Parameters

- **name** – str Name of the Load Balancer
- **load_balancer_type** – LoadBalancerType Type of the Load Balancer
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)
- **location** – Location Location of the Load Balancer
- **network_zone** – str Network Zone of the Load Balancer
- **algorithm** – LoadBalancerAlgorithm (optional) The algorithm the Load Balancer is currently using
- **services** – LoadBalancerService The services the Load Balancer is currently serving
- **targets** – LoadBalancerTarget The targets the Load Balancer is currently serving
- **public_interface** – bool Enable or disable the public interface of the Load Balancer
- **network** – Network Adds the Load Balancer to a Network

Returns
CreateLoadBalancerResponse

delete(*load_balancer:* LoadBalancer | BoundLoadBalancer) → bool

Deletes a Load Balancer.

Parameters
load_balancer – *BoundLoadBalancer* or *LoadBalancer*

Returns
boolean

delete_service(*load_balancer:* LoadBalancer | BoundLoadBalancer, *service:* LoadBalancerService) → *BoundAction*

Deletes a service from a Load Balancer.

Parameters

- **load_balancer** – *BoundLoadBalancer* or *LoadBalancer*

- **service** – *LoadBalancerService* The LoadBalancerService you want to delete from the Load Balancer

Returns
BoundAction

detach_from_network(*load_balancer:* LoadBalancer | BoundLoadBalancer, *network:* Network | BoundNetwork) → *BoundAction*

Detaches a Load Balancer from a Network.

Parameters

- **load_balancer** – :class:` <hcloud.load_balancers.client.BoundLoadBalancer>` or *LoadBalancer*
- **network** – *BoundNetwork* or *Network*

Returns
BoundAction

disable_public_interface(*load_balancer:* LoadBalancer | BoundLoadBalancer) → *BoundAction*

Disables the public interface of a Load Balancer.

Parameters
load_balancer – :class:` <hcloud.load_balancers.client.BoundLoadBalancer>` or *LoadBalancer*

Returns
BoundAction

enable_public_interface(*load_balancer:* LoadBalancer | BoundLoadBalancer) → *BoundAction*

Enables the public interface of a Load Balancer.

Parameters
load_balancer – :class:` <hcloud.load_balancers.client.BoundLoadBalancer>` or *LoadBalancer*

Returns
BoundAction

get_actions(*load_balancer:* LoadBalancer | BoundLoadBalancer, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Load Balancer.

Parameters

- **load_balancer** – Load Balancer to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*load_balancer:* LoadBalancer | BoundLoadBalancer, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Load Balancer.

Parameters

- **load_balancer** – Load Balancer to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_all(*name: str | None = None*, *label_selector: str | None = None*) → list[*BoundLoadBalancer*]

Get all Load Balancers from this account

Parameters

- **name** – str (optional) Can be used to filter Load Balancers by their name.
- **label_selector** – str (optional) Can be used to filter Load Balancers by labels. The response will only contain Load Balancers matching the label selector.

Returns

List[*BoundLoadBalancer*]

get_by_id(*id: int*) → *BoundLoadBalancer*

Get a specific Load Balancer

Parameters

id – int

Returns

BoundLoadBalancer

get_by_name(*name: str*) → *BoundLoadBalancer* | None

Get Load Balancer by name

Parameters

name – str Used to get Load Balancer by name.

Returns

BoundLoadBalancer

get_list(*name: str | None = None*, *label_selector: str | None = None*, *page: int | None = None*, *per_page: int | None = None*) → LoadBalancersPageResult

Get a list of Load Balancers from this account

Parameters

- **name** – str (optional) Can be used to filter Load Balancers by their name.
- **label_selector** – str (optional) Can be used to filter Load Balancers by labels. The response will only contain Load Balancers matching the label selector.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page

Returns

(List[*BoundLoadBalancer*], Meta)

get_metrics(*load_balancer: LoadBalancer | BoundLoadBalancer*, *type: Literal['open_connections', 'connections_per_second', 'requests_per_second', 'bandwidth'] | list[Literal['open_connections', 'connections_per_second', 'requests_per_second', 'bandwidth']]*, *start: datetime | str*, *end: datetime | str*, *step: float | None = None*) → GetMetricsResponse

Get Metrics for a LoadBalancer.

Parameters

- **load_balancer** – The Load Balancer to get the metrics for.
- **type** – Type of metrics to get.
- **start** – Start of period to get Metrics for (in ISO-8601 format).
- **end** – End of period to get Metrics for (in ISO-8601 format).
- **step** – Resolution of results in seconds.

remove_target(*load_balancer:* LoadBalancer | BoundLoadBalancer, *target:* LoadBalancerTarget) → *BoundAction*

Removes a target from a Load Balancer.

Parameters

- **load_balancer** – *BoundLoadBalancer* or *LoadBalancer*
- **target** – *LoadBalancerTarget* The LoadBalancerTarget you want to remove from the Load Balancer

Returns
BoundAction

update(*load_balancer:* LoadBalancer | BoundLoadBalancer, *name: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundLoadBalancer*

Updates a LoadBalancer. You can update a LoadBalancer's name and a LoadBalancer's labels.

Parameters

- **load_balancer** – *BoundLoadBalancer* or *LoadBalancer*
- **name** – str (optional) New name to set
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundLoadBalancer

update_service(*load_balancer:* LoadBalancer | BoundLoadBalancer, *service:* LoadBalancerService) → *BoundAction*

Updates a service of an Load Balancer.

Parameters

- **load_balancer** – *BoundLoadBalancer* or *LoadBalancer*
- **service** – *LoadBalancerService* The LoadBalancerService with updated values within for the Load Balancer

Returns
BoundAction

class BoundLoadBalancer(*client:* LoadBalancersClient, *data: dict[str, Any]*, *complete: bool = True*)

add_service(*service:* LoadBalancerService) → *BoundAction*

Adds a service to a Load Balancer.

Parameters
service – *LoadBalancerService* The LoadBalancerService you want to add to the Load Balancer

Returns
BoundAction

add_target(*target:* LoadBalancerTarget) → *BoundAction*

Adds a target to a Load Balancer.

Parameters
target – *LoadBalancerTarget* The LoadBalancerTarget you want to add to the Load Balancer

Returns
BoundAction

attach_to_network(*network:* Network | BoundNetwork, *ip: str | None = None*, *ip_range: str | None = None*) → *BoundAction*

Attaches a Load Balancer to a Network

Parameters

- **network** – *BoundNetwork* or *Network*
- **ip** – str IP to request to be assigned to this Load Balancer
- **ip_range** – str IP range in CIDR block notation of the subnet to attach to.

Returns
BoundAction

change_algorithm(*algorithm:* LoadBalancerAlgorithm) → *BoundAction*

Changes the algorithm used by the Load Balancer

Parameters
algorithm – *LoadBalancerAlgorithm* The LoadBalancerAlgorithm you want to use

Returns
BoundAction

change_dns_ptr(*ip: str*, *dns_ptr: str*) → *BoundAction*

Changes the hostname that will appear when getting the hostname belonging to the public IPs (IPv4 and IPv6) of this Load Balancer.

Parameters

- **ip** – str The IP address for which to set the reverse DNS entry
- **dns_ptr** – str Hostname to set as a reverse DNS PTR entry, will reset to original default value if *None*

Returns
BoundAction

change_protection(*delete: bool*) → *BoundAction*

Changes the protection configuration of a Load Balancer.

Parameters
delete – boolean If True, prevents the Load Balancer from being deleted

Returns
BoundAction

change_type(*load_balancer_type:* LoadBalancerType | *BoundLoadBalancerType*) → *BoundAction*

Changes the type of a Load Balancer.

Parameters
load_balancer_type – BoundLoadBalancerType or *LoadBalancerType* Load Balancer type the Load Balancer should migrate to

Returns
BoundAction

delete() → bool

Deletes a Load Balancer.

Returns
boolean

delete_service(*service:* LoadBalancerService) → *BoundAction*

Deletes a service from a Load Balancer.

Parameters
service – *LoadBalancerService* The LoadBalancerService you want to delete from the Load Balancer

Returns
BoundAction

detach_from_network(*network:* Network | BoundNetwork) → *BoundAction*

Detaches a Load Balancer from a Network.

Parameters
network – *BoundNetwork* or *Network*

Returns
BoundAction

disable_public_interface() → *BoundAction*

Disables the public interface of a Load Balancer.

Returns
BoundAction

enable_public_interface() → *BoundAction*

Enables the public interface of a Load Balancer.

Returns
BoundAction

get_actions(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Load Balancer.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Load Balancer.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_metrics(*type: Literal['open_connections', 'connections_per_second', 'requests_per_second', 'bandwidth']*, *start: datetime | str*, *end: datetime | str*, *step: float | None = None*) → GetMetricsResponse

Get Metrics for a LoadBalancer.

Parameters

- **type** – Type of metrics to get.
- **start** – Start of period to get Metrics for (in ISO-8601 format).
- **end** – End of period to get Metrics for (in ISO-8601 format).
- **step** – Resolution of results in seconds.

model

alias of *LoadBalancer*

remove_target(*target:* LoadBalancerTarget) → *BoundAction*

Removes a target from a Load Balancer.

Parameters
target – *LoadBalancerTarget* The LoadBalancerTarget you want to remove from the Load Balancer

Returns
BoundAction

update(*name: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundLoadBalancer*

Updates a Load Balancer. You can update a Load Balancers name and a Load Balancers labels.

Parameters

- **name** – str (optional) New name to set
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundLoadBalancer

update_service(*service:* LoadBalancerService) → *BoundAction*

Updates a service of an Load Balancer.

Parameters
service – *LoadBalancerService* The LoadBalancerService you want to update

Returns
BoundAction

class LoadBalancer(*id: int*, *name: str | None = None*, *public_net: PublicNetwork | None = None*, *private_net: list[PrivateNet] | None = None*, *location: BoundLocation | None = None*, *algorithm: LoadBalancerAlgorithm | None = None*, *services: list[LoadBalancerService] | None = None*, *load_balancer_type: BoundLoadBalancerType | None = None*, *protection: LoadBalancerProtection | None = None*, *labels: dict[str, str] | None = None*, *targets: list[LoadBalancerTarget] | None = None*, *created: str | None = None*, *outgoing_traffic: int | None = None*, *ingoing_traffic: int | None = None*, *included_traffic: int | None = None*)

LoadBalancer Domain

Parameters

- **id** – int ID of the Load Balancer
- **name** – str Name of the Load Balancer (must be unique per project)
- **created** – datetime Point in time when the Load Balancer was created
- **protection** – dict Protection configuration for the Load Balancer
- **labels** – dict User-defined labels (key-value pairs)
- **location** – Location Location of the Load Balancer
- **public_net** – PublicNetwork Public network information.
- **private_net** – List[PrivateNet <hcloud.load_balancers.domain.PrivateNet] Private networks information.
- **algorithm** – LoadBalancerAlgorithm The algorithm the Load Balancer is currently using
- **services** – List[LoadBalancerService] The services the LoadBalancer is currently serving
- **targets** – LoadBalancerTarget The targets the LoadBalancer is currently serving
- **load_balancer_type** – LoadBalancerType The type of the Load Balancer
- **outgoing_traffic** – int, None Outbound Traffic for the current billing period in bytes
- **ingoing_traffic** – int, None Inbound Traffic for the current billing period in bytes
- **included_traffic** – int Free Traffic for the current billing period in bytes

private_net_for(*network: BoundNetwork | Network*) → PrivateNet | None

Returns the load balancer's network attachment information in the given Network, and None if no attachment was found.

class LoadBalancerService(*protocol: str | None = None*, *listen_port: int | None = None*, *destination_port: int | None = None*, *proxyprotocol: bool | None = None*, *health_check: LoadBalancerHealthCheck | None = None*, *http: LoadBalancerServiceHttp | None = None*)

LoadBalancerService Domain

Parameters

- **protocol** – str Protocol of the service Choices: tcp, http, https
- **listen_port** – int Required when protocol is tcp, must be unique per Load Balancer.
- **destination_port** – int Required when protocol is tcp
- **proxyprotocol** – bool Enable proxyprotocol
- **health_check** – LoadBalancerHealthCheck Configuration for health checks

- **http** – LoadBalancerServiceHttp Configuration for http/https protocols, required when protocol is http/https

to_payload() → dict[str, Any]

Generates the request payload from this domain object.

class LoadBalancerServiceHttp(*cookie_name: str | None = None*, *cookie_lifetime: str | None = None*, *certificates: list[BoundCertificate] | None = None*, *redirect_http: bool | None = None*, *sticky_sessions: bool | None = None*, *timeout_idle: int | None = None*)

LoadBalancerServiceHttp Domain

Parameters

- **cookie_name** – str Name of the cookie used for Session Stickness
- **cookie_lifetime** – str Lifetime of the cookie used for Session Stickness
- **certificates** – list IDs of the Certificates to use for TLS/SSL termination by the Load Balancer; empty for TLS/SSL passthrough or if protocol is "http"
- **redirect_http** – bool Redirect traffic from http port 80 to port 443
- **sticky_sessions** – bool Use sticky sessions. Only available if protocol is "http" or "https".
- **timeout_idle** – int Idle timeout in seconds for HTTP connections. Must be between 30 and 300 seconds.

class LoadBalancerHealthCheck(*protocol: str | None = None*, *port: int | None = None*, *interval: int | None = None*, *timeout: int | None = None*, *retries: int | None = None*, *http: LoadBalancerHealthCheckHttp | None = None*)

LoadBalancerHealthCheck Domain

Parameters

- **protocol** – str Protocol of the service Choices: tcp, http, https
- **port** – int Port the healthcheck will be performed on
- **interval** – int Interval we trigger health check in
- **timeout** – int Timeout in sec after a try is assumed as timeout
- **retries** – int Retries we perform until we assume a target as unhealthy
- **http** – LoadBalancerHealthCheckHttp HTTP Config

class LoadBalancerHealthCheckHttp(*domain: str | None = None*, *path: str | None = None*, *response: str | None = None*, *status_codes: list[str] | None = None*, *tls: bool | None = None*)

LoadBalancerHealthCheckHttp Domain

Parameters

- **domain** – str Domain name to send in HTTP request. Can be null: In that case we will not send a domain name
- **path** – str HTTP Path send in Request
- **response** – str Optional HTTP response to receive in order to pass the health check
- **status_codes** – list List of HTTP status codes to receive in order to pass the health check
- **tls** – bool Type of health check

class LoadBalancerTarget(*type: str | None = None*, *server: BoundServer | None = None*, *label_selector: LoadBalancerTargetLabelSelector | None = None*, *ip: LoadBalancerTargetIP | None = None*, *use_private_ip: bool | None = None*, *health_status: list[LoadBalancerTargetHealthStatus] | None = None*, *targets: list[LoadBalancerTarget] | None = None*)

LoadBalancerTarget Domain

Parameters

- **type** – str Type of the resource, can be server or label_selector
- **server** – Server Target server
- **label_selector** – LoadBalancerTargetLabelSelector Target label selector
- **ip** – LoadBalancerTargetIP Target IP
- **use_private_ip** – bool use the private IP instead of primary public IP
- **health_status** – list List of health statuses of the services on this target. Only present for target types "server" and "ip".
- **targets** – list List of resolved label selector targets. Only present for target types "label_selector".

to_payload() → dict[str, Any]

Generates the request payload from this domain object.

class LoadBalancerTargetHealthStatus(*listen_port: int | None = None*, *status: str | None = None*)

LoadBalancerTargetHealthStatus Domain

Parameters

- **listen_port** – Load Balancer Target listen port
- **status** – Load Balancer Target status. Choices: healthy, unhealthy, unknown

class LoadBalancerTargetLabelSelector(*selector: str | None = None*)

LoadBalancerTargetLabelSelector Domain

Parameters

selector – str Target label selector

class LoadBalancerTargetIP(*ip: str | None = None*)

LoadBalancerTargetIP Domain

Parameters

ip – str Target IP

class LoadBalancerAlgorithm(*type: str | None = None*)

LoadBalancerAlgorithm Domain

Parameters

type – str Algorithm of the Load Balancer. Choices: round_robin, least_connections

4.2.10 LocationsClient

class LocationsClient(*client: Client*)

get_all(*name: str | None = None*) → list[*BoundLocation*]

Get all locations

Parameters
name – str (optional) Can be used to filter locations by their name.

Returns
List[*BoundLocation*]

get_by_id(*id: int*) → *BoundLocation*

Get a specific location by its ID.

Parameters
id – int

Returns
BoundLocation

get_by_name(*name: str*) → *BoundLocation* | None

Get location by name

Parameters
name – str Used to get location by name.

Returns
BoundLocation

get_list(*name: str | None = None*, *page: int | None = None*, *per_page: int | None = None*) → LocationsPageResult

Get a list of locations

Parameters

- **name** – str (optional) Can be used to filter locations by their name.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page

Returns
(List[*BoundLocation*], Meta)

class BoundLocation(*client: ResourceClientBase*, *data: dict[str, Any]*, *complete: bool = True*)

model

alias of *Location*

class Location(*id: int | None = None*, *name: str | None = None*, *description: str | None = None*, *country: str | None = None*, *city: str | None = None*, *latitude: float | None = None*, *longitude: float | None = None*, *network_zone: str | None = None*)

Location Domain

Parameters

- **id** – int ID of location
- **name** – str Name of location
- **description** – str Description of location
- **country** – str ISO 3166-1 alpha-2 code of the country the location resides in
- **city** – str City the location is closest to

- **latitude** – float Latitude of the city closest to the location
- **longitude** – float Longitude of the city closest to the location
- **network_zone** – str Name of network zone this location resides in

4.2.11 NetworksClient

class NetworksClient(*client:* Client)

actions: *ResourceActionsClient*

Networks scoped actions client

Type
ResourceActionsClient

add_route(*network:* Network | BoundNetwork, *route:* NetworkRoute) → *BoundAction*

Adds a route entry to a network.

Parameters

- **network** – *BoundNetwork* or *Network*
- **route** – *NetworkRoute* The NetworkRoute you want to add to the Network

Returns
BoundAction

add_subnet(*network:* Network | BoundNetwork, *subnet:* NetworkSubnet) → *BoundAction*

Adds a subnet entry to a network.

Parameters

- **network** – *BoundNetwork* or *Network*
- **subnet** – *NetworkSubnet* The NetworkSubnet you want to add to the Network

Returns
BoundAction

change_ip_range(*network:* Network | BoundNetwork, *ip_range: str*) → *BoundAction*

Changes the IP range of a network.

Parameters

- **network** – *BoundNetwork* or *Network*
- **ip_range** – str The new prefix for the whole network.

Returns
BoundAction

change_protection(*network:* Network | BoundNetwork, *delete: bool | None = None*) → *BoundAction*

Changes the protection configuration of a network.

Parameters

- **network** – *BoundNetwork* or *Network*
- **delete** – boolean If True, prevents the network from being deleted

Returns
BoundAction

create(*name: str*, *ip_range: str*, *subnets: list[NetworkSubnet] | None = None*, *routes: list[NetworkRoute] | None = None*, *expose_routes_to_vswitch: bool | None = None*, *labels: dict[str, str] | None = None*) → *BoundNetwork*

Creates a network with range ip_range.

Parameters

- **name** – str Name of the network
- **ip_range** – str IP range of the whole network which must span all included subnets and route destinations
- **subnets** – List[*NetworkSubnet*] Array of subnets allocated
- **routes** – List[*NetworkRoute*] Array of routes set in this network
- **expose_routes_to_vswitch** – Optional[bool] Indicates if the routes from this network should be exposed to the vSwitch connection. The exposing only takes effect if a vSwitch connection is active.
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundNetwork

delete(*network: Network | BoundNetwork*) → bool

Deletes a network.

Parameters
network – *BoundNetwork* or *Network*

Returns
boolean

delete_route(*network: Network | BoundNetwork*, *route: NetworkRoute*) → *BoundAction*

Removes a route entry to a network.

