GitLab

GitLab as Cloud Native

Complex Suite Made Simple with Helm

GitLab

Let’s set some groundwork

Scale

- **5** external dependencies
- **4** forked / parallel charts
- **10** internal charts
- **40+** separate, DRY templates (tpl)

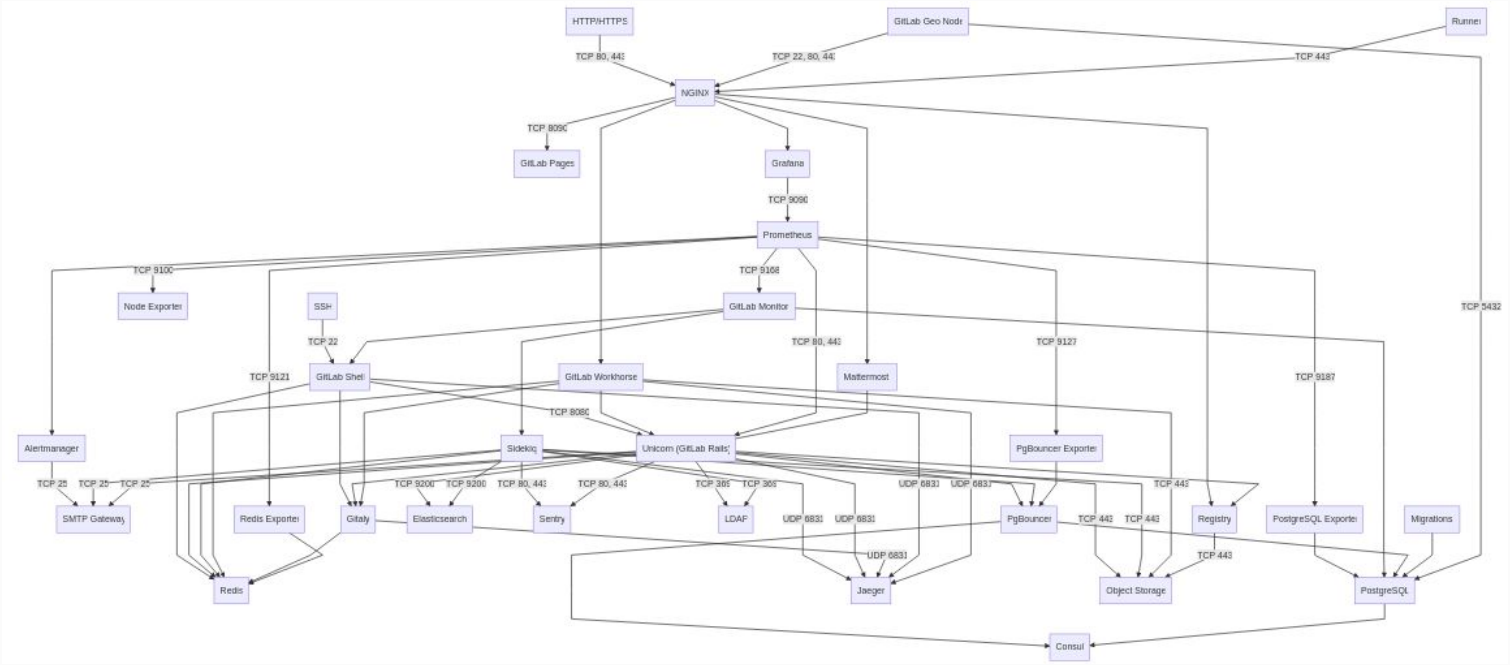
How big is this behemoth?

Including only GA, viable, tested portions

~ 6000 lines

And growing ...

Complexity



@GitLab @WarheadsSE

Target Consumer

Simple to Use

New fangled can’t be harder, or it will never be used.

Expectations of Knowledge

We have to acknowledge that some users will *not be familiar* with Helm. Sometimes, not with Kubernetes

The Challenge

“One command, any cloud”

Production ready, scalable installation of GitLab, in under 10 minutes.

