Helm Provider

The Helm provider is used to deploy software packages in Kubernetes. The provider needs to be configured with the proper credentials before it can be used.

Resources

- Resource: helm_release (/docs/providers/helm/release.html)
- Resource: helm_repository (/docs/providers/helm/repository.html)

Example Usage

```
resource "helm_release" "mydatabase" {
    name      = "mydatabase"
    chart     = "stable/mariadb"

    set {
        name  = "mariadbUser"
        value = "foo"
    }

    set {
        name = "mariadbPassword"
        value = "qux"
    }
}
```

Requirements

- You must have Kubernetes installed. We recommend version 1.4.1 or later.
- You should also have a local configured copy of kubectl.

Authentication

There are generally two ways to configure the Helm provider.

File config

The provider always first tries to load **a config file** (usually `$HOME/.kube/config`), for access kubenetes and reads all the Helm files from home (usually `$HOME/.helm`).

Statically defined credentials

The other way is **statically** define all the credentials:

```
provider "helm" {
    kubernetes {
        host     = "https://104.196.242.174"
        username = "ClusterMaster"
        password = "MindTheGap"

        client_certificate     = "${file("~/.kube/client-cert.pem")}"
        client_key             = "${file("~/.kube/client-key.pem")}"
        cluster_ca_certificate = "${file("~/.kube/cluster-ca-cert.pem")}"
    }
}
```

If you have **both** valid configuration in a config file and static configuration, the static one is used as override. i.e. any static field will override its counterpart loaded from the config.

Argument Reference

The following arguments are supported:

- `host` - (Required) Set an alternative Tiller host. The format is host:port. Can be sourced from `HELM_HOST`.
- `home` - (Required) Set an alternative location for Helm files. By default, these are stored in '$HOME/.helm'. Can be sourced from `HELM_HOME`.
- `namespace` - (Optional) Set an alternative Tiller namespace.
- `install_tiller` - (Optional) Install Tiller if it is not already installed.
- `tiller_image` - (Optional) Tiller image to install.
- `service_account` - (Optional) Service account to install Tiller with.
- `automount_service_account_token` - (Optional) Auto-mount the given service account to tiller.
- `debug` - (Optional)
- `plugins_disable` - (Optional) Disable plugins. Set HELM_NO_PLUGINS=1 to disable plugins.
- `enable_tls` - (Optional) Enables TLS communications with the Tiller.
- `insecure` - (Optional) Whether server should be accessed without verifying the TLS certificate. Can be sourced from `HELM_HOME`.
- `client_key` - (Optional) PEM-encoded client certificate key for TLS authentication. By default read from `$HELM_HOME/key.pem`.
- `client_certificate` - (Optional) PEM-encoded client certificate for TLS authentication. By default read from `$HELM_HOME/cert.pem`.
- `ca_certificate` - (Optional) PEM-encoded root certificates bundle for TLS authentication. CBy default read from `$HELM_HOME/ca.pem`.
- `kubernetes` - Kubernetes configuration.

The `kubernetes` block supports:

- `config_path` - (Optional) Path to the kube config file, defaults to `~/.kube/config`. Can be sourced from `KUBE_CONFIG`.
- `host` - (Optional) The hostname (in form of URI) of Kubernetes master. Can be sourced from `KUBE_HOST`.
- `username` - (Optional) The username to use for HTTP basic authentication when accessing the Kubernetes master endpoint. Can be sourced from `KUBE_USER`.
- `password` - (Optional) The password to use for HTTP basic authentication when accessing the Kubernetes master endpoint. Can be sourced from `KUBE_PASSWORD`.
- `token` - (Optional) The bearer token to use for authentication when accessing the Kubernetes master endpoint. Can be sourced from `KUBE_BEARER_TOKEN`.
- `insecure` - (Optional) Whether server should be accessed without verifying the TLS certificate. Can be sourced from `KUBE_INSECURE`.
- `client_certificate` - (Optional) PEM-encoded client certificate for TLS authentication. Can be sourced from `KUBE_CLIENT_CERT_DATA`.
- `client_key` - (Optional) PEM-encoded client certificate key for TLS authentication. Can be sourced from `KUBE_CLIENT_KEY_DATA`.
- `cluster_ca_certificate` - (Optional) PEM-encoded root certificates bundle for TLS authentication. Can be sourced from `KUBE_CLUSTER_CA_CERT_DATA`.
- `config_context` - (Optional) Context to choose from the config file. Can be sourced from `KUBE_CTX`.