Parameters

- **network** – *BoundNetwork* or *Network*
- **route** – *NetworkRoute* The NetworkRoute you want to remove from the Network

Returns
BoundAction

delete_subnet(*network: Network | BoundNetwork*, *subnet: NetworkSubnet*) → *BoundAction*

Removes a subnet entry from a network

Parameters

- **network** – *BoundNetwork* or *Network*
- **subnet** – *NetworkSubnet* The NetworkSubnet you want to remove from the Network

Returns
BoundAction

get_actions(*network: Network | BoundNetwork*, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Network.

Parameters

- **network** – Network to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*network: Network | BoundNetwork, status: list[Literal['running', 'success', 'error']] | None = None, sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None, page: int | None = None, per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Network.

Parameters

- **network** – Network to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_all(*name: str | None = None, label_selector: str | None = None*) → list[*BoundNetwork*]

Get all networks from this account

Parameters

- **name** – str (optional) Can be used to filter networks by their name.
- **label_selector** – str (optional) Can be used to filter networks by labels. The response will only contain networks matching the label selector.

Returns
List[*BoundNetwork*]

get_by_id(*id: int*) → *BoundNetwork*

Get a specific network

Parameters
id – int

Returns
BoundNetwork

get_by_name(*name: str*) → *BoundNetwork* | None

Get network by name

Parameters
name – str Used to get network by name.

Returns
BoundNetwork

get_list(*name: str | None = None, label_selector: str | None = None, page: int | None = None, per_page: int | None = None*) → NetworksPageResult

Get a list of networks from this account

Parameters

- **name** – str (optional) Can be used to filter networks by their name.

- **label_selector** – str (optional) Can be used to filter networks by labels. The response will only contain networks matching the label selector.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page

Returns
(List[*BoundNetwork*], Meta)

update(*network:* Network | BoundNetwork, *name: str | None = None*, *expose_routes_to_vswitch: bool | None = None*, *labels: dict[str, str] | None = None*) → *BoundNetwork*

Updates a network. You can update a network's name and a network's labels.

Parameters

- **network** – *BoundNetwork* or *Network*
- **name** – str (optional) New name to set
- **expose_routes_to_vswitch** – Optional[bool] Indicates if the routes from this network should be exposed to the vSwitch connection. The exposing only takes effect if a vSwitch connection is active.
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundNetwork

class BoundNetwork(*client:* NetworksClient, *data: dict[str, Any]*, *complete: bool = True*)

add_route(*route:* NetworkRoute) → *BoundAction*

Adds a route entry to a network.

Parameters
route – *NetworkRoute* The NetworkRoute you want to add to the Network

Returns
BoundAction

add_subnet(*subnet:* NetworkSubnet) → *BoundAction*

Adds a subnet entry to a network.

Parameters
subnet – *NetworkSubnet* The NetworkSubnet you want to add to the Network

Returns
BoundAction

change_ip_range(*ip_range: str*) → *BoundAction*

Changes the IP range of a network.

Parameters
ip_range – str The new prefix for the whole network.

Returns
BoundAction

change_protection(*delete: bool | None = None*) → *BoundAction*

Changes the protection configuration of a network.

Parameters
delete – boolean If True, prevents the network from being deleted

Returns
BoundAction

delete() → bool

Deletes a network.

Returns
boolean

delete_route(*route:* NetworkRoute) → *BoundAction*

Removes a route entry to a network.

Parameters
route – *NetworkRoute* The NetworkRoute you want to remove from the Network

Returns
BoundAction

delete_subnet(*subnet:* NetworkSubnet) → *BoundAction*

Removes a subnet entry from a network

Parameters
subnet – *NetworkSubnet* The NetworkSubnet you want to remove from the Network

Returns
BoundAction

get_actions(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Network.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Network.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

model

alias of *Network*

update(*name: str | None = None*, *expose_routes_to_vswitch: bool | None = None*, *labels: dict[str, str] | None = None*) → *BoundNetwork*

Updates a network. You can update a network's name and a networks's labels.

Parameters

- **name** – str (optional) New name to set
- **expose_routes_to_vswitch** – Optional[bool] Indicates if the routes from this network should be exposed to the vSwitch connection. The exposing only takes effect if a vSwitch connection is active.
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns

BoundNetwork

class Network(*id: int*, *name: str | None = None*, *created: str | None = None*, *ip_range: str | None = None*, *subnets: list[NetworkSubnet] | None = None*, *routes: list[NetworkRoute] | None = None*, *expose_routes_to_vswitch: bool | None = None*, *servers: list[BoundServer] | None = None*, *protection: NetworkProtection | None = None*, *labels: dict[str, str] | None = None*)

Network Domain

Parameters

- **id** – int ID of the network
- **name** – str Name of the network
- **ip_range** – str IPv4 prefix of the whole network
- **subnets** – List[*NetworkSubnet*] Subnets allocated in this network
- **routes** – List[*NetworkRoute*] Routes set in this network
- **expose_routes_to_vswitch** – bool Indicates if the routes from this network should be exposed to the vSwitch connection.
- **servers** – List[*BoundServer*] Servers attached to this network
- **protection** – dict Protection configuration for the network
- **labels** – dict User-defined labels (key-value pairs)

class NetworkSubnet(*ip_range: str*, *type: str | None = None*, *network_zone: str | None = None*, *gateway: str | None = None*, *vswitch_id: int | None = None*)

Network Subnet Domain

Parameters

- **type** – str Type of sub network.
- **ip_range** – str Range to allocate IPs from.
- **network_zone** – str Name of network zone.
- **gateway** – str Gateway for the route.
- **vswitch_id** – int ID of the vSwitch.

TYPE_CLOUD = 'cloud'

Used to connect cloud servers and load balancers.

property TYPE_SERVER: str

Used to connect cloud servers and load balancers.

Deprecated since version 2.2.0: Use *NetworkSubnet.TYPE_CLOUD* instead.

TYPE_VSWITCH = 'vswitch'

Used to connect cloud servers and load balancers with dedicated servers.

See https://docs.hetzner.com/networking/networks/connect-dedi-vswitch/

class NetworkRoute(*destination: str*, *gateway: str*)

Network Route Domain

Parameters

- **destination** – str Destination network or host of this route.
- **gateway** – str Gateway for the route.

class CreateNetworkResponse(*network:* BoundNetwork, *action:* BoundAction)

Create Network Response Domain

Parameters

- **network** – *BoundNetwork* The network which was created
- **action** – *BoundAction* The Action which shows the progress of the network Creation

4.2.12 PlacementGroupsClient

class PlacementGroupsClient(*client:* Client)

create(*name: str*, *type: str*, *labels: dict[str, str] | None = None*) → *CreatePlacementGroupResponse*

Creates a new Placement Group.

Parameters

- **name** – str Placement Group Name
- **type** – str Type of the Placement Group
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
CreatePlacementGroupResponse

delete(*placement_group:* PlacementGroup | BoundPlacementGroup) → bool

Deletes a Placement Group.

Parameters
placement_group – *BoundPlacementGroup* or *PlacementGroup*

Returns
boolean

get_all(*label_selector: str | None = None*, *name: str | None = None*, *sort: list[str] | None = None*) → list[*BoundPlacementGroup*]

Get all Placement Groups

Parameters

- **label_selector** – str (optional) Can be used to filter Placement Groups by labels. The response will only contain Placement Groups matching the label selector values.
- **name** – str (optional) Can be used to filter Placement Groups by their name.
- **sort** – List[str] (optional) Choices: id name created (You can add one of ":asc", ":desc" to modify sort order. (":asc" is default))

Returns
List[*BoundPlacementGroup*]

get_by_id(*id: int*) → *BoundPlacementGroup*

Returns a specific Placement Group object

Parameters
id – int

Returns
BoundPlacementGroup

get_by_name(*name: str*) → *BoundPlacementGroup* | None

Get Placement Group by name

Parameters
name – str Used to get Placement Group by name

Returns
class:*BoundPlacementGroup <hcloud.placement_groups.client.BoundPlacementGroup>*

get_list(*label_selector: str | None = None*, *page: int | None = None*, *per_page: int | None = None*, *name: str | None = None*, *sort: list[str] | None = None*, *type: str | None = None*) → PlacementGroupsPageResult

Get a list of Placement Groups

Parameters

- **label_selector** – str (optional) Can be used to filter Placement Groups by labels. The response will only contain Placement Groups matching the label selector values.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page
- **name** – str (optional) Can be used to filter Placement Groups by their name.
- **sort** – List[str] (optional) Choices: id name created (You can add one of ":asc", ":desc" to modify sort order. (":asc" is default))

Returns
(List[*BoundPlacementGroup*], Meta)

update(*placement_group: PlacementGroup | BoundPlacementGroup*, *labels: dict[str, str] | None = None*, *name: str | None = None*) → *BoundPlacementGroup*

Updates the description or labels of a Placement Group.

Parameters

- **placement_group** – *BoundPlacementGroup* or *PlacementGroup*
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)
- **name** – str (optional) New name to set

Returns
BoundPlacementGroup

class BoundPlacementGroup(*client: ResourceClientBase*, *data: dict[str, Any]*, *complete: bool = True*)

delete() → bool

Deletes a Placement Group

Returns
boolean

model

alias of *PlacementGroup*

update(*labels: dict[str, str] | None = None*, *name: str | None = None*) → *BoundPlacementGroup*

Updates the name or labels of a Placement Group

Parameters

- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)
- **name** – str, (optional) New Name to set

Returns
BoundPlacementGroup

class PlacementGroup(*id: int | None = None*, *name: str | None = None*, *labels: dict[str, str] | None = None*, *servers: list[int] | None = None*, *type: str | None = None*, *created: str | None = None*)

Placement Group Domain

Parameters

- **id** – int ID of the Placement Group
- **name** – str Name of the Placement Group
- **labels** – dict User-defined labels (key-value pairs)
- **servers** – List[int] List of server IDs assigned to the Placement Group
- **type** – str Type of the Placement Group
- **created** – datetime Point in time when the image was created

class CreatePlacementGroupResponse(*placement_group:* BoundPlacementGroup, *action:* BoundAction | *None*)

Create Placement Group Response Domain

Parameters

- **placement_group** – *BoundPlacementGroup* The Placement Group which was created
- **action** – *BoundAction* The Action which shows the progress of the Placement Group Creation

4.2.13 PrimaryIPsClient

class PrimaryIPsClient(*client:* Client)

actions: *ResourceActionsClient*

Primary IPs scoped actions client

Type
ResourceActionsClient

assign(*primary_ip:* PrimaryIP | BoundPrimaryIP, *assignee_id: int*, *assignee_type: str = 'server'*) → *BoundAction*

Assigns a Primary IP to a assignee_id.

Parameters

- **primary_ip** – *BoundPrimaryIP* or *PrimaryIP*

- **assignee_id** – int Assignee the Primary IP shall be assigned to
- **assignee_type** – str Assignee the Primary IP shall be assigned to

Returns
BoundAction

change_dns_ptr(*primary_ip:* PrimaryIP | BoundPrimaryIP, *ip: str*, *dns_ptr: str*) → *BoundAction*

Changes the dns ptr that will appear when getting the dns ptr belonging to this Primary IP.

Parameters

- **primary_ip** – *BoundPrimaryIP* or *PrimaryIP*
- **ip** – str The IP address for which to set the reverse DNS entry
- **dns_ptr** – str Hostname to set as a reverse DNS PTR entry, will reset to original default value if *None*

Returns
BoundAction

change_protection(*primary_ip:* PrimaryIP | BoundPrimaryIP, *delete: bool* | *None* = *None*) → *BoundAction*

Changes the protection configuration of the Primary IP.

Parameters

- **primary_ip** – *BoundPrimaryIP* or *PrimaryIP*
- **delete** – boolean If true, prevents the Primary IP from being deleted

Returns
BoundAction

create(*type: str*, *name: str*, *datacenter:* Datacenter | BoundDatacenter | *None* = *None*, *location:* Location | BoundLocation | *None* = *None*, *assignee_type: str* | *None* = *None*, *assignee_id: int* | *None* = *None*, *auto_delete: bool* | *None* = *False*, *labels: dict[str, str]* | *None* = *None*) → CreatePrimaryIPResponse

Creates a new Primary IP assigned to a server.

Parameters

- **type** – str Primary IP type Choices: ipv4, ipv6
- **name** – str
- **datacenter** – Datacenter (optional)
- **location** – Location (optional)
- **assignee_type** – str (optional)
- **assignee_id** – int (optional)
- **auto_delete** – bool (optional)
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
CreatePrimaryIPResponse

delete(*primary_ip:* PrimaryIP | BoundPrimaryIP) → bool

Deletes a Primary IP. If it is currently assigned to an assignee it will automatically get unassigned.

Parameters
primary_ip – *BoundPrimaryIP* or *PrimaryIP*

Returns
boolean

get_actions(*primary_ip: PrimaryIP | BoundPrimaryIP, status: list[Literal['running', 'success', 'error']] | None = None, sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Primary IP.

Parameters

- **primary_ip** – Primary IP to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*primary_ip: PrimaryIP | BoundPrimaryIP, status: list[Literal['running', 'success', 'error']] | None = None, sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None, page: int | None = None, per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Primary IP.

Parameters

- **primary_ip** – Primary IP to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_all(*label_selector: str | None = None, name: str | None = None*) → list[*BoundPrimaryIP*]

Get all primary ips from this account

Parameters

- **label_selector** – str (optional) Can be used to filter Primary IPs by labels. The response will only contain Primary IPs matching the label selector.able values.
- **name** – str (optional) Can be used to filter networks by their name.

Returns
List[*BoundPrimaryIP*]

get_by_id(*id: int*) → *BoundPrimaryIP*

Returns a specific Primary IP object.

Parameters
id – int

Returns
BoundPrimaryIP

get_by_name(*name: str*) → *BoundPrimaryIP* | None

Get Primary IP by name

Parameters
name – str Used to get Primary IP by name.

Returns
BoundPrimaryIP

get_list(*label_selector: str | None = None*, *page: int | None = None*, *per_page: int | None = None*, *name: str | None = None*, *ip: str | None = None*) → PrimaryIPsPageResult

Get a list of primary ips from this account

Parameters

- **label_selector** – str (optional) Can be used to filter Primary IPs by labels. The response will only contain Primary IPs matching the label selectorable values.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page
- **name** – str (optional) Can be used to filter networks by their name.
- **ip** – str (optional) Can be used to filter resources by their ip. The response will only contain the resources matching the specified ip.

Returns
(List[*BoundPrimaryIP*], Meta)

unassign(*primary_ip: PrimaryIP | BoundPrimaryIP*) → *BoundAction*

Unassigns a Primary IP, resulting in it being unreachable. You may assign it to a server again at a later time.

Parameters
primary_ip – *BoundPrimaryIP* or *PrimaryIP*

Returns
BoundAction

update(*primary_ip: PrimaryIP | BoundPrimaryIP*, *auto_delete: bool | None = None*, *labels: dict[str, str] | None = None*, *name: str | None = None*) → *BoundPrimaryIP*

Updates the name, auto_delete or labels of a Primary IP.

Parameters

- **primary_ip** – *BoundPrimaryIP* or *PrimaryIP*
- **auto_delete** – bool (optional) Delete this Primary IP when the resource it is assigned to is deleted
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)
- **name** – str (optional) New name to set

Returns
BoundPrimaryIP

class BoundPrimaryIP(*client: PrimaryIPsClient*, *data: dict[str, Any]*, *complete: bool = True*)

assign(*assignee_id: int*, *assignee_type: str*) → *BoundAction*

Assigns a Primary IP to a assignee.

Parameters

- **assignee_id** – int` Id of an assignee the Primary IP shall be assigned to
- **assignee_type** – string` Assignee type (e.g server) the Primary IP shall be assigned to

Returns
BoundAction

change_dns_ptr(*ip: str*, *dns_ptr: str*) → *BoundAction*

Changes the hostname that will appear when getting the hostname belonging to this Primary IP.

Parameters

- **ip** – str The IP address for which to set the reverse DNS entry
- **dns_ptr** – str Hostname to set as a reverse DNS PTR entry, will reset to original default value if *None*

Returns
BoundAction

change_protection(*delete: bool | None = None*) → *BoundAction*

Changes the protection configuration of the Primary IP.

Parameters
delete – boolean If true, prevents the Primary IP from being deleted

Returns
BoundAction

delete() → bool

Deletes a Primary IP. If it is currently assigned to a server it will automatically get unassigned.

Returns
boolean

get_actions(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Primary IP.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Primary IP.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

model

alias of *PrimaryIP*

unassign() → *BoundAction*

Unassigns a Primary IP, resulting in it being unreachable. You may assign it to a server again at a later time.

Returns
BoundAction

update(*auto_delete: bool | None = None*, *labels: dict[str, str] | None = None*, *name: str | None = None*) → *BoundPrimaryIP*

Updates the description or labels of a Primary IP.

Parameters

- **auto_delete** – bool (optional) Auto delete IP when assignee gets deleted
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)
- **name** – str (optional) New Name to set

Returns
BoundPrimaryIP

class PrimaryIP(*id: int | None = None*, *type: str | None = None*, *ip: str | None = None*, *dns_ptr: list[DNSPtr] | None = None*, *datacenter:* BoundDatacenter *| None = None*, *location:* BoundLocation *| None = None*, *blocked: bool | None = None*, *protection: PrimaryIPProtection | None = None*, *labels: dict[str, str] | None = None*, *created: str | None = None*, *name: str | None = None*, *assignee_id: int | None = None*, *assignee_type: str | None = None*, *auto_delete: bool | None = None*)

Primary IP Domain

Parameters

- **id** – int ID of the Primary IP
- **ip** – str IP address of the Primary IP
- **type** – str Type of Primary IP. Choices: *ipv4*, *ipv6*
- **dns_ptr** – List[Dict] Array of reverse DNS entries
- **datacenter** – *Datacenter* Datacenter the Primary IP was created in.

  This property is deprecated and will be removed after 1 July 2026. Please use the `location` property instead.

  See https://docs.hetzner.cloud/changelog#2025-12-16-phasing-out-datacenters.
- **location** – *Location* Location the Primary IP was created in.
- **blocked** – boolean Whether the IP is blocked
- **protection** – dict Protection configuration for the Primary IP
- **labels** – dict User-defined labels (key-value pairs)
- **created** – datetime Point in time when the Primary IP was created
- **name** – str Name of the Primary IP
- **assignee_id** – int Assignee ID the Primary IP is assigned to
- **assignee_type** – str Assignee Type of entity the Primary IP is assigned to
- **auto_delete** – bool Delete the Primary IP when the Assignee it is assigned to is deleted.

4.2.14 ServerTypesClient

class ServerTypesClient(*client: Client*)

get_all(*name: str | None = None*) → list[*BoundServerType*]

Get all Server types

Parameters
name – str (optional) Can be used to filter server type by their name.

Returns
List[*BoundServerType*]

get_by_id(*id: int*) → *BoundServerType*

Returns a specific Server Type.

Parameters
id – int

Returns
BoundServerType

get_by_name(*name: str*) → *BoundServerType* | None

Get Server type by name

Parameters
name – str Used to get Server type by name.

