

vultr_load_balancer

Manages a Vultr Load Balancer instance for distributing traffic across multiple servers with configurable protocols, algorithms, and health checks.

Contents

01	Introduction	3
02	Example Usage	3
03	Argument Reference	3
04	Attributes Reference	4

vultr_load_balancer

Introduction

Get information about a Vultr load balancer.

Example Usage

Get the information for a load balancer by `label` :

HCL

```
data "vultr_load_balancer" "my_lb" {
  filter {
    name    = "label"
    values = ["my-lb-label"]
  }
}
```

Argument Reference

The following arguments are supported:

- `filter` - (Required) Query parameters for finding load balancers.

The `filter` block supports the following:

- `name` - Attribute name to filter with.
- `values` - One or more values filter with.

Attributes Reference

The following attributes are exported:

- `region` - The region your load balancer is deployed in.
- `label` - The load balancers label.
- `balancing_algorithm` - The balancing algorithm for your load balancer.
- `proxy_protocol` - Boolean value that indicates if Proxy Protocol is enabled.
- `cookie_name` - Name for your given sticky session.
- `ssl_redirect` - Boolean value that indicates if HTTP calls will be redirected to HTTPS.
- `has_ssl` - Boolean value that indicates if SSL is enabled.
- `attached_instances` - Array of instances that are currently attached to the load balancer.
- `status` - Current status for the load balancer
- `ipv4` - IPv4 address for your load balancer.
- `ipv6` - IPv6 address for your load balancer.
- `health_check` - Defines the way load balancers should check for health. The configuration of a `health_check` is listed below.
- `forwarding_rules` - Defines the forwarding rules for a load balancer. The configuration of a `forwarding_rules` is listened below.
- `private_network` - (Deprecated: use `vpc` instead) Defines the private network the load balancer is attached to.
- `vpc` - Defines the VPCthe load balancer is attached to.

`health_check` supports the following

- `protocol` - The protocol used to traffic requests to the load balancer. Possible values are "http", "https", or "tcp".
- `path` - The path on the attached instances that the load balancer should check against.
- `port` - The assigned port (integer) on the attached instances that the load balancer should check against.
- `check_interval` - Time in seconds to perform health check. Default value is 15.

- `response_timeout` - Time in seconds to wait for a health check response. Default value is 5.
- `unhealthy_threshold` - Number of failed attempts encountered before failover. Default value is 5.
- `healthy_threshold` - Number of failed attempts encountered before failover. Default value is 5.

`forwarding_rules` supports the following

- `frontend_protocol` - Protocol on load balancer side. Possible values: "http", "https", "tcp".
- `frontend_port` - Port on load balancer side.
- `backend_protocol` - Protocol on instance side. Possible values: "http", "https", "tcp".
- `target_port` - Port on instance side.

`firewall_rules` supports the following

- `frontend_port` - (Required) Port on load balancer side.
- `ip_type` - (Required) The type of ip this rule is - may be either v4 or v6.
- `source` - (Required) IP address with subnet that is allowed through the firewall. You may also pass in `cloudflare` which will allow only CloudFlares IP range.



VULTR

