# Linode Provider

The Linode provider exposes resources and data sources to interact with Linode (https://www.linode.com/) services. 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 data sources.

## Example Usage

```
# Configure the Linode provider
provider "linode" {
  token = "$LINODE_TOKEN"
}

resource "linode_instance" "foobar" {
  # ...
}
```

## Configuration Reference

The following keys can be used to configure the provider.

- `token` - (Required) This is your Linode APIv4 Token (https://developers.linode.com/api/v4#section/Personal-Access-Token).

The Linode Token can also be specified using the `LINODE_TOKEN` environment variable.

- `url` - (Optional) The HTTP(S) API address of the Linode API to use.

The Linode API URL can also be specified using the `LINODE_URL` environment variable.

- `ua_prefix` - (Optional) An HTTP User-Agent Prefix to prepend in API requests.

The User-Agent Prefix can also be specified using the `LINODE_UA_PREFIX` environment variable.

## Linode Guides

Several Linode Guides & Tutorials (https://www.linode.com/docs/) are available that explore Terraform usage with Linode resources:

- A Beginner's Guide to Terraform (https://www.linode.com/docs/applications/configuration-management/beginners-guide-to-terraform/)
- Introduction to HashiCorp Configuration Language (HCL) (https://www.linode.com/docs/applications/configuration-management/introduction-to-hcl/)
- Use Terraform to Provision Linode Environments (https://www.linode.com/docs/applications/configuration-management/how-to-build-your-infrastructure-using-terraform-and-linode/)

- Deploy a WordPress Site Using Terraform and Linode StackScripts (https://www.linode.com/docs/applications/configuration-management/deploy-a-wordpress-site-using-terraform-and-linode-stackscripts/)
- Create a NodeBalancer with Terraform (https://www.linode.com/docs/applications/configuration-management/create-a-nodebalancer-with-terraform/)
- Import Existing Infrastructure to Terraform (https://www.linode.com/docs/applications/configuration-management/import-existing-infrastructure-to-terraform/)
- Create a Terraform Module (https://www.linode.com/docs/applications/configuration-management/create-terraform-module/)
- Secrets Management with Terraform (https://www.linode.com/docs/applications/configuration-management/secrets-management-with-terraform/)

These guides are maintained by Linode and are not officially endorsed by HashiCorp.

# Debugging

The Linode APIv4 wrapper (https://github.com/linode/linodego) used by this provider accepts a `LINODE_DEBUG` environment variable. If this variable is assigned to `1`, the request and response of all Linode API traffic will be reported through Terraform debugging and logging facilities (/docs/internals/debugging.html).

Use of the `LINODE_DEBUG` variable in production settings is **strongly discouraged** with the `linode_account` datasource. While Terraform does not directly store sensitive data from this datasource, the Linode Account API endpoint returns **sensitive data** such as the account `tax_id` (VAT) and the credit card `last_four` and `expiry`. Be very cautious about storing this debug output.

# Data Source: linode_account

Provides information about a Linode account.

This data source should not be used in conjuction with the `LINODE_DEBUG` option. See the debugging notes (/docs/providers/linode/index.html#debugging) for more details.

## Example Usage

The following example shows how one might use this data source to access account details.

```
data "linode_account" "account" {}
```

## Argument Reference

There are no supported arguments because the provider `token` can only access the associated account.

## Attributes

The Linode Account resource exports the following attributes:

- `email` - The email address for this Account, for account management communications, and may be used for other communications as configured.
- `first_name` - The first name of the person associated with this Account.
- `last_name` - The last name of the person associated with this Account.
- `company` - The company name associated with this Account.
- `address_1` - First line of this Account's billing address.
- `address_2` - Second line of this Account's billing address.
- `phone` - The phone number associated with this Account.
- `city` - The city for this Account's billing address.
- `state` - If billing address is in the United States, this is the State portion of the Account's billing address. If the address is outside the US, this is the Province associated with the Account's billing address.
- `country` - The two-letter country code of this Account's billing address.
- `zip` - The zip code of this Account's billing address.
- `balance` - This Account's balance, in US dollars.

# Data Source: linode_domain

Provides information about a Linode domain.

## Example Usage

The following example shows how one might use this data source to access information about a Linode domain.

```
data "linode_domain" "foo" {
    id = "1234567"
}

data "linode_domain" "bar" {
    domain = "bar.example.com"
}
```

## Argument Reference

The following arguments are supported, at least one is required:

- `id` - (Optional) The unique numeric ID of the Domain record to query.
- `domain` - (Optional) The unique domain name of the Domain record to query.

## Attributes

The Linode Domain resource exports the following attributes:

- `id` - The unique ID of this Domain.
- `domain` - The domain this Domain represents. These must be unique in our system; you cannot have two Domains representing the same domain
- `type` - If this Domain represents the authoritative source of information for the domain it describes, or if it is a read-only copy of a master (also called a slave)
- `group` - The group this Domain belongs to.
- `status` - Used to control whether this Domain is currently being rendered.
- `description` - A description for this Domain.
- `master_ips` - The IP addresses representing the master DNS for this Domain.
- `axfr_ips` - The list of IPs that may perform a zone transfer for this Domain.
- `ttl_sec` - 'Time to Live'-the amount of time in seconds that this Domain's records may be cached by resolvers or other domain servers.

- `retry_sec` - The interval, in seconds, at which a failed refresh should be retried. *
- `expire_sec` - The amount of time in seconds that may pass before this Domain is no longer authoritative.
- `refresh_sec` - The amount of time in seconds before this Domain should be refreshed.
- `soa_email` - Start of Authority email address.
- `tags` - An array of tags applied to this object.

# Data Source: linode_image

Provides information about a Linode image

## Example Usage

The following example shows how one might use this data source to access information about a Linode image.

```
data "linode_image" "k8_master" {
    id = "linode/debian8"
}
```

## Argument Reference

The following arguments are supported:

- `id` - (Required) The unique ID of this Image. The ID of private images begin with `private/` followed by the numeric identifier of the private image, for example `private/12345`.

## Attributes

The Linode Image resource exports the following attributes:

- `label` - A short description of the Image.
- `created` - When this Image was created.
- `created_by` - The name of the User who created this Image, or "linode" for official Images.
- `deprecated` - Whether or not this Image is deprecated. Will only be true for deprecated public Images.
- `description` - A detailed description of this Image.
- `is_public` - True if the Image is public.
- `size` - The minimum size this Image needs to deploy. Size is in MB. example: 2500
- `type` - How the Image was created. Manual Images can be created at any time. image"Automatic" Images are created automatically from a deleted Linode.
- `vendor` - The upstream distribution vendor. `None` for private Images.

# Data Source: linode_instance_type

Provides information about a Linode instance type

## Example Usage

The following example shows how one might use this data source to access information about a Linode Instance type.

```
data "linode_instance_type" "default" {
    id = "g6-standard-2"
}
```

## Argument Reference

The following arguments are supported:

- `id` - (Required) Label used to identify instance type

## Attributes

The Linode Instance Type resource exports the following attributes:

- `id` - The ID representing the Linode Type
- `label` - The Linode Type's label is for display purposes only
- `class` - The class of the Linode Type
- `disk` - The Disk size, in MB, of the Linode Type
- `price.0.hourly` - Cost (in US dollars) per hour.
- `price.0.monthly` - Cost (in US dollars) per month.
- `addons.0.backups.0.price.0.hourly` - The cost (in US dollars) per hour to add Backups service.
- `addons.0.backups.0.price.0.monthly` - The cost (in US dollars) per month to add Backups service.