Returns
BoundServerType

get_list(*name: str | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ServerTypesPageResult

Get a list of Server types

Parameters

- **name** – str (optional) Can be used to filter server type by their name.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page

Returns
(List[*BoundServerType*], Meta)

class BoundServerType(*client: ServerTypesClient*, *data: dict[str, Any]*, *complete: bool = True*)

model

alias of *ServerType*

class ServerType(*id: int | None = None*, *name: str | None = None*, *description: str | None = None*, *category: str | None = None*, *cores: int | None = None*, *memory: int | None = None*, *disk: int | None = None*, *prices: list[dict[str, Any]] | None = None*, *storage_type: str | None = None*, *cpu_type: str | None = None*, *architecture: str | None = None*, *deprecated: bool | None = None*, *deprecation: dict[str, Any] | None = None*, *included_traffic: int | None = None*, *locations: list[ServerTypeLocation] | None = None*)

ServerType Domain

Parameters

- **id** – int ID of the server type

- **name** – str Unique identifier of the server type
- **description** – str Description of the server type
- **category** – str Category of the Server Type.
- **cores** – int Number of cpu cores a server of this type will have
- **memory** – int Memory a server of this type will have in GB
- **disk** – int Disk size a server of this type will have in GB
- **prices** – List of dict Prices in different locations
- **storage_type** – str Type of server boot drive. Local has higher speed. Network has better availability. Choices: *local*, *network*
- **cpu_type** – string Type of cpu. Choices: *shared*, *dedicated*
- **architecture** – string Architecture of cpu. Choices: *x86*, *arm*
- **deprecated** – bool True if server type is deprecated. This field is deprecated. Use *deprecation* instead.
- **deprecation** – *DeprecationInfo*, None Describes if, when & how the resources was deprecated. If this field is set to None the resource is not deprecated. If it has a value, it is considered deprecated.
- **included_traffic** – int Free traffic per month in bytes
- **locations** – Supported Location of the Server Type.

property deprecated: bool | None

Deprecated since version 2.6.0: The 'deprecated' property is deprecated and will gradually be phased starting 24 September 2025. Please refer to the '.locations[].deprecation' property instead.

See https://docs.hetzner.cloud/changelog#2025-09-24-per-location-server-types.

property deprecation: *DeprecationInfo* | None

Deprecated since version 2.6.0: The 'deprecation' property is deprecated and will gradually be phased starting 24 September 2025. Please refer to the '.locations[].deprecation' property instead.

See https://docs.hetzner.cloud/changelog#2025-09-24-per-location-server-types.

property included_traffic: int | None

Deprecated since version 2.1.0: The 'included_traffic' property is deprecated and will be set to 'None' on 5 August 2024. Please refer to the 'prices' property instead.

See https://docs.hetzner.cloud/changelog#2024-07-25-cloud-api-returns-traffic-information-in-different-format.

4.2.15 ServersClient

class ServersClient(*client:* Client)

actions: *ResourceActionsClient*

Servers scoped actions client

Type
ResourceActionsClient

add_to_placement_group(*server:* Server | BoundServer, *placement_group:* PlacementGroup | BoundPlacementGroup) → *BoundAction*

Adds a server to a placement group.

Parameters

- **server** – *BoundServer* or *Server*
- **placement_group** – *BoundPlacementGroup* or *Network*

Returns
BoundAction

attach_iso(*server:* Server | BoundServer, *iso:* Iso | BoundIso) → *BoundAction*

Attaches an ISO to a server.

Parameters

- **server** – *BoundServer* or *Server*
- **iso** – *BoundIso* or *Server*

Returns
BoundAction

attach_to_network(*server:* Server | BoundServer, *network:* Network | BoundNetwork, *ip: str | None = None*, *alias_ips: list[str] | None = None*, *ip_range: str | None = None*) → *BoundAction*

Attaches a server to a network

Parameters

- **server** – *BoundServer* or *Server*
- **network** – *BoundNetwork* or *Network*
- **ip** – str IP to request to be assigned to this server
- **alias_ips** – List[str] New alias IPs to set for this server.
- **ip_range** – str IP range in CIDR block notation of the subnet to attach to.

Returns
BoundAction

change_alias_ips(*server:* Server | BoundServer, *network:* Network | BoundNetwork, *alias_ips: list[str]*) → *BoundAction*

Changes the alias IPs of an already attached network.

Parameters

- **server** – *BoundServer* or *Server*
- **network** – *BoundNetwork* or *Network*
- **alias_ips** – List[str] New alias IPs to set for this server.

Returns
BoundAction

change_dns_ptr(*server:* Server | BoundServer, *ip: str*, *dns_ptr: str | None*) → *BoundAction*

Changes the hostname that will appear when getting the hostname belonging to the primary IPs (ipv4 and ipv6) of this server.

Parameters

- **server** – *BoundServer* or *Server*
- **ip** – str The IP address for which to set the reverse DNS entry

- **dns_ptr** – Hostname to set as a reverse DNS PTR entry, will reset to original default value if *None*

Returns
BoundAction

change_protection(*server:* Server | BoundServer, *delete: bool | None = None*, *rebuild: bool | None = None*) → *BoundAction*

Changes the protection configuration of the server.

Parameters

- **server** – *BoundServer* or *Server*
- **delete** – boolean If true, prevents the server from being deleted (currently delete and rebuild attribute needs to have the same value)
- **rebuild** – boolean If true, prevents the server from being rebuilt (currently delete and rebuild attribute needs to have the same value)

Returns
BoundAction

change_type(*server:* Server | BoundServer, *server_type:* ServerType | BoundServerType, *upgrade_disk: bool*) → *BoundAction*

Changes the type (Cores, RAM and disk sizes) of a server.

Parameters

- **server** – *BoundServer* or *Server*
- **server_type** – *BoundServerType* or *ServerType* Server type the server should migrate to
- **upgrade_disk** – boolean If false, do not upgrade the disk. This allows downgrading the server type later.

Returns
BoundAction

create(*name: str*, *server_type:* ServerType | BoundServerType, *image:* Image, *ssh_keys: list[*SSHKey | BoundSSHKey*] | None = None*, *volumes: list[*Volume | BoundVolume*] | None = None*, *firewalls: list[*Firewall | BoundFirewall*] | None = None*, *networks: list[*Network | BoundNetwork*] | None = None*, *user_data: str | None = None*, *labels: dict[str, str] | None = None*, *location:* Location | BoundLocation | *None = None*, *datacenter:* Datacenter | BoundDatacenter | *None = None*, *start_after_create: bool | None = True*, *automount: bool | None = None*, *placement_group:* PlacementGroup | BoundPlacementGroup | *None = None*, *public_net:* ServerCreatePublicNetwork | *None = None*) → *CreateServerResponse*

Creates a new server. Returns preliminary information about the server as well as an action that covers progress of creation.

Parameters

- **name** – str Name of the server to create (must be unique per project and a valid hostname as per RFC 1123)
- **server_type** – *BoundServerType* or *ServerType* Server type this server should be created with
- **image** – *BoundImage* or *Image* Image the server is created from
- **ssh_keys** – List[*BoundSSHKey* or *SSHKey*] (optional) SSH keys which should be injected into the server at creation time

- **volumes** – List[*BoundVolume* or *Volume*] (optional) Volumes which should be attached to the server at the creation time. Volumes must be in the same location.
- **networks** – List[*BoundNetwork* or *Network*] (optional) Networks which should be attached to the server at the creation time.
- **user_data** – str (optional) Cloud-Init user data to use during server creation. This field is limited to 32KiB.
- **labels** – Dict[str,str] (optional) User-defined labels (key-value pairs)
- **location** – *BoundLocation* or *Location*
- **datacenter** – *BoundDatacenter* or *Datacenter*
- **start_after_create** – boolean (optional) Start Server right after creation. Defaults to True.
- **automount** – boolean (optional) Auto mount volumes after attach.
- **placement_group** – *BoundPlacementGroup* or *Location* Placement Group where server should be added during creation
- **public_net** – *ServerCreatePublicNetwork* Options to configure the public network of a server on creation

Returns
CreateServerResponse

create_image(*server:* Server | BoundServer, *description: str | None = None*, *type: str | None = None*, *labels: dict[str, str] | None = None*) → *CreateImageResponse*

Creates an image (snapshot) from a server by copying the contents of its disks.

Parameters

- **server** – *BoundServer* or *Server*
- **description** – str (optional) Description of the image. If you do not set this we auto-generate one for you.
- **type** – str (optional) Type of image to create (default: snapshot) Choices: snapshot, backup
- **labels** – Dict[str, str] User-defined labels (key-value pairs)

Returns
CreateImageResponse

delete(*server:* Server | BoundServer) → *BoundAction*

Deletes a server. This immediately removes the server from your account, and it is no longer accessible.

Parameters
server – *BoundServer* or *Server*

Returns
BoundAction

detach_from_network(*server:* Server | BoundServer, *network:* Network | BoundNetwork) → *BoundAction*

Detaches a server from a network.

Parameters

- **server** – *BoundServer* or *Server*
- **network** – *BoundNetwork* or *Network*

Returns
BoundAction

detach_iso(*server:* Server | BoundServer) → *BoundAction*

Detaches an ISO from a server.

Parameters
server – *BoundServer* or *Server*

Returns
BoundAction

disable_backup(*server:* Server | BoundServer) → *BoundAction*

Disables the automatic backup option and deletes all existing Backups for a Server.

Parameters
server – *BoundServer* or *Server*

Returns
BoundAction

disable_rescue(*server:* Server | BoundServer) → *BoundAction*

Disables the Hetzner Rescue System for a server.

Parameters
server – *BoundServer* or *Server*

Returns
BoundAction

enable_backup(*server:* Server | BoundServer) → *BoundAction*

Enables and configures the automatic daily backup option for the server. Enabling automatic backups will increase the price of the server by 20%.

Parameters
server – *BoundServer* or *Server*

Returns
BoundAction

enable_rescue(*server:* Server | BoundServer, *type: str | None = None*, *ssh_keys: list[str] | None = None*) → *EnableRescueResponse*

Enable the Hetzner Rescue System for this server.

Parameters

- **server** – *BoundServer* or *Server*
- **type** – str Type of rescue system to boot (default: linux64) Choices: linux64, linux32, freebsd64
- **ssh_keys** – List[str] Array of SSH key IDs which should be injected into the rescue system. Only available for types: linux64 and linux32.

Returns
EnableRescueResponse

get_actions(*server:* Server | BoundServer, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Server.

Parameters

- **server** – Server to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*server: Server | BoundServer, status: list[Literal['running', 'success', 'error']] | None = None, sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None, page: int | None = None, per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Server.

Parameters

- **server** – Server to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_all(*name: str | None = None, label_selector: str | None = None, status: list[str] | None = None*) → list[*BoundServer*]

Get all servers from this account

Parameters

- **name** – str (optional) Can be used to filter servers by their name.
- **label_selector** – str (optional) Can be used to filter servers by labels. The response will only contain servers matching the label selector.
- **status** – List[str] (optional) Can be used to filter servers by their status. The response will only contain servers matching the status.

Returns
List[*BoundServer*]

get_by_id(*id: int*) → *BoundServer*

Get a specific server

Parameters
id – int

Returns
BoundServer

get_by_name(*name: str*) → *BoundServer* | None

Get server by name

Parameters
name – str Used to get server by name.

Returns
BoundServer

get_list(*name: str | None = None*, *label_selector: str | None = None*, *page: int | None = None*, *per_page: int | None = None*, *status: list[str] | None = None*) → ServersPageResult

Get a list of servers from this account

Parameters

- **name** – str (optional) Can be used to filter servers by their name.
- **label_selector** – str (optional) Can be used to filter servers by labels. The response will only contain servers matching the label selector.
- **status** – List[str] (optional) Can be used to filter servers by their status. The response will only contain servers matching the status.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page

Returns
(List[*BoundServer*], Meta)

get_metrics(*server: Server | BoundServer*, *type: Literal['cpu', 'disk', 'network'] | list[Literal['cpu', 'disk', 'network']]*, *start: datetime | str*, *end: datetime | str*, *step: float | None = None*) → GetMetricsResponse

Get Metrics for a Server.

Parameters

- **server** – The Server to get the metrics for.
- **type** – Type of metrics to get.
- **start** – Start of period to get Metrics for (in ISO-8601 format).
- **end** – End of period to get Metrics for (in ISO-8601 format).
- **step** – Resolution of results in seconds.

power_off(*server: Server | BoundServer*) → *BoundAction*

Cuts power to the server. This forcefully stops it without giving the server operating system time to gracefully stop

Parameters
server – *BoundServer* or *Server*

Returns
BoundAction

power_on(*server: Server | BoundServer*) → *BoundAction*

Starts a server by turning its power on.

Parameters
server – *BoundServer* or *Server*

Returns
BoundAction

reboot(*server: Server | BoundServer*) → *BoundAction*

Reboots a server gracefully by sending an ACPI request.

Parameters
server – *BoundServer* or *Server*

Returns
BoundAction

rebuild(*server:* Server | BoundServer, *image:* Image | BoundImage, *user_data: str | None = None*, ***kwargs: Any*) → RebuildResponse

Rebuilds a server overwriting its disk with the content of an image, thereby destroying all data on the target server.

Parameters

- **server** – Server to rebuild
- **image** – Image to use for the rebuilt server
- **user_data** – Cloud-Init user data to use during Server rebuild (optional)

remove_from_placement_group(*server:* Server | BoundServer) → *BoundAction*

Removes a server from a placement group.

Parameters
server – *BoundServer* or *Server*

Returns
BoundAction

request_console(*server:* Server | BoundServer) → *RequestConsoleResponse*

Requests credentials for remote access via vnc over websocket to keyboard, monitor, and mouse for a server.

Parameters
server – *BoundServer* or *Server*

Returns
RequestConsoleResponse

reset(*server:* Server | BoundServer) → *BoundAction*

Cuts power to a server and starts it again.

Parameters
server – *BoundServer* or *Server*

Returns
BoundAction

reset_password(*server:* Server | BoundServer) → *ResetPasswordResponse*

Resets the root password. Only works for Linux systems that are running the qemu guest agent.

Parameters
server – *BoundServer* or *Server*

Returns
ResetPasswordResponse

shutdown(*server:* Server | BoundServer) → *BoundAction*

Shuts down a server gracefully by sending an ACPI shutdown request.

Parameters
server – *BoundServer* or *Server*

Returns
BoundAction

update(*server: Server | BoundServer, name: str | None = None, labels: dict[str, str] | None = None*) → *BoundServer*

Updates a server. You can update a server's name and a server's labels.

Parameters

- **server** – *BoundServer* or *Server*
- **name** – str (optional) New name to set
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundServer

class BoundServer(*client: ServersClient, data: dict[str, Any], complete: bool = True*)

add_to_placement_group(*placement_group: PlacementGroup | BoundPlacementGroup*) → *BoundAction*

Adds a server to a placement group.

Parameters
placement_group – *BoundPlacementGroup* or *Network*

Returns
BoundAction

attach_iso(*iso: Iso | BoundIso*) → *BoundAction*

Attaches an ISO to a server.

Parameters
iso – *BoundIso* or *Server*

Returns
BoundAction

attach_to_network(*network: Network | BoundNetwork, ip: str | None = None, alias_ips: list[str] | None = None, ip_range: str | None = None*) → *BoundAction*

Attaches a server to a network

Parameters

- **network** – *BoundNetwork* or *Network*
- **ip** – str IP to request to be assigned to this server
- **alias_ips** – List[str] New alias IPs to set for this server.
- **ip_range** – str IP range in CIDR block notation of the subnet to attach to.

Returns
BoundAction

change_alias_ips(*network: Network | BoundNetwork, alias_ips: list[str]*) → *BoundAction*

Changes the alias IPs of an already attached network.

Parameters

- **network** – *BoundNetwork* or *Network*
- **alias_ips** – List[str] New alias IPs to set for this server.

Returns
BoundAction

change_dns_ptr(*ip: str*, *dns_ptr: str | None*) → *BoundAction*

Changes the hostname that will appear when getting the hostname belonging to the primary IPs (ipv4 and ipv6) of this server.

Parameters

- **ip** – str The IP address for which to set the reverse DNS entry
- **dns_ptr** – Hostname to set as a reverse DNS PTR entry, will reset to original default value if *None*

Returns

BoundAction

change_protection(*delete: bool | None = None*, *rebuild: bool | None = None*) → *BoundAction*

Changes the protection configuration of the server.

Parameters

- **server** – *BoundServer* or *Server*
- **delete** – boolean If true, prevents the server from being deleted (currently delete and rebuild attribute needs to have the same value)
- **rebuild** – boolean If true, prevents the server from being rebuilt (currently delete and rebuild attribute needs to have the same value)

Returns

BoundAction

change_type(*server_type:* ServerType | BoundServerType, *upgrade_disk: bool*) → *BoundAction*

Changes the type (Cores, RAM and disk sizes) of a server.

Parameters

- **server_type** – *BoundServerType* or *ServerType* Server type the server should migrate to
- **upgrade_disk** – boolean If false, do not upgrade the disk. This allows downgrading the server type later.

Returns

BoundAction

create_image(*description: str | None = None*, *type: str | None = None*, *labels: dict[str, str] | None = None*) → *CreateImageResponse*

Creates an image (snapshot) from a server by copying the contents of its disks.

Parameters

- **description** – str (optional) Description of the image. If you do not set this we auto-generate one for you.
- **type** – str (optional) Type of image to create (default: snapshot) Choices: snapshot, backup
- **labels** – Dict[str, str] User-defined labels (key-value pairs)

Returns

CreateImageResponse

delete() → *BoundAction*

Deletes a server. This immediately removes the server from your account, and it is no longer accessible.

Returns
BoundAction

detach_from_network(*network:* Network | BoundNetwork) → *BoundAction*

Detaches a server from a network.

Parameters
network – *BoundNetwork* or *Network*

Returns
BoundAction

detach_iso() → *BoundAction*

Detaches an ISO from a server.

Returns
BoundAction

disable_backup() → *BoundAction*

Disables the automatic backup option and deletes all existing Backups for a Server.

Returns
BoundAction

disable_rescue() → *BoundAction*

Disables the Hetzner Rescue System for a server.

Returns
BoundAction

enable_backup() → *BoundAction*

Enables and configures the automatic daily backup option for the server. Enabling automatic backups will increase the price of the server by 20%.

Returns
BoundAction

enable_rescue(*type: str | None = None*, *ssh_keys: list[str] | None = None*) → *EnableRescueResponse*

Enable the Hetzner Rescue System for this server.

Parameters

- **type** – str Type of rescue system to boot (default: linux64) Choices: linux64, linux32, freebsd64
- **ssh_keys** – List[str] Array of SSH key IDs which should be injected into the rescue system. Only available for types: linux64 and linux32.

Returns
EnableRescueResponse

get_actions(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Server.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Server.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_metrics(*type: Literal['cpu', 'disk', 'network'] | list[Literal['cpu', 'disk', 'network']]*, *start: datetime | str*, *end: datetime | str*, *step: float | None = None*) → GetMetricsResponse

Get Metrics for a Server.

Parameters

- **server** – The Server to get the metrics for.
- **type** – Type of metrics to get.
- **start** – Start of period to get Metrics for (in ISO-8601 format).
- **end** – End of period to get Metrics for (in ISO-8601 format).
- **step** – Resolution of results in seconds.

model

alias of *Server*

power_off() → *BoundAction*

Cuts power to the server. This forcefully stops it without giving the server operating system time to gracefully stop

Returns
BoundAction

power_on() → *BoundAction*

Starts a server by turning its power on.

Returns
BoundAction

reboot() → *BoundAction*

Reboots a server gracefully by sending an ACPI request.

Returns
BoundAction

rebuild(*image:* Image | BoundImage, *user_data: str | None = None*, ***kwargs: Any*) → RebuildResponse

Rebuilds a server overwriting its disk with the content of an image, thereby destroying all data on the target server.

Parameters

- **image** – Image to use for the rebuilt server
- **user_data** – Cloud-Init user data to use during Server rebuild (optional)

remove_from_placement_group() → *BoundAction*

Removes a server from a placement group.

Returns
BoundAction

request_console() → *RequestConsoleResponse*

Requests credentials for remote access via vnc over websocket to keyboard, monitor, and mouse for a server.

Returns
RequestConsoleResponse

reset() → *BoundAction*

Cuts power to a server and starts it again.

Returns
BoundAction

reset_password() → *ResetPasswordResponse*

Resets the root password. Only works for Linux systems that are running the qemu guest agent.

Returns
ResetPasswordResponse

shutdown() → *BoundAction*

Shuts down a server gracefully by sending an ACPI shutdown request.

Returns
BoundAction

update(*name: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundServer*

Updates a server. You can update a server's name and a server's labels.

