OVH Provider

The OVH provider is used to interact with the many resources supported by OVH. The provider needs to be configured with the proper credentials before it can be used.

Use the navigation to the left to read about the available resources.

Configuration of the provider

Requests to OVH APIs need to configure secrets keys in the provider, either fetching them from `~/.ovh.conf` file, in configuration of OVH provider or from your environment.

It is recommend to install ovh-cli (https://github.com/ovh/ovh-cli) to handle and manage all your secret keys.

Follow installation (https://github.com/ovh/ovh-cli#installation) then setup (https://github.com/ovh/ovh-cli#getting-started) steps of ovh-cli to initialize your environment (secret keys and `~/.ovh.conf` file).

Then, you can just declare a minimal configuration of the OVH provider:

```
# Configure the OVH Provider
provider "ovh" {
  endpoint = "ovh-eu"
}
```

Secret keys `endpoint`, `application_key`, `application_secret` or `consumer_key` will be fetched from the `~/.ovh.conf` file.

Or you can declare them in provider configuration:

```
# Configure the OVH Provider
provider "ovh" {
  endpoint           = "ovh-eu"
  application_key    = "yyyyyy"
  application_secret = "xxxxxxxxxxxxxx"
  consumer_key       = "zzzzzzzzzzzzzz"
}
```

Or let the provider fetching them from your environment (see "Configuration reference").

Example Usage

```
# Create a public cloud user
resource "ovh_publiccloud_user" "user-test" {
  # ...
}
```

Configuration Reference

The following arguments are supported:

- `endpoint` - (Required) Specify which API endpoint to use. It can be set using the `OVH_ENDPOINT` environment variable. e.g. `ovh-eu` or `ovh-ca`.
- `application_key` - (Optional) The API Application Key. If omitted, the `OVH_APPLICATION_KEY` environment variable is used.
- `application_secret` - (Optional) The API Application Secret. If omitted, the `OVH_APPLICATION_SECRET` environment variable is used.
- `consumer_key` - (Optional) The API Consumer key. If omitted, the `OVH_CONSUMER_KEY` environment variable is used.

Testing and Development

In order to run the Acceptance Tests for development, the following environment variables must also be set:

- `OVH_ENDPOINT` - possible value are: `ovh-eu`, `ovh-ca`, `ovh-us`, `soyoustart-eu`, `soyoustart-ca`, `kimsufi-ca`, `kimsufi-eu`, `runabove-ca`
- `OVH_IPLB_SERVICE` - The ID of the IP Load Balancer to use
- `OVH_VRACK` - The ID of the vRack to use.
- `OVH_PUBLIC_CLOUD` - The ID of your public cloud project.
- `OVH_ZONE` - The domain you own to test the domain_zone resource.

You will also need to generate an OVH token (https://api.ovh.com/createToken/?GET=/*&POST=/*&PUT=/*&DELETE=/*) and use it to set the following environment variables:

- `OVH_APPLICATION_KEY`
- `OVH_APPLICATION_SECRET`
- `OVH_CONSUMER_KEY`

You should be able to use any OVH environment to develop on as long as the above environment variables are set.

ovh_cloud_region

Use this data source to retrieve information about a region associated with a public cloud project. The region must be associated with the project.

Example Usage

```
data "ovh_cloud_region" "GRA1" {
   project_id = "XXXXXX"
   name = "GRA1"
}
```

Argument Reference

- `project_id` - (Required) The id of the public cloud project. If omitted, the `OVH_PROJECT_ID` environment variable is used.
- `name` - (Required) The name of the region associated with the public cloud project.

Attributes Reference

`id` is set to the ID of the project concatenated with the name of the region. In addition, the following attributes are exported:

- `continent_code` - the code of the geographic continent the region is running. E.g.: EU for Europe, US for America...
- `datacenter_location` - The location code of the datacenter. E.g.: "GRA", meaning Gravelines, for region "GRA1"
- `continentCode` - (Deprecated) Use `continent_code` instead.
- `datacenterLocation` - (Deprecated) Use `datacenter_location` instead.
- `services` - The list of public cloud services running within the region
  - `name` - the name of the public cloud service
  - `status` - the status of the service

ovh_cloud_regions

Use this data source to get the regions of a public cloud project.

Example Usage

```
data "ovh_cloud_regions" "regions" {
  project_id = "XXXXXX"
}
```

Argument Reference

- `project_id` - (Required) The id of the public cloud project. If omitted, the `OVH_PROJECT_ID` environment variable is used.

Attributes Reference

`id` is set to the ID of the project. In addition, the following attributes are exported:

- `names` - The list of regions associated with the project

ovh_domain_zone

Use this data source to retrieve information about a domain zone.