# Data Source: linode_network_ip

Provides information about a Linode Networking IP Address

## Example Usage

The following example shows how one might use this data source to access information about a Linode Networking IP Address.

```
data "linode_network_ip" "ns1_linode_com" {
    address = "162.159.27.72"
}
```

## Argument Reference

The following arguments are supported:

- `address` - (Required) The IP Address to access. The address must be associated with the account and a resource that the user has access to view.

## Attributes

The Linode Network IP Address resource exports the following attributes:

- `address` - The IP address.
- `gateway` - The default gateway for this address.
- `subnet_mask` - The mask that separates host bits from network bits for this address.
- `prefix` - The number of bits set in the subnet mask.
- `type` - The type of address this is (ipv4, ipv6, ipv6/pool, ipv6/range).
- `public` - Whether this is a public or private IP address.
- `rdns` - The reverse DNS assigned to this address. For public IPv4 addresses, this will be set to a default value provided by Linode if not explicitly set.
- `linode_id` - The ID of the Linode this address currently belongs to.
- `region` - The Region this IP address resides in.

# Data Source: linode_profile

Provides information about a Linode profile.

## Example Usage

The following example shows how one might use this data source to access profile details.

```
data "linode_profile" "profile" {}
```

## Argument Reference

There are no supported arguments because the provider `token` can only access the associated profile.

## Attributes

The Linode Profile resource exports the following attributes:

- `email` - The profile email address. This address will be used for communication with Linode as necessary.
- `timezone` - The profile's preferred timezone. This is not used by the API, and is for the benefit of clients only. All times the API returns are in UTC.
- `email_notifications` - If true, email notifications will be sent about account activity. If false, when false business-critical communications may still be sent through email.
- `username` - The username for logging in to Linode services.
- `ip_whitelist_enabled` - If true, logins for the user will only be allowed from whitelisted IPs. This setting is currently deprecated, and cannot be enabled.
- `lish_auth_method` - The methods of authentication allowed when connecting via Lish. 'keys_only' is the most secure with the intent to use Lish, and 'disabled' is recommended for users that will not use Lish at all.
- `authorized_keys` - The list of SSH Keys authorized to use Lish for this user. This value is ignored if lish_auth_method is 'disabled'.
- `two_factor_auth` - If true, logins from untrusted computers will require Two Factor Authentication.
- `restricted` - If true, the user has restrictions on what can be accessed on the Account.
- `referrals` - Credit Card information associated with this Account.
- `referrals.0.total` - The number of users who have signed up with the referral code.
- `referrals.0.credit` - The amount of account credit in US Dollars issued to the account through the referral program.
- `referrals.0.completed` - The number of completed signups with the referral code.

- `referrals.0.pending` - The number of pending signups for the referral code. To receive credit the signups must be completed.
- `referrals.0.code` - The Profile referral code. If new accounts use this when signing up for Linode, referring account will receive credit.
- `referrals.0.url` - The referral URL.

# Data Source: linode_region

`linode_region` provides details about a specific Linode region.

## Example Usage

The following example shows how the resource might be used to obtain additional information about a Linode region.

```
data "linode_region" "region" {
  id = "us-east"
}
```

## Argument Reference

- `id` - (Required) The code name of the region to select.

## Attributes Reference

In addition to all arguments above, the following attributes are exported:

- `country` - The country the region resides in.

# Data Source: linode_sshkey

`linode_sshkey` provides access to a specifically labeled SSH Key in the Profile of the User identified by the access token.

## Example Usage

The following example shows how the resource might be used to obtain the name of the SSH Key configured on the Linode user profile.

```
data "linode_sshkey" "foo" {
  label = "foo"
}
```

## Argument Reference

- `label` - (Required) The label of the SSH Key to select.

## Attributes Reference

In addition to all arguments above, the following attributes are exported:

- `id` - The ID of the SSH Key
- `ssh_key` - The public SSH Key, which is used to authenticate to the root user of the Linodes you deploy.
- `created` - The date this key was added.

# Data Source: linode_user

Provides information about a Linode user

## Example Usage

The following example shows how one might use this data source to access information about a Linode user.

```
data "linode_user" "foo" {
    username = "foo"
}
```

## Argument Reference

The following arguments are supported:

- `username` - (Required) The unique username of this User.

## Attributes

The Linode User resource exports the following attributes:

- `ssh_keys` - A list of SSH Key labels added by this User. These are the keys that will be deployed if this User is included in the authorized_users field of a create Linode, rebuild Linode, or create Disk request.
- `email` - The email address for this User, for account management communications, and may be used for other communications as configured.
- `restricted` - If true, this User must be granted access to perform actions or access entities on this Account.

# linode_domain

Provides a Linode Domain resource. This can be used to create, modify, and delete Linode Domains through Linode's managed DNS service. For more information, see DNS Manager (https://www.linode.com/docs/platform/manager/dns-manager/) and the Linode APIv4 docs (https://developers.linode.com/api/v4#operation/createDomain).

The Linode Guide, Deploy a WordPress Site Using Terraform and Linode StackScripts (https://www.linode.com/docs/applications/configuration-management/deploy-a-wordpress-site-using-terraform-and-linode-stackscripts/), demonstrates the management of Linode Domain resources in the context of Linode Instance running WordPress.

## Example Usage

The following example shows how one might use this resource to configure a Domain Record attached to a Linode Domain.

```
resource "linode_domain" "foobar" {
    type = "master"
    domain = "foobar.example"
    soa_email = "example@foobar.example"
    tags = ["foo", "bar"]
}

resource "linode_domain_record" "foobar" {
    domain_id = "${linode_domain.foobar.id}"
    name = "www"
    record_type = "CNAME"
    target = "foobar.example"
}
```

## Argument Reference

The following arguments are supported:

- `domain` - (Required) The domain this Domain represents. These must be unique in our system; you cannot have two Domains representing the same domain.
- `type` - (Required) If this Domain represents the authoritative source of information for the domain it describes, or if it is a read-only copy of a master (also called a slave).
- `soa_email` - (Required) Start of Authority email address. This is required for master Domains.
- `master_ips` - (Required for type="slave") The IP addresses representing the master DNS for this Domain.

---

- `status` - (Optional) Used to control whether this Domain is currently being rendered (defaults to "active").
- `description` - (Optional) A description for this Domain. This is for display purposes only.
- `group` - (Optional) The group this Domain belongs to. This is for display purposes only.