Parameters

- **name** – str (optional) New name to set
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundServer

class Server(*id: int*, *name: str | None = None*, *status: str | None = None*, *created: str | None = None*, *public_net: PublicNetwork | None = None*, *server_type: BoundServerType | None = None*, *datacenter: BoundDatacenter | None = None*, *location: BoundLocation | None = None*, *image: BoundImage | None = None*, *iso: BoundIso | None = None*, *rescue_enabled: bool | None = None*, *locked: bool | None = None*, *backup_window: str | None = None*, *outgoing_traffic: int | None = None*, *ingoing_traffic: int | None = None*, *included_traffic: int | None = None*, *protection: ServerProtection | None = None*, *labels: dict[str, str] | None = None*, *volumes: list[BoundVolume] | None = None*, *private_net: list[PrivateNet] | None = None*, *primary_disk_size: int | None = None*, *placement_group: BoundPlacementGroup | None = None*)

Server Domain

Parameters

- **id** – int ID of the server
- **name** – str Name of the server (must be unique per project and a valid hostname as per RFC 1123)
- **status** – str Status of the server Choices: *running*, *initializing*, *starting*, *stopping*, *off*, *deleting*, *migrating*, *rebuilding*, *unknown*

- **created** – datetime Point in time when the server was created
- **public_net** – *PublicNetwork* Public network information.
- **server_type** – *BoundServerType*
- **datacenter** – *BoundDatacenter*

  This property is deprecated and will be removed after 1 July 2026. Please use the `location` property instead.

  See https://docs.hetzner.cloud/changelog#2025-12-16-phasing-out-datacenters.
- **location** – *BoundLocation*
- **image** – *BoundImage*, None
- **iso** – *BoundIso*, None
- **rescue_enabled** – bool True if rescue mode is enabled: Server will then boot into rescue system on next reboot.
- **locked** – bool True if server has been locked and is not available to user.
- **backup_window** – str, None Time window (UTC) in which the backup will run, or None if the backups are not enabled
- **outgoing_traffic** – int, None Outbound Traffic for the current billing period in bytes
- **ingoing_traffic** – int, None Inbound Traffic for the current billing period in bytes
- **included_traffic** – int Free Traffic for the current billing period in bytes
- **primary_disk_size** – int Size of the primary Disk
- **protection** – dict Protection configuration for the server
- **labels** – dict User-defined labels (key-value pairs)
- **volumes** – List[*BoundVolume*] Volumes assigned to this server.
- **private_net** – List[`PrivateNet`] Private networks information.

STATUS_DELETING = 'deleting'

Server Status deleting

STATUS_INIT = 'initializing'

Server Status initializing

STATUS_MIGRATING = 'migrating'

Server Status migrating

STATUS_OFF = 'off'

Server Status off

STATUS_REBUILDING = 'rebuilding'

Server Status rebuilding

STATUS_RUNNING = 'running'

Server Status running

STATUS_STARTING = 'starting'

Server Status starting

STATUS_STOPPING = 'stopping'

Server Status stopping

STATUS_UNKNOWN = 'unknown'

Server Status unknown

private_net_for(*network:* BoundNetwork | Network) → PrivateNet | None

Returns the server's network attachment information in the given Network, and None if no attachment was found.

class PublicNetwork(*ipv4:* IPv4Address | *None*, *ipv6:* IPv6Network | *None*, *floating_ips: list[*BoundFloatingIP*]*, *primary_ipv4:* BoundPrimaryIP | *None*, *primary_ipv6:* BoundPrimaryIP | *None*, *firewalls: list[PublicNetworkFirewall]* | *None = None*)

Public Network Domain

Parameters

- **ipv4** – *IPv4Address*
- **ipv6** – *IPv6Network*
- **floating_ips** – List[*BoundFloatingIP*]
- **primary_ipv4** – BoundPrimaryIP
- **primary_ipv6** – BoundPrimaryIP
- **firewalls** – List[PublicNetworkFirewall]

class IPv4Address(*ip: str*, *blocked: bool*, *dns_ptr: str*)

IPv4 Address Domain

Parameters

- **ip** – str The IPv4 Address
- **blocked** – bool Determine if the IP is blocked
- **dns_ptr** – str DNS PTR for the ip

class IPv6Network(*ip: str*, *blocked: bool*, *dns_ptr: list[DNSPtr]*)

IPv6 Network Domain

Parameters

- **ip** – str The IPv6 Network as CIDR Notation
- **blocked** – bool Determine if the Network is blocked
- **dns_ptr** – dict DNS PTR Records for the Network as Dict
- **network** – str The network without the network mask
- **network_mask** – str The network mask

class CreateServerResponse(*server:* BoundServer, *action:* BoundAction, *next_actions: list[*BoundAction*]*, *root_password: str* | *None*)

Create Server Response Domain

Parameters

- **server** – *BoundServer* The created server
- **action** – *BoundAction* Shows the progress of the server creation

- **next_actions** – List[*BoundAction*] Additional actions like a *start_server* action after the server creation
- **root_password** – str, None The root password of the server if no SSH-Key was given on server creation

class ServerCreatePublicNetwork(*ipv4: PrimaryIP | None = None*, *ipv6: PrimaryIP | None = None*, *enable_ipv4: bool = True*, *enable_ipv6: bool = True*)

Server Create Public Network Domain

Parameters

- **ipv4** – Optional[*PrimaryIP*]
- **ipv6** – Optional[*PrimaryIP*]
- **enable_ipv4** – bool
- **enable_ipv6** – bool

class ResetPasswordResponse(*action: BoundAction*, *root_password: str*)

Reset Password Response Domain

Parameters

- **action** – *BoundAction* Shows the progress of the server passwort reset action
- **root_password** – str The root password of the server

class EnableRescueResponse(*action: BoundAction*, *root_password: str*)

Enable Rescue Response Domain

Parameters

- **action** – *BoundAction* Shows the progress of the server enable rescue action
- **root_password** – str The root password of the server in the rescue mode

class RequestConsoleResponse(*action: BoundAction*, *wss_url: str*, *password: str*)

Request Console Response Domain

Parameters

- **action** – *BoundAction* Shows the progress of the server request console action
- **wss_url** – str URL of websocket proxy to use. This includes a token which is valid for a limited time only.
- **password** – str VNC password to use for this connection. This password only works in combination with a wss_url with valid token.

4.2.16 SSHKeysClient

class SSHKeysClient(*client: Client*)

create(*name: str*, *public_key: str*, *labels: dict[str, str] | None = None*) → *BoundSSHKey*

Creates a new SSH key with the given name and public_key.

Parameters

- **name** – str
- **public_key** – str Public Key of the SSH Key you want create
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundSSHKey

delete(*ssh_key:* SSHKey | BoundSSHKey) → bool

Deletes an SSH key. It cannot be used anymore.

Parameters
ssh_key – *BoundSSHKey* or *SSHKey*

Returns
True

get_all(*name: str | None = None*, *fingerprint: str | None = None*, *label_selector: str | None = None*) → list[*BoundSSHKey*]

Get all SSH keys from the account

Parameters

- **name** – str (optional) Can be used to filter SSH keys by their name. The response will only contain the SSH key matching the specified name.
- **fingerprint** – str (optional) Can be used to filter SSH keys by their fingerprint. The response will only contain the SSH key matching the specified fingerprint.
- **label_selector** – str (optional) Can be used to filter SSH keys by labels. The response will only contain SSH keys matching the label selector.

Returns
List[*BoundSSHKey*]

get_by_fingerprint(*fingerprint: str*) → *BoundSSHKey* | None

Get ssh key by fingerprint

Parameters
fingerprint – str Used to get ssh key by fingerprint.

Returns
BoundSSHKey

get_by_id(*id: int*) → *BoundSSHKey*

Get a specific SSH Key by its ID

Parameters
id – int

Returns
BoundSSHKey

get_by_name(*name: str*) → *BoundSSHKey* | None

Get ssh key by name

Parameters
name – str Used to get ssh key by name.

Returns
BoundSSHKey

get_list(*name: str | None = None*, *fingerprint: str | None = None*, *label_selector: str | None = None*, *page: int | None = None*, *per_page: int | None = None*) → SSHKeysPageResult

Get a list of SSH keys from the account

Parameters

- **name** – str (optional) Can be used to filter SSH keys by their name. The response will only contain the SSH key matching the specified name.
- **fingerprint** – str (optional) Can be used to filter SSH keys by their fingerprint. The response will only contain the SSH key matching the specified fingerprint.
- **label_selector** – str (optional) Can be used to filter SSH keys by labels. The response will only contain SSH keys matching the label selector.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page

Returns
(List[*BoundSSHKey*], Meta)

update(*ssh_key: SSHKey | BoundSSHKey*, *name: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundSSHKey*

Updates an SSH key. You can update an SSH key name and an SSH key labels.

Parameters

- **ssh_key** – *BoundSSHKey* or *SSHKey*
- **name** – str (optional) New Description to set
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundSSHKey

class BoundSSHKey(*client: ResourceClientBase*, *data: dict[str, Any]*, *complete: bool = True*)

delete() → bool

Deletes an SSH key. It cannot be used anymore. :return: boolean

model

alias of *SSHKey*

update(*name: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundSSHKey*

Updates an SSH key. You can update an SSH key name and an SSH key labels.

Parameters

- **description** – str (optional) New Description to set
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundSSHKey

class SSHKey(*id: int | None = None*, *name: str | None = None*, *fingerprint: str | None = None*, *public_key: str | None = None*, *labels: dict[str, str] | None = None*, *created: str | None = None*)

SSHKey Domain

Parameters

- **id** – int ID of the SSH key
- **name** – str Name of the SSH key (must be unique per project)
- **fingerprint** – str Fingerprint of public key
- **public_key** – str Public Key

- **labels** – Dict User-defined labels (key-value pairs)
- **created** – datetime Point in time when the SSH Key was created

4.2.17 StorageBoxTypesClient

class StorageBoxTypesClient(*client:* Client)

A client for the Storage Box Types API.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-types.

get_all(*name: str | None = None*) → list[*BoundStorageBoxType*]

Returns all Storage Box Types.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-types-list-storage-box-types

Parameters
name – Name of the Storage Box Type.

get_by_id(*id: int*) → *BoundStorageBoxType*

Returns a specific Storage Box Type.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-types-get-a-storage-box-type

Parameters
id – ID of the Storage Box Type.

get_by_name(*name: str*) → *BoundStorageBoxType* | None

Returns a specific Storage Box Type.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-types-list-storage-box-types

Parameters
name – Name of the Storage Box Type.

get_list(*name: str | None = None*, *page: int | None = None*, *per_page: int | None = None*) → *StorageBoxTypesPageResult*

Returns a list of Storage Box Types for a specific page.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-types-list-storage-box-types

Parameters

- **name** – Name of the Storage Box Type.
- **page** – Page number to return.
- **per_page** – Maximum number of entries returned per page.

class StorageBoxTypesPageResult(*storage_box_types*, *meta*)

meta: Meta

Alias for field number 1

storage_box_types: list[*BoundStorageBoxType*]

Alias for field number 0

class BoundStorageBoxType(*client: ResourceClientBase*, *data: dict[str, Any]*, *complete: bool = True*)

model

alias of *StorageBoxType*

class StorageBoxType(*id: int | None = None*, *name: str | None = None*, *description: str | None = None*, *snapshot_limit: int | None = None*, *automatic_snapshot_limit: int | None = None*, *subaccounts_limit: int | None = None*, *size: int | None = None*, *prices: list[dict[str, Any]] | None = None*, *deprecation: dict[str, Any] | None = None*)

Storage Box Type Domain.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-types.

4.2.18 StorageBoxesClient

class StorageBoxesClient(*client:* Client)

A client for the Storage Boxes API.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes.

actions: *ResourceActionsClient*

Storage Boxes scoped actions client

Type
ResourceActionsClient

change_protection(*storage_box:* StorageBox | BoundStorageBox, *, *delete: bool | None = None*) → *BoundAction*

Changes the protection of a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-change-protection

Parameters

- **storage_box** – Storage Box to update.
- **delete** – Prevents the Storage Box from being deleted.

change_subaccount_home_directory(*subaccount:* StorageBoxSubaccount | BoundStorageBoxSubaccount, *home_directory: str*) → *BoundAction*

Change the home directory of a Storage Box Subaccount.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccount-actions-change-home-directory

Parameters

- **subaccount** – Storage Box Subaccount to update.
- **home_directory** – Home directory for the Subaccount.

change_type(*storage_box:* StorageBox | BoundStorageBox, *storage_box_type:* StorageBoxType | BoundStorageBoxType) → *BoundAction*

Changes the type of a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-change-type

Parameters

- **storage_box** – Storage Box to update.
- **storage_box_type** – Storage Box Type to change to.

create(*, *name: str*, *password: str*, *location:* BoundLocation | Location, *storage_box_type:* BoundStorageBoxType | StorageBoxType, *ssh_keys: list[str | SSHKey | BoundSSHKey] | None = None*, *access_settings:* StorageBoxAccessSettings | *None = None*, *labels: dict[str, str] | None = None*) → *CreateStorageBoxResponse*

Creates a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes-create-a-storage-box

Parameters

- **name** – Name of the Storage Box.
- **password** – Password of the Storage Box.
- **location** – Location of the Storage Box.
- **storage_box_type** – Type of the Storage Box.
- **ssh_keys** – SSH public keys of the Storage Box.
- **access_settings** – Access settings of the Storage Box.
- **labels** – User-defined labels (key/value pairs) for the Storage Box.

create_snapshot(*storage_box: StorageBox | BoundStorageBox*, ***, *description: str | None = None*, *labels: dict[str, str] | None = None*) → *CreateStorageBoxSnapshotResponse*

Creates a Snapshot of the Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-create-a-snapshot

Parameters

- **storage_box** – Storage Box to create a Snapshot from.
- **description** – Description of the Snapshot.
- **labels** – User-defined labels (key/value pairs) for the Snapshot.

create_subaccount(*storage_box: StorageBox | BoundStorageBox*, ***, *name: str | None = None*, *home_directory: str*, *password: str*, *access_settings: StorageBoxSubaccountAccessSettings | None = None*, *description: str | None = None*, *labels: dict[str, str] | None = None*) → *CreateStorageBoxSubaccountResponse*

Creates a Subaccount for the Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-create-a-subaccount

Parameters

- **storage_box** – Storage Box to create a Subaccount for.
- **name** – Name of the Subaccount.
- **home_directory** – Home directory of the Subaccount.
- **password** – Password of the Subaccount.
- **access_settings** – Access settings of the Subaccount.
- **description** – Description of the Subaccount.
- **labels** – User-defined labels (key/value pairs) for the Subaccount.

delete(*storage_box: BoundStorageBox | StorageBox*) → *DeleteStorageBoxResponse*

Deletes a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes-delete-a-storage-box

Parameters

storage_box – Storage Box to delete.

delete_snapshot(*snapshot:* StorageBoxSnapshot | BoundStorageBoxSnapshot) → *DeleteStorageBoxSnapshotResponse*

Deletes a Storage Box Snapshot.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-delete-a-snapshot

Parameters
snapshot – Storage Box Snapshot to delete.

delete_subaccount(*subaccount:* StorageBoxSubaccount | BoundStorageBoxSubaccount) → *DeleteStorageBoxSubaccountResponse*

Deletes a Storage Box Subaccount.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-delete-a-subaccount

Parameters
subaccount – Storage Box Subaccount to delete.

disable_snapshot_plan(*storage_box:* StorageBox | BoundStorageBox) → *BoundAction*

Disable the snapshot plan of a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-disable-snapshot-plan

Parameters
storage_box – Storage Box to update.

enable_snapshot_plan(*storage_box:* StorageBox | BoundStorageBox, *snapshot_plan:* StorageBoxSnapshotPlan) → *BoundAction*

Enable the snapshot plan of a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-enable-snapshot-plan

Parameters

- **storage_box** – Storage Box to update.
- **snapshot_plan** – Snapshot Plan to enable.

get_actions(*storage_box:* StorageBox | BoundStorageBox, *, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-list-actions-for-a-storage-box

Parameters

- **storage_box** – Storage Box to get the Actions for.
- **status** – Filter the actions by status. The response will only contain actions matching the specified statuses.
- **sort** – Sort resources by field and direction.

get_actions_list(*storage_box:* StorageBox | BoundStorageBox, *, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-list-actions-for-a-storage-box

Parameters

- **storage_box** – Storage Box to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_all(*, *name: str | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*) → list[*BoundStorageBox*]

Returns all Storage Boxes.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes-list-storage-boxes

Parameters

- **name** – Name of the Storage Box.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

get_by_id(*id: int*) → *BoundStorageBox*

Returns a specific Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes-get-a-storage-box

Parameters
id – ID of the Storage Box.

get_by_name(*name: str*) → *BoundStorageBox* | None

Returns a specific Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes-list-storage-boxes

Parameters
name – Name of the Storage Box.

get_folders(*storage_box:* BoundStorageBox | StorageBox, *, *path: str | None = None*) → *StorageBoxFoldersResponse*

Lists the (sub)folders contained in a Storage Box.

Files are not part of the response.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes-list-folders-of-a-storage-box

Parameters

- **storage_box** – Storage Box to list the folders from.
- **path** – Relative path to list the folders from.

get_list(*, *name: str | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → *StorageBoxesPageResult*

Returns a paginated list of Storage Boxes for a specific page.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes-list-storage-boxes

Parameters

- **name** – Name of the Storage Box.

- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.
- **page** – Page number to return.
- **per_page** – Maximum number of entries returned per page.

get_snapshot_all(*storage_box: StorageBox | BoundStorageBox*, ***, *name: str | None = None*, *is_automatic: bool | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*) → list[*BoundStorageBoxSnapshot*]

Returns all Snapshots for a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-list-snapshots

Parameters

- **storage_box** – Storage Box to get the Snapshots from.
- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **is_automatic** – Filter whether the snapshot was made by a Snapshot Plan.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

get_snapshot_by_id(*storage_box: StorageBox | BoundStorageBox*, *id: int*) → *BoundStorageBoxSnapshot*

Returns a single Snapshot from a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-get-a-snapshot

Parameters

- **storage_box** – Storage Box to get the Snapshot from.
- **id** – ID of the Snapshot.

get_snapshot_by_name(*storage_box: StorageBox | BoundStorageBox*, *name: str*) → *BoundStorageBoxSnapshot* | None

Returns a single Snapshot from a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-list-snapshots

Parameters

- **storage_box** – Storage Box to get the Snapshot from.
- **name** – Name of the Snapshot.

get_snapshot_list(*storage_box: StorageBox | BoundStorageBox*, ***, *name: str | None = None*, *is_automatic: bool | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*) → *StorageBoxSnapshotsPageResult*

Returns all Snapshots for a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-list-snapshots

Parameters

- **storage_box** – Storage Box to get the Snapshots from.
- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.

