Helm 3 Sneak Preview

What you can expect to see in Helm version 3

JOSH DOLITSKY | SR. SOFTWARE ENGINEER - CODEFRESH | @JDOLITSKY
“k8splace” is developed at a Deis hackathon

Helm v1 is announced at the first KubeCon

Helm merges with Google Deployment Manager

Helm 2.0.0 is released

Helm Summit takes place to kick off discussion on Helm 3

Helm project joins the CNCF
Major changes to expect in Helm 3

- Embedded Lua engine
- Removal of Tiller
- Chart repo auth & upload
Major changes to expect in Helm 3

- Removal of Tiller

- Chart repo auth & upload

Embedded Lua engine
Major changes to expect in Helm 3

- Removal of Tiller
- Embedded Lua engine
- Chart repo auth & upload
Embedded Lua engine

Making Helm charts more robust and programmable
Embedded Lua engine

- Lua can be used with, or in place of, YAML templates
- Treat Kubernetes resources as objects, not strings
- Intercept lifecycle events, modify chart on-the-fly
- Sandboxed - optionally limit access to network/io
- Develop and import reusable “library charts”
- Helm plugins will have access to the Lua runtime
apiVersion: v1
type: Pod
metadata:
  name: {{ template "alpine.fullname" }}
  labels:
    heritage: {{ .Release.Service }}
    release: {{ .Release.Name }}
    app: {{ template "alpine.name" }}
spec:
  restartPolicy: {{ .Values.restartPolicy }}
  containers:
  - name: waiter
    image: "{{ .Values.img }}:{{ .Values.img.tag }}"
    imagePullPolicy: {{ .Values.img.pullPolicy }}
    command: ["/bin/sleep", "9000"]
```yaml
apiVersion: v1
kind: Pod
metadata:
  name: {{ template "alpine.fullname" . }}
labels:
  heritage: {{ .Release.Service }}
  release: {{ .Release.Name }}
  app: {{ template "alpine.name" . }}
spec:
  restartPolicy: {{ .Values.restartPolicy }}
  containers:
    - name: waiter
      image: "{{ .Values.img }}:{{ .Values.img.tag }}"
      imagePullPolicy: {{ .Values.img.pullPolicy }}
      command: ["/bin/sleep", "9000"]
```

Helm 3

```python
function create_alpine_pod(_)
    local pod = {
        apiVersion = "v1",
        kind = "Pod",
        metadata = {
            name = alpine_fullname(_),
            labels = {
                heritage = __.Release.Service or "helm",
                release = __.Release.Name,
                chart = __.Chart.Name .. "-" .. __.Chart.Version,
                app = alpine_name(_)
            }
        },
        spec = {
            restartPolicy = __.Values.restartPolicy,
            containers = {
                {
                    name = waiter,
                    image = __.Values.image.repository .. "",
                    imagePullPolicy = __.Values.image.pullPolicy,
                    command = {
                        "/bin/sleep",
                        "9000"
                    }
                }
            }
        }
    }
    _.resources.add(pod)
end
```
Helm 2

```yaml
apiVersion: v1
type: Pod

metadata:
  name: {{ template "alpine.fullname" . }}

labels:
  heritage: {{ .Release.Service }}
  release: {{ .Release.Name }}

spec:
  restartPolicy: {{ .Values.restartPolicy }}

containers:
  - name: waiter
    image: "{{ .Values.img }}:{{ .Values.img.tag }}"
    imagePullPolicy: {{ .Values.img.pullPolicy }}
    command: ["/bin/sleep", "9000"]
```

Helm 3

```javascript
// Example of using a "library chart"
local pods = require("mylib.pods");

function create_alpine_pod(_)
  myPod = pods.new("alpine:3.7", _)
  myPod.spec.restartPolicy = "Always"
  -- set any other properties
  _.Manifests.add(myPod)
end

```
Removal of Tiller

Improving security by delegating auth to Kubernetes RBAC
Removal of Tiller

- Shrinks the security model for Helm, now client-only
- Auth is delegated to Kubernetes RBAC
- Release history maintained using ULIDs vs. integers
- “Release” CRD will store instance of an application
- “ReleaseVersion” CRD will store version of release
Chart repo auth & upload

New commands and API spec for working with chart repositories
Chart repo auth & upload

- "helm push" command to upload chart to a repo
- API spec for HTTP uploads, based on ChartMuseum
- Plugins can supply custom protocols (e.g. s3://)
- "helm login" command to authenticate against a repo
- OAuth2 authorization flow, bearer/token auth
- Limit which users can upload/install which charts
Helm 2

$ helm repo add myrepo      
   https://site.com/myrepo   
   --username=josh         
   --password=*****

$ helm package mychart/

$ ./my-custom-uploader.sh     
   mychart-0.1.0.tgz

$ helm repo update

$ helm install myrepo/mychart

Helm 3
$ helm repo add myrepo      
  https://site.com/myrepo 
  --username=josh         
  --password=*****

$ helm login https://site.com

$ helm repo add myrepo 
  https://site.com/myrepo

$ helm push mychart/ myrepo

$ helm repo update

$ helm repo add myrepo 
  https://site.com/myrepo

$ helm login https://site.com

$ helm push mychart/ myrepo

$ helm repo update

$ helm repo update

$ helm install myrepo/mychart
Other Helm 3 changes

- “Managed” hooks - if Helm creates something, it will delete it
- index.yaml will move to index.json, and be partitioned for performance
- Schematize your values by including a values.schema.yaml file
- “helm serve” and “helm reset” commands will be removed
- ???
Want to know more about Helm 3?

Helm 3 Design Proposal
https://github.com/helm/community/tree/master/helm-v3

“A First Look at the Helm 3 Plan” by Matt Butcher
https://sweetcode.io/a-first-look-at-the-helm-3-plan/
The platform committed to supporting HELM 3 on day 1!
Thank you!
Want to watch the video for this presentation? Click below!