- `ttl_sec` - (Optional) 'Time to Live' - the amount of time in seconds that this Domain's records may be cached by resolvers or other domain servers. Valid values are 300, 3600, 7200, 14400, 28800, 57600, 86400, 172800, 345600, 604800, 1209600, and 2419200 - any other value will be rounded to the nearest valid value.
- `retry_sec` - (Optional) The interval, in seconds, at which a failed refresh should be retried. Valid values are 300, 3600, 7200, 14400, 28800, 57600, 86400, 172800, 345600, 604800, 1209600, and 2419200 - any other value will be rounded to the nearest valid value.
- `expire_sec` - (Optional) The amount of time in seconds that may pass before this Domain is no longer authoritative. Valid values are 300, 3600, 7200, 14400, 28800, 57600, 86400, 172800, 345600, 604800, 1209600, and 2419200 - any other value will be rounded to the nearest valid value.
- `refresh_sec` - (Optional) The amount of time in seconds before this Domain should be refreshed. Valid values are 300, 3600, 7200, 14400, 28800, 57600, 86400, 172800, 345600, 604800, 1209600, and 2419200 - any other value will be rounded to the nearest valid value.
- `axfr_ips` - (Optional) The list of IPs that may perform a zone transfer for this Domain. This is potentially dangerous, and should be set to an empty list unless you intend to use it.
- `tags` - (Optional) A list of tags applied to this object. Tags are for organizational purposes only.

## Attributes

This resource exports no additional attributes, however `status` may reflect degraded states.

## Import

Linodes Domains can be imported using the Linode Domain `id`, e.g.

```
terraform import linode_domain_record.foobar 1234567
```

The Linode Guide, Import Existing Infrastructure to Terraform (https://www.linode.com/docs/applications/configuration-management/import-existing-infrastructure-to-terraform/), offers resource importing examples for Domains and other Linode resource types.

# linode_domain

Provides a Linode Domain Record resource. This can be used to create, modify, and delete Linodes Domain Records. For more information, see DNS Manager (https://www.linode.com/docs/platform/manager/dns-manager/) and the Linode APIv4 docs (https://developers.linode.com/api/v4#operation/createDomainRecord).

## Example Usage

The following example shows how one might use this resource to configure a Domain Record attached to a Linode Domain.

```
resource "linode_domain" "foobar" {
    type = "master"
    domain = "foobar.example"
    soa_email = "example@foobar.example"
}

resource "linode_domain_record" "foobar" {
    domain_id = "${linode_domain.foobar.id}"
    name = "www"
    record_type = "CNAME"
    target = "foobar.example"
}
```

## Argument Reference

The following arguments are supported:

- `name` - (Required) The name of this Record. This field's actual usage depends on the type of record this represents. For A and AAAA records, this is the subdomain being associated with an IP address.
- `domain_id` - (Required) The ID of the Domain to access. *Changing `domain_id` forces the creation of a new Linode Domain Record.*.
- `record_type` - (Required) The type of Record this is in the DNS system. For example, A records associate a domain name with an IPv4 address, and AAAA records associate a domain name with an IPv6 address. *Changing `record_type` forces the creation of a new Linode Domain Record.*.
- `target` - (Required) The target for this Record. This field's actual usage depends on the type of record this represents. For A and AAAA records, this is the address the named Domain should resolve to.

---

- `ttl_sec` - (Optional) 'Time to Live' - the amount of time in seconds that this Domain's records may be cached by resolvers or other domain servers. Valid values are 300, 3600, 7200, 14400, 28800, 57600, 86400, 172800, 345600, 604800, 1209600, and 2419200 - any other value will be rounded to the nearest valid value.
- `priority` - (Optional) The priority of the target host. Lower values are preferred.
- `protocol` - (Optional) The protocol this Record's service communicates with. Only valid for SRV records.
- `service` - (Optional) The service this Record identified. Only valid for SRV records.

- `tag` - (Optional) The tag portion of a CAA record. It is invalid to set this on other record types.
- `port` - (Optional) The port this Record points to.
- `weight` - (Optional) The relative weight of this Record. Higher values are preferred.

## Attributes

This resource exports no additional attributes.

## Import

Linodes Domain Records can be imported using the Linode Domain `id` followed by the Domain Record `id` separated by a comma, e.g.

```
terraform import linode_domain_record.www-foobar 1234567,7654321
```

The Linode Guide, Import Existing Infrastructure to Terraform (https://www.linode.com/docs/applications/configuration-management/import-existing-infrastructure-to-terraform/), offers resource importing examples for Domain Records and other Linode resource types.

# linode_image

Provides a Linode Image resource. This can be used to create, modify, and delete Linodes Images. Linode Images are snapshots of a Linode Instance Disk which can then be used to provision more Linode Instances. Images can be used across regions.

For more information, see Linode's documentation on Images (https://www.linode.com/docs/platform/disk-images/linode-images/) and the Linode APIv4 docs (https://developers.linode.com/api/v4#operation/createImage).

## Example Usage

The following example shows how one might use this resource to create an Image from a Linode Instance Disk and then deploy a new Linode Instance in another region using that Image.

```
resource "linode_instance" "foo" {
    type = "g6-nanode-1"
    region = "us-central"
}

resource "linode_image" "bar" {
    label = "foo-sda-image"
    description = "Image taken from foo"
    disk_id = "${linode_instance.foo.disk.0.id}"
    linode_id = "${linode_instance.foo.id}"
}

resource "linode_instance" "bar_based" {
    type = "${linode_instance.foo.type}"
    region = "eu-west"
    image = "${linode_image.bar.id}"
}
```

## Argument Reference

The following arguments are supported:

- `label` - (Required) A short description of the Image. Labels cannot contain special characters.
- `disk_id` - (Required) The ID of the Linode Disk that this Image will be created from.
- `linode_id` - (Required) The ID of the Linode that this Image will be created from.

---

- `description` - (Optional) A detailed description of this Image.

### Timeouts

The `timeouts` block allows you to specify timeouts (https://www.terraform.io/docs/configuration/resources.html#timeouts) for certain actions:

- `create` - (Defaults to 20 mins) Used when creating the instance image (until the instance is available)

## Attributes

This resource exports the following attributes:

- `id` - The unique ID of this Image. The ID of private images begin with `private/` followed by the numeric identifier of the private image, for example `private/12345`.
- `created` - When this Image was created.
- `created_by` - The name of the User who created this Image.
- `deprecated` - Whether or not this Image is deprecated. Will only be True for deprecated public Images.
- `is_public` - True if the Image is public.
- `size` - The minimum size this Image needs to deploy. Size is in MB.
- `type` - How the Image was created. 'Manual' Images can be created at any time. 'Automatic' images are created automatically from a deleted Linode.
- `expiry` - Only Images created automatically (from a deleted Linode; type=automatic) will expire.
- `vendor` - The upstream distribution vendor. Nil for private Images.

## Import

Linodes Images can be imported using the Linode Image `id`, e.g.

```
terraform import linode_image.myimage 1234567
```

# linode_instance

Provides a Linode Instance resource. This can be used to create, modify, and delete Linodes. For more information, see Getting Started with Linode (https://linode.com/docs/getting-started/) and the Linode APIv4 docs (https://developers.linode.com/api/v4#operation/createLinodeInstance).

The Linode Guide, Use Terraform to Provision Linode Environments (https://www.linode.com/docs/applications/configuration-management/how-to-build-your-infrastructure-using-terraform-and-linode/), provides step-by-step guidance and additional examples.

Linode Instances can also use provisioners (/docs/provisioners/index.html).