- **is_automatic** – Filter whether the snapshot was made by a Snapshot Plan.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

get_subaccount_all(*storage_box: StorageBox | BoundStorageBox*, ***, *name: str | None = None*, *username: str | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*) → list[*BoundStorageBoxSubaccount*]

Returns all Subaccounts for a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-list-subaccounts

Parameters

- **storage_box** – Storage Box to get the Subaccount from.
- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **username** – Filter resources by their username. The response will only contain the resources matching exactly the specified username.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

get_subaccount_by_id(*storage_box: StorageBox | BoundStorageBox*, *id: int*) → *BoundStorageBoxSubaccount*

Returns a single Subaccount from a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-get-a-subaccount

Parameters

- **storage_box** – Storage Box to get the Subaccount from.
- **id** – ID of the Subaccount.

get_subaccount_by_name(*storage_box: StorageBox | BoundStorageBox*, *name: str*) → *BoundStorageBoxSubaccount* | None

Returns a single Subaccount from a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-list-subaccounts

Parameters

- **storage_box** – Storage Box to get the Subaccount from.
- **name** – Name of the Subaccount.

get_subaccount_by_username(*storage_box: StorageBox | BoundStorageBox*, *username: str*) → *BoundStorageBoxSubaccount* | None

Returns a single Subaccount from a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-list-subaccounts

Parameters

- **storage_box** – Storage Box to get the Subaccount from.
- **username** – User name of the Subaccount.

get_subaccount_list(*storage_box: StorageBox | BoundStorageBox*, ***, *name: str | None = None*, *username: str | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*) → *StorageBoxSubaccountsPageResult*

Returns all Subaccounts for a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-list-subaccounts

Parameters

- **storage_box** – Storage Box to get the Subaccount from.
- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **username** – Filter resources by their username. The response will only contain the resources matching exactly the specified username.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

reset_password(*storage_box: StorageBox | BoundStorageBox*, *password: str*) → *BoundAction*

Reset the password of a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-reset-password

Parameters

- **storage_box** – Storage Box to update.
- **password** – New password.

reset_subaccount_password(*subaccount: StorageBoxSubaccount | BoundStorageBoxSubaccount*, *password: str*) → *BoundAction*

Reset the password of a Storage Box Subaccount.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccount-actions-reset-password

Parameters

- **subaccount** – Storage Box Subaccount to update.
- **password** – Password for the Subaccount.

rollback_snapshot(*storage_box: StorageBox | BoundStorageBox*, *snapshot: StorageBoxSnapshot | BoundStorageBoxSnapshot*) → *BoundAction*

Rollback the Storage Box to the given snapshot.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-rollback-snapshot

Parameters

- **storage_box** – Storage Box to update.
- **snapshot** – Snapshot to rollback to.

update(*storage_box: BoundStorageBox | StorageBox*, ***, *name: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundStorageBox*

Updates a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes-update-a-storage-box

Parameters

- **storage_box** – Storage Box to update.
- **name** – Name of the Storage Box.
- **labels** – User-defined labels (key/value pairs) for the Storage Box.

update_access_settings(*storage_box: StorageBox | BoundStorageBox*, *access_settings: StorageBoxAccessSettings*) → *BoundAction*

Update the access settings of a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-update-access-settings

Parameters

- **storage_box** – Storage Box to update.
- **access_settings** – New access settings for the Storage Box.

update_snapshot(*snapshot: StorageBoxSnapshot | BoundStorageBoxSnapshot*, ***, *description: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundStorageBoxSnapshot*

Updates a Storage Box Snapshot.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-update-a-snapshot

Parameters

- **snapshot** – Storage Box Snapshot to update.
- **description** – Description of the Snapshot.
- **labels** – User-defined labels (key/value pairs) for the Snapshot.

update_subaccount(*subaccount: StorageBoxSubaccount | BoundStorageBoxSubaccount*, ***, *name: str | None = None*, *description: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundStorageBoxSubaccount*

Updates a Storage Box Subaccount.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-update-a-subaccount

Parameters

- **subaccount** – Storage Box Subaccount to update.
- **name** – Name of the Subaccount.
- **description** – Description of the Subaccount.
- **labels** – User-defined labels (key/value pairs) for the Subaccount.

update_subaccount_access_settings(*subaccount: StorageBoxSubaccount | BoundStorageBoxSubaccount*, *access_settings: StorageBoxSubaccountAccessSettings*) → *BoundAction*

Update the access settings of a Storage Box Subaccount.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccount-actions-update-access-settings

Parameters

- **subaccount** – Storage Box Subaccount to update.
- **access_settings** – Access settings for the Subaccount.

class StorageBoxesPageResult(*storage_boxes*, *meta*)

meta: Meta

Alias for field number 1

storage_boxes: list[*BoundStorageBox*]

Alias for field number 0

class StorageBoxSnapshotsPageResult(*snapshots*, *meta*)

meta: Meta

Alias for field number 1

snapshots: list[*BoundStorageBoxSnapshot*]

Alias for field number 0

class StorageBoxSubaccountsPageResult(*subaccounts*, *meta*)

meta: Meta

Alias for field number 1

subaccounts: list[*BoundStorageBoxSubaccount*]

Alias for field number 0

class BoundStorageBox(*client:* StorageBoxesClient, *data: dict[str, Any]*, *complete: bool = True*)

change_protection(*, *delete: bool | None = None*) → *BoundAction*

Changes the protection of a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-change-protection

Parameters

delete – Prevents the Storage Box from being deleted.

change_type(*storage_box_type:* StorageBoxType | BoundStorageBoxType) → *BoundAction*

Changes the type of a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-change-type

Parameters

storage_box_type – Storage Box Type to change to.

create_snapshot(*, *description: str | None = None*, *labels: dict[str, str] | None = None*) → *CreateStorageBoxSnapshotResponse*

Creates a Snapshot of the Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-create-a-snapshot

Parameters

- **description** – Description of the Snapshot.
- **labels** – User-defined labels (key/value pairs) for the Snapshot.

create_subaccount(*, *name: str | None = None*, *home_directory: str*, *password: str*, *access_settings:* StorageBoxSubaccountAccessSettings | *None = None*, *description: str | None = None*, *labels: dict[str, str] | None = None*) → *CreateStorageBoxSubaccountResponse*

Creates a Subaccount for the Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-create-a-subaccount

Parameters

- **storage_box** – Storage Box to create a Subaccount for.

- **name** – Name of the Subaccount.
- **home_directory** – Home directory of the Subaccount.
- **password** – Password of the Subaccount.
- **access_settings** – Access settings of the Subaccount.
- **description** – Description of the Subaccount.
- **labels** – User-defined labels (key/value pairs) for the Subaccount.

delete() → *DeleteStorageBoxResponse*

Deletes a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes-delete-a-storage-box

disable_snapshot_plan() → *BoundAction*

Disable the snapshot plan of a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-disable-snapshot-plan

enable_snapshot_plan(*snapshot_plan:* StorageBoxSnapshotPlan) → *BoundAction*

Enable the snapshot plan of a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-enable-snapshot-plan

Parameters

snapshot_plan – Snapshot Plan to enable.

get_actions(*, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-list-actions-for-a-storage-box

Parameters

- **status** – Filter the actions by status. The response will only contain actions matching the specified statuses.
- **sort** – Sort resources by field and direction.

get_actions_list(*, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-list-actions-for-a-storage-box

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_folders(*, path: str | None = None) → StorageBoxFoldersResponse

Lists the (sub)folders contained in a Storage Box.

Files are not part of the response.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes-list-folders-of-a-storage-box

Parameters
path – Relative path to list the folders from.

get_snapshot_all(*, name: str | None = None, is_automatic: bool | None = None, label_selector: str | None = None, sort: list[str] | None = None) → list[BoundStorageBoxSnapshot]

Returns all Snapshots for a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-list-snapshots

Parameters

- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **is_automatic** – Filter whether the snapshot was made by a Snapshot Plan.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

get_snapshot_by_id(id: int) → BoundStorageBoxSnapshot

Returns a single Snapshot from a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-get-a-snapshot

Parameters
id – ID of the Snapshot.

get_snapshot_by_name(name: str) → BoundStorageBoxSnapshot | None

Returns a single Snapshot from a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-list-snapshots

Parameters
name – Name of the Snapshot.

get_snapshot_list(*, name: str | None = None, is_automatic: bool | None = None, label_selector: str | None = None, sort: list[str] | None = None) → StorageBoxSnapshotsPageResult

Returns all Snapshots for a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-list-snapshots

Parameters

- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **is_automatic** – Filter wether the snapshot was made by a Snapshot Plan.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

get_subaccount_all(*, *name: str | None = None*, *username: str | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*) → list[*BoundStorageBoxSubaccount*]

Returns all Subaccounts for a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-list-subaccounts

Parameters

- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **username** – Filter resources by their username. The response will only contain the resources matching exactly the specified username.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

get_subaccount_by_id(*id: int*) → *BoundStorageBoxSubaccount*

Returns a single Subaccount from a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-get-a-subaccount

Parameters

id – ID of the Subaccount.

get_subaccount_by_name(*name: str*) → *BoundStorageBoxSubaccount* | None

Returns a single Subaccount from a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-list-subaccounts

Parameters

name – Name of the Subaccount.

get_subaccount_by_username(*username: str*) → *BoundStorageBoxSubaccount* | None

Returns a single Subaccount from a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-list-subaccounts

Parameters

username – User name of the Subaccount.

get_subaccount_list(*, *name: str | None = None*, *username: str | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*) → *StorageBoxSubaccountsPageResult*

Returns all Subaccounts for a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-list-subaccounts

Parameters

- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **username** – Filter resources by their username. The response will only contain the resources matching exactly the specified username.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

model

alias of *StorageBox*

reset_password(*password: str*) → *BoundAction*

Reset the password of a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-reset-password

Parameters
password – New password.

rollback_snapshot(*snapshot:* StorageBoxSnapshot | BoundStorageBoxSnapshot) → *BoundAction*

Rollback the Storage Box to the given snapshot.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-rollback-snapshot

Parameters
snapshot – Snapshot to rollback to.

update(*, *name: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundStorageBox*

Updates a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes-update-a-storage-box

Parameters

- **name** – Name of the Storage Box.
- **labels** – User-defined labels (key/value pairs) for the Storage Box.

update_access_settings(*access_settings:* StorageBoxAccessSettings) → *BoundAction*

Update the access settings of a Storage Box.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-actions-update-access-settings

Parameters
access_settings – New access settings for the Storage Box.

class BoundStorageBoxSnapshot(*client:* StorageBoxesClient, *data: dict[str, Any]*, *complete: bool = True*)

delete() → *DeleteStorageBoxSnapshotResponse*

Deletes a Storage Box Snapshot.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-delete-a-snapshot

model

alias of *StorageBoxSnapshot*

update(*, *description: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundStorageBoxSnapshot*

Updates a Storage Box Snapshot.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-snapshots-update-a-snapshot

Parameters

- **description** – Description of the Snapshot.
- **labels** – User-defined labels (key/value pairs) for the Snapshot.

class BoundStorageBoxSubaccount(*client:* StorageBoxesClient, *data: dict[str, Any]*, *complete: bool = True*)

change_home_directory(*home_directory: str*) → *BoundAction*

Change the home directory of a Storage Box Subaccount.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccount-actions-change-home-directory

Parameters
home_directory – Home directory for the Subaccount.

delete() → *DeleteStorageBoxSubaccountResponse*

Deletes a Storage Box Subaccount.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-delete-a-subaccount

model

alias of *StorageBoxSubaccount*

reset_password(*password: str*) → *BoundAction*

Reset the password of a Storage Box Subaccount.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccount-actions-reset-password

Parameters
password – Password for the Subaccount.

update(*, *name: str | None = None*, *description: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundStorageBoxSubaccount*

Updates a Storage Box Subaccount.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccounts-update-a-subaccount

Parameters

- **name** – Name of the Subaccount.
- **description** – Description of the Subaccount.
- **labels** – User-defined labels (key/value pairs) for the Subaccount.

update_access_settings(*access_settings:* StorageBoxSubaccountAccessSettings) → *BoundAction*

Update the access settings of a Storage Box Subaccount.

See https://docs.hetzner.cloud/reference/hetzner#storage-box-subaccount-actions-update-access-settings

Parameters
access_settings – Access settings for the Subaccount.

class StorageBox(*id: int | None = None*, *name: str | None = None*, *storage_box_type:* BoundStorageBoxType | StorageBoxType | *None = None*, *location:* BoundLocation | Location | *None = None*, *system: str | None = None*, *server: str | None = None*, *username: str | None = None*, *labels: dict[str, str] | None = None*, *protection: dict[str, bool] | None = None*, *snapshot_plan:* StorageBoxSnapshotPlan | *None = None*, *access_settings:* StorageBoxAccessSettings | *None = None*, *stats:* StorageBoxStats | *None = None*, *status: Literal['active', 'initializing', 'locked'] | None = None*, *created: str | None = None*)

Storage Box Domain.

See https://docs.hetzner.cloud/reference/hetzner#storage-boxes.

class StorageBoxAccessSettings(*reachable_externally: bool | None = None*, *samba_enabled: bool | None = None*, *ssh_enabled: bool | None = None*, *webdav_enabled: bool | None = None*, *zfs_enabled: bool | None = None*)

Storage Box Access Settings Domain.

to_payload() → dict[str, Any]

Generates the request payload from this domain object.

class StorageBoxSnapshotPlan(*max_snapshots: int*, *hour: int*, *minute: int*, *day_of_week: int | None = None*, *day_of_month: int | None = None*)

Storage Box Snapshot Plan Domain.

to_payload() → dict[str, Any]

Generates the request payload from this domain object.

class StorageBoxStats(*size: int | None = None*, *size_data: int | None = None*, *size_snapshots: int | None = None*)

Storage Box Stats Domain.

StorageBoxStatus

alias of Literal['active', 'initializing', 'locked']

class StorageBoxSnapshot(*id: int | None = None*, *name: str | None = None*, *description: str | None = None*, *is_automatic: bool | None = None*, *labels: dict[str, str] | None = None*, *storage_box:* BoundStorageBox | StorageBox | *None = None*, *created: str | None = None*, *stats:* StorageBoxSnapshotStats | *None = None*)

Storage Box Snapshot Domain.

class StorageBoxSnapshotStats(*size: int*, *size_filesystem: int*)

Storage Box Snapshot Stats Domain.

class StorageBoxSubaccount(*id: int | None = None*, *name: str | None = None*, *username: str | None = None*, *description: str | None = None*, *server: str | None = None*, *home_directory: str | None = None*, *access_settings:* StorageBoxSubaccountAccessSettings | *None = None*, *labels: dict[str, str] | None = None*, *storage_box:* BoundStorageBox | StorageBox | *None = None*, *created: str | None = None*)

Storage Box Subaccount Domain.

class StorageBoxSubaccountAccessSettings(*reachable_externally: bool | None = None*, *samba_enabled: bool | None = None*, *ssh_enabled: bool | None = None*, *webdav_enabled: bool | None = None*, *readonly: bool | None = None*)

Storage Box Subaccount Access Settings Domain.

to_payload() → dict[str, Any]

Generates the request payload from this domain object.

class CreateStorageBoxResponse(*storage_box:* BoundStorageBox, *action:* BoundAction)

Create Storage Box Response Domain.

class CreateStorageBoxSnapshotResponse(*snapshot:* BoundStorageBoxSnapshot, *action:* BoundAction)

Create Storage Box Snapshot Response Domain.

class CreateStorageBoxSubaccountResponse(*subaccount:* BoundStorageBoxSubaccount, *action:* BoundAction)

Create Storage Box Subaccount Response Domain.

class StorageBoxFoldersResponse(*folders: list[str]*)

Storage Box Folders Response Domain.

class DeleteStorageBoxResponse(*action:* BoundAction)

Delete Storage Box Response Domain.

class DeleteStorageBoxSnapshotResponse(*action:* BoundAction)

Delete Storage Box Snapshot Response Domain.

class DeleteStorageBoxSubaccountResponse(*action:* BoundAction)

Delete Storage Box Subaccount Response Domain.

4.2.19 VolumesClient

class VolumesClient(*client:* Client)

actions: *ResourceActionsClient*

Volumes scoped actions client

Type
ResourceActionsClient

attach(*volume:* Volume | BoundVolume, *server:* Server | BoundServer, *automount: bool | None = None*) → *BoundAction*

Attaches a volume to a server. Works only if the server is in the same location as the volume.

Parameters

- **volume** – *BoundVolume* or *Volume*
- **server** – *BoundServer* or *Server*
- **automount** – boolean

Returns
BoundAction

change_protection(*volume:* Volume | BoundVolume, *delete: bool | None = None*) → *BoundAction*

Changes the protection configuration of a volume.

Parameters

- **volume** – *BoundVolume* or *Volume*
- **delete** – boolean If True, prevents the volume from being deleted

Returns
BoundAction

create(*size: int*, *name: str*, *labels: str | None = None*, *location:* Location | *None = None*, *server:* Server | *None = None*, *automount: bool | None = None*, *format: str | None = None*) → *CreateVolumeResponse*

Creates a new volume attached to a server.

Parameters

- **size** – int Size of the volume in GB
- **name** – str Name of the volume
- **labels** – Dict[str,str] (optional) User-defined labels (key-value pairs)
- **location** – *BoundLocation* or *Location*
- **server** – *BoundServer* or *Server*
- **automount** – boolean (optional) Auto mount volumes after attach.
- **format** – str (optional) Format volume after creation. One of: xfs, ext4

Returns
CreateVolumeResponse

delete(*volume:* Volume | BoundVolume) → bool

Deletes a volume. All volume data is irreversibly destroyed. The volume must not be attached to a server and it must not have delete protection enabled.

Parameters
volume – *BoundVolume* or *Volume*

Returns
boolean

detach(*volume:* Volume | BoundVolume) → *BoundAction*

Detaches a volume from the server it's attached to. You may attach it to a server again at a later time.

Parameters
volume – *BoundVolume* or *Volume*

Returns
BoundAction

get_actions(*volume:* Volume | BoundVolume, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Volume.

Parameters

- **volume** – Volume to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*volume:* Volume | BoundVolume, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Volume.

Parameters

- **volume** – Volume to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_all(*label_selector: str | None = None*, *status: list[str] | None = None*) → list[*BoundVolume*]

Get all volumes from this account

Parameters

- **label_selector** – Can be used to filter volumes by labels. The response will only contain volumes matching the label selector.

- **status** – List[str] (optional) Can be used to filter volumes by their status. The response will only contain volumes matching the status.

Returns
List[*BoundVolume*]

get_by_id(*id: int*) → *BoundVolume*

Get a specific volume by its id

Parameters
id – int

Returns
BoundVolume

get_by_name(*name: str*) → *BoundVolume* | None

Get volume by name

Parameters
name – str Used to get volume by name.

Returns
BoundVolume

get_list(*name: str | None = None*, *label_selector: str | None = None*, *page: int | None = None*, *per_page: int | None = None*, *status: list[str] | None = None*) → VolumesPageResult

Get a list of volumes from this account

Parameters

- **name** – str (optional) Can be used to filter volumes by their name.
- **label_selector** – str (optional) Can be used to filter volumes by labels. The response will only contain volumes matching the label selector.
- **status** – List[str] (optional) Can be used to filter volumes by their status. The response will only contain volumes matching the status.
- **page** – int (optional) Specifies the page to fetch
- **per_page** – int (optional) Specifies how many results are returned by page

Returns
(List[*BoundVolume*], Meta)

resize(*volume:* Volume | BoundVolume, *size: int*) → *BoundAction*

Changes the size of a volume. Note that downsizing a volume is not possible.

Parameters

- **volume** – *BoundVolume* or *Volume*
- **size** – int New volume size in GB (must be greater than current size)

Returns
BoundAction

update(*volume:* Volume | BoundVolume, *name: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundVolume*

Updates the volume properties.

Parameters

- **volume** – *BoundVolume* or *Volume*

- **name** – str (optional) New volume name
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns
BoundAction

class **BoundVolume**(*client: VolumesClient*, *data: dict[str, Any]*, *complete: bool = True*)

attach(*server: Server | BoundServer*, *automount: bool | None = None*) → *BoundAction*

Attaches a volume to a server. Works only if the server is in the same location as the volume.

Parameters

- **server** – *BoundServer* or *Server*
- **automount** – boolean

Returns
BoundAction

change_protection(*delete: bool | None = None*) → *BoundAction*

Changes the protection configuration of a volume.

Parameters
delete – boolean If True, prevents the volume from being deleted

Returns
BoundAction

delete() → bool

Deletes a volume. All volume data is irreversibly destroyed. The volume must not be attached to a server and it must not have delete protection enabled.

Returns
boolean

detach() → *BoundAction*

Detaches a volume from the server it's attached to. You may attach it to a server again at a later time.

Returns
BoundAction

get_actions(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Volume.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Volume.

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

model

alias of *Volume*

resize(*size: int*) → *BoundAction*

Changes the size of a volume. Note that downsizing a volume is not possible.

Parameters

size – int New volume size in GB (must be greater than current size)

Returns

BoundAction

update(*name: str | None = None*, *labels: dict[str, str] | None = None*) → *BoundVolume*

Updates the volume properties.