Example Usage

```
data "ovh_domain_zone" "rootzone" {
    name = "mysite.ovh"
}
```

Argument Reference

- `name` - (Required) The name of the domain zone.

Attributes Reference

`id` is set to the domain zone name. In addition, the following attributes are exported:

- `last_update` - Last update date of the DNS zone
- `has_dns_anycast` - hasDnsAnycast flag of the DNS zone
- `name_servers` - Name servers that host the DNS zone
- `dnssec_supported` - Is DNSSEC supported by this zone

ovh_iploadbalancing

Use this data source to retrieve information about an IP Load Balancing product

Example Usage

```
data "ovh_iploadbalancing" "lb" {
   service_name = "xxx"
   state        = "ok"
}
```

Argument Reference

- `ipv6` - The IPV6 associated to your IP load balancing
- `ipv4` - The IPV4 associated to your IP load balancing
- `zone` - Location where your service is. This takes an array of values.
- `offer` - The offer of your IP load balancing
- `service_name` - The internal name of your IP load balancing
- `ip_loadbalancing` - Your IP load balancing
- `state` - Current state of your IP. Can take any of the following value: "blacklisted", "deleted", "free", "ok", "quarantined", "suspended"
- `vrack_eligibility` - Vrack eligibility. Takes a boolean value.
- `vrack_name` - Name of the vRack on which the current Load Balancer is attached to, as it is named on vRack product
- `ssl_configuration` - Modern oldest compatible clients : Firefox 27, Chrome 30, IE 11 on Windows 7, Edge, Opera 17, Safari 9, Android 5.0, and Java 8. Intermediate oldest compatible clients : Firefox 1, Chrome 1, IE 7, Opera 5, Safari 1, Windows XP IE8, Android 2.3, Java 7. Can take any of the following value: "intermediate", "modern"
- `display_name` - the name displayed in ManagerV6 for your iplb (max 50 chars)

Attributes Reference

`id` is set to the service_name of your IP load balancing In addition, the following attributes are exported:

- `metrics_token` - The metrics token associated with your IP load balancing This attribute is sensitive.
- `orderable_zone` - Available additional zone for your Load Balancer
  - `name` - The zone three letter code
  - `plan_code` - The billing planCode for this zone

ovh_me_paymentmean_bankaccount

Use this data source to retrieve information about a bank account payment mean associated with an OVH account.

Example Usage

```
data "ovh_me_paymentmean_bankaccount" "ba" {
   use_default = true
}
```

Argument Reference

- `description_regexp` - (Optional) a regexp used to filter bank accounts on their `description` attributes.
- `use_default` - (Optional) Retrieve bank account marked as default payment mean.
- `use_oldest` - (Optional) Retrieve oldest bank account. project.
- `state` - (Optional) Filter bank accounts on their `state` attribute. Can be "blockedForIncidents", "valid", "pendingValidation"

Attributes Reference

`id` is set to the ID of the bank account payment mean

- `description` - the description attribute of the bank account
- `default` - a boolean which tells if the retrieved bank account is marked as the default payment mean

ovh_me_paymentmean_creditcard

Use this data source to retrieve information about a credit card payment mean associated with an OVH account.

Example Usage

```
data "ovh_me_paymentmean_creditcard" "cc" {
   use_default = true
}
```

Argument Reference

- `description_regexp` - (Optional) a regexp used to filter credit cards on their `description` attributes.
- `use_default` - (Optional) Retrieve credit card marked as default payment mean.
- `use_last_to_expire` - (Optional) Retrieve the credit card that will be the last to expire according to its expiration date.
- `states` - (Optional) Filter credit cards on their `state` attribute. Can be "expired", "valid", "tooManyFailures"

Attributes Reference

`id` is set to the ID of the credit card payment mean

- `description` - the description attribute of the credit card
- `state` - the state attribute of the credit card
- `default` - a boolean which tells if the retrieved credit card is marked as the default payment mean

ovh_publiccloud_region

DEPRECATED: Use `ovh_cloud_region` (/docs/providers/ovh/d/cloud_region.html) instead.

Use this data source to retrieve information about a region associated with a public cloud project. The region must be associated with the project.

Example Usage

```
data "ovh_publiccloud_region" "GRA1" {
   project_id = "XXXXXX"
   region = "GRA1"
}
```

Argument Reference

- `project_id` - (Required) The id of the public cloud project. If omitted, the `OVH_PROJECT_ID` environment variable is used.
- `region` - (Required) The name of the region associated with the public cloud project.