## Example Usage

### Simple Linode Instance

The following example shows how one might use this resource to configure a Linode instance.

```
resource "linode_instance" "web" {
    label = "simple_instance"
    image = "linode/ubuntu18.04"
    region = "us-central"
    type = "g6-standard-1"
    authorized_keys = ["ssh-rsa AAAA...Gw== user@example.local"]
    root_pass = "terr4form-test"

    group = "foo"
    tags = [ "foo" ]
    swap_size = 256
    private_ip = true
}
```

### Linode Instance with explicit Configs and Disks

Using explicit Instance Configs and Disks it is possible to create a more elaborate Linode instance. This can be used to provision multiple disks and volumes during Instance creation.

```
data "linode_profile" "me" {}

resource "linode_instance" "web" {
  label      = "complex_instance"
  group      = "foo"
  tags = [ "foo" ]
  region     = "us-central"
  type       = "g6-nanode-1"
  private_ip = true

  disk {
    label = "boot"
    size = 3000
    image  = "linode/ubuntu18.04"

    # Any of authorized_keys, authorized_users, and root_pass
    # can be used for provisioning.
    authorized_keys = [ "ssh-rsa AAAA...Gw== user@example.local" ]
    authorized_users = [ "${data.linode_profile.me.username}" ]
    root_pass = "terr4form-test"
  }

  config {
    label = "boot_config"
    kernel = "linode/latest-64bit"
    devices {
      sda {
        disk_label = "boot"
      }
      sdb {
        volume_id = "${linode_volume.web_volume.id}"
      }
    }
    root_device = "/dev/sda"
  }

  boot_config_label = "boot_config"
}

resource "linode_volume" "web_volume" {
  label = "web_volume"
  size = 20
  region = "us-central"
}
```

# Argument Reference

The following arguments are supported:

- `region` - (Required) This is the location where the Linode is deployed. Examples are `"us-east"`, `"us-west"`, `"ap-south"`, etc. *Changing `region` forces the creation of a new Linode Instance.*.
- `type` - (Required) The Linode type defines the pricing, CPU, disk, and RAM specs of the instance. Examples are `"g6-nanode-1"`, `"g6-standard-2"`, `"g6-highmem-16"`, `"g6-dedicated-16"`, etc.

- `label` - (Optional) The Linode's label is for display purposes only. If no label is provided for a Linode, a default will be assigned.
- `group` - (Optional) The display group of the Linode instance.
- `tags` - (Optional) A list of tags applied to this object. Tags are for organizational purposes only.
- `private_ip` - (Optional) If true, the created Linode will have private networking enabled, allowing use of the 192.168.128.0/17 network within the Linode's region. It can be enabled on an existing Linode but it can't be disabled.
- `alerts.0.cpu` - (Optional) The percentage of CPU usage required to trigger an alert. If the average CPU usage over two hours exceeds this value, we'll send you an alert. If this is set to 0, the alert is disabled.
- `alerts.0.network_in` - (Optional) The amount of incoming traffic, in Mbit/s, required to trigger an alert. If the average incoming traffic over two hours exceeds this value, we'll send you an alert. If this is set to 0 (zero), the alert is disabled.
- `alerts.0.network_out` - (Optional) The amount of outbound traffic, in Mbit/s, required to trigger an alert. If the average outbound traffic over two hours exceeds this value, we'll send you an alert. If this is set to 0 (zero), the alert is disabled.
- `alerts.0.transfer_quota` - (Optional) The percentage of network transfer that may be used before an alert is triggered. When this value is exceeded, we'll alert you. If this is set to 0 (zero), the alert is disabled.
- `alerts.0.io` - (Optional) The amount of disk IO operation per second required to trigger an alert. If the average disk IO over two hours exceeds this value, we'll send you an alert. If set to 0, this alert is disabled.
- `backups_enabled` - (Optional) If this field is set to true, the created Linode will automatically be enrolled in the Linode Backup service. This will incur an additional charge. The cost for the Backup service is dependent on the Type of Linode deployed.
- `watchdog_enabled` - (Optional) The watchdog, named Lassie, is a Shutdown Watchdog that monitors your Linode and will reboot it if it powers off unexpectedly. It works by issuing a boot job when your Linode powers off without a shutdown job being responsible. To prevent a loop, Lassie will give up if there have been more than 5 boot jobs issued within 15 minutes.

## Simplified Resource Arguments

Just as the Linode API provides, these fields are for the most common provisioning use case, a single data disk, a single swap disk, and a single config. These arguments are not compatible with `disk` and `config` fields, described later.

- `authorized_keys` - (Optional with `image`) A list of SSH public keys to deploy for the root user on the newly created Linode. *This value can not be imported. Changing `authorized_keys` forces the creation of a new Linode Instance.*
- `authorized_users` - (Optional with `image`) A list of Linode usernames. If the usernames have associated SSH keys, the keys will be appended to the `root` user's `~/.ssh/authorized_keys` file automatically. *This value can not be imported. Changing `authorized_users` forces the creation of a new Linode Instance.*
- `root_pass` - (Optional) The initial password for the `root` user account. *This value can not be imported. Changing `root_pass` forces the creation of a new Linode Instance. If omitted, a random password will be generated but will not be stored in Terraform state.*
- `image` - (Optional) An Image ID to deploy the Disk from. Official Linode Images start with linode/, while your Images start with `private/`. See images (https://api.linode.com/v4/images) for more information on the Images available for

you to use. Examples are `linode/debian9`, `linode/fedora28`, `linode/ubuntu16.04lts`, `linode/arch`, and `private/12345`. *This value can not be imported. Changing `image` forces the creation of a new Linode Instance.*

- `stackscript_id` - (Optional) The StackScript to deploy to the newly created Linode. If provided, 'image' must also be provided, and must be an Image that is compatible with this StackScript. *This value can not be imported. Changing `stackscript_id` forces the creation of a new Linode Instance.*
- `stackscript_data` - (Optional) An object containing responses to any User Defined Fields present in the StackScript being deployed to this Linode. Only accepted if 'stackscript_id' is given. The required values depend on the StackScript being deployed. *This value can not be imported. Changing `stackscript_data` forces the creation of a new Linode Instance.*
- `swap_size` - (Optional) When deploying from an Image, this field is optional with a Linode API default of 512mb, otherwise it is ignored. This is used to set the swap disk size for the newly-created Linode.
- `backup_id` - (Optional) A Backup ID from another Linode's available backups. Your User must have read_write access to that Linode, the Backup must have a status of successful, and the Linode must be deployed to the same region as the Backup. See /linode/instances/{linodeId}/backups for a Linode's available backups. This field and the image field are mutually exclusive. *This value can not be imported. Changing `backup_id` forces the creation of a new Linode Instance.*

## Disk and Config Arguments

By specifying the `disk` and `config` fields for a Linode instance, it is possible to use non-standard kernels, boot with and provision multiple disks, and modify the boot behaviors ( `helpers` ) of the Linode.