Parameters

- **name** – str (optional) New volume name
- **labels** – Dict[str, str] (optional) User-defined labels (key-value pairs)

Returns

BoundAction

class Volume(*id: int*, *name: str | None = None*, *server:* Server | BoundServer | *None = None*, *created: str | None = None*, *location:* Location | BoundLocation | *None = None*, *size: int | None = None*, *linux_device: str | None = None*, *format: str | None = None*, *protection: VolumeProtection | None = None*, *labels: dict[str, str] | None = None*, *status: str | None = None*)

Volume Domain

Parameters

- **id** – int ID of the Volume
- **name** – str Name of the Volume
- **server** – *BoundServer*, None Server the Volume is attached to, None if it is not attached at all.
- **created** – datetime Point in time when the Volume was created
- **location** – *BoundLocation* Location of the Volume. Volume can only be attached to Servers in the same location.
- **size** – int Size in GB of the Volume
- **linux_device** – str Device path on the file system for the Volume
- **protection** – dict Protection configuration for the Volume
- **labels** – dict User-defined labels (key-value pairs)
- **status** – str Current status of the volume Choices: *creating*, *available*
- **format** – str, None Filesystem of the volume if formatted on creation, None if not formatted on creation.

STATUS_AVAILABLE = 'available'

Volume Status available

STATUS_CREATING = 'creating'

Volume Status creating

class CreateVolumeResponse(*volume:* BoundVolume, *action:* BoundAction, *next_actions: list[*BoundAction*]*)

Create Volume Response Domain

Parameters

- **volume** – *BoundVolume* The created volume
- **action** – *BoundAction* The action that shows the progress of the Volume Creation
- **next_actions** – List[*BoundAction*] List of actions that are performed after the creation, like attaching to a server

4.2.20 ZonesClient

class ZonesClient(*client:* Client)

ZoneClient is a client for the Zone (DNS) API.

See https://docs.hetzner.cloud/reference/cloud#zones and https://docs.hetzner.cloud/reference/cloud#zone-rrsets.

actions: *ResourceActionsClient*

Zones scoped actions client

Type
ResourceActionsClient

add_rrset_records(*rrset:* ZoneRRSet | BoundZoneRRSet, *records: list[*ZoneRecord*]*, *ttl: int | None = None*) → *BoundAction*

Adds records to a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-add-records-to-an-rrset

Parameters

- **rrset** – RRSet to update.
- **records** – Records to add to the RRSet.
- **ttl** – Time To Live (TTL) of the RRSet.

change_primary_nameservers(*zone:* Zone | BoundZone, *primary_nameservers: list[*ZonePrimaryNameserver*]*) → *BoundAction*

Changes the primary nameservers of a Zone.

See https://docs.hetzner.cloud/reference/cloud#zone-actions-change-a-zones-primary-nameservers

Parameters

- **zone** – Zone to update.
- **primary_nameservers** – Primary nameservers of the Zone.

change_protection(*zone:* Zone | BoundZone, *, *delete: bool | None = None*) → *BoundAction*

Changes the protection of a Zone.

See https://docs.hetzner.cloud/reference/cloud#zone-actions-change-a-zones-protection

Parameters

- **zone** – Zone to update.
- **delete** – Prevents the Zone from being deleted.

change_rrset_protection(*rrset: ZoneRRSet | BoundZoneRRSet*, *, *change: bool | None = None*) → *BoundAction*

Changes the protection of a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-change-an-rrsets-protection

Parameters

- **rrset** – RRSet to update.
- **change** – Prevent the Zone from being changed (deletion and updates).

change_rrset_ttl(*rrset: ZoneRRSet | BoundZoneRRSet*, *ttl: int | None*) → *BoundAction*

Changes the TTL of a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-change-an-rrsets-ttl

Parameters

- **rrset** – RRSet to update.
- **change** – Time To Live (TTL) of the RRSet.

change_ttl(*zone: Zone | BoundZone*, *ttl: int*) → *BoundAction*

Changes the TTL of a Zone.

See https://docs.hetzner.cloud/reference/cloud#zone-actions-change-a-zones-default-ttl

Parameters

- **zone** – Zone to update.
- **ttl** – Default Time To Live (TTL) of the Zone.

create(*, *name: str*, *mode: Literal['primary', 'secondary']*, *ttl: int | None = None*, *labels: dict[str, str] | None = None*, *primary_nameservers: list[ZonePrimaryNameserver] | None = None*, *rrsets: list[ZoneRRSet] | None = None*, *zonefile: str | None = None*) → *CreateZoneResponse*

Creates a Zone.

A default SOA and three NS resource records with the assigned Hetzner nameservers are created automatically.

See https://docs.hetzner.cloud/reference/cloud#zones-create-a-zone

Parameters

- **name** – Name of the Zone.
- **mode** – Mode of the Zone.
- **ttl** – Default Time To Live (TTL) of the Zone.
- **labels** – User-defined labels (key/value pairs) for the Resource.
- **primary_nameservers** – Primary nameservers of the Zone.
- **rrsets** – RRSets to be added to the Zone.
- **zonefile** – Zone file to import.

create_rrset(*zone:* Zone | BoundZone, *, *name: str*, *type: Literal['A', 'AAAA', 'CAA', 'CNAME', 'DS', 'HINFO', 'HTTPS', 'MX', 'NS', 'PTR', 'RP', 'SOA', 'SRV', 'SVCB', 'TLSA', 'TXT']*, *ttl: int | None = None*, *labels: dict[str, str] | None = None*, *records: list[*ZoneRecord*] | None = None*) → CreateZoneRRSetResponse

Creates a ZoneRRSet in the Zone.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-create-an-rrset

Parameters

- **zone** – Zone to create the RRSets in.
- **name** – Name of the RRSet.
- **type** – Type of the RRSet.
- **ttl** – Time To Live (TTL) of the RRSet.
- **labels** – User-defined labels (key/value pairs) for the Resource.
- **records** – Records of the RRSet.

delete(*zone:* Zone | BoundZone) → DeleteZoneResponse

Deletes a Zone.

See https://docs.hetzner.cloud/reference/cloud#zones-delete-a-zone

Parameters
zone – Zone to delete.

delete_rrset(*rrset:* ZoneRRSet | BoundZoneRRSet) → DeleteZoneRRSetResponse

Deletes a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-delete-an-rrset

Parameters
rrset – RRSet to delete.

export_zonefile(*zone:* Zone | BoundZone) → ExportZonefileResponse

Returns a generated Zone file in BIND (RFC 1034/1035) format.

See https://docs.hetzner.cloud/reference/cloud#zones-export-a-zone-file

Parameters
zone – Zone to export the zone file from.

get(*id_or_name: int | str*) → *BoundZone*

Returns a single Zone.

See https://docs.hetzner.cloud/reference/cloud#zones-get-a-zone

Parameters
id_or_name – ID or Name of the Zone.

get_actions(*zone:* Zone | BoundZone, *, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Zone.

See https://docs.hetzner.cloud/reference/cloud#zones-list-zones

Parameters

- **zone** – Zone to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*zone: Zone | BoundZone*, ***, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Zone.

See https://docs.hetzner.cloud/reference/cloud#zones-list-zones

Parameters

- **zone** – Zone to get the Actions for.
- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_all(***, *name: str | None = None*, *mode: Literal['primary', 'secondary'] | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*) → list[*BoundZone*]

Returns a list of all Zone.

See https://docs.hetzner.cloud/reference/cloud#zones-list-zones

Parameters

- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **mode** – Filter resources by their mode. The response will only contain the resources matching exactly the specified mode.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

get_list(***, *name: str | None = None*, *mode: Literal['primary', 'secondary'] | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ZonesPageResult

Returns a list of Zone for a specific page.

See https://docs.hetzner.cloud/reference/cloud#zones-list-zones

Parameters

- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **mode** – Filter resources by their mode. The response will only contain the resources matching exactly the specified mode.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

- **page** – Page number to return.
- **per_page** – Maximum number of entries returned per page.

get_rrset(*zone: Zone | BoundZone*, *name: str*, *type: Literal['A', 'AAAA', 'CAA', 'CNAME', 'DS', 'HINFO', 'HTTPS', 'MX', 'NS', 'PTR', 'RP', 'SOA', 'SRV', 'SVCB', 'TLSA', 'TXT']*) → *BoundZoneRRSet*

Returns a single ZoneRRSet from the Zone.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-get-an-rrset

Parameters

- **zone** – Zone to fetch the RRSet from.
- **name** – Name of the RRSet.
- **type** – Type of the RRSet.

get_rrset_all(*zone: Zone | BoundZone*, ***, *name: str | None = None*, *type: list[Literal['A', 'AAAA', 'CAA', 'CNAME', 'DS', 'HINFO', 'HTTPS', 'MX', 'NS', 'PTR', 'RP', 'SOA', 'SRV', 'SVCB', 'TLSA', 'TXT']] | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*) → list[*BoundZoneRRSet*]

Returns all ZoneRRSet in the Zone.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-list-rrsets

Parameters

- **zone** – Zone to fetch the RRSets from.
- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **type** – Filter resources by their type. The response will only contain the resources matching exactly the specified type.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

get_rrset_list(*zone: Zone | BoundZone*, ***, *name: str | None = None*, *type: list[Literal['A', 'AAAA', 'CAA', 'CNAME', 'DS', 'HINFO', 'HTTPS', 'MX', 'NS', 'PTR', 'RP', 'SOA', 'SRV', 'SVCB', 'TLSA', 'TXT']] | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ZoneRRSetsPageResult

Returns all ZoneRRSet in the Zone for a specific page.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-list-rrsets

Parameters

- **zone** – Zone to fetch the RRSets from.
- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **type** – Filter resources by their type. The response will only contain the resources matching exactly the specified type.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.
- **page** – Page number to return.

- **per_page** – Maximum number of entries returned per page.

import_zonefile(*zone:* Zone | BoundZone, *zonefile: str*) → *BoundAction*

Imports a zone file, replacing all resource record sets (ZoneRRSet).

See https://docs.hetzner.cloud/reference/cloud#zone-actions-import-a-zone-file

Parameters

- **zone** – Zone to import the zone file into.
- **zonefile** – Zone file to import.

remove_rrset_records(*rrset:* ZoneRRSet | BoundZoneRRSet, *records: list[*ZoneRecord*]*) → *BoundAction*

Removes records from a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-remove-records-from-an-rrset

Parameters

- **rrset** – RRSet to update.
- **records** – Records to remove from the RRSet.

set_rrset_records(*rrset:* ZoneRRSet | BoundZoneRRSet, *records: list[*ZoneRecord*]*) → *BoundAction*

Sets the records of a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-set-records-of-an-rrset

Parameters

- **rrset** – RRSet to update.
- **records** – Records to set in the RRSet.

update(*zone:* Zone | BoundZone, *, *labels: dict[str, str] | None = None*) → *BoundZone*

Updates a Zone.

See https://docs.hetzner.cloud/reference/cloud#zones-update-a-zone

Parameters

- **zone** – Zone to update.
- **labels** – User-defined labels (key/value pairs) for the Resource.

update_rrset(*rrset:* ZoneRRSet | BoundZoneRRSet, *, *labels: dict[str, str] | None = None*) → *BoundZoneRRSet*

Updates a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-update-an-rrset

Parameters

- **rrset** – RRSet to update.
- **labels** – User-defined labels (key/value pairs) for the Resource.

update_rrset_records(*rrset:* ZoneRRSet | BoundZoneRRSet, *records: list[*ZoneRecord*]*) → *BoundAction*

Updates records in a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-update-records-of-an-rrset

Parameters

- **rrset** – RRSet to update.

- **records** – Records to update in the RRSet.

class **BoundZone**(*client: ZonesClient*, *data: dict[str, Any]*, *complete: bool = True*)

add_rrset_records(*rrset: ZoneRRSet | BoundZoneRRSet*, *records: list[ZoneRecord]*, *ttl: int | None = None*) → *BoundAction*

Adds records to a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-add-records-to-an-rrset

Parameters

- **rrset** – RRSet to update.
- **records** – Records to add to the RRSet.
- **ttl** – Time To Live (TTL) of the RRSet.

change_primary_nameservers(*primary_nameservers: list[ZonePrimaryNameserver]*) → *BoundAction*

Changes the primary nameservers of the Zone.

See https://docs.hetzner.cloud/reference/cloud#zone-actions-change-a-zones-primary-nameservers

Parameters

primary_nameservers – Primary nameservers of the Zone.

change_protection(*, *delete: bool | None = None*) → *BoundAction*

Changes the protection of the Zone.

See https://docs.hetzner.cloud/reference/cloud#zone-actions-change-a-zones-protection

Parameters

delete – Prevents the Zone from being deleted.

change_rrset_protection(*rrset: ZoneRRSet | BoundZoneRRSet*, *, *change: bool | None = None*) → *BoundAction*

Changes the protection of a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-change-an-rrsets-protection

Parameters

- **rrset** – RRSet to update.
- **change** – Prevent the Zone from being changed (deletion and updates).

change_rrset_ttl(*rrset: ZoneRRSet | BoundZoneRRSet*, *ttl: int | None*) → *BoundAction*

Changes the TTL of a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-change-an-rrsets-ttl

Parameters

- **rrset** – RRSet to update.
- **change** – Time To Live (TTL) of the RRSet.

change_ttl(*ttl: int*) → *BoundAction*

Changes the TTL of the Zone.

See https://docs.hetzner.cloud/reference/cloud#zone-actions-change-a-zones-default-ttl

Parameters

ttl – Default Time To Live (TTL) of the Zone.

create_rrset(*, *name: str*, *type: Literal['A', 'AAAA', 'CAA', 'CNAME', 'DS', 'HINFO', 'HTTPS', 'MX', 'NS', 'PTR', 'RP', 'SOA', 'SRV', 'SVCB', 'TLSA', 'TXT']*, *ttl: int | None = None*, *labels: dict[str, str] | None = None*, *records: list[ZoneRecord] | None = None*) → CreateZoneRRSetResponse

Creates a ZoneRRSet in the Zone.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-create-an-rrset

Parameters

- **name** – Name of the RRSet.
- **type** – Type of the RRSet.
- **ttl** – Time To Live (TTL) of the RRSet.
- **labels** – User-defined labels (key/value pairs) for the Resource.
- **records** – Records of the RRSet.

delete() → DeleteZoneResponse

Deletes the Zone.

See https://docs.hetzner.cloud/reference/cloud#zones-delete-a-zone

delete_rrset(*rrset: ZoneRRSet | BoundZoneRRSet*) → DeleteZoneRRSetResponse

Deletes a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-delete-an-rrset

Parameters

rrset – RRSet to delete.

export_zonefile() → ExportZonefileResponse

Returns a generated Zone file in BIND (RFC 1034/1035) format.

See https://docs.hetzner.cloud/reference/cloud#zones-export-a-zone-file

get_actions(*, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*) → list[*BoundAction*]

Returns all Actions for a Zone.

See https://docs.hetzner.cloud/reference/cloud#zones-list-zones

Parameters

- **status** – Filter the Actions by status.
- **sort** – Sort Actions by field and direction.

get_actions_list(*, *status: list[Literal['running', 'success', 'error']] | None = None*, *sort: list[Literal['id', 'id:asc', 'id:desc', 'command', 'command:asc', 'command:desc', 'status', 'status:asc', 'status:desc', 'started', 'started:asc', 'started:desc', 'finished', 'finished:asc', 'finished:desc']] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ActionsPageResult

Returns a paginated list of Actions for a Zone.

See https://docs.hetzner.cloud/reference/cloud#zones-list-zones

Parameters

- **status** – Filter the Actions by status.

- **sort** – Sort Actions by field and direction.
- **page** – Page number to get.
- **per_page** – Maximum number of Actions returned per page.

get_rrset(*name: str*, *type: Literal['A', 'AAAA', 'CAA', 'CNAME', 'DS', 'HINFO', 'HTTPS', 'MX', 'NS', 'PTR', 'RP', 'SOA', 'SRV', 'SVCB', 'TLSA', 'TXT']*) → *BoundZoneRRSet*

Returns a single ZoneRRSet from the Zone.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-get-an-rrset

Parameters

- **name** – Name of the RRSet.
- **type** – Type of the RRSet.

get_rrset_all(*, *name: str | None = None*, *type: list[Literal['A', 'AAAA', 'CAA', 'CNAME', 'DS', 'HINFO', 'HTTPS', 'MX', 'NS', 'PTR', 'RP', 'SOA', 'SRV', 'SVCB', 'TLSA', 'TXT']] | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*) → list[*BoundZoneRRSet*]

Returns all ZoneRRSet in the Zone.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-list-rrsets

Parameters

- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **type** – Filter resources by their type. The response will only contain the resources matching exactly the specified type.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.

get_rrset_list(*, *name: str | None = None*, *type: list[Literal['A', 'AAAA', 'CAA', 'CNAME', 'DS', 'HINFO', 'HTTPS', 'MX', 'NS', 'PTR', 'RP', 'SOA', 'SRV', 'SVCB', 'TLSA', 'TXT']] | None = None*, *label_selector: str | None = None*, *sort: list[str] | None = None*, *page: int | None = None*, *per_page: int | None = None*) → ZoneRRSetsPageResult

Returns all ZoneRRSet in the Zone for a specific page.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-list-rrsets

Parameters

- **name** – Filter resources by their name. The response will only contain the resources matching exactly the specified name.
- **type** – Filter resources by their type. The response will only contain the resources matching exactly the specified type.
- **label_selector** – Filter resources by labels. The response will only contain resources matching the label selector.
- **sort** – Sort resources by field and direction.
- **page** – Page number to return.
- **per_page** – Maximum number of entries returned per page.

import_zonefile(*zonefile: str*) → *BoundAction*

Imports a zone file, replacing all resource record sets (ZoneRRSet).

See https://docs.hetzner.cloud/reference/cloud#zone-actions-import-a-zone-file

Parameters
zonefile – Zone file to import.

model

alias of *Zone*

remove_rrset_records(*rrset: ZoneRRSet | BoundZoneRRSet*, *records: list[ZoneRecord]*) → *BoundAction*

Removes records from a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-remove-records-from-an-rrset

Parameters

- **rrset** – RRSet to update.
- **records** – Records to remove from the RRSet.

set_rrset_records(*rrset: ZoneRRSet | BoundZoneRRSet*, *records: list[ZoneRecord]*) → *BoundAction*

Sets the records of a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-set-records-of-an-rrset

Parameters

- **rrset** – RRSet to update.
- **records** – Records to set in the RRSet.

update(*, *labels: dict[str, str] | None = None*) → *BoundZone*

Updates the Zone.

See https://docs.hetzner.cloud/reference/cloud#zones-update-a-zone

Parameters
labels – User-defined labels (key/value pairs) for the Resource.

update_rrset(*rrset: ZoneRRSet | BoundZoneRRSet*, *, *labels: dict[str, str] | None = None*) → *BoundZoneRRSet*

Updates a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-update-an-rrset

Parameters

- **rrset** – RRSet to update.
- **labels** – User-defined labels (key/value pairs) for the Resource.

update_rrset_records(*rrset: ZoneRRSet | BoundZoneRRSet*, *records: list[ZoneRecord]*) → *BoundAction*

Updates records in a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-update-records-of-an-rrset

Parameters

- **rrset** – RRSet to update.
- **records** – Records to update in the RRSet.

class BoundZoneRRSet(*client: ZonesClient*, *data: dict[str, Any]*, *complete: bool = True*)

add_rrset_records(*records: list[ZoneRecord]*, *ttl: int | None = None*) → *BoundAction*

Adds records to the ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-add-records-to-an-rrset

Parameters

- **records** – Records to add to the RRSet.
- **ttl** – Time To Live (TTL) of the RRSet.

change_rrset_protection(*, *change: bool | None = None*) → *BoundAction*

Changes the protection of the ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-change-an-rrsets-protection

Parameters

change – Prevent the Zone from being changed (deletion and updates).

change_rrset_ttl(*ttl: int | None*) → *BoundAction*

Changes the TTL of the ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-change-an-rrsets-ttl

Parameters

change – Time To Live (TTL) of the RRSet.

delete_rrset() → DeleteZoneRRSetResponse

Deletes the ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-delete-an-rrset

model

alias of *ZoneRRSet*

remove_rrset_records(*records: list[ZoneRecord]*) → *BoundAction*

Removes records from the ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-remove-records-from-an-rrset

Parameters

records – Records to remove from the RRSet.

set_rrset_records(*records: list[ZoneRecord]*) → *BoundAction*

Sets the records of the ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-set-records-of-an-rrset

Parameters

records – Records to set in the RRSet.

update_rrset(*, *labels: dict[str, str] | None = None*) → *BoundZoneRRSet*

Updates the ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets-update-an-rrset

Parameters

labels – User-defined labels (key/value pairs) for the Resource.

update_rrset_records(*records: list[ZoneRecord]*) → *BoundAction*

Updates records in a ZoneRRSet.