Attributes Reference

`id` is set to the ID of the project concatenated with the name of the region. In addition, the following attributes are exported:

- `continent_code` - the code of the geographic continent the region is running. E.g.: EU for Europe, US for America...
- `datacenter_location` - The location code of the datacenter. E.g.: "GRA", meaning Gravelines, for region "GRA1"
- `continentCode` - (Deprecated) Use `continent_code` instead.
- `datacenterLocation` - (Deprecated) Use `datacenter_location` instead.
- `services` - The list of public cloud services running within the region
  - `name` - the name of the public cloud service
  - `status` - the status of the service

ovh_publiccloud_regions

DEPRECATED: Use `ovh_cloud_regions` (/docs/providers/ovh/d/cloud_regions.html) instead.

Use this data source to get the regions of a public cloud project.

Example Usage

```
data "ovh_publiccloud_regions" "regions" {
  project_id = "XXXXXX"
}
```

Argument Reference

- `project_id` - (Required) The id of the public cloud project. If omitted, the `OVH_PROJECT_ID` environment variable is used.

Attributes Reference

`id` is set to the ID of the project. In addition, the following attributes are exported:

- `names` - The list of regions associated with the project

ovh_cloud_network_private

Creates a private network in a public cloud project.

Example Usage

```
resource "ovh_cloud_network_private" "net" {
   project_id = "67890"
   name       = "admin_network"
   regions    = ["GRA1", "BHS1"]
}
```

Argument Reference

The following arguments are supported:

- `project_id` - (Required) The id of the public cloud project. If omitted, the `OVH_PROJECT_ID` environment variable is used.
- `name` - (Required) The name of the network.
- `vlan_id` - a vlan id to associate with the network. Changing this value recreates the resource. Defaults to 0.
- `regions` - an array of valid OVH public cloud region ID in which the network will be available. Ex.: "GRA1". Defaults to all public cloud regions.

Attributes Reference

The following attributes are exported:

- `project_id` - See Argument Reference above.
- `name` - See Argument Reference above.
- `vlan_id` - See Argument Reference above.
- `regions` - See Argument Reference above.
- `regions_status` - A map representing the status of the network per region.
- `regions_status/region` - The id of the region.
- `regions_status/status` - The status of the network in the region.
- `status` - the status of the network. should be normally set to 'ACTIVE'.
- `type` - the type of the network. Either 'private' or 'public'.

ovh_cloud_network_private_subnet

Creates a subnet in a private network of a public cloud project.

Example Usage

```
resource "ovh_cloud_network_private_subnet" "subnet" {
   project_id = "67890"
   network_id = "0234543"
   region     = "GRA1"
   start      = "192.168.168.100"
   end        = "192.168.168.200"
   network    = "192.168.168.0/24"
   dhcp       = true
   no_gateway = false
}
```

Argument Reference

The following arguments are supported:

- `project_id` - (Required) The id of the public cloud project. If omitted, the `OVH_PROJECT_ID` environment variable is used. Changing this forces a new resource to be created.
- `network_id` - (Required) The id of the network. Changing this forces a new resource to be created.
- `dhcp` - (Optional) Enable DHCP. Changing this forces a new resource to be created. Defaults to false. _
- `start` - (Required) First ip for this region. Changing this value recreates the subnet.
- `end` - (Required) Last ip for this region. Changing this value recreates the subnet.
- `network` - (Required) Global network in CIDR format. Changing this value recreates the subnet
- `region` - The region in which the network subnet will be created. Ex.: "GRA1". Changing this value recreates the resource.
- `no_gateway` - Set to true if you don't want to set a default gateway IP. Changing this value recreates the resource. Defaults to false.

Attributes Reference

The following attributes are exported:

- `project_id` - See Argument Reference above.
- `network_id` - See Argument Reference above.
- `dhcp_id` - See Argument Reference above.

- `start` - See Argument Reference above.
- `end` - See Argument Reference above.
- `network` - See Argument Reference above.
- `region` - See Argument Reference above.
- `gateway_ip` - The IP of the gateway
- `no_gateway` - See Argument Reference above.
- `cidr` - Ip Block representing the subnet cidr.
- `ip_pools` - List of ip pools allocated in the subnet.
- `ip_pools/network` - Global network with cidr.
- `ip_pools/region` - Region where this subnet is created.
- `ip_pools/dhcp` - DHCP enabled.
- `ip_pools/end` - Last ip for this region.
- `ip_pools/start` - First ip for this region.

ovh_cloud_user

Creates a user in a public cloud project.