- `boot_config_label` - (Optional) The Label of the Instance Config that should be used to boot the Linode instance. If there is only one `config`, the `label` of that `config` will be used as the `boot_config_label`. *This value can not be imported.*

### Disks

- `disk`
  - `label` - (Required) The disks label, which acts as an identifier in Terraform. This must be unique within each Linode Instance.
  - `size` - (Required) The size of the Disk in MB.
  - `id` - (Computed) The ID of the disk in the Linode API.
  - `filesystem` - (Optional) The Disk filesystem can be one of: `"raw"`, `"swap"`, `"ext3"`, `"ext4"`, or `"initrd"` which has a max size of 32mb and can be used in the config `initrd` (not currently supported in this Terraform Provider).
  - `readonly` - (Optional) If true, this Disk is read-only.
  - `image` - (Optional) An Image ID to deploy the Disk from. Official Linode Images start with linode/, while your Images start with private/. See /images for more information on the Images available for you to use. Examples are `linode/debian9`, `linode/fedora28`, `linode/ubuntu16.04lts`, `linode/arch`, and `private/12345`. *Changing `image` forces the creation of a new Linode Instance.*

- `authorized_keys` - (Optional with `image`) A list of SSH public keys to deploy for the root user on the newly created Linode. Only accepted if `image` is provided. *This value can not be imported. Changing `authorized_keys` forces the creation of a new Linode Instance.*
- `authorized_users` - (Optional with `image`) A list of Linode usernames. If the usernames have associated SSH keys, the keys will be appended to the `root` user's `~/.ssh/authorized_keys` file automatically. *This value can not be imported. Changing `authorized_users` forces the creation of a new Linode Instance.*
- `root_pass` - (Optional with `image`) The initial password for the `root` user account. *This value can not be imported. Changing `root_pass` forces the creation of a new Linode Instance. If omitted, a random password will be generated but will not be stored in Terraform state.*
- `stackscript_id` - (Optional with `image`) The StackScript to deploy to the newly created Linode. If provided, 'image' must also be provided, and must be an Image that is compatible with this StackScript. *This value can not be imported. Changing `stackscript_id` forces the creation of a new Linode Instance.*
- `stackscript_data` - (Optional with `image`) An object containing responses to any User Defined Fields present in the StackScript being deployed to this Linode. Only accepted if 'stackscript_id' is given. The required values depend on the StackScript being deployed. *This value can not be imported. Changing `stackscript_data` forces the creation of a new Linode Instance.*

### Configs

Configuration profiles define the VM settings and boot behavior of the Linode Instance. Multiple configurations profiles can be provided but their `label` values must be unique.

- `config`
  - `label` - (Required) The Config's label for display purposes. Also used by `boot_config_label`.
  - `helpers` - (Options) Helpers enabled when booting to this Linode Config.
  - `updatedb_disabled` - (Optional) Disables updatedb cron job to avoid disk thrashing.
  - `distro` - (Optional) Controls the behavior of the Linode Config's Distribution Helper setting.
  - `modules_dep` - (Optional) Creates a modules dependency file for the Kernel you run.
  - `network` - (Optional) Controls the behavior of the Linode Config's Network Helper setting, used to automatically configure additional IP addresses assigned to this instance.
  - `devices` - (Optional) A list of `disk` or `volume` attachments for this `config`. If the `boot_config_label` omits a `devices` block, the Linode will not be booted.
  - `sda` ... `sdh` - (Optional) The SDA-SDH slots, represent the Linux block device nodes for the first 8 disks attached to the Linode. Each device must be suplied sequentially. The device can be either a Disk or a Volume identified by `disk_label` or `volume_id`. Only one disk identifier is permitted per slot. Devices mapped from `sde` through `sdh` are unavailable in `"fullvirt" virt_mode`.
    - `disk_label` - (Optional) The `label` of the `disk` to map to this `device` slot.
    - `volume_id` - (Optional) The Volume ID to map to this `device` slot.
    - `disk_id` - (Computed) The Disk ID of the associated `disk_label`, if used.
  - `kernel` - (Optional) - A Kernel ID to boot a Linode with. Default is based on image choice. (examples:

linode/latest-64bit, linode/grub2, linode/direct-disk)

- `run_level` - (Optional) - Defines the state of your Linode after booting. Defaults to `"default"`.
- `virt_mode` - (Optional) - Controls the virtualization mode. Defaults to `"paravirt"`.
- `root_device` - (Optional) - The root device to boot. The corresponding disk must be attached to a `device` slot. Example: `"/dev/sda"`
- `comments` - (Optional) - Arbitrary user comments about this `config`.
- `memory_limit` - (Optional) - Defaults to the total RAM of the Linode

### Timeouts

The `timeouts` block allows you to specify timeouts (https://www.terraform.io/docs/configuration/resources.html#timeouts) for certain actions:

- `create` - (Defaults to 10 mins) Used when launching the instance (until it reaches the initial `running` state)
- `update` - (Defaults to 20 mins) Used when stopping and starting the instance when necessary during update - e.g. when changing instance type
- `delete` - (Defaults to 10 mins) Used when terminating the instance

## Attributes

This Linode Instance resource exports the following attributes:

- `status` - The status of the instance, indicating the current readiness state. (`running`, `offline`, ...)
- `ip_address` - A string containing the Linode's public IP address.
- `private_ip_address` - This Linode's Private IPv4 Address, if enabled. The regional private IP address range, 192.168.128.0/17, is shared by all Linode Instances in a region.
- `ipv6` - This Linode's IPv6 SLAAC addresses. This address is specific to a Linode, and may not be shared. The prefix (`/64`) is included in this attribute.
- `ipv4` - This Linode's IPv4 Addresses. Each Linode is assigned a single public IPv4 address upon creation, and may get a single private IPv4 address if needed. You may need to open a support ticket to get additional IPv4 addresses.
- `specs.0.disk` - The amount of storage space, in GB. this Linode has access to. A typical Linode will divide this space between a primary disk with an image deployed to it, and a swap disk, usually 512 MB. This is the default configuration created when deploying a Linode with an image through POST /linode/instances.
- `specs.0.memory` - The amount of RAM, in MB, this Linode has access to. Typically a Linode will choose to boot with all of its available RAM, but this can be configured in a Config profile.
- `specs.0.vcpus` - The number of vcpus this Linode has access to. Typically a Linode will choose to boot with all of its available vcpus, but this can be configured in a Config Profile.
- `specs.0.transfer` - The amount of network transfer this Linode is allotted each month.

- `backups` - Information about this Linode's backups status.
    - `enabled` - If this Linode has the Backup service enabled.
    - `schedule`
    - `day` - The day of the week that your Linode's weekly Backup is taken. If not set manually, a day will be chosen for you. Backups are taken every day, but backups taken on this day are preferred when selecting backups to retain for a longer period. If not set manually, then when backups are initially enabled, this may come back as "Scheduling" until the day is automatically selected.
    - `window` - The window ('W0'-'W22') in which your backups will be taken, in UTC. A backups window is a two-hour span of time in which the backup may occur. For example, 'W10' indicates that your backups should be taken between 10:00 and 12:00. If you do not choose a backup window, one will be selected for you automatically. If not set manually, when backups are initially enabled this may come back as Scheduling until the window is automatically selected.

# Import

Linodes Instances can be imported using the Linode `id`, e.g.

```
terraform import linode_instance.mylinode 1234567
```

When importing an instance, all `disk` and `config` values must be represented.

Imported disks must include their `label` value. **Any disk that is not precisely represented may be removed resulting in data loss.**

Imported configs should include all `devices`, and must include `label`, `kernel`, and the `root_device`. The instance must include a `boot_config_label` referring to the correct configuration profile.