See https://docs.hetzner.cloud/reference/cloud#zone-rrset-actions-update-records-of-an-rrset

Parameters
records – Records to update in the RRSet.

class Zone(*id: int | None = None*, *name: str | None = None*, *created: str | None = None*, *mode: Literal['primary', 'secondary'] | None = None*, *ttl: int | None = None*, *labels: dict[str, str] | None = None*, *protection: ZoneProtection | None = None*, *status: Literal['ok', 'updating', 'error'] | None = None*, *record_count: int | None = None*, *registrar: Literal['hetzner', 'other', 'unknown'] | None = None*, *primary_nameservers: list[ZonePrimaryNameserver] | None = None*, *authoritative_nameservers: ZoneAuthoritativeNameservers | None = None*)

Zone Domain.

See https://docs.hetzner.cloud/reference/cloud#zones.

MODE_PRIMARY = 'primary'

Zone in primary mode, resource record sets (RRSets) and resource records (RRs) are managed via the Cloud API or Cloud Console.

MODE_SECONDARY = 'secondary'

Zone in secondary mode, Hetzner's nameservers query RRSets and RRs from given primary nameservers via AXFR.

STATUS_ERROR = 'error'

The Zone could not be published to the authoritative nameservers.

STATUS_OK = 'ok'

The Zone is pushed to the authoritative nameservers.

STATUS_UPDATING = 'updating'

The Zone is currently being published to the authoritative nameservers.

class ZoneAuthoritativeNameservers(*assigned: list[str] | None = None*, *delegated: list[str] | None = None*, *delegation_last_check: str | None = None*, *delegation_status: Literal['valid', 'partially-valid', 'invalid', 'lame', 'unregistered', 'unknown'] | None = None*)

Zone Authoritative Nameservers Domain.

class ZonePrimaryNameserver(*address: str*, *port: int | None = None*, *tsig_algorithm: Literal['hmac-md5', 'hmac-sha1', 'hmac-sha256'] | None = None*, *tsig_key: str | None = None*)

Zone Primary Nameserver Domain.

TSIG_ALGORITHM_HMAC_MD5 = 'hmac-md5'

Transaction signature (TSIG) algorithm used to generate the TSIG key.

TSIG_ALGORITHM_HMAC_SHA1 = 'hmac-sha1'

Transaction signature (TSIG) algorithm used to generate the TSIG key.

TSIG_ALGORITHM_HMAC_SHA256 = 'hmac-sha256'

Transaction signature (TSIG) algorithm used to generate the TSIG key.

to_payload() → dict[str, Any]

Generates the request payload from this domain object.

class ZoneRecord(*value: str*, *comment: str | None = None*)

Zone Record Domain.

to_payload() → dict[str, Any]

Generates the request payload from this domain object.

class ZoneRRSet(*name: str | None = None*, *type: ZoneRRSetType | None = None*, *ttl: int | None = None*, *labels: dict[str, str] | None = None*, *protection: ZoneRRSetProtection | None = None*, *records: list[ZoneRecord] | None = None*, *id: str | None = None*, *zone:* BoundZone | Zone | *None = None*)

Zone RRSet Domain.

See https://docs.hetzner.cloud/reference/cloud#zone-rrsets

to_payload() → dict[str, Any]

Generates the request payload from this domain object.

class CreateZoneResponse(*zone:* BoundZone, *action:* BoundAction)

Create Zone Response Domain.

4.3 Exceptions

class HCloudException

There was an error while using the hcloud library.

All exceptions in the hcloud library inherit from this exception. It may be used as catch-all exception.

class APIException(*code: int | str*, *message: str*, *details: Any*, ***, *correlation_id: str | None = None*)

There was an error while performing an API Request.

class ActionException(*action:* Action | BoundAction)

A generic action exception

class ActionFailedException(*action:* Action | BoundAction)

The pending action failed

class ActionTimeoutException(*action:* Action | BoundAction)

The pending action timed out

4.4 Other

4.4.1 Helpers

class LabelValidator

static validate(*labels: dict[str, str]*) → bool

Validates Labels. If you want to know which key/value pair of the dict is not correctly formatted use `validate_verbose()`.

Returns
bool

static validate_verbose(*labels: dict[str, str]*) → tuple[bool, str]

Validates Labels and returns the corresponding error message if something is wrong. Returns True, <empty string> if everything is fine.

Returns
bool, str

4.4.2 Deprecation Info

class DeprecationInfo(*announced: str | None = None*, *unavailable_after: str | None = None*)

Describes if, when & how the resources was deprecated. If this field is set to `None` the resource is not deprecated. If it has a value, it is considered deprecated.

Parameters

- **announced** – datetime Date of when the deprecation was announced.
- **unavailable_after** – datetime After the time in this field, the resource will not be available from the general listing endpoint of the resource type, and it can not be used in new resources. For example, if this is an image, you can not create new servers with this image after the mentioned date.

CHAPTER

FIVE

CONTRIBUTING

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at https://github.com/hetznercloud/hcloud-python/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it.

5.1.4 Write Documentation

Hetzner Cloud Python could always use more documentation, whether as part of the official Hetzner Cloud Python docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/hetznercloud/hcloud-python/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.2 Get Started!

Ready to contribute? Here's how to set up `hcloud-python` for local development.

1. Fork the `hcloud-python` repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/hcloud-python.git
```

3. Read the `Development` section in the `README.md`, to setup your development environment.
4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

5. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

6. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.md.
3. The pull request should work for all the versions of Python the library supports, and for PyPy.

CHAPTER

SIX

UPGRADING

This project adheres to Semantic Versioning.

Before upgrading, make sure to resolve any deprecation warnings.

6.1 Upgrading to v2

- #397: The package version was moved from `hcloud.__version__.VERSION` to `hcloud.__version__`, make sure to update your import paths:

```
-from hcloud.__version__ import VERSION
+from hcloud import __version__ as VERSION
```

- #401: The deprecated `hcloud.hcloud` module was removed, make sure to update your import paths:

```
-from hcloud.hcloud import Client
+from hcloud import Client
```

- #398: The `Client.poll_interval` property is now private, make sure to configure it while creating the `Client`:

```
-client = Client(token=token)
-client.poll_interval = 2
+client = Client(
+    token=token,
+    poll_interval=2,
+)
```

- #400: The `Client.request` method now returns an empty dict instead of an empty string when the API response is empty:

```
 response = client.request(method="DELETE", url="/primary_ips/123456")
-assert response == ""
+assert response == {}
```

- #402: In the `Client.isos.get_list` and `Client.isos.get_all` methods, the deprecated `include_wildcard_architecture` argument was removed, make sure to use the `include_architecture_wildcard` argument instead:

```
 client.isos.get_all(
-    include_wildcard_architecture=True,
+    include_architecture_wildcard=True,
 )
```

- #363: In the `Client.primary_ips.create` method, the `datacenter` argument was moved after `name` argument and is now optional:

```
 client.primary_ips.create(
    "ipv4",
-   None,
    "my-ip",
    assignee_id=12345,
 )
```

```
 client.primary_ips.create(
    "ipv4",
-   Datacenter(name="fsn1-dc14"),
    "my-ip",
+   datacenter=Datacenter(name="fsn1-dc14"),
 )
```

- #406: In the `Client.servers.rebuild` method, the single action return value was deprecated and is now removed. The method now returns a full response wrapping the action and an optional root password:

```
-action = client.servers.rebuild(server, image)
+resp = client.servers.rebuild(server, image)
+action = resp.action
+root_password = resp.root_password
```

CHAPTER

SEVEN

CHANGELOG

7.1 v2.20.0

7.1.1 Load Balancer HTTP Services now support `timeout_idle`

HTTP Services now support the field `timeout_idle`, which controls the time a HTTP connection is allowed to idle before it is being dropped.

See the changelog for more information.

7.1.2 Features

- **load-balancer**: support `timeout_idle` http service field (#649)

7.2 v2.19.0

7.2.1 Primary IPs `assignee_type` behavior change

In the create Primary IP call, the `assignee_type` argument is now only send when the `assignee_id` argument is set. The `assignee_type` argument will stop defaulting to `server` in the near future, consider explicitly setting this argument when needed.

As of 1 August 2026, the behavior of the Primary IP `assignee_type` property will change, and will return `unassigned` when the Primary IP is not assigned (when `assignee_id` is `null`). The goal is to eventually assign Primary IPs to other resource types, not only to `server`.

See the changelog for more details.

In addition, the Primary IP request body `assignee_type` property of the operation `POST /v1/primary_ips` is now optional. Primary IPs created without `assignee_type` return `server` until 1 August 2026, after this date, its value will be `unassigned`.

See the changelog for more details.

7.2.2 Features

- **primary-ip**: `assignee_type` behavior changed when creating a primary ip (#647)

7.3 v2.18.0

7.3.1 Available and recommended Server Types have been moved

`Datacenter.server_types` has been deprecated in favor of the new `ServerType.locations[].available` and `ServerType.locations[].recommended` properties.

See the changelog for more details.

7.3.2 Features

- **datacenter, server_type**: move available and recommended to server_type (#645)

7.4 v2.17.1

7.4.1 Bug Fixes

- missing `__api_properties__` on LoadBalancerService (#639)

7.5 v2.17.0

7.5.1 Features

- parse nested load balancer `label_selector` targets (#633)

7.6 v2.16.0

7.6.1 Storage Boxes support is now generally available

The experimental phase for Storage Boxes is over, and Storage Boxes support is now generally available.

7.6.2 Features

- **servers**: allow setting user_data for rebuild (#627)
- Storage Box support no longer experimental (#626)

7.7 v2.15.0

7.7.1 Features

- add name to Storage Box Subaccount (#621)

7.8 v2.14.0

7.8.1 Features

- retry requests when the api returns a `timeout` error (#617)

7.9 v2.13.0

7.9.1 Features

- add per primary ip actions list operations (#608)
- deprecate datacenter in `primary ips` and `servers` (#609)

7.10 v2.12.0

7.10.1 Storage Box API Experimental

This release adds support for the Storage Box API.

The Storage Box integration will be introduced as an **experimental** feature. This experimental phase is expected to last at least until **12 January 2026**. During this period, upcoming minor releases of the project may include breaking changes to features related to Storage Boxes.

This release includes all changes from the recent Storage Box API changelog entry.

7.10.1.1 Examples

```
response = client.storage_boxes.create(
    name="string",
    location=Location(name="fsn1"),
    storage_box_type=StorageBoxType(name="bx11"),
    labels={
        "environment": "prod",
        "example.com/my": "label",
        "just-a-key": "",
    },
    password="my-password",
    access_settings=StorageBoxAccessSettings(
        reachable_externally=False,
        samba_enabled=False,
        ssh_enabled=False,
        webdav_enabled=False,
        zfs_enabled=False,
    ),
    ssh_keys=[SSHKey(public_key="ssh-rsa AAAjjk76kgf...Xt")],
)

response.action.wait_until_finished()

storage_box = response.storage_box
```

7.10.2 Features

- add update rrset records action to zone client (#597)
- add support for Storage Boxes (#524)

7.11 v2.11.1

7.11.1 Bug Fixes

- support reloading sub resource bound models (#590)

7.12 v2.11.0

7.12.1 DNS API is now generally available

The DNS API is now generally available, as well as support for features in this project that are related to the DNS API.

To migrate existing zones to the new DNS API, see the DNS migration guide.

See the changelog for more details.

7.12.2 Features

- DNS support is now generally available (#581)

7.13 v2.10.0

7.13.1 Features

- **exp**: add zone format txt record helper (#578)
- add server and load balancer `private_net_for` helper method (#580)

7.14 v2.9.0

7.14.1 Features

- support python 3.14 (#566)
- drop support for python 3.9 (#574)

7.15 v2.8.0

7.15.1 DNS API Beta

This release adds support for the new DNS API.

The DNS API is currently in **beta**, which will likely end on 10 November 2025. After the beta ended, it will no longer be possible to create new zones in the old DNS system. See the DNS Beta FAQ for more details.

Future minor releases of this project may include breaking changes for features that are related to the DNS API.

See the DNS API Beta changelog for more details.

Examples

```
resp = client.zones.create(
    name="example.com",
    mode="primary",
    labels={"key": "value"},
    rrsets=[
        ZoneRRSet(
            name="@",
            type="A",
            records=[
                ZoneRecord(value="201.180.75.2", comment="server1")
            ],
```

(continues on next page)

(continued from previous page)

```
        )
    ],
)

resp.action.wait_until_finished()
zone = resp.zone
```

7.15.2 Features

- add new `ip_range` param to load balancer `attach_to_network` (#562)
- add new `ip_range` param to server `attach_to_network` (#561)
- support the new DNS API (#568)

7.15.3 Bug Fixes

- source_ips property is optional in firewall rule (#567)

7.16 v2.7.0

Server Types now depend on Locations.

- We added a new `locations` property to the Server Types resource. The new property defines a list of supported Locations and additional per Locations details such as deprecations information.
- We deprecated the `deprecation` property from the Server Types resource. The property will gradually be phased out as per Locations deprecations are being announced. Please use the new per Locations deprecation information instead.

See our changelog for more details.

Upgrading

```
# Before
def validate_server_type(server_type: ServerType):
    if server_type.deprecation is not None:
        raise ValueError(f"server type {server_type.name} is deprecated")
```

```
# After
def validate_server_type(server_type: ServerType, location: Location):
    found = [o for o in server_type.locations if location.name == o.location.name]
    if not found:
        raise ValueError(
            f"server type {server_type.name} is not supported in location {location.name}
↪"
        )

    server_type_location = found[0]