Example Usage

```
resource "ovh_cloud_user" "user1" {
   project_id = "67890"
}
```

Argument Reference

The following arguments are supported:

- `project_id` - (Required) The id of the public cloud project. If omitted, the `OVH_PROJECT_ID` environment variable is used.
- `description` - A description associated with the user.

Attributes Reference

The following attributes are exported:

- `project_id` - See Argument Reference above.
- `description` - See Argument Reference above.
- `username` - the username generated for the user. This username can be used with the Openstack API.
- `password` - (Sensitive) the password generated for the user. The password can be used with the Openstack API. This attribute is sensitive and will only be retrieve once during creation.
- `status` - the status of the user. should be normally set to 'ok'.
- `creation_date` - the date the user was created.
- `openstack_rc` - a convenient map representing an openstack_rc file. Note: no password nor sensitive token is set in this map.

ovh_iploadbalancing_http_route

Manage http route for a loadbalancer service

Example Usage

Route which redirect all url to https.

```
resource "ovh_iploadbalancing_http_route" "httpsredirect" {
  service_name = "loadbalancer-xxxxxxxxxxxxxxxxx"
  display_name = "Redirect to HTTPS"
  weight = 1

  action {
    status = 302
    target = "https://${host}${path}${arguments}"
    type = "redirect"
  }
}
```

Argument Reference

The following arguments are supported:

- `service_name` - (Required) The internal name of your IP load balancing
- `display_name` - Human readable name for your route, this field is for you
- `weight` - Route priority ([0..255]). 0 if null. Highest priority routes are evaluated first. Only the first matching route will trigger an action
- `action.status` - HTTP status code for "redirect" and "reject" actions
- `action.target` - Farm ID for "farm" action type or URL template for "redirect" action. You may use ${uri}, ${protocol}, ${host}, ${port} and ${path} variables in redirect target
- `action.type` - (Required) Action to trigger if all the rules of this route matches
- `frontend_id` - Route traffic for this frontend

Attributes Reference

The following attributes are exported:

- `service_name` - See Argument Reference above.
- `display_name` - See Argument Reference above.
- `weight` - See Argument Reference above.

- `action.status` - See Argument Reference above.
- `action.target` - See Argument Reference above.
- `action.type` - See Argument Reference above.
- `frontend_id` - See Argument Reference above.

ovh_iploadbalancing_http_route_rule

Manage rules for HTTP route.

Example Usage

Route which redirect all url to https for example.com (Vhost).

```
resource "ovh_iploadbalancing_http_route" "httpsredirect" {
  service_name = "loadbalancer-xxxxxxxxxxxxxxxxx"
  display_name = "Redirect to HTTPS"
  weight       = 1
  frontend_id  = 11111

  action {
    status = 302
    target = "https://$${host}$${path}$${arguments}"
    type = "redirect"
  }
}

resource "ovh_iploadbalancing_http_route_rule" "examplerule" {
  service_name = "loadbalancer-xxxxxxxxxxxxxxxxx"
  route_id     = "${ovh_iploadbalancing_http_route.httpsredirect.id}"
  display_name = "Match example.com host"
  field        = "host"
  match        = "is"
  negate       = false
  pattern      = "example.com"
}
```

Rule which match a specific header (same effect as the host match above).

```
resource "ovh_iploadbalancing_http_route_rule" "examplerule" {
  service_name = "loadbalancer-xxxxxxxxxxxxxxxxx"
  route_id     = "${ovh_iploadbalancing_http_route.httpsredirect.id}"
  display_name = "Match example.com Host header"
  field        = "headers"
  match        = "is"
  negate       = false
  pattern      = "example.com"
  sub_field    = "Host"
}
```

Argument Reference

The following arguments are supported:

- `service_name` - (Required) The internal name of your IP load balancing

- `route_id` - (Required) The route to apply this rule
- `display_name` - Human readable name for your rule, this field is for you
- `field` - (Required) Name of the field to match like "protocol" or "host". See "/ipLoadbalancing/{serviceName}/availableRouteRules" for a list of available rules
- `match` - (Required) Matching operator. Not all operators are available for all fields. See "/ipLoadbalancing/{serviceName}/availableRouteRules"
- `negate` - Invert the matching operator effect
- `pattern` - Value to match against this match. Interpretation if this field depends on the match and field
- `sub_field` - Name of sub-field, if applicable. This may be a Cookie or Header name for instance

Attributes Reference

The following attributes are exported:

- `service_name` - See Argument Reference above.
- `route_id` - See Argument Reference above.
- `display_name` - See Argument Reference above.
- `field` - See Argument Reference above.
- `match` - See Argument Reference above.
- `negate` - See Argument Reference above.
- `pattern` - See Argument Reference above.
- `sub_field` - See Argument Reference above.

ovh_iploadbalancing_refresh

Applies changes from other ovh_iploadbalancing_* resourcesto the production configuration of loadbalancers.