The Linode Guide, Import Existing Infrastructure to Terraform (https://www.linode.com/docs/applications/configuration-management/import-existing-infrastructure-to-terraform/), offers resource importing examples for Instances and other Linode resource types.

# linode_nodebalancer_config

Provides a Linode NodeBalancer Config resource. This can be used to create, modify, and delete Linodes NodeBalancer Configs. For more information, see Getting Started with NodeBalancers (https://www.linode.com/docs/platform/nodebalancer/getting-started-with-nodebalancers/) and the Linode APIv4 docs (https://developers.linode.com/api/v4#operation/createNodeBalancerConfig).

The Linode Guide, Create a NodeBalancer with Terraform (https://www.linode.com/docs/applications/configuration-management/create-a-nodebalancer-with-terraform/), provides step-by-step guidance and additional examples.

## Example Usage

The following example shows how one might use this resource to configure a NodeBalancer Config attached to a Linode instance.

```
resource "linode_nodebalancer" "foobar" {
    label = "mynodebalancer"
    region = "us-east"
    client_conn_throttle = 20
}

resource "linode_nodebalancer_config" "foofig" {
    nodebalancer_id = "${linode_nodebalancer.foobar.id}"
    port = 8088
    protocol = "http"
    check = "http"
    check_path = "/foo"
    check_attempts = 3
    check_timeout = 30
    stickiness = "http_cookie"
    algorithm = "source"
}
```

## Argument Reference

The following arguments are supported:

- `nodebalancer_id` - (Required) The ID of the NodeBalancer to access.
- `region` - (Required) The region where this nodebalancer_config will be deployed. Examples are `"us-east"`, `"us-west"`, `"ap-south"`, etc. *Changing `region` forces the creation of a new Linode NodeBalancer Config.*.

---

- `protocol` - (Optional) The protocol this port is configured to serve. If this is set to https you must include an ssl_cert and an ssl_key. (Defaults to "http")

- `port` - (Optional) The TCP port this Config is for. These values must be unique across configs on a single NodeBalancer (you can't have two configs for port 80, for example). While some ports imply some protocols, no enforcement is done and you may configure your NodeBalancer however is useful to you. For example, while port 443 is generally used for HTTPS, you do not need SSL configured to have a NodeBalancer listening on port 443. (Defaults to 80)
- `algorithm` - (Optional) What algorithm this NodeBalancer should use for routing traffic to backends: roundrobin, leastconn, source
- `stickiness` - (Optional) Controls how session stickiness is handled on this port: 'none', 'table', 'http_cookie'
- `check` - (Optional) The type of check to perform against backends to ensure they are serving requests. This is used to determine if backends are up or down. If none no check is performed. connection requires only a connection to the backend to succeed. http and http_body rely on the backend serving HTTP, and that the response returned matches what is expected.
- `check_interval` - (Optional) How often, in seconds, to check that backends are up and serving requests.
- `check_timeout` - (Optional) How long, in seconds, to wait for a check attempt before considering it failed. (1-30)
- `check_attempts` - (Optional) How many times to attempt a check before considering a backend to be down. (1-30)
- `check_path` - (Optional) The URL path to check on each backend. If the backend does not respond to this request it is considered to be down.
- `check_passive` - (Optional) If true, any response from this backend with a 5xx status code will be enough for it to be considered unhealthy and taken out of rotation.
- `cipher_suite` - (Optional) What ciphers to use for SSL connections served by this NodeBalancer. `legacy` is considered insecure and should only be used if necessary.
- `ssl_cert` - (Optional) The certificate this port is serving. This is not returned. If set, this field will come back as `<REDACTED>` . Please use the ssl_commonname and ssl_fingerprint to identify the certificate.
- `ssl_key` - (Optional) The private key corresponding to this port's certificate. This is not returned. If set, this field will come back as `<REDACTED>` . Please use the ssl_commonname and ssl_fingerprint to identify the certificate.

# Attributes

This resource exports the following attributes:

- `ssl_commonname` - The common name for the SSL certification this port is serving if this port is not configured to use SSL.
- `ssl_fingerprint` - The fingerprint for the SSL certification this port is serving if this port is not configured to use SSL.
- `node_status_up` - The number of backends considered to be 'UP' and healthy, and that are serving requests.
- `node_status_down` - The number of backends considered to be 'DOWN' and unhealthy. These are not in rotation, and not serving requests.

# Import

NodeBalancer Configs can be imported using the NodeBalancer `nodebalancer_id` followed by the NodeBalancer Config `id` separated by a comma, e.g.

```
terraform import linode_nodebalancer_config.http-foobar 1234567,7654321
```

The Linode Guide, Import Existing Infrastructure to Terraform (https://www.linode.com/docs/applications/configuration-management/import-existing-infrastructure-to-terraform/), offers resource importing examples for NodeBalancer Configs and other Linode resource types.

# linode_nodebalancer

Provides a Linode NodeBalancer resource. This can be used to create, modify, and delete Linodes NodeBalancers in Linode's managed load balancer service. For more information, see Getting Started with NodeBalancers (https://www.linode.com/docs/platform/nodebalancer/getting-started-with-nodebalancers/) and the Linode APIv4 docs (https://developers.linode.com/api/v4#operation/createNodeBalancer).

The Linode Guide, Create a NodeBalancer with Terraform (https://www.linode.com/docs/applications/configuration-management/create-a-nodebalancer-with-terraform/), provides step-by-step guidance and additional examples.

## Example Usage

The following example shows how one might use this resource to configure a NodeBalancer.

```
resource "linode_nodebalancer" "foobar" {
    label = "mynodebalancer"
    region = "us-east"
    client_conn_throttle = 20
    tags = ["foobar"]

}
```

## Argument Reference

The following arguments are supported:

- `region` - (Required) The region where this NodeBalancer will be deployed. Examples are `"us-east"`, `"us-west"`, `"ap-south"`, etc. *Changing `region` forces the creation of a new Linode NodeBalancer.*.

---

- `label` - (Optional) The label of the Linode NodeBalancer
- `client_conn_throttle` - (Optional) Throttle connections per second (0-20). Set to 0 (default) to disable throttling.
- `linode_id` - (Optional) The ID of a Linode Instance where the the NodeBalancer should be attached.
- `tags` - (Optional) A list of tags applied to this object. Tags are for organizational purposes only.

## Attributes

This resource exports the following attributes:

- `hostname` - This NodeBalancer's hostname, ending with .nodebalancer.linode.com
- `ipv4` - The Public IPv4 Address of this NodeBalancer
- `ipv6` - The Public IPv6 Address of this NodeBalancer

# Import

Linodes NodeBalancers can be imported using the Linode NodeBalancer `id`, e.g.

```
terraform import linode_nodebalancer.mynodebalancer 1234567
```

The Linode Guide, Import Existing Infrastructure to Terraform (https://www.linode.com/docs/applications/configuration-management/import-existing-infrastructure-to-terraform/), offers resource importing examples for NodeBalancers and other Linode resource types.

# linode_nodebalancer_node

Provides a Linode NodeBalancer Node resource. This can be used to create, modify, and delete Linodes NodeBalancer Nodes. For more information, see Getting Started with NodeBalancers (https://www.linode.com/docs/platform/nodebalancer/getting-started-with-nodebalancers/) and the Linode APIv4 docs (https://developers.linode.com/api/v4#operation/createNodeBalancerNode).

