API

- Load balancers

Load balancers

Load Balancers

RackCorp Load Balancers are designed to be easy to use for automation, while also being globally relevant. By selecting the scope, they can be deployed both globally using public IPs, or can act locally acting as a LAN-based load balancer.

cmd style:

loadbalancer.create
loadbalancer.update
loadbalancer.delete
loadbalancer.get
loadbalancer.getall

REST style:

POST /loadbalancer/
PUT /loadbalancer/{id}
DELETE /loadbalancer/{id}
GET /loadbalancer/{id}
GET /loadbalancer/

> **Load Balancer Types:**
> **GLB** - A geographically spread set of HTTP Load Balancers which can connect to public IP address backends. This provides fast TLS handshakes for customers, Anti-DDoS, and edge WAF and caching capabilities.
> **HTTP** - A single-region, redundant HTTP/HTTPS Load Balancer solution. Can connect to public or private IP backends.
> **TCP** - A geographically spread set of TCP load balancers. Can connect to public or private IP backends.

> **SSL / TLS Certificates**
> **GLB** and **HTTP** types automatically handle TLS certificates within 30 seconds of deployment.

GLB / HTTP / TCP Load Balancer:

HTTP Load Balancers are useful for public and private services where TLS certificates are handled automatically, and multiple backends can be used for failover.

TCP Load Balancers are useful for public and private services where only TCP-layer is required.

HTTP and TCP Load Balancers can also be deployed with direct LAN access, bridging the public and private network layers by way of application proxy.

```
{
  "id": __LBID__,
  "cmd": "loadbalancer.XXXXXXXX",
  "name": "Test GLB",
  "type": "GLB",
  "hostname": "XXXXXX.glb.XXXXXXXXX.com",
  "checkmode": "TCP",
  "aliases": [
    "glbtest1.frontendhostname.com",
    "glbtest2.frontendhostname.com"
  ],
  "backends": [
    {
      "hostname": "www.hostnameoriptoconnectto.com",
      "tcpproxy": 2,
      "weight": 100,
      "port": 443,
      "tls": true
    }
  ]
}
```

id:

Required for: update, delete, get
Integer: The ID of the load balancer instance, as returned by a create, update, get, getall operation

cmd:

Only required for cmd style, otherwise auto-determined via REST style

name:

String: Name, useful for customer to understand in GUI / billing

type:

String: GLB, HTTP, TCP

hostname: *(Returned by API only)*

String: Hostname for customer to point their DNS to as a CNAME. Effective immediately upon creation. Regional static global IPs are available for root domains (contact support).

checkmode:

String: "TCP" ensures a TCP connection can be made to the backend else it is deemed offline. "HTTP" performs HTTP healthcheck using checkmethod/checkurl/checkhost below. "" means backend is always treated as online (can cause delays if a backend is offline)

checkmethod:

String: GET or POST. Defaults to GET

checkurl:

String: URI to use for healthchecks. defaults to "/"

checkhost:

String: Host header to use for healthchecks. defaults to HTTP/1.0 (No host header)

backend_hostname: (Optional, applicable for type = GLB or HTTP)

String: override Host header. Defaults to off (http Host header pass-through)

aliases: (Required, applicable for type: GLB and HTTP)

Array of String:

String: hostname of frontend to use. Domain base must exist in domain approvals.

scope:

String: "global" (default). Provides access to a global set of anycasted load balancers directly connecting to the noted backends.
"local". Provides a local network load balancer instance. Requires scope_networkid.

scope_networkid: (Required, only applicable for scope = local)

The IPNetwork ID of the local network where this is to be deployed to. Network must have available IPs, and you must be using portal-registered IPs to avoid collissions. By default this will use the highest available IPs for the requested IPNetwork. If no IPs are available, it will fail.

scope_instances: (Optional, defaults 1, only applicable for scope = local)

Number of load balancer instances to run simultaneously. Note that Load Balancer fees are charged per-instance.

backends: (Required)

Array of backend:

hostname:
String: hostname or IP address of the backend server to use. Hostname minimum resolution is 30 seconds.
tcpproxy:
Integer: Enable TCP PROXY protocol. Only supports blank (no proxy), or 2 for version 2
weight:
Integer: Value 1-100 indicating the probability relative to all backend total weight that this backend will be used
port:
Integer: TCP port number
tls:
Boolean: true / false if TLS is meant to be used for the backend
uuid:
String: Optionally customer-supplied UUID that can be used for backend update commands (See backend updating)
ttl:
Integer: If a backend is not updated for ttl seconds, then it is automatically removed (See backend updating)
portmask: (Required for backend type UDP or TCP)
Array of Integer: List of frontend port numbers that this backend is applicable for

Backend Updating:

A lightweight API call can be used to update the backends using the UUID. Using this, all other UUID's remain untouched, only if there is a matching UUID will that backend be updated. *If there is no matching UUID, then a new one is created.*

> Combine this function with use of backend TTLs to enable auto-registration / availability / removal of backends.

```
{
  "cmd": "loadbalancer.backend.update",
  "id": __LBID__,
```

```
  "backends": [
    {
      "hostname": "1.2.3.4",
      "port": 80,
      "weight": 100
      "tls": false,
      "ttl": 600,
      "uuid": "XXXXX-XXXXX-XXXXX-XXXXX-XXXXX"
    },
    {
      "hostname": "5.6.7.8",
      "port": 80,
      "weight": 100,
      "tls": "false"
      "ttl": 600,
      "uuid": "XXXXX-XXXXX-XXXXX-XXXXX-XXXXX"
    }
  ]
}
```