Resource: helm_release

A Release is an instance of a chart running in a Kubernetes cluster. A Chart is a Helm package. It contains all of the resource definitions necessary to run an application, tool, or service inside of a Kubernetes cluster.

`helm_release` describes the desired status of a chart in a kubernetes cluster.

Example Usage

```
resource "helm_release" "example" {
  name = "my_redis"
  chart = "redis"
  values = [
    "${file("values.yaml")}"
  ]
}
```

Argument Reference

The following arguments are supported:

- `name` - (Required) Release name.
- `repository` - (Optional) Repository where to locate the requested chart. If is an URL the chart is installed without install the repository.
- `chart` - (Required) Chart name to be installed.
- `devel` - (Optional) Use chart development versions, too. Equivalent to version '>0.0.0-0'. If version is set, this is ignored.
- `version` - (Optional) Specify the exact chart version to install. If this is not specified, the latest version is installed.
- `values` - (Optional) List of values in raw yaml to pass to helm. Values will be merged, in order, as Helm does with multiple `-f` options.
- `set` - (Optional) Value block with custom values to be merge with the values.yaml.
- `namespace` - (Optional) Namespace to install the release into.
- `verify` - (Optional) Verify the package before installing it.
- `keyring` - (Optional) Location of public keys used for verification.
- `timeout` - (Optional) Time in seconds to wait for any individual kubernetes operation.
- `disable_webhooks` - (Optional) Prevent hooks from running.
- `force_update` - (Optional) Force resource update through delete/recreate if needed.
- `recreate_pods` - (Optional) On update performs pods restart for the resource if applicable.
- `reuse_values` - (Optional) Reuse values from previous revision when upgrading a release. Same as `--reuse-values` flag in Helm CLI. Default is false.

- `reuse` - (Optional) Instructs Tiller to re-use an existing name. Default is true.

The `set` block supports:

- `name` - (Required) full name of the variable to be set.
- `value` - (Required) value of the variable to be set.

Attributes Reference

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

- `metadata` - Block status of the deployed release.

The `metadata` block supports:

- `chart` - The name of the chart.
- `name` - Name is the name of the release.
- `namespace` - Namespace is the kubernetes namespace of the release.
- `revision` - Version is an int32 which represents the version of the release.
- `status` - Status of the release.
- `version` - A SemVer 2 conformant version string of the chart.
- `values` - The compounded values from `values` and `set`

Import

helm_release can be imported using the , e.g.

```
$ terraform import helm_release.example ...
```

Resource: helm_repository

A chart repository is a location where packaged charts can be stored and shared.

`helm_repository` describes the desired status of a helm repository.

Example Usage

```
resource "helm_repository" "incubator" {
    name = "incubator"
    url  = "https://kubernetes-charts-incubator.storage.googleapis.com"
}

resource "helm_release" "my_cache" {
    name       = "my_cache"
    repository = "${helm_repository.incubator.metadata.0.name}"
    chart      = "redis-cache"
}
```

Argument Reference

The following arguments are supported:

- `name` - (Required) Chart repository name.
- `url` - (Required) Chart repository URL.
- `key_file` - (Optional) Identify HTTPS client using this SSL key file
- `cert_file` - (Optional) Identify HTTPS client using this SSL certificate file.
- `ca_file` - (Optional) Verify certificates of HTTPS-enabled servers using this CA bundle

Attributes Reference

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

- `metadata` - Status of the deployed release.

The `metadata` block supports:

- `name` - Name of the repository read from the home.
- `url` - URL of the repository read from the home.

Import

helm_release can be imported using the , e.g.

```
$ terraform import helm_release.example ...
```