The Linode Guide, Create a NodeBalancer with Terraform (https://www.linode.com/docs/applications/configuration-management/create-a-nodebalancer-with-terraform/), provides step-by-step guidance and additional examples.

## Example Usage

The following example shows how one might use this resource to configure NodeBalancer Nodes attached to Linode instances.

```
resource "linode_instance" "web" {
    count = "3"
    label = "web-${count.index + 1}"
    image = "linode/ubuntu18.04"
    region = "us-east"
    type = "g6-standard-1"
    authorized_keys = ["ssh-rsa AAAA...Gw== user@example.local"]
    root_pass = "terraform-test"

    private_ip = true
}

resource "linode_nodebalancer" "foobar" {
    label = "mynodebalancer"
    region = "us-east"
    client_conn_throttle = 20
}

resource "linode_nodebalancer_config" "foofig" {
    nodebalancer_id = "${linode_nodebalancer.foobar.id}"
    port = 80
    protocol = "http"
    check = "http"
    check_path = "/foo"
    check_attempts = 3
    check_timeout = 30
    stickiness = "http_cookie"
    algorithm = "source"
}

resource "linode_nodebalancer_node" "foonode" {
    count = "3"
    nodebalancer_id = "${linode_nodebalancer.foobar.id}"
    config_id = "${linode_nodebalancer_config.foofig.id}"
    address = "${element(linode_instance.web.*.private_ip_address, count.index)}:80"
    label = "mynodebalancernode"
    weight = 50
}
```

## Argument Reference

The following arguments are supported:

- `label` - (Required) The label of the Linode NodeBalancer Node. This is for display purposes only.
- `nodebalancer_id` - (Required) The ID of the NodeBalancer to access.
- `config_id` - (Required) The ID of the NodeBalancerConfig to access.
- `address` - (Required) The private IP Address where this backend can be reached. This must be a private IP address.

---

- `mode` - (Optional) The mode this NodeBalancer should use when sending traffic to this backend. If set to `accept` this backend is accepting traffic. If set to `reject` this backend will not receive traffic. If set to `drain` this backend will not receive new traffic, but connections already pinned to it will continue to be routed to it

- `weight` - (Optional) Used when picking a backend to serve a request and is not pinned to a single backend yet. Nodes with a higher weight will receive more traffic. (1-255).

## Attributes

This resource exports the following attributes:

- `status` - The current status of this node, based on the configured checks of its NodeBalancer Config. (unknown, UP, DOWN).
- `config_id` - The ID of the NodeBalancerConfig this NodeBalancerNode is attached to.
- `nodebalancer_id` - The ID of the NodeBalancer this NodeBalancerNode is attached to.

## Import

NodeBalancer Nodes can be imported using the NodeBalancer `nodebalancer_id` followed by the NodeBalancer Config `config_id` followed by the NodeBalancer Node `id`, separated by a comma, e.g.

```
terraform import linode_nodebalancer_node.https-foobar-1 1234567,7654321,9999999
```

The Linode Guide, Import Existing Infrastructure to Terraform (https://www.linode.com/docs/applications/configuration-management/import-existing-infrastructure-to-terraform/), offers resource importing examples for NodeBalancer Nodes and other Linode resource types.

# linode_rdns

Provides a Linode RDNS resource. This can be used to create and modify RDNS records.

Linode RDNS names must have a matching address value in an A or AAAA record. This A or AAAA name must be resolvable at the time the RDNS resource is being associated.

For more information, see the Linode APIv4 docs (https://developers.linode.com/api/docs/v4#operation/updateIP) and the Configure your Linode for Reverse DNS (https://www.linode.com/docs/networking/dns/configure-your-linode-for-reverse-dns-classic-manager/) guide.

## Example Usage

The following example shows how one might use this resource to configure an RDNS address for an IP address.

```
resource "linode_rdns" "foo" {
  address = "${linode_instance.foo.ip_address}"
  rdns = "${linode_instance.foo.ip_address}.nip.io"
}

resource "linode_instance" "foo" {
   image = "linode/alpine3.9"
   region = "ca-east"
   type = "g6-dedicated-2"
}
```

## Argument Reference

The following arguments are supported:

- `address` - The Public IPv4 or IPv6 address that will receive the `PTR` record. A matching `A` or `AAAA` record must exist.
- `rdns` - The name of the RDNS address.

## Import

Linodes RDNS resources can be imported using the address as the `id`.

```
terraform import linode_rdns.foo 123.123.123.123
```

# linode_sshkey

Provides a Linode SSH Key resource. This can be used to create, modify, and delete Linodes SSH Keys. Managed SSH Keys allow instances to be created with a list of Linode usernames, whose SSH keys will be automatically applied to the root account's `~/.ssh/authorized_keys` file. For more information, see the Linode APIv4 docs (https://developers.linode.com/api/v4#operation/getSSHKeys).

## Example Usage

The following example shows how one might use this resource to configure a SSH Key for access to a Linode Instance.

```
resource "linode_sshkey" "foo" {
  label = "foo"
  ssh_key = "${chomp(file("~/.ssh/id_rsa.pub"))}"
}

resource "linode_instance" "foo" {
  image  = "linode/ubuntu18.04"
  label  = "foo"
  region = "us-east"
  type   = "g6-nanode-1"
  authorized_keys    = ["${linode_sshkey.foo.ssh_key}"]
  root_pass      = "..."
}
```

## Argument Reference

The following arguments are supported:

- `label` - A label for the SSH Key.
- `ssh_key` - The public SSH Key, which is used to authenticate to the root user of the Linodes you deploy.

## Attributes

This resource exports the following attributes:

- `created` - The date this SSH Key was created.

## Import

Linodes SSH Keys can be imported using the Linode SSH Key `id` , e.g.

```
terraform import linode_sshkey.mysshkey 1234567
```

# linode_stackscript

Provides a Linode StackScript resource. This can be used to create, modify, and delete Linode StackScripts. StackScripts are private or public managed scripts which run within an instance during startup. StackScripts can include variables whose values are specified when the Instance is created.

For more information, see Automate Deployment with StackScripts (https://www.linode.com/docs/platform/stackscripts/) and the Linode APIv4 docs (https://developers.linode.com/api/v4#tag/StackScripts).

The Linode Guide, Deploy a WordPress Site Using Terraform and Linode StackScripts (https://www.linode.com/docs/applications/configuration-management/deploy-a-wordpress-site-using-terraform-and-linode-stackscripts/), shows how a public StackScript can be used to provision a Linode Instance. The guide, Create a Terraform Module (https://www.linode.com/docs/applications/configuration-management/create-terraform-module/), demonstrates StackScript use through a wrapping module.