    if server_type_location.deprecation is not None:
        raise ValueError(
            f"server type {server_type.name} is deprecated in location {location.name}"
        )
```

7.16.1 Features

- per location server types (#558)

7.17 v2.6.0

7.17.1 Features

- add category property to server type (#549)

7.17.2 Bug Fixes

- rename `ClientEntityBase` to `ResourceClientBase` (#532)

7.18 v2.5.4

7.18.1 Bug Fixes

- typo in `LoadBalancerHealthCheckHttp` class name (#511)
- equality for some domain classes (#510)
- use valid license identifier (SPDX) (#514)

7.19 v2.5.3

7.19.1 Bug Fixes

- invalid placement group id casting (#501)
- handle string id when checking has_id_or_name (#504)

7.20 v2.5.2

7.20.1 Bug Fixes

- listing page result always provide meta (#496)

7.21 v2.5.1

7.21.1 Bug Fixes

- missing slots and api_properties for FirewallResourceLabelSelector (#492)

7.22 v2.5.0

7.22.1 Features

- improve exception messages (#488)

7.23 v2.4.0

7.23.1 Features

- drop support for python 3.8 (#458)
- add equality checks to domains (#481)

7.23.2 Bug Fixes

- server public ipv4 and ipv6 properties are nullable (#455)

7.24 2.3.0 (2024-10-09)

7.24.1 Features

- support python 3.13 (#451) (4a514c7)

7.24.2 Bug Fixes

- change floating ip labels type to `dict[str, str]` (#444) (1f6da4e)

7.25 2.2.1 (2024-08-19)

7.25.1 Bug Fixes

- prices properties are list of dict (#438) (9621604), closes #437

7.26 2.2.0 (2024-08-06)

7.26.1 Features

- retry requests when the api gateway errors (#430) (f63ac8b)
- retry requests when the api returns a conflict error (#430) (f63ac8b)
- retry requests when the network timed outs (#430) (f63ac8b)
- retry requests when the rate limit was reached (#430) (f63ac8b)

7.26.2 Bug Fixes

- update network subnet types (#431) (c32a615)

7.27 2.1.1 (2024-07-30)

7.27.1 Bug Fixes

- do not sleep before checking for the reloaded action status (#426) (3e0a85b)
- mark client retry backoff function as static (#429) (14ed130)

7.27.2 Documentation

- add api changes note in changelog (#424) (5cbe188)

7.28 2.1.0 (2024-07-25)

7.28.1 API Changes for Traffic Prices and Server Type Included Traffic

There will be a breaking change in the API regarding Traffic Prices and Server Type Included Traffic on 2024-08-05. This release marks the affected fields as `Deprecated`. Please check if this affects any of your code and switch to the replacement fields where necessary.

You can learn more about this change in our changelog.

7.28.2 Features

- add exponential and constant backoff function (#416) (fe7ddf6)
- deprecate `ServerType` `included_traffic` property (#423) (3d56ac5)
- use exponential backoff when retrying requests (#417) (f306073)

7.29 2.0.1 (2024-07-03)

7.29.1 Bug Fixes

- `assignee_type` is required when creating a primary ip (#409) (bce5e94)
- clean unused arguments in the `Client.servers.rebuild` method (#407) (6d33c3c)
- details are optional in API errors (#411) (f1c6594)
- rename `trace_id` variable to `correlation_id` (#408) (66a0f54)

7.30 2.0.0 (2024-07-03)

7.30.1 BREAKING CHANGES

- return full rebuild response in `Client.servers.rebuild` (#406)
- make `datacenter` argument optional when creating a primary ip (#363)
- remove deprecated `include_wildcard_architecture` argument in `IsosClient.get_list` and `IsosClient.get_all` (#402)
- make `Client.request` `tries` a private argument (#399)
- make `Client.poll_interval` a private property (#398)
- return empty dict on empty responses in `Client.request` (#400)
- remove deprecated `hcloud.hcloud` module (#401)
- move `hcloud.__version__.VERSION` to `hcloud.__version__` (#397)

7.30.2 Features

- add `trace_id` to API exceptions (#404) (8375261)
- allow using a custom poll_interval function (#403) (93eb56b)
- make `Client.poll_interval` a private property (#398) (d5f24db)
- make `Client.request` `tries` a private argument (#399) (428ea7e)
- move `hcloud.__version__.VERSION` to `hcloud.__version__` (#397) (4e3f638), closes #234
- remove deprecated `hcloud.hcloud` module (#401) (db37e63)
- remove deprecated `include_wildcard_architecture` argument in `IsosClient.get_list` and `IsosClient.get_all` (#402) (6b977e2)
- return empty dict on empty responses in `Client.request` (#400) (9f46adb)
- return full rebuild response in `Client.servers.rebuild` (#406) (1970d84)

7.30.3 Bug Fixes

- make `datacenter` argument optional when creating a primary ip (#363) (ebef774)

7.30.4 Dependencies

- update dependency coverage to >=7.5,<7.6 (#386) (5660691)
- update dependency mypy to >=1.10,<1.11 (#387) (35c933b)
- update dependency myst-parser to v3 (#385) (9f18270)
- update dependency pylint to >=3,<3.3 (#391) (4a6f005)
- update dependency pytest to >=8,<8.3 (#390) (584a36b)
- update dependency sphinx to >=7.3.4,<7.4 (#383) (69c2e16)
- update pre-commit hook asottile/pyupgrade to v3.16.0 (0ce5fbc)
- update pre-commit hook pre-commit/pre-commit-hooks to v4.6.0 (5ef25ab)
- update pre-commit hook psf/black-pre-commit-mirror to v24.4.0 (0941fbf)
- update pre-commit hook psf/black-pre-commit-mirror to v24.4.1 (fec08c5)
- update pre-commit hook psf/black-pre-commit-mirror to v24.4.2 (#389) (2b2e21f)
- update pre-commit hook pycqa/flake8 to v7.1.0 (3bc651d)

7.30.5 Documentation

- add v2 upgrade notes (#405) (c77f771)
- cx11 is name, not an id (#381) (b745d40)

7.31 1.35.0 (2024-04-02)

7.31.1 Features

- add `include_deprecated` option when fetching images by name (#375) (6d86f86)

7.31.2 Bug Fixes

- raise warnings for the `ImagesClient.get_by_name` deprecation (#376) (b24de80)

7.32 1.34.0 (2024-03-27)

7.32.1 Features

- add `has_id_or_name` to `DomainIdentityMixin` (#373) (8facaf6)

7.33 1.33.3 (2024-03-27)

7.33.1 Bug Fixes

- invalid type for load balancer private network property (#372) (903e92f)

7.33.2 Dependencies

- update codecov/codecov-action action to v4 (#359) (a798979)
- update dependency mypy to >=1.9,<1.10 (#368) (4b9328c)
- update dependency pylint to >=3,<3.2 (#364) (d71d17f)
- update dependency pytest to >=8,<8.2 (#366) (8665dcf)
- update dependency pytest to v8 (#357) (f8f756f)
- update dependency pytest-cov to v5 (#371) (04a6a42)
- update dependency watchdog to v4 (#360) (cb8d383)
- update pre-commit hook asottile/pyupgrade to v3.15.1 (#362) (dd2a521)
- update pre-commit hook asottile/pyupgrade to v3.15.2 (3d02ad7)
- update pre-commit hook psf/black-pre-commit-mirror to v24 (#356) (b46397d)
- update pre-commit hook psf/black-pre-commit-mirror to v24.1.1 (#358) (7e4645e)
- update pre-commit hook psf/black-pre-commit-mirror to v24.2.0 (#361) (5b56ace)
- update pre-commit hook psf/black-pre-commit-mirror to v24.3.0 (3bbac5d)
- update pre-commit hook pycqa/flake8 to v7 (#354) (66a582f)
- update pypa/gh-action-pypi-publish action to v1.8.12 (#365) (55db255)
- update pypa/gh-action-pypi-publish action to v1.8.14 (#367) (0cb615f)

7.34 1.33.2 (2024-01-02)

7.34.1 Bug Fixes

- publish package to PyPI using OIDC auth (1a0e93b)

7.35 1.33.1 (2024-01-02)

7.35.1 Bug Fixes

- private object not exported in top level module (#346) (5281b05)

7.35.2 Dependencies

- update dependency coverage to >=7.4,<7.5 (#348) (3ac5711)
- update dependency mypy to >=1.8,<1.9 (#343) (984022f)
- update pre-commit hook psf/black-pre-commit-mirror to v23.12.1 (#347) (2c24efe)

7.36 1.33.0 (2023-12-19)

7.36.1 Features

- add metrics endpoint for load balancers and servers (#331) (ee3c54f)

7.36.2 Bug Fixes

- fallback to error code when message is unset (#328) (1c94153)

7.36.3 Dependencies

- update actions/setup-python action to v5 (#335) (2ac252d)
- update dependency sphinx-rtd-theme to v2 (#330) (7cc4335)
- update pre-commit hook psf/black-pre-commit-mirror to v23.12.0 (#338) (38e4748)
- update pre-commit hook pycqa/isort to v5.13.0 (#336) (3244cfe)
- update pre-commit hook pycqa/isort to v5.13.1 (#337) (020a0ef)
- update pre-commit hook pycqa/isort to v5.13.2 (#339) (b46df8c)

7.37 1.32.0 (2023-11-17)

7.37.1 Features

- allow returning root_password in servers rebuild (#276) (38e098a)

7.37.2 Dependencies

- update dependency mypy to >=1.7,<1.8 (#325) (7b59a2d)
- update pre-commit hook pre-commit/mirrors-prettier to v3.1.0 (#326) (213b661)
- update pre-commit hook psf/black-pre-commit-mirror to v23.10.1 (#322) (999afe3)
- update pre-commit hook psf/black-pre-commit-mirror to v23.11.0 (#324) (7b2a24e)

7.38 1.31.0 (2023-10-23)

7.38.1 Features

- prepare for iso deprecated field removal (#320) (beae328)

7.38.2 Dependencies

- update pre-commit hook psf/black-pre-commit-mirror to v23.10.0 (#319) (184bbe6)

7.39 1.30.0 (2023-10-13)

7.39.1 Features

- add deprecation field to Iso (#318) (036b52f)
- support python 3.12 (#311) (7e8cd1d)

7.39.2 Dependencies

- update dependency mypy to >=1.6,<1.7 (#317) (d248bbd)
- update dependency pylint to v3 (#307) (277841d)
- update pre-commit hook asottile/pyupgrade to v3.14.0 (#308) (07a4513)
- update pre-commit hook asottile/pyupgrade to v3.15.0 (#312) (c544639)
- update pre-commit hook pre-commit/pre-commit-hooks to v4.5.0 (#313) (e51eaa9)
- update python docker tag to v3.12 (#309) (3a1ee67)

7.40 1.29.1 (2023-09-26)

7.40.1 Bug Fixes

- prevent api calls when printing bound models (#305) (c1de7ef)

7.41 1.29.0 (2023-09-25)

7.41.1 Features

- add domain attribute type hints to bound models (#300) (6d46d06)
- **firewalls:** add `applied_to_resources` to `FirewallResource` (#297) (55d2b20)

7.41.2 Bug Fixes

- missing BaseDomain base class inheritance (#303) (0ee7598)

7.41.3 Dependencies

- update actions/checkout action to v4 (#295) (c02b446)
- update dependency sphinx to >=7.2.2,<7.3 (#291) (10234ea)
- update dependency sphinx to v7 (#211) (f635c94)

- update pre-commit hook asottile/pyupgrade to v3.11.0 (#298) (4bbd0cc)
- update pre-commit hook asottile/pyupgrade to v3.11.1 (#299) (2f9fcd7)
- update pre-commit hook asottile/pyupgrade to v3.13.0 (#301) (951dbf3)
- update pre-commit hook pre-commit/mirrors-prettier to v3.0.3 (#294) (381e336)
- update pre-commit hook psf/black to v23.9.1 (#296) (4374a7b)

7.41.4 Documentation

- load token from env in examples scripts (#302) (f18c9a6)

7.42 1.28.0 (2023-08-17)

7.42.1 Features

- add load balancer target health status field (#288) (5780418)
- implement resource actions clients (#252) (4bb9a97)

7.42.2 Dependencies

- update dependency coverage to >=7.3,<7.4 (#286) (a4df4fa)
- update dependency mypy to >=1.5,<1.6 (#284) (9dd5c81)
- update pre-commit hook pre-commit/mirrors-prettier to v3.0.2 (#287) (6bf03cb)

7.42.3 Documentation

- fail on warning (#289) (e61300e)

7.43 1.27.2 (2023-08-09)

7.43.1 Documentation

- fix python references (#281) (0c0518e)

7.44 1.27.1 (2023-08-08)

7.44.1 Bug Fixes

- missing long_description content_type in setup.py (#279) (6d79d1d)

7.45 1.27.0 (2023-08-08)

7.45.1 Features

- add global request timeout option (#271) (07a663f)
- reexport references in parent ressources modules (#256) (854c12b)
- the package is now typed (#265) (da8baa5)

7.45.2 Bug Fixes

- allow omitting `datacenter` when creating a primary ip (#171) (4375dc6)
- ineffective doc strings (#266) (bb34df9)
- invalid attribute in placement group (#258) (23b3607)

7.45.3 Dependencies

- update pre-commit hook asottile/pyupgrade to v3.10.1 (#261) (efa5780)
- update pre-commit hook pre-commit/mirrors-prettier to v3.0.1 (#269) (2239b0b)
- update pre-commit hook pycqa/flake8 to v6.1.0 (#260) (fd01384)

7.45.4 Documentation

- update documentation (#247) (e63741f)
- update hetzner logo (#264) (ee79851)

7.46 1.26.0 (2023-07-19)

7.46.1 Features

- add **repr** method to domains (#246) (4c22765)
- drop support for python 3.7 (#242) (2ce71e9)

7.47 1.25.0 (2023-07-14)

7.47.1 Features

- add details to raise exceptions (#240) (cf64e54)
- move hcloud.hcloud module to hcloud._client (#243) (413472d)

7.47.2 Dependencies

- update pre-commit hook asottile/pyupgrade to v3.9.0 (#238) (0053ded)
- update pre-commit hook pre-commit/mirrors-prettier to v3 (#235) (047d4e1)
- update pre-commit hook psf/black to v23.7.0 (#239) (443bf26)

7.48 1.24.0 (2023-07-03)

7.48.1 Features

- revert remove python-dateutil dependency (#231) (945bfde), closes #226

7.48.2 Dependencies

- update pre-commit hook asottile/pyupgrade to v3.8.0 (#232) (27f21bc)

7.49 1.23.1 (2023-06-30)

7.49.1 Bug Fixes

- handle Z timezone in ISO8601 datetime format (#228) (6a5c3f4), closes #226

7.50 1.23.0 (2023-06-26)

7.50.1 Features

- remove python-dateutil dependency (#221) (8ea4aa0)

7.50.2 Bug Fixes

- **isos:** invalid name for include_wildcard_architecture argument (#222) (c3dfcab)

7.50.3 Dependencies

- update dependency pytest to >=7.4,<7.5 (#217) (11e1f45)

7.51 1.22.0 (2023-06-22)

7.51.1 Features

- adhere to PEP 517 (#213) (7a19add)
- bump required python version to >=3.7 (#198) (62d89f9)
- **network:** add field expose_routes_to_vswitch (#208) (5321182)
- setup exception hierarchy (#199) (8466645)

7.51.2 Dependencies

- update actions/setup-python action to v4 (#209) (aeee575)
- update actions/stale action to v8 (#210) (cb13230)
- update pre-commit hook asottile/pyupgrade to v3.7.0 (#205) (c46c5a4)

7.52 1.21.0 (2023-06-19)

7.52.1 Features

- add deprecation field to ServerType (#192) (4a0fce7)

7.52.2 Bug Fixes

- adjust label validation for max length of 63 characters (#194) (3cba96d)

7.52.3 Documentation

- improve branding, design & fix warnings (#191) (47eb9f1)
- use venv for the dev setup (#196) (93f48ff)

7.53 1.20.0 (2023-05-12)

7.53.1 Features

- **server_type:** add field for included traffic (#185) (8ae0bc6)

7.54 v1.19.0 (2023-04-12)

- docs: link to PrivateNet broken by @apricote in #177
- feat: add support for ARM APIs by @apricote in #182

7.55 v1.18.2 (2022-12-27)

- fix: remove unused future dependency by @apricote in #173
- chore: update tests to use released python-3.11 by @apricote in #175
- chore: prepare release 1.18.2 by @apricote in #174

7.56 v1.18.1 (2022-10-25)

- Update Github Actions by @LKaemmerling in #165
- Add tests for Python 3.11 by @LKaemmerling in #167

7.57 v1.18.0 (2022-08-17)

- Remove use of external mock module by @s-t-e-v-e-n-k in #162
- document installation path via conda-forge by @s-m-e in #149
- Drop # – coding: utf-8 – from files by @jonasdlindner in #154
- Simplify Requirement Constraints by @LKaemmerling in #163
- Add validation helper for Label Values/Keys by @LKaemmerling in #164

7.58 v1.17.0 (2022-06-29)

- Add primary IP support by @LKaemmerling in #160

7.59 v1.16.0 (2021-08-17)

- Feature: Add support for Load Balancer DNS PTRs

7.60 v1.15.0 (2021-08-16)

- Feature: Add support for Placement Groups

7.61 v1.14.1 (2021-08-10)

- Bugfix: Fix crash on extra fields in public_net response
- Improvement: Format code with black

7.62 v1.14.0 (2021-08-03)

- Feature: Add support for Firewall rule descriptions

7.63 v1.13.0 (2021-07-16)

- Feature: Add support for Firewall Protocols ESP and GRE
- Feature: Add support for Image Type APP
- Feature: Add support for creating Firewalls with Firewalls
- Feature: Add support for Label Selectors in Firewalls
- Improvement: Improve handling of underlying TCP connections. Now for every client instance a single TCP connection is used instead of one per call.
- Note: Support for Python 2.7 and Python 3.5 was removed

7.64 v1.12.0 (2021-04-06)

- Feature: Add support for managed Certificates

7.65 v1.11.0 (2021-03-11)

- Feature: Add support for Firewalls
- Feature: Add `primary_disk_size` to `Server` Domain

7.66 v1.10.0 (2020-11-03)

- Feature: Add `include_deprecated` filter to `get_list` and `get_all` on `ImagesClient`
- Feature: Add vSwitch support to `add_subnet` on `NetworksClient`
- Feature: Add subnet type constants to `NetworkSubnet` domain (`NetworkSubnet.TYPE_CLOUD`, `NetworkSubnet.TYPE_VSWITCH`)

7.67 v1.9.1 (2020-08-11)

- Bugfix: BoundLoadBalancer serialization failed when using IP targets

7.68 v1.9.0 (2020-08-10)

- Feature: Add `included_traffic`, `outgoing_traffic` and `ingoing_traffic` properties to Load Balancer domain
- Feature: Add `change_type`-method to `LoadBalancersClient`

- Feature: Add support for `LoadBalancerTargetLabelSelector`
- Feature: Add support for `LoadBalancerTargetLabelSelector`

7.69 v1.8.2 (2020-07-20)

- Fix: Loosen up the requirements.

7.70 v1.8.1 (2020-06-29)

- Fix Load Balancer Client.
- Fix: Unify setting of request parameters within `get_list` methods.

7.71 1.8.0 (2020-06-22)

- Feature: Add Load Balancers **Attention: The Load Balancer support in v1.8.0 is kind of broken. Please use v1.8.1**
- Feature: Add Certificates

7.72 1.7.1 (2020-06-15)

- Feature: Add requests 2.23 support

7.73 1.7.0 (2020-06-05)

- Feature: Add support for the optional 'networks' parameter on server creation.
- Feature: Add python 3.9 support
- Feature: Add subnet type `cloud`

7.74 1.6.3 (2020-01-09)

- Feature: Add 'created' property to SSH Key domain
- Fix: Remove ISODatetime Descriptor because it leads to wrong dates

7.75 1.6.2 (2019-10-15)

- Fix: future dependency requirement was too strict

7.76 1.6.1 (2019-10-01)

- Fix: python-dateutil dependency requirement was too strict

7.77 1.6.0 (2019-09-17)

- Feature: Add missing `get_by_name` on `FloatingIPsClient`

7.78 1.5.0 (2019-09-16)

- Fix: ServersClient.create_image fails when specifying the `labels`
- Feature: Add support for `name` on Floating IPs

7.79 1.4.1 (2019-08-19)

- Fix: Documentation for `NetworkRoute` domain was missing
- Fix: `requests` dependency requirement was to strict

7.80 1.4.0 (2019-07-29)

- Feature: Add `mac_address` to Server PrivateNet domain
- Feature: Add python 3.8 support

7.81 1.3.0 (2019-07-10)

- Feature: Add status filter for servers, images and volumes
- Feature: Add 'created' property to Floating IP domain
- Feature: Add 'Networks' support

7.82 1.2.1 (2019-03-13)

- Fix: BoundVolume.server server property now casted to the 'BoundServer'.

7.83 1.2.0 (2019-03-06)

- Feature: Add `get_by_fingerprint`-method for ssh keys
- Fix: Create Floating IP with location raises an error because no action was given.

7.84 1.1.0 (2019-02-27)

- Feature: Add `STATUS`-constants for server and volume status

7.85 1.0.1 (2019-02-22)

Fix: Ignore unknown fields in API response instead of raising an error

7.86 1.0.0 (2019-02-21)

- First stable release.

  You can find the documentation under https://hcloud-python.readthedocs.io/en/stable/

7.87 0.1.0 (2018-12-20)

- First release on GitHub.

CHAPTER

EIGHT

HETZNER CLOUD PYTHON

Official Hetzner Cloud python library.

The library's documentation is available at hcloud-python.readthedocs.io, the public API documentation is available at docs.hetzner.cloud.

Important

Make sure to follow our API changelog available at docs.hetzner.cloud/changelog (or the RSS feed available at docs.hetzner.cloud/changelog/feed.rss) to be notified about additions, deprecations and removals.

8.1 Usage

Install the `hcloud` library:

```
pip install hcloud
```

For more installation details, please see the installation docs.

Here is an example that creates a server and list them:

```
from hcloud import Client
from hcloud.images import Image
from hcloud.server_types import ServerType

client = Client(
    token="{YOUR_API_TOKEN}", # Please paste your API token here
    application_name="my-app",
    application_version="v1.0.0",
)

# Create a server named my-server
response = client.servers.create(
    name="my-server",
    server_type=ServerType(name="cx23"),
    image=Image(name="ubuntu-22.04"),
)
server = response.server
print(f"{server.id=} {server.name=} {server.status=}")
```

(continues on next page)

(continued from previous page)

```
print(f"root password: {response.root_password}")

# List your servers
servers = client.servers.get_all()
for server in servers:
    print(f"{server.id=} {server.name=} {server.status=}")
```

- To upgrade the package, please read the instructions available in the documentation.
- For more details on the API, please see the API reference.
- You can find some more examples under the `examples/` directory.

8.2 Supported Python versions

We support python versions until `end-of-life`.

8.3 Experimental features

Experimental features are published as part of our regular releases (e.g. a product public beta). During an experimental phase, breaking changes on those features may occur within minor releases.

The stability of experimental features is not related to the stability of its upstream API.

Experimental features have different levels of maturity (e.g. experimental, alpha, beta) based on the maturity of the upstream API.

While experimental features will be announced in the release notes, you can also find whether a python class or function is experimental in its docstring:

```
Experimental:
    $PRODUCT is $MATURITY, breaking changes may occur within minor releases.
    See https://docs.hetzner.cloud/changelog#$SLUG for more details.
```

8.4 Development

First, create a virtual environment and activate it:

```
make venv
source venv/bin/activate
```

You may setup `pre-commit` to run before you commit changes, this removes the need to run it manually afterwards:

```
pre-commit install
```

You can then run different tasks defined in the `Makefile`, below are the most important ones:

Build the documentation and open it in your browser:

```
make docs
```

Lint the code:

```
make lint
```

Run tests using the current `python3` version:

```
make test
```

You may also run the tests for multiple `python3` versions using `tox`:

```
tox .
```

8.4.1 Deprecations implementation

When deprecating a module or a function, you must:

- Update the docstring with a `deprecated` notice:

```
"""Get image by name

.. deprecated:: 1.19
    Use :func:`hcloud.images.client.ImagesClient.get_by_name_and_architecture` instead.
"""
```

- Raise a warning when the deprecated module or function is being used:

```
warnings.warn(
    "The 'hcloud.images.client.ImagesClient.get_by_name' method is deprecated, please␣
↪use the "
    "'hcloud.images.client.ImagesClient.get_by_name_and_architecture' method instead.",
    DeprecationWarning,
    stacklevel=2,
)
```

8.4.2 Releasing experimental features

To publish experimental features as part of regular releases:

- an announcement, including a link to a changelog entry, must be added to the release notes.
- an `Experimental` notice, including a link to a changelog entry, must be added to the python classes and functions that are experimental:

```
"""
Experimental:
    $PRODUCT is $MATURITY, breaking changes may occur within minor releases.
    See https://docs.hetzner.cloud/changelog#$SLUG for more details.
"""
```

8.5 License

The MIT License (MIT). Please see `License File` for more information.

INDEX

C

D

E

F

G

H

I

L

M

N

P

R

S

T

U

V

W

Z