Example Usage

```
data "ovh_iploadbalancing" "lb" {
  service_name = "ip-1.2.3.4"
   state       = "ok"
}

resource "ovh_iploadbalancing_tcp_farm" "farmname" {
  service_name = "${data.ovh_iploadbalancing.lb.id}"
  port = 8080
  zone = "all"
}

resource "ovh_iploadbalancing_tcp_farm_server" "backend" {
  service_name           = "${data.ovh_iploadbalancing.lb.id}"
  farm_id                = "${ovh_iploadbalancing_tcp_farm.farmname.id}"
  display_name           = "mybackend"
  address                = "4.5.6.7"
  status                 = "active"
  port                   = 80
  proxy_protocol_version = v2
  weight                 = 2
  probe                  = true
  ssl                    = false
  backup                 = true
}

resource "ovh_iploadbalancing_refresh" "mylb" {
  service_name = "${data.ovh_iploadbalancing.lb.id}"
  keepers = [
    "${ovh_iploadbalancing_tcp_farm_server.backend.*.address}",
    ]
}
```

Argument Reference

The following arguments are supported:

- `service_name` - (Required) The internal name of your IP load balancing
- `keepers` - List of values traccked to trigger refresh, used also to form implicit dependencies

Attributes Reference

The following attributes are exported:

- `service_name` - See Argument Reference above.
- `keepers` - See Argument Reference above.

ovh_iploadbalancing_tcp_farm

Creates a backend server group (farm) to be used by loadbalancing frontend(s)

Example Usage

```
data "ovh_iploadbalancing" "lb" {
  service_name = "ip-1.2.3.4"
   state       = "ok"
}

resource "ovh_iploadbalancing_tcp_farm" "farmname" {
  service_name = "${data.ovh_iploadbalancing.lb.id}"
  display_name = "ingress-8080-gra"
  zone = "GRA"
}
```

Argument Reference

The following arguments are supported:

- `service_name` - (Required) The internal name of your IP load balancing
- `balance` - Load balancing algorithm. `roundrobin` if null (`first`, `leastconn`, `roundrobin`, `source`)
- `display_name` - Readable label for loadbalancer farm
- `port` - Port attached to your farm ([1..49151]). Inherited from frontend if null
- `stickiness` - Stickiness type. No stickiness if null (`sourceIp`)
- `vrack_network_id` - Internal Load Balancer identifier of the vRack private network to attach to your farm, mandatory when your Load Balancer is attached to a vRack
- `zone` - (Required) Zone where the farm will be defined (ie. `GRA`, `BHS` also supports `ALL`)
- `probe` - define a backend healthcheck probe
  - `type` - (Required) Valid values : `http`, `internal`, `mysql`, `oko`, `pgsql`, `smtp`, `tcp`
  - `interval` - probe interval, Value between 30 and 3600 seconds, default 30
  - `match` - What to mach `pattern` against (`contains`, `default`, `internal`, `matches`, `status`)
  - `port` - Port for backends to recieve traffic on.
  - `negate` - Negate probe result
  - `pattern` - Pattern to match against `match`
  - `force_ssl` - Force use of SSL (TLS)
  - `url` - URL for HTTP probe type.

    - `method` - HTTP probe method (`GET` , `HEAD` , `OPTIONS` , `internal`)

Attributes Reference

The following attributes are exported:

- `service_name` - See Argument Reference above.
- `balance` - See Argument Reference above.
- `display_name` - See Argument Reference above.
- `port` - See Argument Reference above.
- `stickiness` - See Argument Reference above.
- `vrack_network_id` - See Argument Reference above.
- `zone` - See Argument Reference above.
- `probe` - See Argument Reference above.
    - `type` - See Argument Reference above.
    - `interval` - See Argument Reference above.
    - `match` - See Argument Reference above.
    - `port` - See Argument Reference above.
    - `negate` - See Argument Reference above.
    - `pattern` - See Argument Reference above.
    - `force_ssl` - See Argument Reference above.
    - `url` - See Argument Reference above.
    - `method` - See Argument Reference above.

ovh_iploadbalancing_tcp_farm_server

Creates a backend server entry linked to loadbalancing group (farm)