## Example Usage

The following example shows how one might use this resource to configure a StackScript attached to a Linode Instance. As shown below, StackScripts must begin with a shebang ( `#!/` ). The `<UDF ...>` element provided in the Bash comment block defines a variable whose value is provided when creating the Instance (or disk) using the `stackscript_data` field.

```
resource "linode_stackscript" "foo" {
  label = "foo"
  description = "Installs a Package"
  script = <<EOF
#!/bin/bash
# <UDF name="package" label="System Package to Install" example="nginx" default="">
apt-get -q update && apt-get -q -y install $PACKAGE
EOF
  images = ["linode/ubuntu18.04", "linode/ubuntu16.04lts"]
  rev_note = "initial version"
}

resource "linode_instance" "foo" {
  image  = "linode/ubuntu18.04"
  label  = "foo"
  region = "us-east"
  type   = "g6-nanode-1"
  authorized_keys    = ["..."]
  root_pass      = "..."

  stackscript_id = "${linode_stackscript.install-nginx.id}"
  stackscript_data = {
    "package" = "nginx"
  }
}
```

## Argument Reference

The following arguments are supported:

- `label` - (Required) The StackScript's label is for display purposes only.
- `script` - (Required) The script to execute when provisioning a new Linode with this StackScript.
- `description` - (Required) A description for the StackScript.

---

- `rev_note` - (Optional) This field allows you to add notes for the set of revisions made to this StackScript.
- `is_public` - (Optional) This determines whether other users can use your StackScript. Once a StackScript is made public, it cannot be made private. *Changing `is_public` forces the creation of a new StackScript*
- `images` - (Optional) An array of Image IDs representing the Images that this StackScript is compatible for deploying with.

## Attributes

This resource exports the following attributes:

- `deployments_active` - Count of currently active, deployed Linodes created from this StackScript.
- `user_gravatar_id` - The Gravatar ID for the User who created the StackScript.
- `deployments_total` - The total number of times this StackScript has been deployed.
- `username` - The User who created the StackScript.
- `created` - The date this StackScript was created.
- `updated` - The date this StackScript was updated.
- `user_defined_fields` - This is a list of fields defined with a special syntax inside this StackScript that allow for supplying customized parameters during deployment.
    - `label` - A human-readable label for the field that will serve as the input prompt for entering the value during deployment.
    - `name` - The name of the field.
    - `example` - An example value for the field.
    - `one_of` - A list of acceptable single values for the field.
    - `many_of` - A list of acceptable values for the field in any quantity, combination or order.
    - `default` - The default value. If not specified, this value will be used.

## Import

Linodes StackScripts can be imported using the Linode StackScript `id` , e.g.

```
terraform import linode_stackscript.mystackscript 1234567
```

# linode_token

Provides a Linode Token resource. This can be used to create, modify, and delete Linode API Personal Access Tokens. Personal Access Tokens proxy user credentials for Linode API access. This is necessary for tools, such as Terraform, to interact with Linode services on a user's behalf.

It is common for Terraform itself to be configured with broadly scoped Personal Access Tokens. Provisioning scripts or tools configured within a Linode Instance should follow the principle of least privilege to afford only the required roles for tools to perform their necessary tasks. The `linode_token` resource allows for the management of Personal Access Tokens with scopes mirroring or narrowing the scope of the parent token.

For more information, see the Linode APIv4 docs (https://developers.linode.com/api/v4#operation/getTokens).

## Example Usage

The following example shows how one might use this resource to configure a token for use in another tool that needs access to Linode resources.

```
resource "linode_token" "foo" {
  label  = "token"
  scopes = "linodes:read_only"
  expiry = "2100-01-02T03:04:05Z"
}

resource "linode_instance" "foo" {
  # Configure the linode-cli and use it to add other Linode Instances to the hosts file
  provisioner "remote-exec" {
    inline = <<EOF
echo -e "[DEFAULT]\n token = ${linode_token.foo.token}\n region=${self.region}\n type=${self.type}" > ~/.
linode-cli
pip install linode-cli
linode-cli linodes list --format "ipv6,label" --text --no-headers >> /etc/hosts
EOF
  }
}
```

## Argument Reference

The following arguments are supported:

- `label` - A label for the Token.
- `scopes` - The scopes this token was created with. These define what parts of the Account the token can be used to access. Many command-line tools, such as the Linode CLI, require tokens with access to *. Tokens with more restrictive scopes are generally more secure.
- `expiry` - When this token will expire. Personal Access Tokens cannot be renewed, so after this time the token will be completely unusable and a new token will need to be generated. Tokens may be created with 'null' as their expiry and will never expire unless revoked.

## Attributes

This resource exports the following attributes:

- `token` - The token used to access the API.
- `created` - The date this Token was created.

## Import

Linodes Tokens can be imported using the Linode Token `id` , e.g. The secret token will not be imported.

```
terraform import linode_token.mytoken 1234567
```

# linode_volume

Provides a Linode Volume resource. This can be used to create, modify, and delete Linodes Block Storage Volumes. Block Storage Volumes are removable storage disks that persist outside the life-cycle of Linode Instances. These volumes can be attached to and detached from Linode instances throughout a region.

For more information, see How to Use Block Storage with Your Linode (https://www.linode.com/docs/platform/block-storage/how-to-use-block-storage-with-your-linode/) and the Linode APIv4 docs (https://developers.linode.com/api/v4#operation/createVolume).

## Example Usage

The following example shows how one might use this resource to configure a Block Storage Volume attached to a Linode Instance.

```
resource "linode_instance" "foobaz" {
    root_pass = "3X4mp13"
    type = "g6-nanode-1"
    region = "us-west"
    tags = ["foobaz"]

}

resource "linode_volume" "foobar" {
    label = "foo-volume"
    region = "${linode_instance.foobaz.region}"
    linode_id = "${linode_instance.foobaz.id}"
}
```

Volumes can also be attached using the Linode Instance config device map.

```
resource "linode_instance" "foo" {
  region             = "us-east"
  type               = "g6-nanode-1"

  config {
    label = "boot-existing-volume"
    kernel = "linode/latest-64bit"
    devices {
      sda {
        volume_id = "123"
      }
    }
  }
}
```

## Argument Reference

The following arguments are supported:

- `label` - (Required) The label of the Linode Volume
- `region` - (Required) The region where this volume will be deployed. Examples are `"us-east"`, `"us-west"`, `"ap-south"`, etc. *Changing `region` forces the creation of a new Linode Volume.*.

---

- `size` - (Optional) Size of the Volume in GB.
- `linode_id` - (Optional) The ID of a Linode Instance where the the Volume should be attached.
- `tags` - (Optional) A list of tags applied to this object. Tags are for organizational purposes only.

### Timeouts

The `timeouts` block allows you to specify timeouts (https://www.terraform.io/docs/configuration/resources.html#timeouts) for certain actions:

- `create` - (Defaults to 10 mins) Used when creating the volume (until the volume is reaches the initial `active` state)
- `update` - (Defaults to 20 mins) Used when updating the volume when necessary during update - e.g. when resizing the volume
- `delete` - (Defaults to 10 mins) Used when deleting the volume

## Attributes

This resource exports the following attributes:

- `status` - The label of the Linode Volume.
- `filesystem_path` - The full filesystem path for the Volume based on the Volume's label. The path is "/dev/disk/by-id/scsi-0Linode_Volume_" + the Volume label

## Import

Linodes Volumes can be imported using the Linode Volume `id`, e.g.

```
terraform import linode_volume.myvolume 1234567
```

The Linode Guide, Import Existing Infrastructure to Terraform (https://www.linode.com/docs/applications/configuration-management/import-existing-infrastructure-to-terraform/), offers resource importing examples for Block Storage Volumes and other Linode resource types.