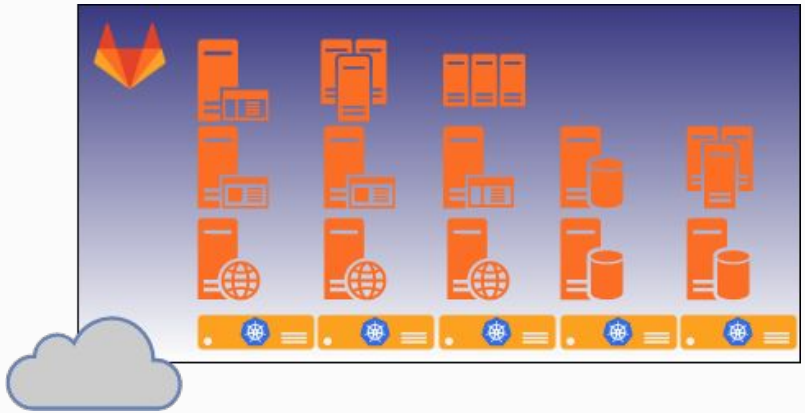
On the provider of your choice.

At the scale you need.

Without the hassle.

```
helm repo add gitlab https://charts.gitlab.io/
helm repo update
helm upgrade --install gitlab gitlab/gitlab \
  --timeout 600 \
  --set global.hosts.domain=example.com \
  --set global.hosts.externalIP=10.10.10.10 \
  --set certmanager-issuer.email=me@example.com
```

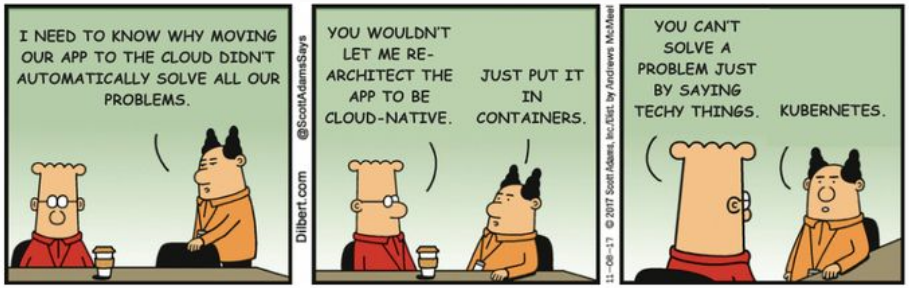
Moving to Cloud Native



Grounding

Head out of the clouds, please!

- Why make a chart *so massive*?
- Why not piece by piece?



Targets

1. Maintainable
2. Flexible
3. Industry Acceptance

Gory Details

Let’s face it.

They’re why you’re here

Templates

Templates Everywhere!

In order to ensure we don't repeat ourselves, we make extensive use of templates *for everything* especially complex or sensitive values.

Remember, Library Charts are not a thin … Yet.

```
$ find . -name "*.tpl"
./templates/_deprecations.tpl
./templates/_migrations.tpl
./templates/_helpers.tpl
./templates/_redis.tpl
./templates/_application.tpl
./templates/_registry.tpl
./templates/_runner.tpl
./templates/_shell.tpl
./templates/_minio.tpl
./templates/_certificates.tpl
./templates/_gitaly.tpl
./templates/_checkConfig.tpl
./templates/_rails.tpl
./templates/_workhorse.tpl
./templates/_runcheck.tpl
./charts/gitlab/templates/_artifacts.tpl
./charts/gitlab/templates/_packages.tpl
./charts/gitlab/templates/_unicorn.tpl
./charts/gitlab/templates/_helpers.tpl
./charts/gitlab/templates/_redis.tpl
./charts/gitlab/templates/_omniauth.tpl
./charts/gitlab/templates/_objectStorage.tpl
./charts/gitlab/templates/_registry.tpl
./charts/gitlab/templates/_configure.tpl
./charts/gitlab/templates/_defaultProjectsFeatures.tpl
./charts/gitlab/templates/_postgresql.tpl
./charts/gitlab/templates/_smtp.tpl
./charts/gitlab/templates/_gitlab.yaml.tpl
./charts/gitlab/templates/_minio.tpl
./charts/gitlab/templates/_operator.tpl
./charts/gitlab/templates/_gitaly.tpl
./charts/gitlab/templates/_uploads.tpl
./charts/gitlab/templates/_externaldiffs.tpl
./charts/gitlab/templates/_pseudonymizer.tpl
./charts/gitlab/templates/_lfs.tpl
./charts/gitlab/templates/_ldap.tpl
./charts/gitlab/charts/unicorn/templates/_helpers.tpl
./charts/gitlab/charts/migrations/templates/_helpers.tpl
./charts/shared-secrets/templates/_helpers.tpl
./charts/redis/templates/_helpers.tpl
./charts/minio/templates/_helpers.tpl
./charts/certmanager-issuer/templates/_helpers.tpl
./charts/registry/templates/_helpers.tpl
./charts/nginx/templates/_helpers.tpl
./charts/redis-ha/templates/_helpers.tpl
```

