Intro

Support for `helmfile` with `argo-cd` .

`argo-cd` already supports `helm` in 2 distinct ways, why is this useful?

- It helps decouple configuration from chart development
- It's similar to using a repo type of `helm` but you can still manage configuration with git.
- Because I like the power afforded using `helmfile` 's features such as `environments` , `selectors` , templates, and being able to use `ENV` vars as conditionals **AND** values.
- https://github.com/roboll/helmfile/blob/master/docs/writing-helmfile.md
- https://github.com/roboll/helmfile/blob/master/docs/shared-configuration-across-teams.md

Security

Please make note that `helmfile` itself allows execution of arbitrary scripts. Due to this feature, execution of arbitrary scripts are allowed by this plugin, both explicitly (see `HELMFILE_INIT_SCRIPT_FILE` env below) and implicity.

Consider these implications for your environment and act appropriately.

- https://github.com/roboll/helmfile#templating (`exec` desciption)
- the execution pod/context is the `argocd-repo-server`

Installation

```
    configManagementPlugins: |
      - name: helmfile
        init:                          # Optional command to initialize
application source directory
          command: ["argo-cd-helmfile.sh"]
          args: ["init"]
        generate:                      # Command to generate manifests YAML
          command: ["argo-cd-helmfile.sh"]
          args: ["generate"]



  volumes:
  - name: custom-tools
    emptyDir: {}

  initContainers:
  - name: download-tools
    image: alpine:3.8
    command: [sh, -c]
    args:
      - wget -qO /custom-tools/argo-cd-helmfile.sh
https://raw.githubusercontent.com/travisghansen/argo-cd-
helmfile/master/src/argo-cd-helmfile.sh &&
        chmod +x /custom-tools/argo-cd-helmfile.sh &&
        wget -qO /custom-tools/helmfile
https://github.com/roboll/helmfile/releases/download/v0.138.7/helmfile_linux_amd64
&&
        chmod +x /custom-tools/helmfile
    volumeMounts:
      - mountPath: /custom-tools
        name: custom-tools
  volumeMounts:
  - mountPath: /usr/local/bin/argo-cd-helmfile.sh
    name: custom-tools
    subPath: argo-cd-helmfile.sh
  - mountPath: /usr/local/bin/helmfile
    name: custom-tools
    subPath: helmfile
```

Usage

Configure your `argo-cd` app to use a repo/directory which holds a valid `helmfile` configuation. This can be a directory which contains a `helmfile.yaml` file **OR** a `helmfile.d` directory containing any number of `*.yaml` files. You cannot have both configurations.

There are a number of specially handled `ENV` variables which can be set (all optional):

- `HELM_BINARY` - custom path to `helm` binary
- `HELM_TEMPLATE_OPTIONS` - pass-through options for the templating operation `helm template --help`
- `HELMFILE_BINARY` - custom path to `helmfile` binary
- `HELMFILE_USE_CONTEXT_NAMESPACE` - do not set helmfile namespace to `ARGOCD_APP_NAMESPACE`, for use with multi-namespace apps
- `HELMFILE_GLOBAL_OPTIONS` - pass-through options for all `helmfile` operations `helmfile --help`
- `HELMFILE_TEMPLATE_OPTIONS` - pass-through options for the templating operation `helmfile template --help`
- `HELMFILE_INIT_SCRIPT_FILE` - path to script to execute during init phase
- `HELMFILE_HELMFILE` - a complete `helmfile.yaml` content
- `HELMFILE_HELMFILE_STRATEGY` - one of `REPLACE` or `INCLUDE`
  - `REPLACE` - the default option, only the content of `HELMFILE_HELMFILE` is rendered, if any valid files exist in the repo they are ignored
  - `INCLUDE` - any valid files in the repo **AND** the content of `HELMFILE_HELMFILE` are rendered, precedence is given to `HELMFILE_HELMFILE` should the same release name be declared in multiple files
- `HELMFILE_CACHE_CLEANUP` - run helmfile cache cleanup on init

Of the above `ENV` variables, the following do variable expansion on the value:

- `HELMFILE_GLOBAL_OPTIONS`
- `HELMFILE_TEMPLATE_OPTIONS`
- `HELM_TEMPLATE_OPTIONS`
- `HELMFILE_INIT_SCRIPT_FILE`
- `HELM_DATA_HOME`

Meaning, you can do things like:

- `HELMFILE_GLOBAL_OPTIONS="--environment ${ARGOCD_APP_NAME} --selector cluster=${CLUSTER_ID}`

Any of the standard `Build Environment` variables can be used as well as variables declared in the application spec.

- https://argoproj.github.io/argo-cd/user-guide/config-management-plugins/#environment
- https://argoproj.github.io/argo-cd/user-guide/build-environment/

Helm Plugins

To use the various helm plugins the recommended approach is the install the plugins using the/an `initContainers` (explicity set the `HELM_DATA_HOME` env var during the `helm plugin add` command) and simply set the `HELM_DATA_HOME` environment variable in your application spec (or globally in the pod). This prevents the plugin(s) from being downloaded over and over each run.

```
# repo server deployment
  volumes:
  ...
  - name: helm-data-home
    emptyDir: {}

# repo-server container
  volumeMounts:
  ...
  - mountPath: /home/argocd/.local/share/helm
    name: helm-data-home

# init container
  volumeMounts:
  ...
  - mountPath: /helm/data
    name: helm-data-home

    [[ ! -d "${HELM_DATA_HOME}/plugins/helm-secrets" ]] && /custom-tools/helm-v3
plugin install https://github.com/jkroepke/helm-secrets --version
${HELM_SECRETS_VERSION}
    chown -R 999:999 "${HELM_DATA_HOME}"

# lastly, in your app definition
...
plugin:
  env:
  - name: HELM_DATA_HOME
    value: /home/argocd/.local/share/helm
```

If the above is not possible/desired, the recommended approach would be to use `HELMFILE_INIT_SCRIPT_FILE` to execute an arbitrary script during the `init` phase. Within the script it's desireable to run `helm plugin list` and only install the plugin only if it's not already installed.

Custom Init

You can use the `HELMFILE_INIT_SCRIPT_FILE` feature to do any kind of *init* logic required including installing helm plugins, downloading external files, etc. The value can be a relative or absolute path and the file itself can be injected using an `initContainers` or stored in the application git repository.

Development

```
# format before commit
shfmt -i 2 -ci -w src/argo-cd-helmfile.sh
```

Releases

2 tags

Packages

No packages published

Contributors 4

travisghansen Travis Glenn Hansen

tpatrascu Tiberiu Patrascu

cenkalti Cenk Altı

dmakeroam Sirinat Paphatsirinatthi

Languages

● **Shell** 100.0%