Example Usage

```
data "ovh_iploadbalancing" "lb" {
  service_name = "ip-1.2.3.4"
   state       = "ok"
}

resource "ovh_iploadbalancing_tcp_farm" "farmname" {
  service_name = "${data.ovh_iploadbalancing.lb.id}"
  port = 8080
  zone = "all"
}

resource "ovh_iploadbalancing_tcp_farm_server" "backend" {
  service_name           = "${data.ovh_iploadbalancing.lb.id}"
  farm_id                = "${ovh_iploadbalancing_tcp_farm.farmname.id}"
  display_name           = "mybackend"
  address                = "4.5.6.7"
  status                 = "active"
  port                   = 80
  proxy_protocol_version = v2
  weight                 = 2
  probe                  = true
  ssl                    = false
  backup                 = true
}
```

Argument Reference

The following arguments are supported:

- `service_name` - (Required) The internal name of your IP load balancing
- `farm_id` - ID of the farm this server is attached to
- `display_name` - Label for the server
- `address` - Address of the backend server (IP from either internal or OVH network)
- `status` - backend status - `active` or `inactive`
- `port` - Port that backend will respond on
- `proxy_protocol_version` - version of the PROXY protocol used to pass origin connection information from loadbalancer to recieving service (`v1` , `v2` , `v2-ssl` , `v2-ssl-cn`)
- `weight` - used in loadbalancing algorithm

- `probe` - defines if backend will be probed to determine health and keep as active in farm if healthy
- `ssl` - is the connection ciphered with SSL (TLS)
- `backup` - is it a backup server used in case of failure of all the non-backup backends

Attributes Reference

The following attributes are exported:

- `service_name` - See Argument Reference above.
- `farm_id` - See Argument Reference above.
- `display_name` - See Argument Reference above.
- `address` - See Argument Reference above.
- `status` - See Argument Reference above.
- `port` - See Argument Reference above.
- `proxy_protocol_version` - See Argument Reference above.
- `weight` - See Argument Reference above.
- `probe` - See Argument Reference above.
- `ssl` - See Argument Reference above.
- `backup` - See Argument Reference above.
- `cookie` - Value of the stickiness cookie used for this backend.

ovh_iploadbalancing_tcp_frontend

Creates a backend server group (frontend) to be used by loadbalancing frontend(s)

Example Usage

```
data "ovh_iploadbalancing" "lb" {
  service_name = "ip-1.2.3.4"
  state       = "ok"
}

resource "ovh_iploadbalancing_tcp_farm" "farm80" {
   service_name = "${data.ovh_iploadbalancing.lb.service_name}"
   display_name = "ingress-8080-gra"
   zone = "all"
   port = 80
}

resource "ovh_iploadbalancing_tcp_frontend" "testfrontend" {
   service_name = "${data.ovh_iploadbalancing.lb.service_name}"
   display_name = "ingress-8080-gra"
   zone = "all"
   port = "80,443"
   default_farm_id = "${ovh_iploadbalancing_tcp_farm.farm80.id}"
}
```

Argument Reference

The following arguments are supported:

- `service_name` - (Required) The internal name of your IP load balancing
- `display_name` - Human readable name for your frontend, this field is for you
- `port` - Port(s) attached to your frontend. Supports single port (numerical value), range (2 dash-delimited increasing ports) and comma-separated list of 'single port' and/or 'range'. Each port must be in the [1;49151] range
- `zone` - (Required) Zone where the frontend will be defined (ie. `gra`, `bhs` also supports `all`)
- `allowed_source` - Restrict IP Load Balancing access to these ip block. No restriction if null. List of IP blocks.
- `dedicated_ipfo` - Only attach frontend on these ip. No restriction if null. List of Ip blocks.
- `default_farm_id` - Default TCP Farm of your frontend
- `default_ssl_id` - Default ssl served to your customer
- `disabled` - Disable your frontend. Default: 'false'
- `ssl` - SSL deciphering. Default: 'false'

Attributes Reference

The following attributes are exported:

- `id` - Id of your frontend
- `display_name` - See Argument Reference above.
- `allowed_source` - See Argument Reference above.
- `dedicated_ipfo` - See Argument Reference above.
- `default_farm_id` - See Argument Reference above.
- `default_ssl_id` - See Argument Reference above.
- `disabled` - See Argument Reference above.
- `ssl` - See Argument Reference above.

ovh_ip_reverse

Provides a OVH IP reverse.

Example Usage

```
# Set the reverse of an IP
resource "ovh_ip_reverse" "test" {
    ip = "192.0.2.0/24"
    ipreverse = "192.0.2.1"
    reverse = "example.com"
}
```

Argument Reference

The following arguments are supported:

- `ip` - (Required) The IP block to which the IP belongs
- `reverse` - (Required) The value of the reverse
- `ipreverse` - (Optional) The IP to set the reverse of, default to `ip` if `ip` is a /32 (IPv4) or a /128 (IPv6)

Attributes Reference

The following attributes are exported:

- `ipreverse` - The IP to set the reverse of
- `reverse` - The value of the reverse

ovh_domain_zone_record

Provides a OVH domain zone record.