Templates

Values

Sections

Globals

```
{{/*
Returns the hostname.
If the hostname is set in `global.hosts.gitlab.name`, that will be returned,
otherwise the hostname will be assembled using `gitlab` as the prefix, and the `gitl
ab.assembleHost` function.
*/}}
{{- define "gitlab.gitlab.hostname" -}}
{{- coalesce .Values.global.hosts.gitlab.name (include "gitlab.assembleHost"  (dict
"name" "gitlab" "context" . )) -}}
{{- end -}}
```

Templates: Ground Rules

Template partials for *gitlab/** should be global whenever possible.

All template partials of the gitlab/* sub-charts should be a part of the global or GitLab sub-chart templates/_helpers.tpl whenever possible. Templates from forked charts will remain a part of those charts. This reduces the maintenance impact of these forks.

The benefits of this are straight-forward:

- Increased DRY behavior, leading to easier maintenance. There should be no reason to have duplicates of the same function across multiple sub-charts when a single entry will suffice.
- Reduction of template naming conflicts. All partials throughout a chart are compiled together, and thus we can treat them like the global behavior they are.

Templates: Ground Rules

Breaking changes via deprecation

During the development of these charts, we occasionally make improvements that require alterations to the properties of existing deployments.

Two examples were the centralization of configuring the use of Minio, and the migration of external object storage configuration from properties to secrets (in observance of our preference).

These notifications will cause the helm install or upgrade commands to stop with a parse error, and output a complete list of items that need to be addressed. We have taken care to ensure a user will not be placed into a loop of error, fix, repeat.

Templates: Ground Rules

Attempt to catch problematic config

Due to the complexity of these charts and their level of flexibility, there are some overlaps where it is possible to produce a configuration that would lead to an unpredictable, or entirely non-functional deployment. In an effort to prevent known problematic settings combinations, we have implemented template logic designed to detect and warn the user that their configuration will not work.

This is very similar to Deprecations, but specific to ensuring functional configurations.

Globalization

Pervasive use of .Values.global

All sub-charts of this repository are designed to be deployed via the global chart. Each component can still be deployed individually, but make use of a common set of properties facilitated by the global chart.

Secrets are Secret

ENV is BAD, Sorry!

Much of the container ecosystem has, or expects, the capability to be configured through environment variables. This configuration practice stems from the concept of The Twelve-Factor App.

But why??

```
initContainers:
  {{ include "gitlab.certificates.initContainer" . | nindent 8 }}
  - name: configure
    image: {{ .Values.init.image }}:{{ .Values.init.tag}}
    imagePullPolicy: {{ .Values.init.pullPolicy }}
    command: ["sh", "/config/configure"]
    volumeMounts:
      - name: registry-secrets
        mountPath: /config
      - name: registry-server-config
        mountPath: /registry
    resources:
```

Almost done!

Play Nice Kids!

AKA Thank You Partners!

NGINX

bitnami

JETSTACK

@GitLab @WarheadsSE

Questions?

@GitLab @WarheadsSE

Thank You Helm Community!

@GitLab @WarheadsSE