Example Usage

```
# Add a record to a sub-domain
resource "ovh_domain_zone_record" "test" {
    zone = "testdemo.ovh"
    subdomain = "test"
    fieldtype = "A"
    ttl = "3600"
    target = "0.0.0.0"
}
```

Argument Reference

The following arguments are supported:

- `zone` - (Required) The domain to add the record to
- `subdomain` - (Required) The name of the record
- `target` - (Required) The value of the record
- `fieldtype` - (Required) The type of the record
- `ttl` - (Optional) The TTL of the record

Attributes Reference

The following attributes are exported:

- `id` - The record ID
- `zone` - The domain to add the record to
- `subDomain` - The name of the record
- `target` - The value of the record
- `fieldType` - The type of the record
- `ttl` - The TTL of the record

Import

OVH record can be imported using the `id` and the `zone` , eg:

```
$ terraform import ovh_domain_zone_record.test 1234OVH_ID.zone.tld
```

ovh_domain_zone_redirection

Provides a OVH domain zone redirection.

Example Usage

```
# Add a redirection to a sub-domain
resource "ovh_domain_zone_redirection" "test" {
    zone = "testdemo.ovh"
    subdomain = "test"
    type = "visiblePermanent"
    target = "http://www.ovh"
}
```

Argument Reference

The following arguments are supported:

- `zone` - (Required) The domain to add the redirection to
- `subdomain` - (Optional) The name of the redirection
- `target` - (Required) The value of the redirection
- `type` - (Required) The type of the redirection, with values:
  - `visible` -> Redirection by http code 302
  - `visiblePermanent` -> Redirection by http code 301
  - `invisible` -> Redirection by html frame
- `description` - (Optional) A description of this redirection
- `keywords` - (Optional) Keywords to describe this redirection
- `title` - (Optional) Title of this redirection

Attributes Reference

The following attributes are exported:

- `id` - The redirection ID
- `zone` - The domain to add the redirection to
- `subDomain` - The name of the redirection
- `target` - The value of the redirection
- `type` - The type of the redirection

- `description` - The description of the redirection
- `keywords` - Keywords of the redirection
- `title` - The title of the redirection

ovh_publiccloud_private_network

DEPRECATED: Use `ovh_cloud_network_private` (/docs/providers/ovh/r/cloud_network_private.html) instead.

Creates a private network in a public cloud project.

Example Usage

```
resource "ovh_publiccloud_private_network" "net" {
   project_id = "67890"
   name       = "admin_network"
   regions    = ["GRA1", "BHS1"]
}
```

Argument Reference

The following arguments are supported:

- `project_id` - (Required) The id of the public cloud project. If omitted, the `OVH_PROJECT_ID` environment variable is used.
- `name` - (Required) The name of the network.
- `vlan_id` - a vlan id to associate with the network. Changing this value recreates the resource. Defaults to 0.
- `regions` - an array of valid OVH public cloud region ID in which the network will be available. Ex.: "GRA1". Defaults to all public cloud regions.

Attributes Reference

The following attributes are exported:

- `project_id` - See Argument Reference above.
- `name` - See Argument Reference above.
- `vlan_id` - See Argument Reference above.
- `regions` - See Argument Reference above.
- `regions_status` - A map representing the status of the network per region.
- `regions_status/region` - The id of the region.
- `regions_status/status` - The status of the network in the region.
- `status` - the status of the network. should be normally set to 'ACTIVE'.

- `type` - the type of the network. Either 'private' or 'public'.

ovh_publiccloud_private_network_subnet

DEPRECATED: Use `ovh_cloud_network_private_subnet` (/docs/providers/ovh/r/cloud_network_private_subnet.html) instead.

Creates a subnet in a private network of a public cloud project.

Example Usage

```
resource "ovh_publiccloud_private_network_subnet" "subnet" {
   project_id = "67890"
   network_id = "0234543"
   region     = "GRA1"
   start      = "192.168.168.100"
   end        = "192.168.168.200"
   network    = "192.168.168.0/24"
   dhcp       = true
   no_gateway = false
}
```

Argument Reference

The following arguments are supported:

- `project_id` - (Required) The id of the public cloud project. If omitted, the `OVH_PROJECT_ID` environment variable is used. Changing this forces a new resource to be created.
- `network_id` - (Required) The id of the network. Changing this forces a new resource to be created.
- `dhcp` - (Optional) Enable DHCP. Changing this forces a new resource to be created. Defaults to false. _
- `start` - (Required) First ip for this region. Changing this value recreates the subnet.
- `end` - (Required) Last ip for this region. Changing this value recreates the subnet.
- `network` - (Required) Global network in CIDR format. Changing this value recreates the subnet
- `region` - The region in which the network subnet will be created. Ex.: "GRA1". Changing this value recreates the resource.
- `no_gateway` - Set to true if you don't want to set a default gateway IP. Changing this value recreates the resource. Defaults to false.

Attributes Reference

The following attributes are exported:

- `project_id` - See Argument Reference above.
- `network_id` - See Argument Reference above.
- `dhcp_id` - See Argument Reference above.
- `start` - See Argument Reference above.
- `end` - See Argument Reference above.
- `network` - See Argument Reference above.
- `region` - See Argument Reference above.
- `gateway_ip` - The IP of the gateway
- `no_gateway` - See Argument Reference above.
- `cidr` - Ip Block representing the subnet cidr.
- `ip_pools` - List of ip pools allocated in the subnet.
- `ip_pools/network` - Global network with cidr.
- `ip_pools/region` - Region where this subnet is created.
- `ip_pools/dhcp` - DHCP enabled.
- `ip_pools/end` - Last ip for this region.
- `ip_pools/start` - First ip for this region.

ovh_publiccloud_user

DEPRECATED: Use `ovh_cloud_user` (/docs/providers/ovh/r/cloud_user.html) instead.

Creates a user in a public cloud project.

Example Usage

```
resource "ovh_publiccloud_user" "user1" {
   project_id = "67890"
}
```

Argument Reference

The following arguments are supported:

- `project_id` - (Required) The id of the public cloud project. If omitted, the `OVH_PROJECT_ID` environment variable is used.
- `description` - A description associated with the user.

Attributes Reference

The following attributes are exported:

- `project_id` - See Argument Reference above.
- `description` - See Argument Reference above.
- `username` - the username generated for the user. This username can be used with the Openstack API.
- `password` - (Sensitive) the password generated for the user. The password can be used with the Openstack API. This attribute is sensitive and will only be retrieve once during creation.
- `status` - the status of the user. should be normally set to 'ok'.
- `creation_date` - the date the user was created.
- `openstack_rc` - a convenient map representing an openstack_rc file. Note: no password nor sensitive token is set in this map.

ovh_vrack_cloudproject

Attach an existing public cloud project to an existing VRack.

Example Usage

```
resource "ovh_vrack_cloudproject" "attach" {
  vrack_id   = "12345"
  project_id = "67890"
}
```

Argument Reference

The following arguments are supported:

- `vrack_id` - (Required) The id of the vrack. If omitted, the `OVH_VRACK_ID` environment variable is used.
- `project_id` - (Required) The id of the public cloud project. If omitted, the `OVH_PROJECT_ID` environment variable is used.

Attributes Reference

The following attributes are exported:

- `vrack_id` - See Argument Reference above.
- `project_id` - See Argument Reference above.

Notes

The vrack attachment isn't a proper resource with an ID. As such, the resource id will be forged from the vrack and project ids and there's no correct way to import the resource in terraform. When the resource is created by terraform, it first checks if the attachment already exists within OVH infrastructure; if it exists it set the resource id without modifying anything. Otherwise, it will try to attach the vrack with the public cloud project.

ovh_vrack_publiccloud_attachment

DEPRECATED: Use `ovh_vrack_cloudproject` (/docs/providers/ovh/r/vrack_cloudproject.html) instead.

Attach an existing PublicCloud project to an existing VRack.

Example Usage

```
resource "ovh_vrack_publiccloud_attachment" "attach" {
  vrack_id   = "12345"
  project_id = "67890"
}
```

Argument Reference

The following arguments are supported:

- `vrack_id` - (Required) The id of the vrack. If omitted, the `OVH_VRACK_ID` environment variable is used.
- `project_id` - (Required) The id of the public cloud project. If omitted, the `OVH_PROJECT_ID` environment variable is used.

Attributes Reference

The following attributes are exported:

- `vrack_id` - See Argument Reference above.
- `project_id` - See Argument Reference above.

Notes

The vrack attachment isn't a proper resource with an ID. As such, the resource id will be forged from the vrack and project ids and there's no correct way to import the resource in terraform. When the resource is created by terraform, it first checks if the attachment already exists within OVH infrastructure; if it exists it set the resource id without modifying anything. Otherwise, it will try to attach the vrack with the public cloud project.